TCP-Aware Network Coding with Opportunistic Scheduling in Wireless Mobile Ad Hoc Networks

By

Tebatso Nage, BEng.

A thesis submitted to
The Faculty of Graduate Studies and Research
in partial fulfillment of
the requirements for the degree of

Master of Applied Science in Electrical Engineering

Ottawa-Carleton Institute of Electrical and Computer Engineering

Department of Systems and Computer Engineering
Carleton University
Ottawa, Canada

July, 2009

The undersigned hereby recommends to the Faculty of Graduate Studies and Research acceptance of the thesis

TCP-Aware Network Coding with Opportunistic Scheduling in Wireless Mobile Ad Hoc Networks

Submitted by

Tebatso Nage, BEng.

In partial fulfillment of the requirements for the degree of
Master of Applied Science in Electrical Engineering

Thesis Co-Supervisor
Prof. F. Richard Yu

Thesis Co-Supervisor
Prof. Marc St-Hilaire

Chair, Department of Systems and Computer Engineering
Prof. Howard Schwartz

Carleton University
Ottawa, Ontario, Canada
July 2009

Abstract

This thesis presents a scheme that employs TCP-Aware network coding with opportunistic scheduling to enhance TCP performance in wireless mobile ad hoc networks. Specifically, it considers a TCP parameter, congestion window size and wireless channel conditions simultaneously to increase TCP throughput. Evaluation of this scheme is carried out by using ns2 simulations in different scenarios such as network traffic increase, network node increase, and low/high mobility environments. The results found show that this scheme gives approximately 35% throughput improvement in a high mobility environment and about 33% throughput improvement in no or low mobility environment as compared to traditional network coding with opportunistic scheduling. This thesis also proposes a new adaptive-W scheme whose objective is to adaptively control waiting time of overheard packets that are stored in a buffer to achieve tradeoff between throughput and overhead.

Dedications

Dedicated to my father, Samuel Matsuokwane Nage and in memory of my mother, Lydia Albertina Nage.

Acknowledgements

I would like to thank the invaluable supervision and support of my supervisors Professor Fei Richard Yu and Professor Marc St-Hilaire during the development of this work. I also would like to thank my family in Botswana for their support and love. My wholehearted thanks go to Monageng Kgwadi for his assistance in ns2.

Contents

List of Figures

List of Tables

List of Acronyms

ACK	Acknowledgement
BER	Bit Error Rate
EM	Electromagnetic
HARQ	Hybrid Automatic Repeat reQuest
IFQ	Interface Queue
IP	Internet Protocol
MAC	Medium Access Control
NC	Network Coding
ns2	Network Simulator 2
OS	Opportunistic Scheduling
OSI	Open System Interface
PER	Packet Error Rate
PNC	Physical layer Network Coding
RTO	Retransmission TimeOut
SINR	Signal to Interference plus Noise Ratio
TCP	Transmission Control Protocol
WiMAX	Worldwide Interoperability for Microwave Access

Chapter 1

Introduction

1.1 Background Information

This section presents background information on network coding (NC), opportunistic scheduling (OS), transmission control protocol (TCP), cross-layer design approach and wireless ad hoc networks which play integral part in this thesis.

1.1.1 Network Coding

Network coding is a new transmission paradigm pioneered by Ahlswede *et al.* [2]. In recent years, it has generated huge research interest especially in wireless communication. The main attraction of this novel concept is that it is bandwidth efficient and achieves high throughput gains [3, 4, 5, 6, 7, 8]. Through experiments [9], it has been found that network coding makes it possible to perform peer-to-peer live multimedia streaming with finer granularity. Network coding can be regarded as an extension of the traditional routing protocol in which intermediate nodes store and forward packets. The basic idea about network coding is that packets are intelligently mixed (or coded) together at an intermediate node into one coded packet which is then broadcast. This process facilitates not only the generation of a few number of transmissions

in the network but also information rich transmissions. Furthermore, with network coding, more bandwidth becomes available for new data to be transmitted resulting in high network throughput [10].

Network coding comes in different forms. Some studies consider what is referred to as random network coding [1, 3, 4, 9, 10, 11]. There is also physical layer network coding [12, 13]. In random network coding, encoding coefficients are randomly chosen from a set of coefficients of finite field. A linear combination of packets is then performed to generate a coded packet. At the receiver, decoding becomes possible if and only if the receiver can generate a full rank transfer matrix made of coefficients extracted from received coded packets [9]. This means that if there are K encoding coefficients in a coded packet, the receiver will have to receive at least K independent versions of the coded packet for successful decoding. Physical layer network coding, on the other hand, facilitates simultaneous reception of electromagnetic (EM) waves of signals at air interface [12] which are then aggregated. This approach achieves higher network throughput than random network coding. However, it introduces interference at the node's air interface hence high design complexity.

Although network coding was initially designed for wired networks which employ multicast transmission mode, it is currently one of the hottest research topics in wireless communications because of its capability to exploit broadcast nature of wireless networks. It does so by letting nodes snoop on transmissions going on in the network. Even though some packets may not be intended to a node of interest, such a node captures and stores packets it can successfully receive. When this node gets an opportunity to transmit its data, it then includes the information about the packets

it has overheard from the network in the outgoing packet. Such information is called reception report. The idea is to share with neighbors information about packets that have been overheard. By so doing, this generates coding opportunities. Coding packets is based on reception reports received from neighboring nodes. Decoding of coded packets is carried out by using packets previously sent or overheard from the network whose information has been shared with neighbors. Combining packets into a single one can substantially enhance performance. Infact, since fewer transmissions are generated, throughput is more likely to increase.

To elaborate more on this, consider Fig. 1.1 and Fig. 1.2. In the traditional routing protocol, a relay node simply receives, stores and forwards packets as shown in Fig. 1.1. As a result of this routing mechanism, Fig. 1.1 illustrates that it takes four transmissions for nodes A and B to send each other only one message. However, by employing XOR network coding, nodes A and B can send each other one message in only three transmissions as shown in Fig. 1.2.

The secret is rooted in the fact that the relay node does not only store and forward messages but also codes packets together and broadcasts a coded version of messages. Suppose packet P_1 has bit stream $10101110011010\ldots$ and packet P_2 has bit stream $010001101010100\ldots$ in their data portions as shown in Fig. 1.3. When the relay node receives packets P_1 and P_2, it combines them by performing bitwise XOR operation to produce a new packet $P_3 = P_1 \oplus P_2$ where $\oplus$ stands for bitwise XOR operation. Note that before bitwise XOR operation is carried out at the data link layer, XOR headers and MAC headers for packets P_1 and P_2 are unwrapped and the remainder is only the data portion which includes internet protocol (IP) header. After the operation,

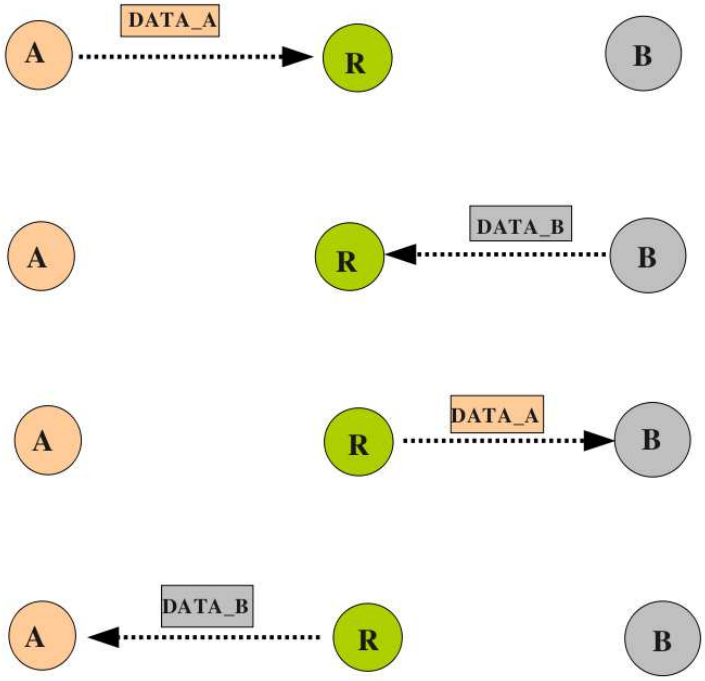

Figure 1.1: Traditional routing protocol.

P_3 has bit stream 111010011001110 . . . in its data portion as shown in Fig. 1.3.

Upon receiving the broadcasted coded message P_3, node A uses a copy of packet P_1 it sent earlier to extract P_2 from P_3 by performing bitwise XOR operation as illustrated in Fig. 1.4. Again, XOR header and MAC header are removed from P_3 before the operation. Node B on the other hand, uses a copy of packet P_2 to extract packet P_1 from P_3 as depicted in Fig. 1.5. Note that the remaining transmission could be used to send new messages hence more efficient bandwidth utilization and high throughput by network coding. It must be noted however that network coding is heavily dependent on network traffic pattern, medium access link scheduling and topology. For example, Fig. 1.2 only shows a scenario in which nodes A and B first send their messages. This link scheduling facilitates network coding. However,

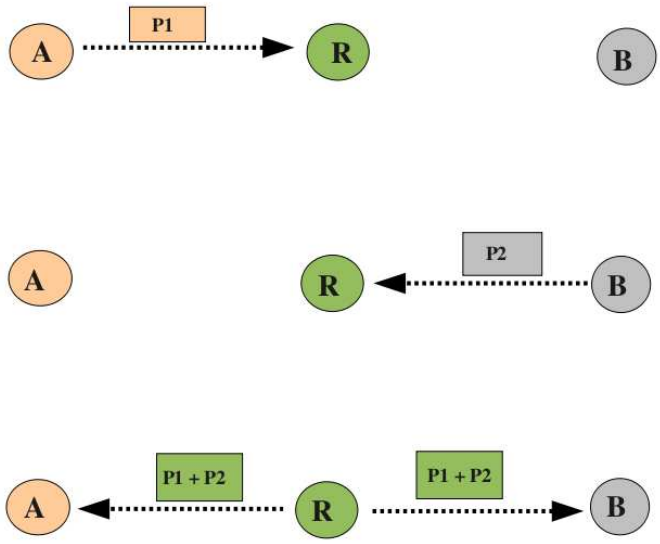

Figure 1.2: Basic principle of XOR network coding.

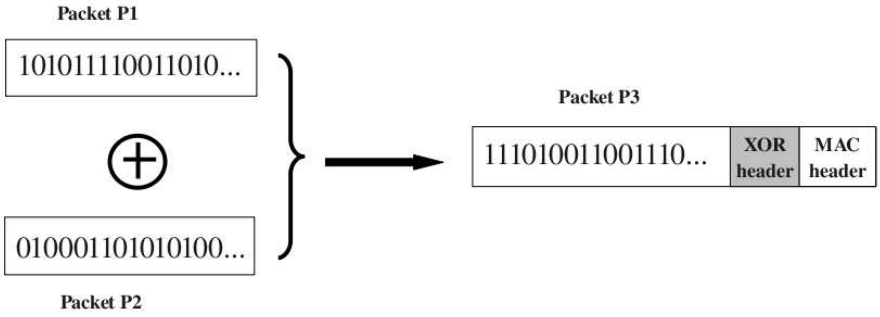

Figure 1.3: Coding packets at a relay node.

this link scheduling scenario is not always the case in real networks. Consider link scheduling scenario in which node A first sends its message followed by relay node and then node B. In this scenario, network coding in not possible because the relay node only has one message to forward. To facilitate network coding, some studies have considered what is referred to as coordinated network coding in which traffic flows are made to overlap so as to generate coding opportunities. However, this findings have been criticized for being unrealistic [14].

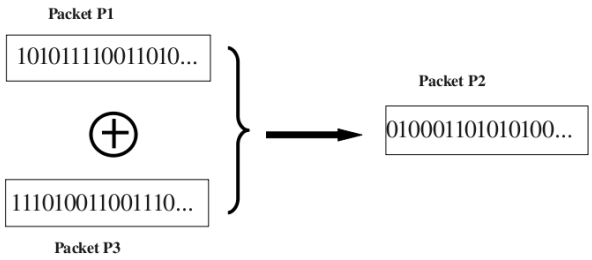

Figure 1.4: Extract p2 from p3 at node A.

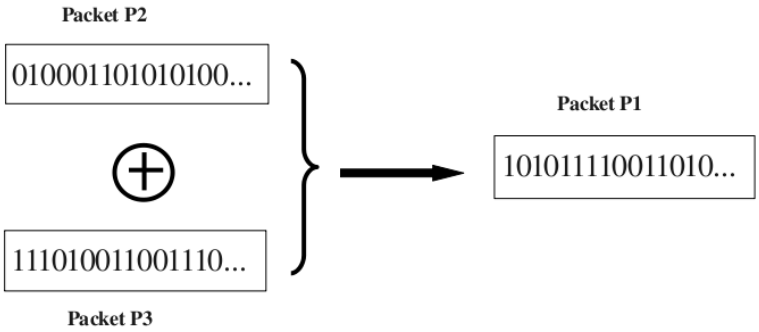

Figure 1.5: Extract p1 from p3 at node B.

1.1.2 Opportunistic Scheduling

In reality, wireless channel conditions fluctuate. At times they are favourable but other times they are unfavourable, a situation referred to as fading. When a node has channel information, it can take advantage of stochastic fluctuation of channel conditions by transmitting at high data rates where applicable to maximize throughput. Also, a node could halt transmission when wireless links are poor to avoid creating interference to other nodes and to save power. In order to have access to channel information, nodes periodically send each other information of outgoing channel conditions using feedback channels. After receiving this information, a node then makes some statistical analysis from which it makes decisions about its outgoing

channel conditions. Although this scheme is promising, the downside of it is that it introduces stochastic halt of data transfer which can degrade network performance as some packets may be kept at a node for too long or that they may be received out of order.

1.1.3 Transmission Control Protocol

Transmission Control Protocol (TCP) is a connection-oriented, reliable transport layer protocol. It is the dorminant transport layer protocol in today's Internet. Many Internet applications such as email, file transfer and web browsers use TCP as their transport layer protocol [15, 16, 17]. TCP was originally designed for wired networks in which bit error rate (BER) is lower than 10^{-8} [15, 16]. This protocol was designed based on the assumption that packet losses due to damage in transit is rare, hence most of packets get lost due to network congestion [16, 18]. Even though this assumption is valid for wired networks, it may not be valid for wireless networks in which it is difficult to provide channel quality of bit error rate of 10^{-6} [16]. The reason being that in wireless networks, most packet losses are due to high bit error rate as opposed to network congestion. Therefore, TCP tends to misinterpret packet losses due to bit errors in wireless networks as congestion thereby slowing down its transmission unnecessarilly resulting in performance deterioration.

1.1.4 Cross-Layer Design

The layered structure in Open System Interface (OSI) model has been praised for its simplicity in maintaining, managing and optimizing protocols. However, it has

been found that this approach introduces inefficiency, redundancy and that it can lead to poor performance more espcially in applications with hard quality of service constraints. Therefore, a promising cross-layer design approach has been introduced. This approach allows information exchange across layers which offers substantial gains in efficiency, throughput and quality of service. It comes handy in wireless networks in which there is mobility, frequent handoffs and fluctuation of channel conditions. For example; suppose a particular node in a wireless mobile ad hoc network has TCP sending rate which is increasing gradually and low data rate is used yet the current wireless channel conditions are favourable such that the highest data rate could be used. If layered structure is employed, such a node may fail to take advantage of an opportunity to transmit at a high data rate due to lack of efficient coordination among protocol layers. However, when cross-layer design approach is used, the data link layer could use the TCP and channel information received from the transport and physical layers respectively to determine the transmission data rate that can help speed up the rate at which TCP sending rate is increasing. Furthermore, if wireless channel conditions are poor such that even the lowest modulation scheme cannot be used, by employing channel information from physical layer, opportunistic scheduling scheme in the data link layer could temporarily halt transmission to allow conditions to improve.

1.1.5 Wireless Mobile Ad Hoc Networks

Wireless mobile ad hoc networks are networks in which there is no infrastructure. Nodes in these kind of networks communicate through each other. That is, a node

could be the sender, the receiver or the intermediate node. There could be several intermediate nodes in one session in which the message goes through several hops from the sender to the receiver. At times, there are no intermediate nodes in which the sender communicates directly with the receiver. Nodes send each other routing messages periodically to determine best routes. These routes dynamically change due to wireless link conditions and mobility. Furthermore, nodes are free to join or leave the network at any time. Since there is no central point of network control, security is a issue. Also, the idea of using one's mobile device to route somebody elses data is a challenge as battery life of a mobile device is limited.

1.2 Objective

The aim of this thesis is to develop TCP-Aware network coding with opportunistic scheduling to increase TCP throughput in wireless mobile ad hoc networks. To that end, this thesis also proposes an adaptive-W scheme whose purpose is to adaptively control packet waiting time in packetPool to achieve tradeoff between throughput and overhead.

1.3 Motivation

The motivation behind this work can be described as follows: If TCP and wireless channel information is considered simultaneously in network coding for determining transmission data rates, throughput may be increased in wireless mobile ad hoc networks. Furthermore, it would be costly to deploy a modified version of TCP. Therefore, an efficient scheme which employs the current TCP variant (TCP Reno)

as is without any changes could cut down deployment cost. Moreover, information exchange required in network coding could degrade network performance. Therefore, an adaptive scheme for controlling overhead could increase network throughput and provide efficient bandwidth utilization.

1.4 Thesis Contributions

We designed a robust and resilient scheme which simultaneously considers TCP and channel information in traditional network coding to increase TCP throughput in wireless mobile ad hoc networks. Also, we designed an adaptive-W scheme whose aim is to adaptively control packet waiting time in the packetPool to achieve tradeoff between throughput and overhead. In addition to this, we submitted two conference papers for publication. These are;

- Tebatso Nage, F. Richard Yu and Marc St-Hilaire,"TCP-Aware Network Coding with Opportunistic Scheduling in wireless mobile ad hoc networks ", *submitted.*
- Tebatso Nage, F. Richard Yu and Marc St-Hilaire,"Adaptive Control of Packet Overhead in XOR Network Coding ", *submitted.*

1.5 Thesis Organization

The rest of this thesis is organized as follows. Related works and problem definition can be found in chapter 2. Chapter 3 presents the system models. TCP-Aware network coding scheme with opportunistic scheduling is presented in chapter 4. Chapter 5 presents adaptive-W scheme and we conclude our work in chapter 6.

Chapter 2

Related Works and Problem Definition

2.1 State of the Art in Network Coding

This section presents related works in network coding. Specifically, it covers methods of network coding and overhead aspect of network coding.

2.1.1 Methods of Network Coding

There are mainly two methods of network coding that have been identified in the literature. These are physical layer network coding and random network coding.

Physical Layer Network Coding

Rather than a blessing, broadcast nature of wireless networks is treated as an interference-inducing nuisance in most of wireless networks today such as in IEEE 802.11 [19]. Zhang *et al.* [19] proposed a physical layer network coding (PNC) scheme which embraces interference to improve performance in a multihop network. Specifically, a relay node does not perform coding arithmetic on digital bit streams after they have been received which is a requirement in straightforward network coding. Rather it makes use of the additive nature of the simultaneously arriving electromagnetic waves by directly mapping the combined signals received simultaneously to a signal to be

relayed. In order to achieve this, 1) a relay node must be able to convert simultaneously received signals into sensible output signals to be relayed; 2) A destination node must be able to extract its intended signal from the relayed signals.

Xue *et al.* [20] proposed a simple adaptive coding scheme that combines channel coding with network coding which greatly simplifies the complexity in code design and decoding as compared to pure random network coding. Specifically, the scheme first designs codes independently for each channel, then combines them with simple network coding to achieve performance comparable to ideal cases. Part of the motivation for [20] is that in XOR network coding, coded data packet first has to be decoded by intended receivers before being XOR-ed with old known packets. According to information theory [20], this limits information rates in the broadcast stage to the weakest link through which one of the intended receivers receives its version of the coded packet.

Zimmermann *et al.* [13] came up with an analytical cross-layer optimization approach for network coding problems and analyzed the network coding solution of exemplarily chosen unicast problems. Their work was inspired by the fact that network coding gains are generally only achieved for multicast problem formulations but many scenarios in a network's daily operations consists mainly of unicast flows.

Random Network Coding

Sundararajan *et al.* [21] came up with a new network coding scheme which is inserted between transport and network layers. This scheme allows random linear combination of packets currently in TCP sender's congestion window. They claim that, this

scheme reacts to packet drops in a smooth manner resulting in a novel and effective approach for congestion control over networks involving lossy links such as wireless links. Since receiver does not receive original packets of the message but receives linear combination of packets, the notion of ordered sequence of packets used by TCP is missing. Therefore, there is need to change TCP mechanism that acknowledges packets at the receiver for this scheme to work.

Due to the resilience and stable transmission offered by random network coding, Jin *et al.* [4] have proposed adaptive random network coding in Worldwide Interoperability for Microwave Access (WiMAX). The objective is to optimize WiMAX network performance in which it has been established that hybrid automatic repeat request (HARQ) is under-utilizing wireless bandwidth because it was designed for point-to-point channel. As an attempt to minimize delay created by waiting for full rank transfer matrix to be formed, [9] has implemented decoding process using Gauss-Jordan elimination technique which allows decoding to be carried out while coded packets are progressively received. Although Gauss-Jordan elimination usually leads to numerical instability [9], it does not affect network coding since the operation is done in the Galois field. It has been found that when Galois field size is increased, success probability of decoding is increased [10]. However, this introduces additional system design and computational complexity. Therefore, a special case of random network coding called XOR network coding which uses Galois field of size 2, has been identified suitable to address this problem. It remains a promising technique for coding packets in future networks [1, 5].

In this type of network coding, packets are coded using bitwise XOR operation

into one coded packet. At the receiver, decoding is then carried out by applying bitwise XOR operation again to the received coded packet using packets which were previously overheard from the network or those that were previously sent. Lin *et al.* [22] analyzed average capacity for a wireless network with joint opportunistic scheduling and wireless network coding. From their analysis, they found out that when more packets are coded together into a single packet, there is high probability of some intended nodes experiencing deep fading. Therefore, they proposed a scheme which uses power control and rate adaptation in network coding to optimize packet delivery of coded packets. Sagduyu *et al.* [23] considered the joint design of network coding and medium access control (MAC) in wireless ad hoc networks. They also outlined an extension of network coding to operate with arbitrary MAC protocols.

Yomo *et al.* [1] addressed a scheduling problem in which they did not only consider information from neighboring nodes for network coding, but also considered instantaneous wireless link conditions. The idea was to find a set of packets whose destinations have good quality links and optimize data rate for the coded packet. Katti *et al.* [5] proposed a new architecture in wireless mesh networks called COPE that employs the traditional network coding. They considered unicast traffic, dynamic and potentially bursty flows in which COPE demonstrated significant performance improvement. However, COPE demonstrated little performance improvement in collision-related packet losses. Authors of [24, 25] realised that opportunistic coding scheme (COPE) introduced by Katti *et al.* misses several coding opportunities due to random link access scheduling at data link layer. They proposed coordinated network coding scheme to fully utilize coding opportunities missed by COPE by

making sure that neighboring nodes transmissions are scheduled such that network coding gain is maximized [14]. Regrettably, according to Koutsonikolas *et al.* [14], these works either provide only theoretical results, making unrealistic assumptions of a slotted MAC layer or only consider a superficial traffic pattern, consisting of pairs of perfectly overlapping flows going towards opposite directions.

2.1.2 Overhead aspect of Network Coding

In network coding, there is need for network nodes to exchange information in order to carry out successful encoding and decoding of packets. However without proper measures, this can lead to high network overhead which can significantly degrade network performance. This is due to the fact that overhead introduces additional delay, congestion, energy consumption and inefficient bandwidth utilization. Research studies show that there are ways by which this overhead can be minimized.

In their work, Chou *et a.l* [26] proposed a buffer model which employs traditional generation-based network coding. Their scheme involves grouping packets according to generations and linear combination of packets that exist withing the same generation. The idea was to minimize or limit packet overhead generated by coefficient vector at an expense of coding opportunities. Although this approach showed some performance improvements according to their findings, it is susceptible to packet loss that is facilitated by flushing old generations. Also, the fact that non-innovative packets are discarded at the receiver is inefficient as it consumes bandwidth.

Halloush *et al.* [27] proposed a better approach which employs Multi-Generation Mixing and eliminates the need to increase buffer size while improving performance.

In their scheme, packets are grouped into generations and generations are grouped into mixing sets. Packets belonging to new generation can be linearly combined with those from old generations. This ensures that receivers which may not have received full rank matrix to perform decoding on previously received coded packets could use information in newly arriving coded packets to perform decoding and thus guaranteeing reliability and reduced overhead. Although this approach is better than the former, it may experience performance deterioration due to increased overhead if network traffic is high such that each generation contains lots of packets to be sent out. In [28], Kim presented concatenated random parity forwarding technique that enables network coding gain to increase with each additional hop in wireless networks. This scheme employs partial network coding in that a relay node combines and sends out parity bits (instead of entire codewords) received from neighboring nodes. The destination node then combines information bits from direct path from source nodes, and parity bits from relay nodes to improve reliability and reduce overhead. However, the main challenge for this scheme is that an extra control information overhead may be required to coordinate relay encodings.

In [4], Jin *et al.* designed an adaptive random network coding in WiMAX by adapting the number of upstream nodes and dynamically adjusting block size in response to channel conditions. In their scheme, a downstream node prematurely sends feedback to a subset of upstream nodes to stop transmission in order to minimize redundant blocks and thus reducing protocol overhead. To effectively deal with overhead, Prasad *et al.* [29] proposed a new encoding strategy, XOR-TOP which employs local topology to effectively estimate the available non-coded or *native* packets at

neighboring nodes. They claim that their scheme which does not employ information exchange among neighboring nodes, can always accurately identify coding opportunities according to local topology. However, their claim on accurate identification of coding opportunities is questionable as wireless link conditions are unpredictable. Therefore, there is a chance of making an error in estimating available native packets at neighboring nodes more especially in high mobility environment in which local network topology frequently changes. Although reliability is crucial, the idea of acknowledging each and every packet sent (as in [29]) can degrade performance because of additional overhead introduced more especially in a network with high TCP traffic in which MAC and transport layer performs retransmissions.

Katti *et al.* [6] employed bit-map format in packet's reception report to report packets which have been overheard by a node. Even though the representation for reception reports in their approach has two advantages of compactness and effectiveness, their approach of handling overheard packets has some shortcomings. Consider a network with high traffic such that many nodes send packets. According to their approach which employs fixed packet waiting time, there is high probability of many packets being overheard by nodes. Therefore, packet overhead is more likely to be high due to long reception reports thereby degrading network performance. Furthermore, since more packets will be overheard, more memory will be required at each node. Also, high power consumption will be required to transmit packets with high overhead thereby increasing cost.

2.2 Problem Statement

The main problem of this thesis is how to efficiently enhance XOR network coding performance in wireless mobile ad hoc networks so as to increase TCP throughput by considering both opportunistic scheduling and TCP dynamics simultaneously.

This thesis employs XOR network coding because it is the simplest and cheapest method of network coding. It is simple because it employs the same technique for encoding and decoding packets. Also, it employs Galois field of size 2 which involves less complex system design and computations than other random network coding schemes whose Galois field sizes are greater than 2. This makes XOR network coding affordable. Although physical layer network coding can significantly improve throughput by its capability to process simulataneously arriving signals, it requires comprehensive changes in the physical layer, data link layer and the network layer architecture. This introduces high design complexity which increases cost.

The poor performance of TCP in lossy links such as wireless links is well known in the literature. Although attempts have been made to maximize its throughput in those lossy links, performance still remains low. The scheme proposed by Sundararajan *et al.* [21] is promising as it involves linear combination of packets in current TCP window. However, the fact that some TCP modifications need to be done at the receiver side introduces some complexity which might make it difficult for this scheme to be deployed in real networks. Also, without opportunistic scheduling, this scheme fails to take advantage of stochastic fluctuations of wireless channel conditions by transmitting at high rates when applicable. Therefore, maximazation of

TCP throughput is limited. [1] and [22] only account for opportunistic scheduling in network coding. Although this improves performance, it may introduce congestion by employing opportunistic scheduling scheme alone. Specifically, suppose TCP congestion window size is large and the channel through which the next TCP packet is transmitted is of low quality such that it permits transmission via the lowest modulation scheme. Since a node only has channel knowledge but has no information on TCP sending rate, it chooses low modulation scheme for such transmission. This inevitably introduces some delay in transmitting such TCP data packet which in turn may facilitate congestion at such a node as more TCP packets accomulate in the node queue. As a result TCP timeout event will be triggered thereby degrading TCP performance.

Furthermore, suppose the link for a TCP data packet is poor. In this case, the node decides not to transmit at all. Although this is efficient in avoiding interference to other nodes and promotes network resource sharing, if it occurs quite frequently, it may lead to TCP timeout due to too much time spent by packets waiting to be transmitted thereby degrading network performance. There is high chances of this occuring in high network traffic environment. In addition to this, suppose now that the node has TCP data packets to be transmitted in its queue. If the link for a packet (say P_1) at the head of the queue is poor, by employing opportunistic scheduling alone, the node will choose a different TCP data packet (say P_2) heading to a different hop with a good link. If this packet belongs to the same TCP connection as packet P_1 and it arrives before packet P_1, then packets will arrive out of order. If no packet re-ordering is performed before passing packets up the protocol stack, TCP's fast

retransmission and fast recovery at the TCP sender may occur there by degrading performance.

Also, opportunistic scheduling on network coding does not perform well on coded TCP data packets since it is only based on channel information. Specifically, suppose TCP data packets $P_1, P_2, \ldots, P_n$ are coded together. By employing opportunistic scheduling alone, the transmission rate that is chosen is limited by the weakest link of one of the intended receivers. This will degrade performance if high percentage of coded TCP packets require high data rates. Note that network coding by itself without opportunistic scheduling and TCP dynamics leads to tremendous performance deterioration which is facilitated by spurious generation of coding opportunities.

From the analysis above, it can be learnt that little performance benefits in TCP is obtained without taking into account TCP dynamics. Therefore, in order to increase TCP throughput in wireless mobile ad hoc networks, both channel and TCP information are essential for network coding.

2.3 Summary

This chapter presented research works related to the work done in this thesis. Specifically, it discussed different methods of network coding found in the literature which are physical layer network coding and random linear network coding. Different strategies proposed in the literature for solving the problem of overhead were also presented. Furthermore, problem definition and contributions from this thesis were discussed.

Chapter 3

System Models

This chapter presents a detailed description of the traditional network coding which is the foundation of the entire scheme. Furthermore, the channel and TCP models employed are discussed. Also, it presents the system design of TCP-Aware network coding with opportunistic scheduling as implemented in a node.

3.1 Network Coding

This approach exploits the broadcast nature of wireless networks to facilitate coding opportunities. Nodes are made to snoop on all transmissions going on in the network. Overheard packets are captured and stored. Each node has three buffers. These are the interface queue (IFQ), packetPool and the lowPriorityBuffer. The IFQ exists in all nodes regardless of whether they use network coding or not. This buffer is used to store packets generated by a node or those to be forwarded. PacketPool and lowPriorityBuffer are used for storing overheard packets from the network. They can store up to 100 packets. See section 3.4.4 for details. Whenever a node gets an opportunity to transmit, it adds reception report in the outgoing packet XOR header as a means of sharing information with other nodes. When other neighboring nodes overhear such a packet, they extract the reception report and update their

local information. In Fig. 3.1, nodes that are within communication range of each other can overhear each other's transmitted packets. For example, nodes A and C are within communication range of node E. Therefore, they can overhear packets sent by node E provided wireless link conditions are favourable.

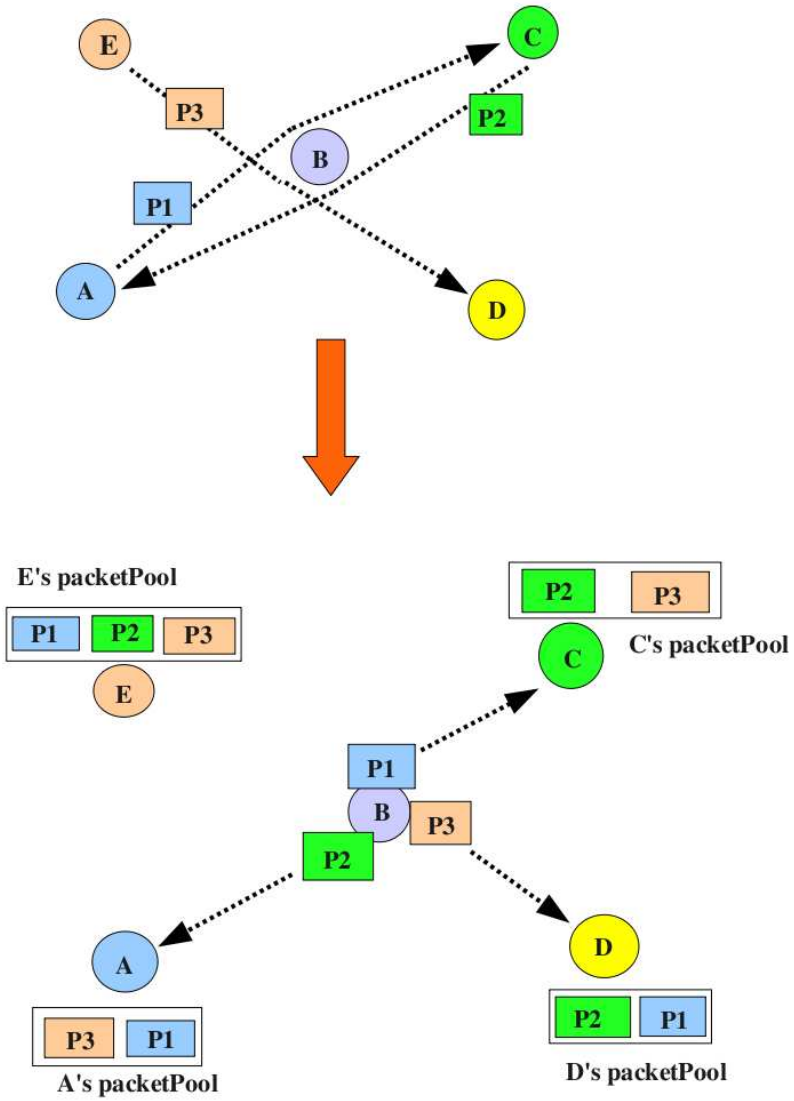

Figure 3.1: XOR network coding [1].

3.1.1 Coding Packets

Assuming that nodes have shared their reception reports; when a node has data to transmit, the scheduler first checks reception reports received from neighboring nodes to determine which packets to code together. A coding opportunity is said to exists if four conditions are met. These are: 1) there has be two or more packets in the set of packets to be coded together. 2) none of the packets in the set has to be generated by the node carrying out network coding [6]. 3) there has to be at most one packet going to one unique next hop in the set [6]. 4) the intended recipients of the coded packet need to have $\kappa - 1$ packets of those in the set where κ is the total number of packets in the set. In Fig. 3.1, node B has packets P_1, P_2 and P_3 to be forwarded to their respective hops. Since all four conditions are met, node B can code packets P_1, P_2 and P_3 together based on reception reports it has previously received from its neighboring nodes. Recall that during encoding of packets, the data portions of packets which include IP and TCP headers are coded together. The XOR and MAC headers are never coded. It must be noted also that before searching for coding opportunities, packets that are still in lowPriorityBuffer but have already reached their destinations are discarded. Last received reception reports are used to determine whether a packet in lowPriorityBuffer has been received by its destination node or not. It is very important for a node to be aware of the number of its neighboring nodes in order to consider XOR network coding. If a node only has one neighboring node to forward its packets to, it cannot perform XOR network coding because of the following reason. Since there must be at most one packet in set to

be sent to one unique hop according to condition 3, it means that condition 1 which requires atleast two packets in the set of packets to be coded will not be satisfied. Therefore, conditions 1 and 3 are both not satisfied. Fig. 3.2 illustrates flowchart of execution of XOR network coding at a node.

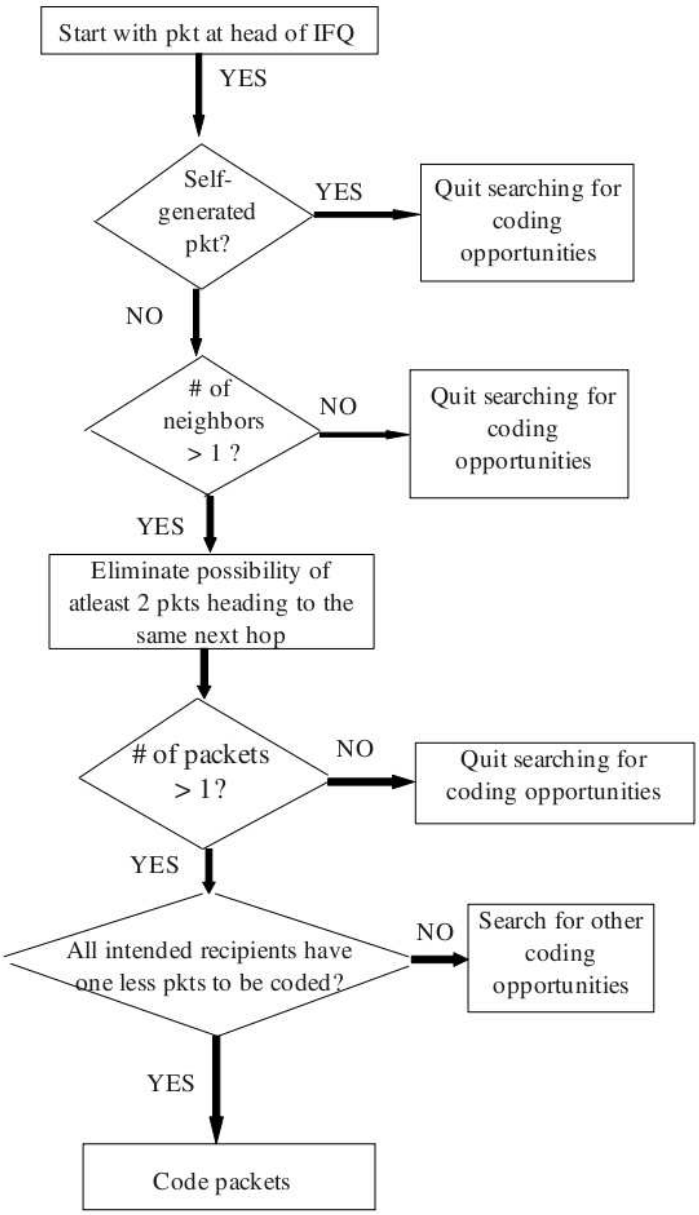

Figure 3.2: Flowchart for XOR network coding.

3.1.2 Searching for Coding Opportunities

Recall that the purpose of network coding is to maximize network throughput. Therefore, performing network coding involves more than just searching for coding opportunities. Nodes have to search for a coding opportunity which maximizes throughput. That is, if there exists a coding opportunity which has more *native* packets than others, it will be chosen. Packets in the IFQ and lowPriorityBuffer are considered for coding. They are arranged according to their next hops as shown in Fig. 3.3 where $P_1, P_2, \ldots, P_i$ represents packets. Using sequence numbers, packets in the first row are to be sent first to their respective next hops followed by those in the second row and so on. P_1 is the packet at the head of the IFQ. Packets heading to the same next hop as P_1 are eliminated from potential set of packets to be coded. This is done because there can only be one packet heading to the same hop in a coded packet.

Note that there are two ways identified through which the number of packets to code can be maximized. These are: 1) by employing exhaustive search algorithm and 2) by using local search algorithm. Exhaustive search algorithm has a large solution space as compared to local search algorithm because it searches for all coding opportunities. For example, Fig. 3.4 shows some coding opportunities which local search algorithm will not consider. Even though exhaustive search algorithm seems to perform better than local search algorithm, it has weaknesses which can severely deteriorate network performance. These are: time delay, high design complexity, high power consumption and node memory.

Time is a valuable resource in wireless networks because of scarce bandwidth.

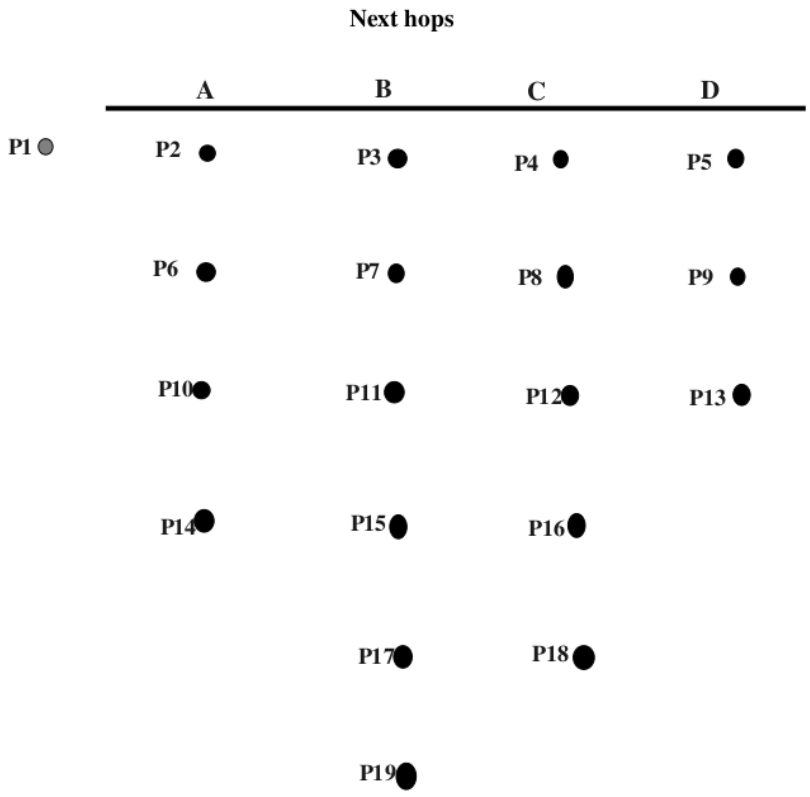

Figure 3.3: Packets arranged according to their next hops.

Once a node has been granted access of wireless medium, it has to transmit its data within a time limit beyond which performance goes down. Therefore, if an exhaustive search algorithm is employed, there are high chances of a node taking too much time searching for a coding opportunity before sending its data. This is more likely to happen in a network with high traffic where packets are overheard or forwarded in abundance. Note that spending more time searching does not necessary mean that a node will eventually find a coding opportunity or that a coding opportunity with high number of packets to code will be found. Furthermore, exhaustive search algorithm requires higher system design complexity which increases cost. Also, to perform exhaustive search requires high power consumption. However, a mobile node has

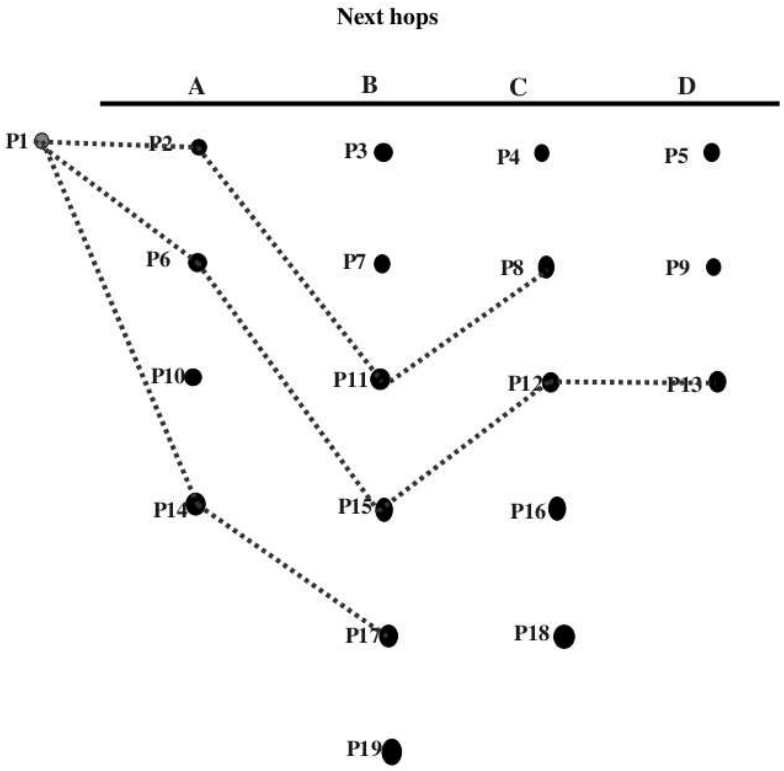

Figure 3.4: Exhaustive search algorithm.

limited battery life which has to be used efficiently. Also, computations performed by exhaustive search algorithm require more node memory thereby increasing cost.

Therefore, there is a need to come up with a searching strategy that can efficiently utilize network and/or node resources and maximize information content of coded packets. This thesis employs local search algorithm which has been identified as a better method as far as network/node resources are concerned. Even though it does not search for all coding opportunities, it uses network/node resources more effectively. Also, it employs maximization strategy on number of packets to code in all coding opportunities found. Fig. 3.5 illustrates how local search algorithm searches for coding opportunities. The search process is performed by varying packets in one

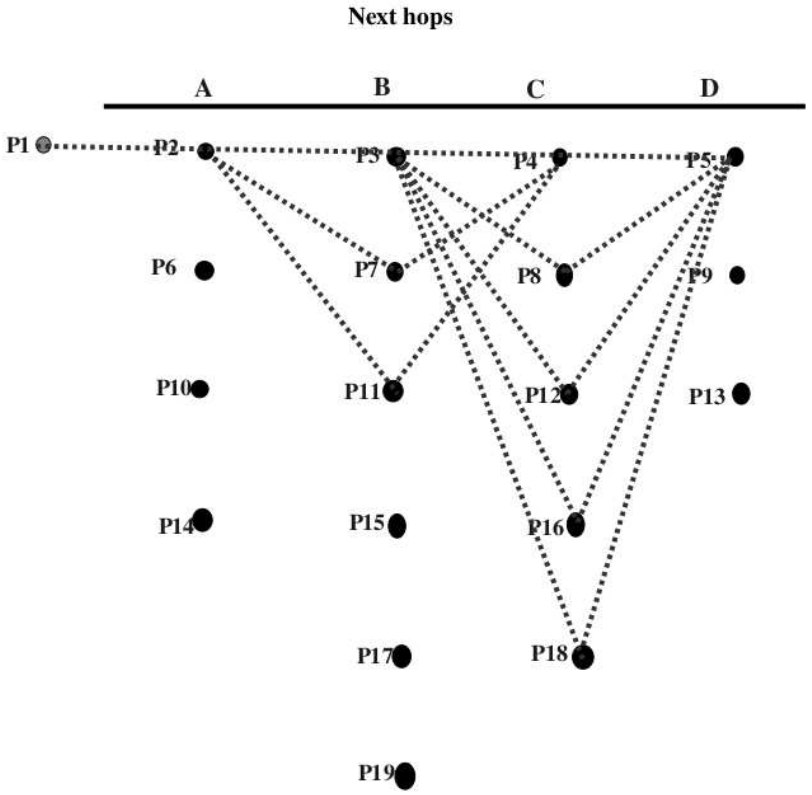

Figure 3.5: Local search algorithm.

of the columns while others are held constant as shown in Fig. 3.5. In each case, neighbors information is employed to check if a set of packets found so far can be coded together. If the end of that column is reached and there is no coding opportunity that maximizes throughput, the search process considers a different column until all columns have been processed.

Assume that according to the maximization strategy employed by local search algorithm, at most five packets can be coded together as shown by unique hops A, B, C, D and that of packet P_1 in Fig. 3.5. Therefore, local search algorithm first searches for a coding opportunity which leads to coding five packets in total. Therefore, for this particular scenario, condition 1 mentioned above will require that

there should be exactly 5 packets in the set of potential packets to be coded together. From Fig. 3.5, suppose that, a set of packets which is currently being processed includes packets P_1, P_2, P_3, P_4 and P_5. If all the four conditions mentioned above for successful encoding are met, then these packets will have passed the first phase of coding. However, if at least one condition is not satisfied, then the entire set of packets is discarded and a new set with packets P_1, P_2, P_3, P_8 and P_5 will be considered as shown in Fig. 3.5.

If all columns have been processed and there is still no set of packets which leads to five packets being coded, the local search algorithm will then look for four packets which can be coded together by repeating the same process. If there is no combination that satisfies four packets, then three packet combination will be considered. This process continues until no combination is found after which the algorithm gives up and just sends the packet at the head of IFQ without coding it with any packet. This approach takes less processing time, less memory and consumes lower power than exhaustive search algorithm.

3.1.3 On Channel and TCP information

Let $\rho \in \psi = \{P_1, P_2, \ldots, P_n\}$ be the potential set of packets to be coded. In other words, set ψ contains packets which have passed the first phase of network coding and now moving on to the next phase where there is a possibility of some packets being eliminated. In the second phase, the scheduler uses the available channel information to determine the quality of wireless links through which the potential coded packet is headed. Packet ρ whose link is poor such that even the lowest modulation scheme

cannot be used, is removed from set ψ. After eliminating packets whose wireless links are poor in phase two, packets still in set ψ move on to the final phase.

In the final phase, the best data rate for transmiting coded packet made of packets in set ψ is chosen. The criteria used is based on each packet's TCP sending rate which is annotated on the XOR header as shown in Fig. 3.8. Furthermore, this criteria takes into account the instantaneous channel conditions for all wireless links involved. Having identified the final set of packets to code, XOR bitwise operation is carried out to code packets. Information of *native* packets contained in the coded packet and that of recipients is annotated in the XOR header of coded packet so that receivers know whether they are the intended receivers or not. Also, it helps receivers in determining which packets should be used for decoding. Reception report is also added before the packet is transmitted.

Note that this network coding implementation does not follow that of [6] which employs pseudo-broadcasting of coded packet and have all intended receivers send acknowledgment packets. Instead, this scheme takes a different approach which uses unicast mode. Suppose packet P was used to carry collective information content of coded packets such that $P = \{P_1, P_2, \ldots, P_n\}$. Packet P will only be addressed to one of the recipients of this coded packet. The advantage of this approach is that it minimizes acknowledgement packets in the network which would be sent if all the recipients had to send back acknowledgements hence facilitates efficient bandwidth utilization. Also, since coded packets only contain native packets whose links are good, success probability of packet delivery and decoding is high.

3.1.4 Decoding at the Receiver

Upon receiving a data packet, a node checks the XOR header of the packet to determine if the packet is coded or not. If it is coded, the node then checks if it is one of the intended receivers by accessing information annotated in the XOR header of the packet. Assuming it is one of intended receivers, the node then checks to see how many and which *native* packets are in this coded packet. If κ is the number of *native* packets in the coded packet, the node needs to have $\kappa - 1$ of those *native* packets in order to have successful decoding. During decoding process, XOR bitwise operation is performed on the coded packet using packets in the local packetPool (see 3.4.4). If some node which is not one of the intended recipients has overheard a coded packet and has $\kappa - 1$ packets required to decode coded packet P, it performs decoding to extract *native* packet it does not have and stores it in its packetPool. Fig. 3.6 illustrates how decoding is performed at the receiver. Note that if the receiver does not have $\kappa - 1$ *native* packets required to perform decoding, the received coded packet is discarded as shown in Fig. 3.6.

3.2 Channel Model

The propagation model used is TwoRayGround model in a 1km X 1km area. Packet error rate is set to 0.01 and the number of MAC retransmissions is set to 6. For simplicity and simulation purposes, TCP packet size is set to 100 bytes. The transmit power is 1 watt. Perfect channel estimation is assumed. It is also assumed that the sender has already received channel information from the receivers. Using equations (3.1), (3.2) and (3.3) from [30], we determine the required SINRs for different

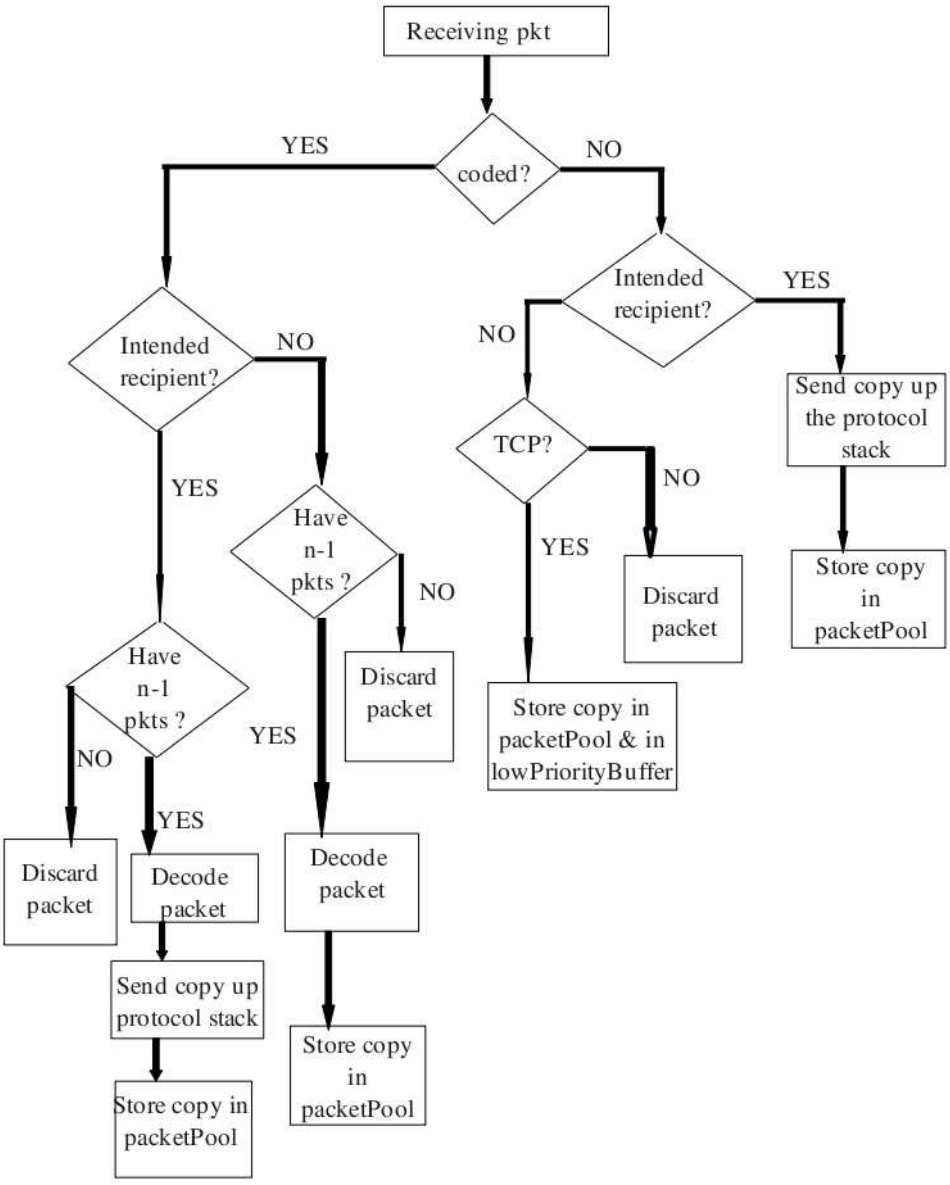

Figure 3.6: Flowchart of decoding process at the receiver.

modulation schemes shown in Table 3.1.

$$P_e = 1 - (1 - F_e^{N_{retrans}+1})^{N_{fr}}, \tag{3.1}$$

$$F_e = 1 - (1 - BER)^{L_{fr}}, \tag{3.2}$$

$$BER = K_1 exp(\frac{-k_2 \gamma P_{tr}}{2^{\rho} - 1}), \tag{3.3}$$

Table 3.1: Modulation schemes and SINR

Modulation scheme	**Required SINR [dB]**	$\mathbf{K_1}$	$\mathbf{K_2}$
BPSK	6.549	0.5	1.5
QPSK	8.471	0.2	2.5
QAM16	10.989	0.2	7
QAM64	16.13	0.2	9

where K_1 and K_2 are constellation and code-specific constants, respectively. $N_{retrans}$ is the number of retransmissions. N_{fr} is the number of frames. L_{fr} is the length of the frame. BER is bit error rate. P_e is packet error rate. F_e is frame error rate. ρ is the number of bits representing each symbol in the constellation space. γ is signal to interference plus noise ratio and P_{tr} is transmit power.

3.3 TCP Model

TCP Reno, which is the most widely deployed version of TCP protocol [31], is used in all simulations. TCP Reno has three phases. These are: slow-start, congestion-avoidance and fast recovery. When a TCP session is established, TCP is said to be in slow-start phase. In this state, congestion window size is increased by doubling its size for every ACK received. When slow start threshold is reached, which indicates transition from slow-start to congestion avoidance state, the window size starts increasing linearly. If three duplicate packets are received by the sender due to corrupted, lost or out of order packet delivery at the receiver, fast retransmission of the lost packet is performed without waiting for retransmission timeout (RTO) [15]. This moves TCP Reno into fast recovery state in which the new RTO equals double the previous RTO and the transmission rate is reduced by half. Reno remains in this state until the entire transmit window has been acknowledged after which it moves

into congestion-avoidance state. In case of timeout, Reno will move into slow-start mode. Even though this strategy is better than the one used in earlier TCP version, it is inefficient in that it only recovers at most 1 lost packet [32]. Therefore, its performance deteriorates a lot in wireless networks where channel errors are frequent [15, 16, 18].

3.4 TCP-Aware Network Coding with Opportunistic Scheduling

TCP-Aware network coding with opportunistic scheduling is implemented at the data link layer of the protocol stack as shown in Fig. 3.7. For full functionality of the system, information about the current TCP window size and channel state has to be known. Therefore, this scheme employ cross-layer design approach in which there is information exchange across the protocol stack.

3.4.1 Accessing TCP window size

At the transport layer, information on the current TCP window size is annotated on the XOR header (see Fig. 3.8 and 3.4.3) of a TCP packet which has just been generated before it is passed down the protocol stack. Upon arrival at the data link layer, the window size information is extracted from the XOR header of the packet and a local variable for keeping current window size is updated before the packet is inserted in the interface queue. Note that some packets coming down the protocol stack from network layer may not be self generated by a node of interest but need to be forwarded. Therefore, to ensure that the correct information is extracted, the

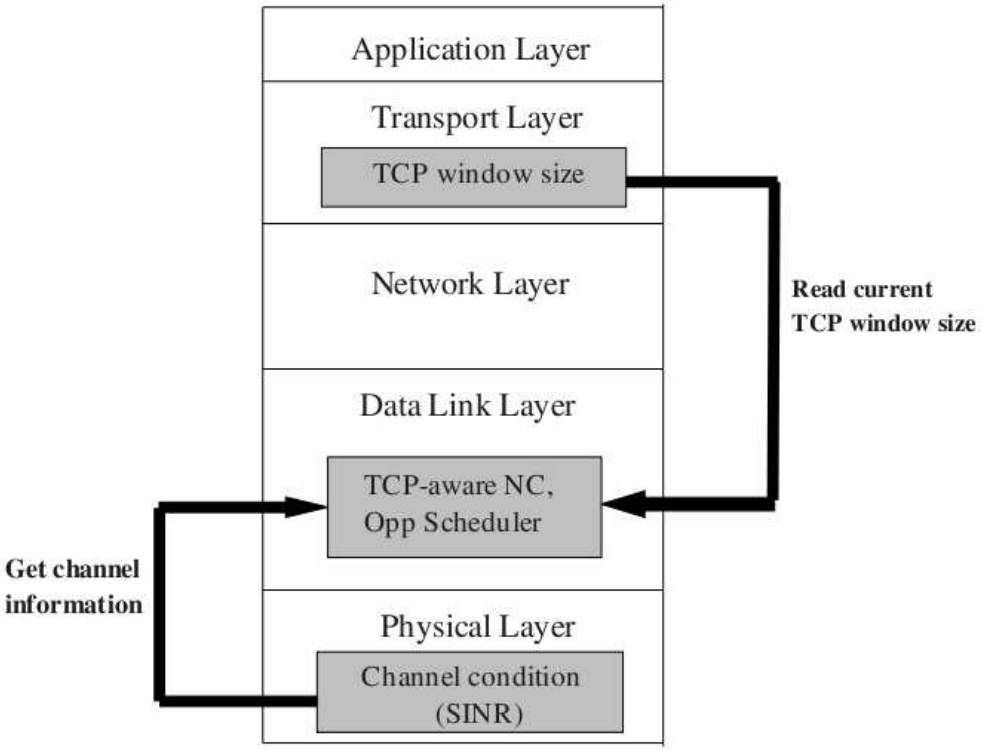

Figure 3.7: Protocol stack.

source IP address of the packet is checked to make sure it is the same as that of the node of interest. If it is different then the local variable is not updated. It has to be noted that this approach assumes that a node can only generate at most one TCP session at a time.

3.4.2 Packet fields for Network Coding

Fig. 3.9 illustrates packet fields for network coding as implemented in this thesis. The number of bytes shown for different fields are for simulation purposes. MAC header is for indicating the MAC source and next hop of the packet where as IP header is for indicating the original source and final destination of the packet. XOR header carries

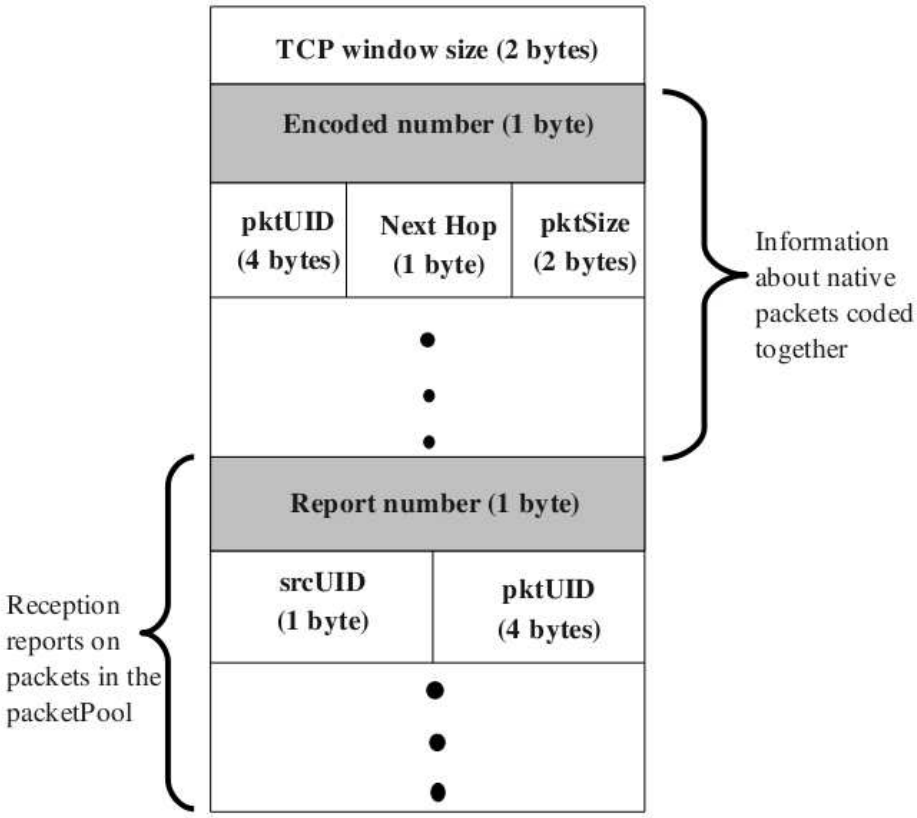

Figure 3.8: XOR header.

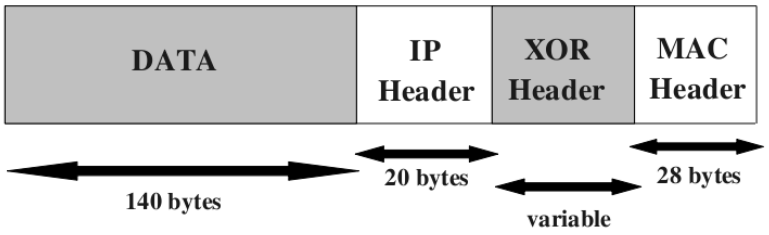

Figure 3.9: Packet fields for network coding.

network coding information only. From Fig. 3.9, it can be seen that XOR header field is variable where as other fields are fixed. This is due to the varying number of reception reports that can be generated at a node. Also, it is due to the changing number of packets that can be coded together. When packets are coded, only their

data portion and IP headers are coded but not XOR and MAC headers.

3.4.3 XOR Header

We adopt the presentation style in [6] used for COPE header. According to our scheme, there are three main blocks in the XOR header. These are: the first block identifies the current TCP window size of the TCP source from which the packet is coming. The second block identifies the number of packets XORed together and specific information about those packets such as packet unique identification number (pktUID), next hop and packet size. The third block identifies the number of reception reports included in the XOR header and specific information of each report. Note that the memory allocated for each field in the XOR header is just for simulation purposes and may not be used in practice. From Fig. 3.8, *Encoded number* field specifies the number of packets coded. In the implementation, it is set to 1 if there is no coding. *PktUID* is a packet unique identification number which in the simulation is generated by ns2 automatically when the packet is formed. For simulation purposes, pktUID is allocated four bytes since many packets can be generated. This pktUID in practice could be an output of hash function of packet IP address and other parameters. Also, memory allocated for pkt UID in practice could be less than four bytes. *Next hop* field identifies the neighboring node which the packet will be forwarded to. For simulation purposes, *next hop* is allocated 1 byte as an indication that a node cannot have more than 256 neighboring nodes due to power, memory and computational contraints. *PktSize* field indicates the size of the non-coded or *native* packet in a coded packet. This assists in padding to facilitate coding. *SrcUID* is the source identification number

which uniquely identifies a node. Again for simulation purpose, we allocated 1 byte to uniquely identify each node in the network because no more than 256 nodes where used in any simulation scenario defined. However, in practice this is not the case as there can be hundreds or even thousands of nodes in the network.

Note that in Fig. 3.8, the *Report number* and *Encoded number* fields are allocated only 1 byte each. The assumption here is that, 1) a node cannot keep (or process) more than 256 overheard packets in its packetPool due to memory, power and computational constraints; 2) a node cannot code more than 256 *native* packets because of constraints mentioned. Also, if nodes had that capability, this would mean that long XOR headers are generated. Therefore, packets would be too long which could degrade the network performance.

3.4.4 Components of TCP-Aware Network Coding

At the data link layer of the protocol stack, there are various components that make up TCP-Aware NC with OS scheme as shown in Fig. 3.10. These are: the encoder and decoder for network coding, opportunistic scheduler, packetPool, lowPriorityBuffer, storage for neighboring nodes information and that of the current TCP window size.

Information of Neighboring Nodes

A node has to keep track of its neighboring nodes. If a node moves away or is inactive for more than 3 sec, it is not considered as one of the neighboring nodes. A node is considered having moved away or inactive if there are no messages received from it. Information of the most recent reception reports received from neighboring nodes

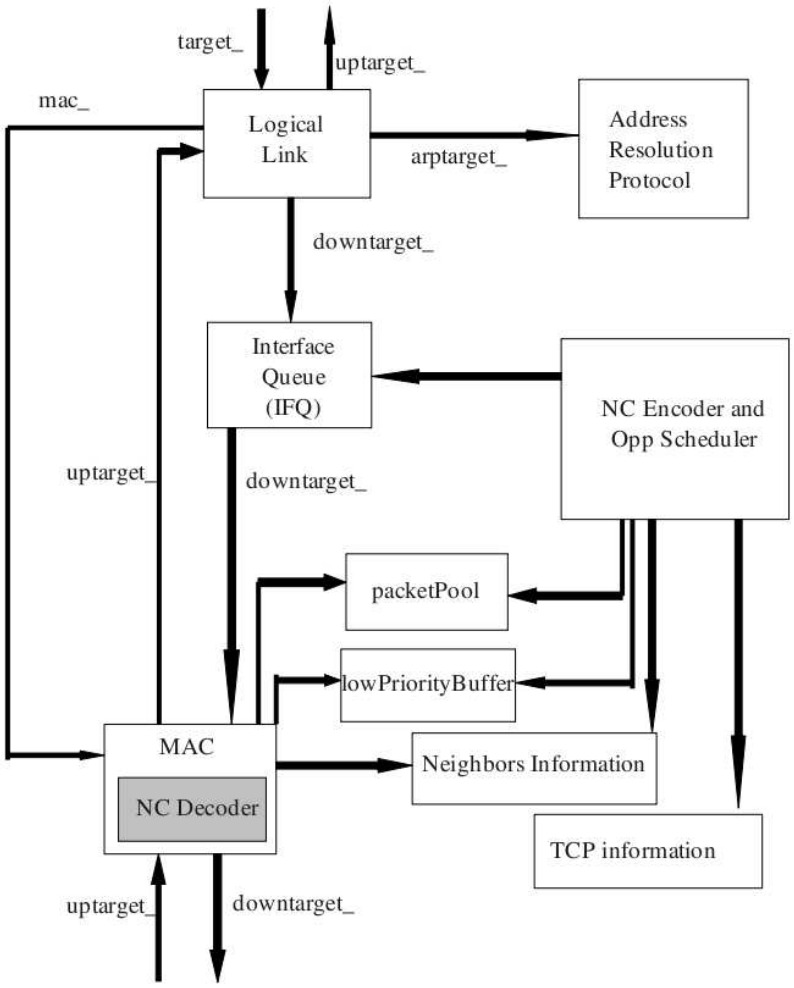

Figure 3.10: Data link layer implementation on ns2.

is kept. Also, channel information of outgoing links to those nodes is also stored. Reception report is a piece of information which a node shares with its neighbors on packets it has overheard from the network. This information is annotated on XOR headers (see Fig. 3.8) of all outgoing data packets.

PacketPool

This a buffer which keeps overheard packets from the network and those which have been sent by a node. The purpose of packets in this buffer is that they are used for decoding incoming coded packets. Also, information of these packets is used to generate reception reports to be included in outgoing data packets. Packets in this buffer are kept for a particular time interval after which they are deleted. Note that packets stored in this buffer may or may not be intended to a node.

LowPriorityBuffer

LowPriorityBuffer stores overheard packets which are *not* intended to a node. This ensures that the IFQ is not overwhelmed by packets from the network. Also, this is carried out to avoid additional processing load which could potentially reduce the node's lifetime. Moreover, this avoids flooding the network by forwarding packets unnecessarily. Only packets generated by the node and those to be forwarded by the node go through the IFQ. Packets in lowPriorityBuffer can *only* be transmitted by being coded with those from IFQ if there is coding opportunity. If there are no packets to be sent from IFQ, those in lowPriorityBuffer are not sent and will only be discarded after a certain time interval.

Network Coding Encoder and Opportunistic Scheduler

The NC encoder processes packets from the IFQ and those from lowPriorityBuffer. The opportunistic scheduler in collaboration with the NC encoder chooses suitable transmisssion data rates on the bases of the current TCP congestion window size,

channel state and reception reports received from neighboring nodes.

Network Coding Decoder

The NC decoder processes received data packets. These packets may or may not be coded packets. The decoder makes decisions on whether a packet is coded or not and acts accordingly. It determines where the packet has to be sent after processing it. Therefore, it has access to packetPool and lowPriorityBuffer. For successful packet decoding, NC decoder need to have $n-1$ packets of the native packets in the coded packet. For example, in Fig. 3.1 taken from [1], the NC decoders for nodes A, C and D can successfully decode the received coded packet made of packets P_1, P_2 and P_3 because they each have $n-1$ packets of n native packets in the coded packet as required. Upon receiving such coded packet, node E will discard such packet since it already has all native packets contained in the coded packet.

3.5 Summary

This chapter described how packet encoding and decoding is carried out. It discussed why and how local search algorithm was chosen in order to code packets. A brief discussion on how network coding incorporates TCP and channel information was presented. Furthermore, packet fields, XOR header, components of TCP-Aware network coding were presented. Also, channel and TCP models were discussed.

Chapter 4

TCP-Aware Network Coding with Opportunistic Scheduling

This chapter presents three schemes which are traditional network coding, traditional network coding with opportunistic scheduling and TCP-Aware network coding with opportunistic scheduling. Simulation results and discussion are also presented. The purpose of this analysis is to determine which scheme provides the best TCP performance in wireless mobile ad hoc networks.

4.1 Traditional Network Coding

In traditional network coding, when a node gets an opportunity to transmit, it first searches for any available coding opportunity. This is performed by *only* using reception reports received from neighboring nodes. In other words, such a node does not have information on conditions of outgoing channels. Moreover, it does not have information on how TCP parameters are changing in response to channel conditions and the capability to react to those changes. Therefore, there is high probability of spurious coding opportunities being found which means some intended recipients may not be able to receive or decode coded packets due to channel errors. As a result, TCP retransmissions and timeout events will be frequent hence decreasing the

performance.

4.2 Traditional NC with Opportunistic Scheduling

A better approach is traditional network coding with opportunistic scheduling. In this scheme, when a node gets an opportunity to transmit, it searches for coding opportunities by employing a much advanced approach. In this approach, reception reports received from neighbors and channel conditions are taken into consideration in order to determine which set of packets to code together. This scheme ensures that native packets contained in the coded packet have good quality links which means high probability of successful packet reception. Also, it takes advantage of instantaneous link conditions by employing rate adaptation. However, it introduces stochastic halt of data transfer controlled by the scheduler when links are poor by waiting until link conditions improve. This leads to frequent TCP timeout event which degrade TCP performance [32]. Furthermore, if link quality is low and low channel rate is allocated when TCP sending rate is high, more packets will accumulate in the node's IFQ. As a result, some packets will have to be discarded due to full interface queue there by reducing performance.

4.3 TCP-Aware NC with Opportunistic Scheduling

A more robust and resilient scheme is TCP aware network coding with opportunistic scheduling. In this scheme, TCP dynamics and link conditions are both considered to determine which packets to code and which data rate to use. This scheme facilitates quick and efficient response of TCP sender upon realizing any changes in

unpredictable link conditions. Specifically, at the sender, let N be the largest TCP window size such that $1 \leqslant \eta < \alpha < \beta \leqslant N$ where η, α and β are some parameters indicating the state of TCP congestion window as shown in Fig. 4.1.

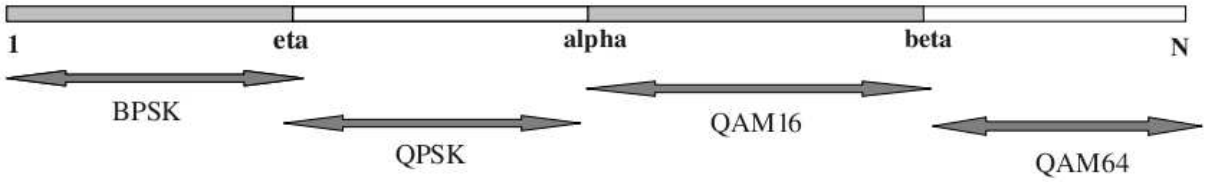

Figure 4.1: Various states of TCP congestion window size.

If $n \in N$ is the current state of the congestion window such that $\beta < n \leqslant N$, it means that TCP sending rate is very high. Therefore the scheduler will allocate the highest channel rate (QAM64) irrespective of the link quality [18]. This is done because if the channel quality is low and low channel rate is allocated when TCP sending rate is high, this will inevitably results in packet loss due to congestion. The advantage of transmitting at high data rate in this situation is that the medium can become available quickly to other nodes with good links. This facilitates efficient resource sharing. However, transmitting at high modulation scheme in a channel of low quality can lead to high bit error rate. This is accounted for by employing adaptive coding to ensure that a particular bit error rate is obtained. Furthermore, in case packets get lost due to low quality channel to the intended receiver, other nodes with good quality links can overhear such packets and could forward them by network coding to the intended receiver. When $\alpha < n \leqslant \beta$, the least channel rate that can be allocated is QAM16. In case the link quality is good such that it allows QAM64

channel rate, the scheduler allocates QAM64 instead of QAM16. This is done to ensure that TCP reacts quickly enough to take advantage of the available bandwidth [33]. For $\eta < n \leqslant \alpha$, QPSK is the least channel rate that can be allocated. Again, if the link quality allows transmission at a higher modulation scheme, the scheduler will allocate the corresponding modulation scheme. The same is also the case for $1 \leqslant n \leqslant \eta$. Fig. 4.2 gives a summary of how a self-generated packet is processed in TCP-Aware network coding with opportunistic scheduling. Note that a delay of $1ms$ is induced in case the wireless link is poor to allow channel conditions to improve after which transmission resumes.

At the intermediate nodes, information on TCP sending rate annotated on XOR header of a packet is used to determine data rate to be used for either coded packet or non-coded packet. In case there is coding opportunity, the intermediate node uses the strategy discussed in chapter 3 to determine a set of packets to code. Note that the final set of packets to be coded may have packets which require different data rates for transmission. This scheme chooses the lowest data rate required by one of the native packets in the set. Even though this decision does not favor packets which require high data rates, it ensures that both TCP dynamics and link conditions requirements are met to enhance performance.

4.4 Example of how decision is made by NC schemes

Consider Fig. 4.3 and Table 4.1. Node B intends to combine packets P_1, P_2 and P_3. Table 4.1 shows the quality of each wireless link involved by indicating the modulation scheme suitable for each link. Also, it shows the state of the current TCP window

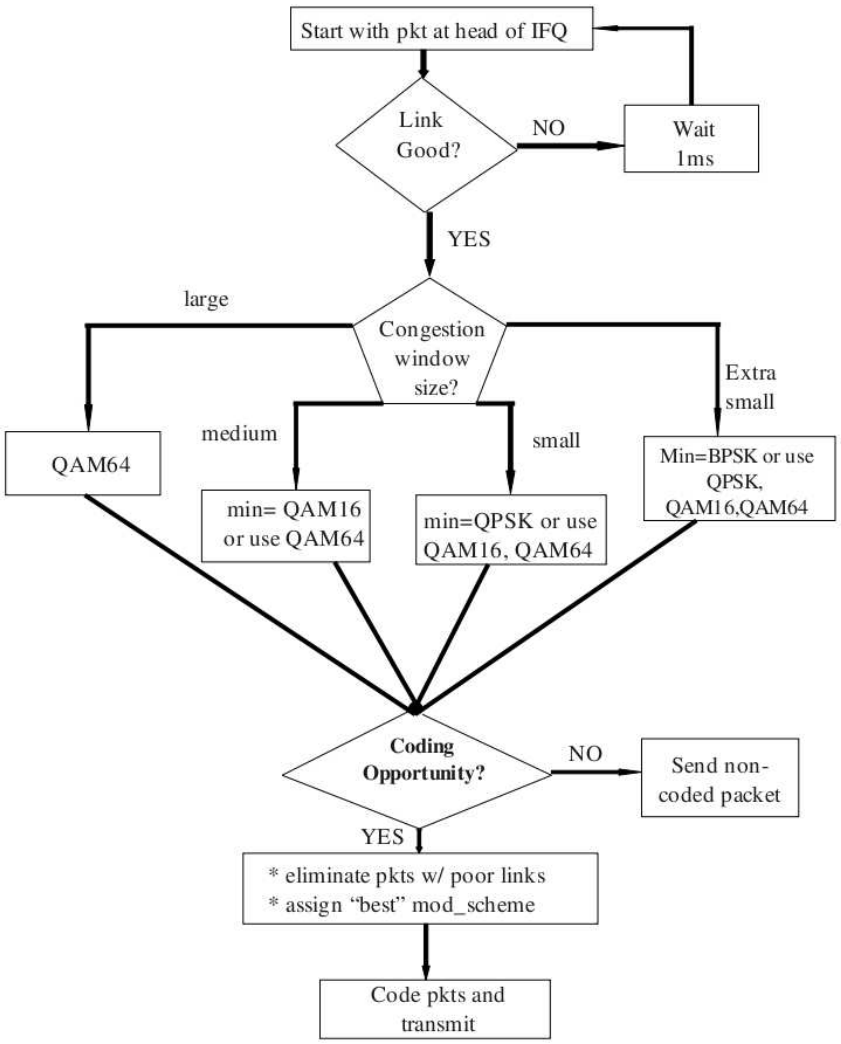

Figure 4.2: Flowchart for tcp-aware nc with opp scheduling.

size from which each packet is coming by indicating the modulation scheme suitable for each state of TCP window size as shown in Fig. 4.1.

Table 4.1: Data rates required by TCP and wireless channels

wireless links	ba	bc	bd
link quality (data rate)	BPSK	QPSK	QAM16
TCP window size (state)	QAM64	BPSK	QPSK

If node B is using traditional network coding only, then it will allocate BPSK channel rate for the coded packet since it does not have TCP and channel information.

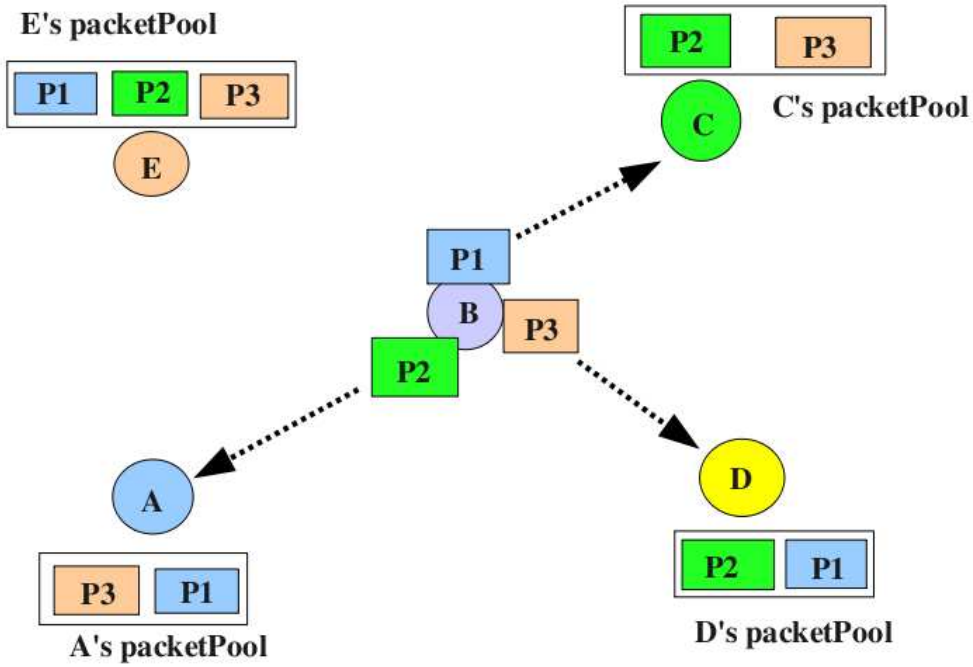

Figure 4.3: Network coding at node b.

On the other hand, node B will choose BPSK channel rate if it uses traditional network coding with opportunistic scheduling. The is due to the weakest link which limits the transmission data rate. In this case, it is link BA. If node B is using TCP-Aware network coding with opportunistic scheduling, it will choose QPSK channel rate. The criteria used specifies that for each packet, the data rate is determined by either TCP information or channel information which maximizes data rates. Having determined data rates for all packets, the least data rate is chosen to satisfy both TCP and channel requirements for coded packet. For example, according to Table 4.1, node B makes decision for each link as shown in Table 4.2 from which it chooses QPSK channel rate which is the least.

Now suppose that link BA is broken or poor as shown in Table 4.3 such that

Table 4.2: Required data rates for TCP-Aware NC

wireless links	ba	bc	bd
decision (on data rate)	QAM64	QPSK	QAM16

even the least modulation scheme cannot be used for transmission. If node B is using traditional network coding alone, it will still code all the three packets and allocate BPSK channel rate for the coded packet. However, node A will not receive the coded packet at all. On the other hand, if node B is using traditional network coding with opportunistic scheduling, it will not code all the three packets. Instead, it will code packets P_1 and P_3 since their links are good. Node B will then choose QPSK channel rate for the coded packet. If node B is using TCP-Aware network coding with opportunistic scheduling, it will code packets P_1 and P_3 and allocate QPSK channel rate for the coded packet. Note that in case there is no coding opportunity and packet P_2 going through the poor link is at the head of the interface queue, then node B will wait for $1ms$ to allow channel conditions to improve after which it resumes transmission.

Table 4.3: Different channel conditions and various TCP states

wireless links	ba	bc	bd
link quality (data rate)	broken	QPSK	QAM16
TCP window size (state)	QAM64	BPSK	QPSK

4.5 Simulation Results and Discussion

4.5.1 Simulation Models

Network Simulator-2.33 version was used to perform all simulations. Each simulation took 30 seconds. The reason being that there was no significant change in the simu-

lation results after 30 seconds. Note that the transmission range was determined by ns-2.33 and that the network was stable. We implemented our scheme in IEEE 802.11 and WirelessPhy extended versions developed by Mercedes-Benz Research and Development North America and University of Karlsruhe for ns-2.33. See the ns-manual at http://www.isi.edu/nsnam/ns/ns-documentation.html for more details. η, α and β were set at $20, 80$ and 140 for simulation purposes. A total of 300 simulations were carried out for each case and an average found. Note that more samples (300 samples) were taken to be sure that the results found truely reflects the population. Three simulation scenarios were defined. These are: mobility, network traffic increase and increasing network nodes. In mobility scenario, there were 9 nodes in the network. Initial locations, routes, speed and final locations for all nodes were fixed. 5 TCP sessions were generated in the network. For network traffic increase scenario, there were 19 nodes in the network which were placed at fixed predefined locations. In the last simulation scenario, nodes were added in the network by being placed at fixed predefined locations. 5 TCP sessions were generated.

4.5.2 Results and Discussions

Confidence Interval

In order to indicate the reliability of sample estimates found in these simulation results, we employed confidence interval. In statistics, confidence interval is an interval estimate of a population parameter. Therefore, instead of estimating the population parameter with a single value, an interval within which the population parameter is

likely to be found is used. The likelihood of finding the population parameter within the confidence interval is determined by confidence level. For these results, we used 95% confidence level represented by vertical lines on the graphs. These vertical lines indicate the confidence interval within which the true population parameter can be found.

Analysis

From Fig. 4.4, we can see that no or low mobility environment exhibits approximately 48% of TCP performance improvement as compared to high mobility environment for TCP-Aware network coding with opportunistic scheduling. This is attributed to relatively constant topology which allows the routing layer to forward packets to best links. However, in a high mobility environment, there is a frequent change in topology which increases the rate at which wireless channel conditions are fluctuating. Therefore some nodes end up forwarding packets to others whose links are poor hence low throughput. Despite this, TCP aware network coding with opportunistic scheduling performs far much better than the traditional network coding with opportunistic scheduling. The reason is that the former uses induced short delay ($1ms$) that facilitates coding opportunities and link quality improvement. Even if the topology is frequently changing and the link is bad, TCP does not suffer that much because other nodes with good links can pick up overheard packets and forward them via network coding. On the other hand, the latter waits until the link is good before it can transmit. Even though this saves power and avoids interference, it does not only reduce chances of coding but also degrade TCP performance be-

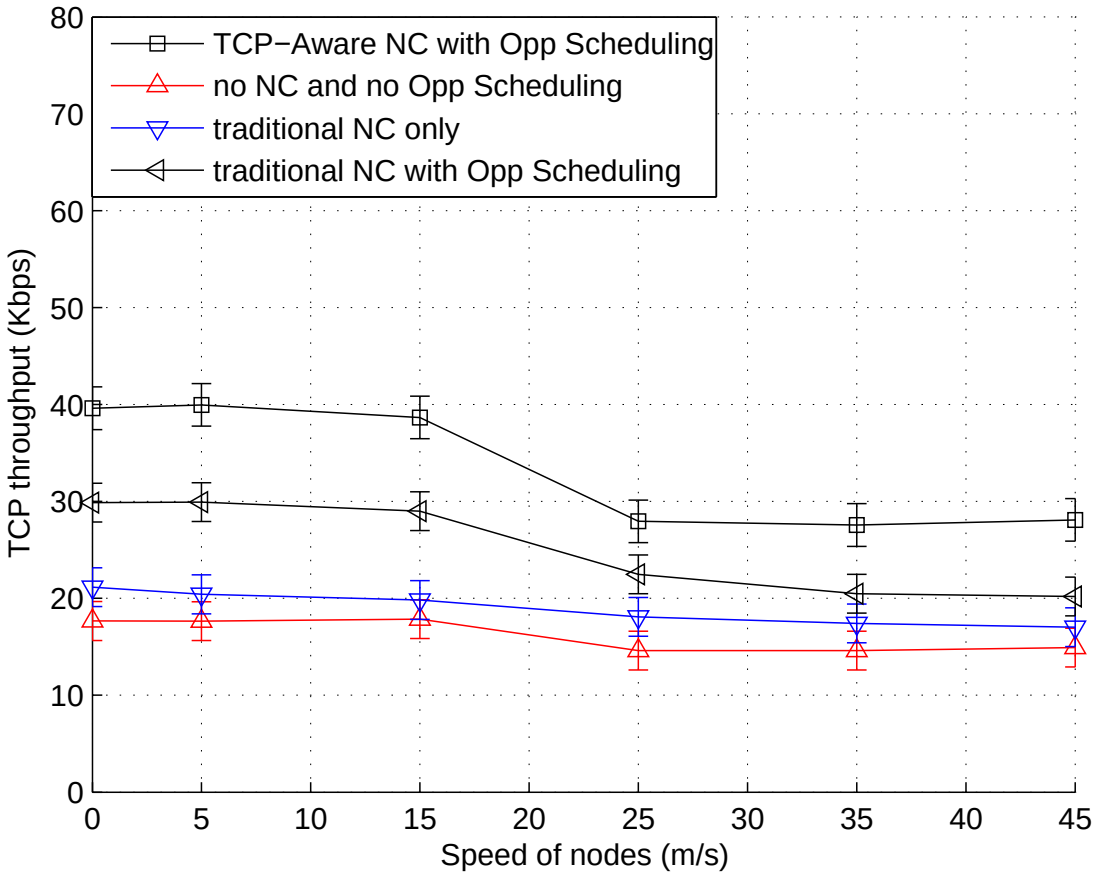

Figure 4.4: Impact of node mobility.

cause of timeouts. The TCP performance of traditional network coding scheme falls much slowly as compared to other schemes. This is explained by coding opportunties which are found in abundance despite transition from low mobility environment to high mobility environment. Even though link conditions change often, more packets are routed via network coding through the network and as a result mobility does not have significant impact provided more nodes are closer together.

Fig. 4.5 illustrates that network traffic increase is directly proportional to network throughput. However, it is expected that when the network traffic reaches saturation point, TCP throughput will stabilize or may even diminish as intermediate nodes be-

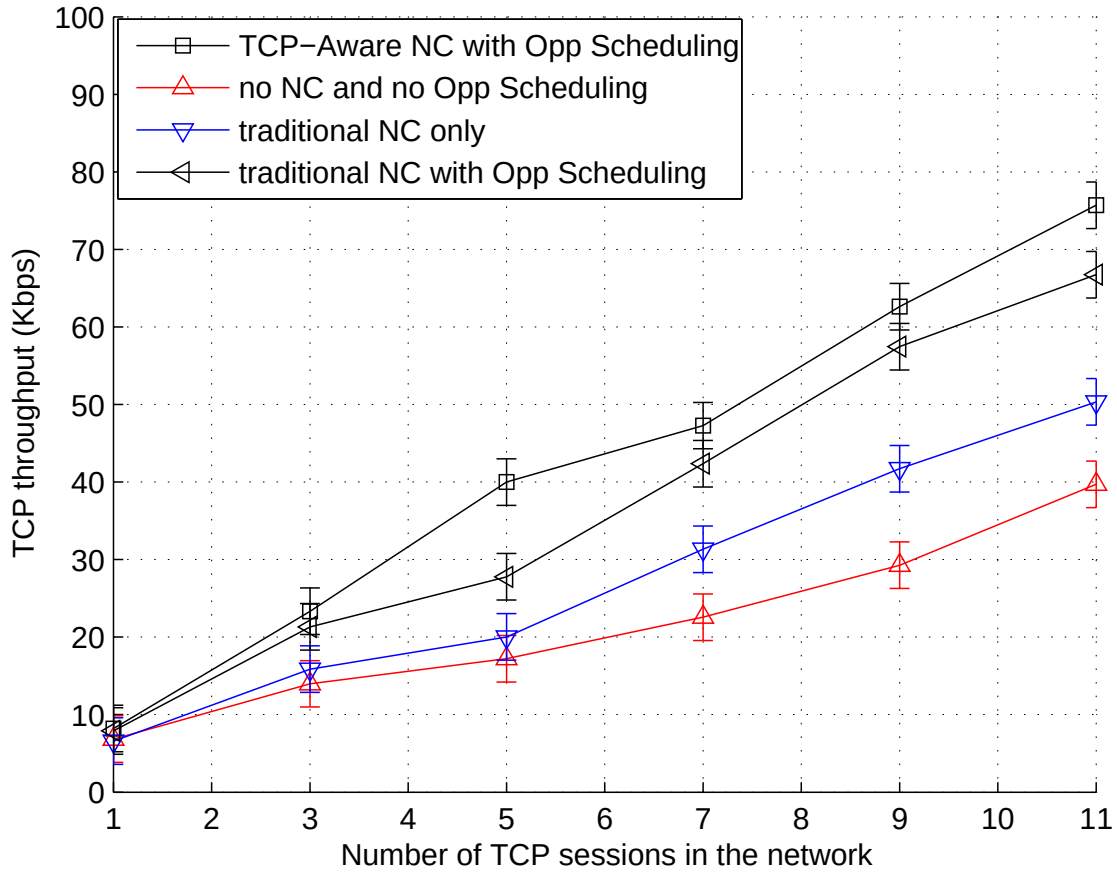

Figure 4.5: Impact of traffic increase.

come overwhelmed by high network traffic to be routed through the network. It can be seen from Fig. 4.5, that when there is low network traffic, there is insignificant performance improvement by any scheme. This is attributed to unavailability of coding opportunities. Also, with low network traffic, the wireless medium is readily available thereby presenting opportunities to all schemes to perform MAC retransmissions without affecting TCP performance. However, if network traffic is high, performing retransmission introduces delay because of time spent waiting to access the medium. Despite this, TCP-Aware network coding with opportunistic scheduling achieves the highest performance as compared to other schemes. This indicates that when the

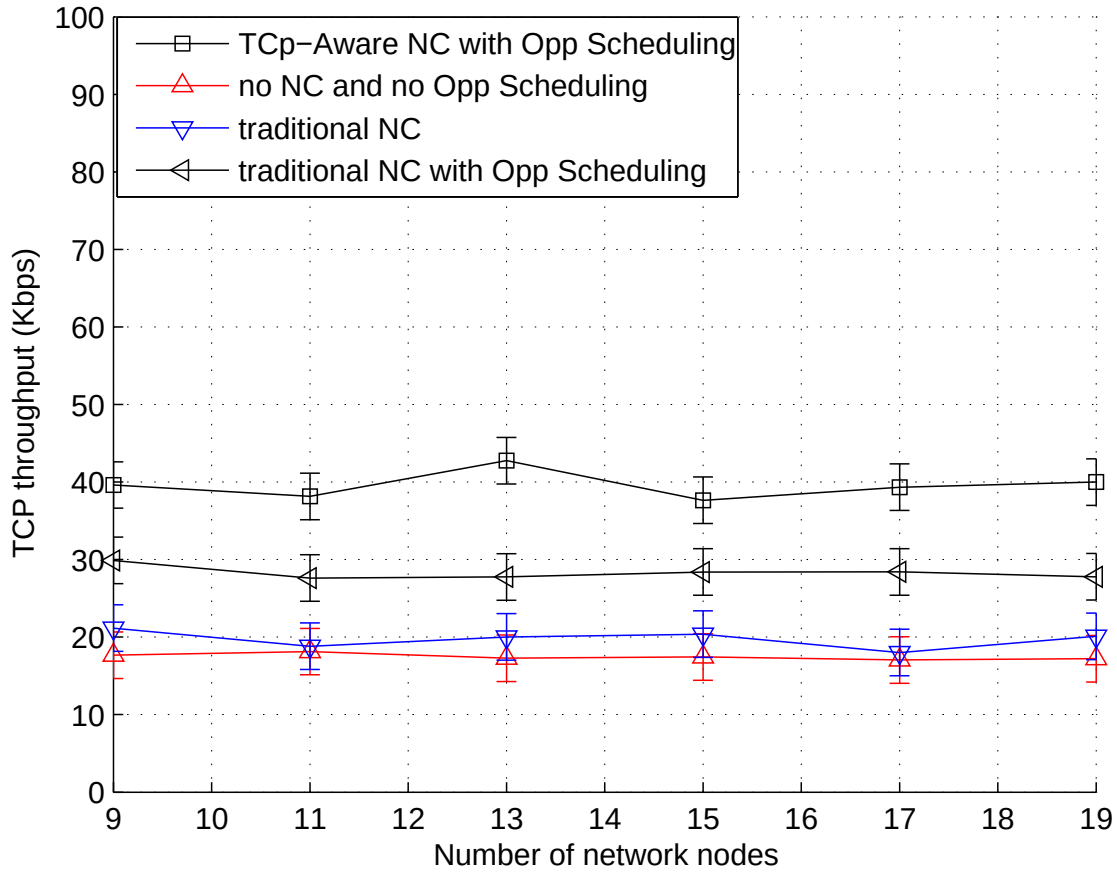

Figure 4.6: Effects of node increase.

traditional network coding is upgraded to TCP aware network coding and combined with opportunistic scheduling in wireless mobile ad hoc networks, performance can significantly improve. Fig. 4.6 exhibits no performance improvement when nodes are added in the network. Although additional nodes in the network create options for routing packets, there are more routing messages that are being generated resulting in few chances of a node being granted channel access per unit time.

The remaining simulation results in this chapter indicates the impact of different node/network parameters on coding opportunities for traditional network coding alone. In this analysis, 25 fixed nodes were placed in crystal lattis shape $100m$ apart. Shadowing model with pathloss exponent = 2.8, standard deviation = $6.0dB$, reference distance = $1m$, Pt = $1watt$ was used. A total of 50 simulations which took 10 seconds was carried out. 16 TCP sessions were generated.

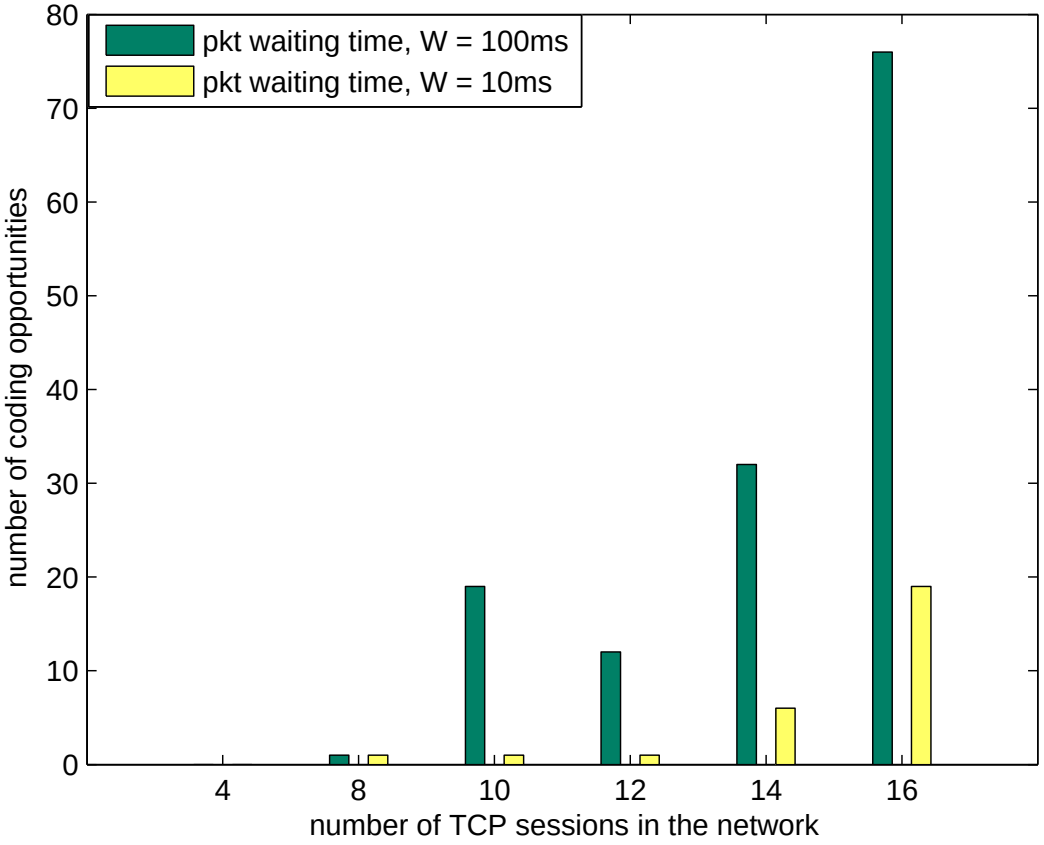

Figure 4.7: Impact of network traffic on coding opportunities.

Fig. 4.7 shows that when network traffic increases, more coding opportunities are generated. The figure also illustrates that if overheard packets are stored in a packetPool longer, more coding opportunities are found.

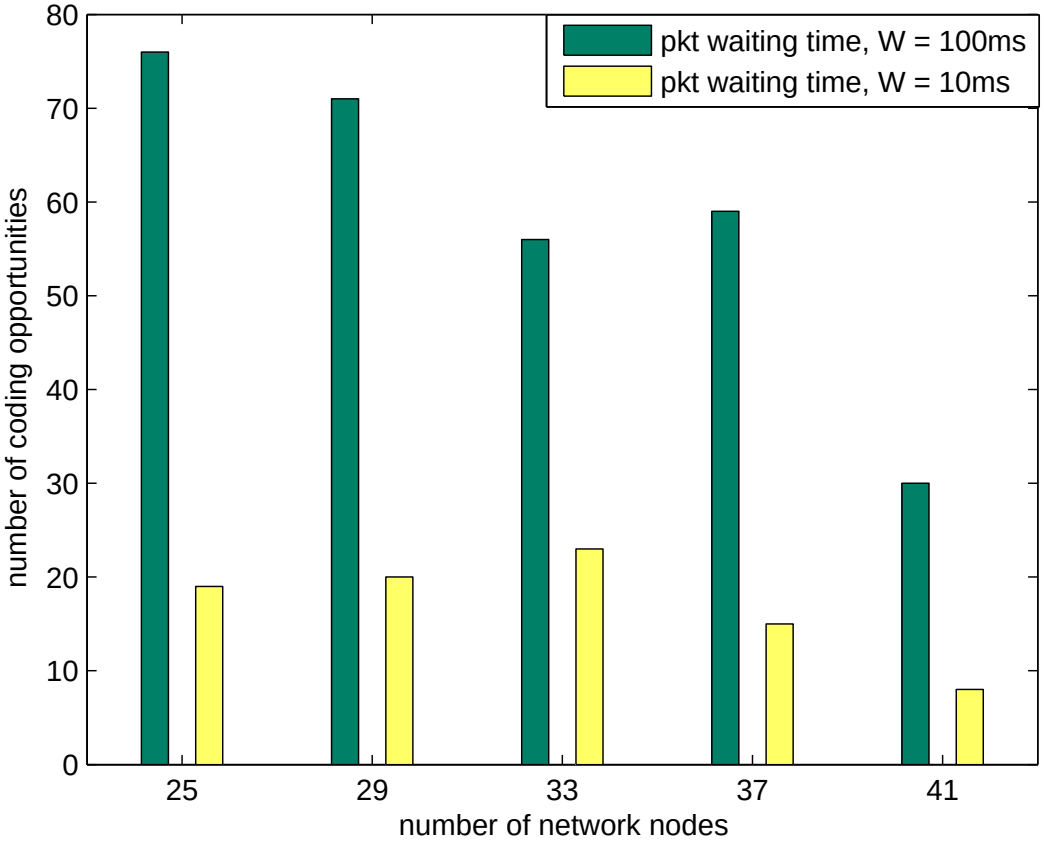

Figure 4.8: Impact of network nodes on coding opportunities.

Fig. 4.8 depicts a reduction in coding opportunities as the number of nodes in the network increases. This is due to availability of more alternative routes for forwarding packets resulting in evenly distributed network traffic. Therefore, intermediate nodes forward fewer packets which leads to fewer coding opportunities. Despite this, more coding opportunities are generated when packets are kept in a packetPool longer. However, as discussed in chapter 5, this increases packet overhead hence low network throughput.

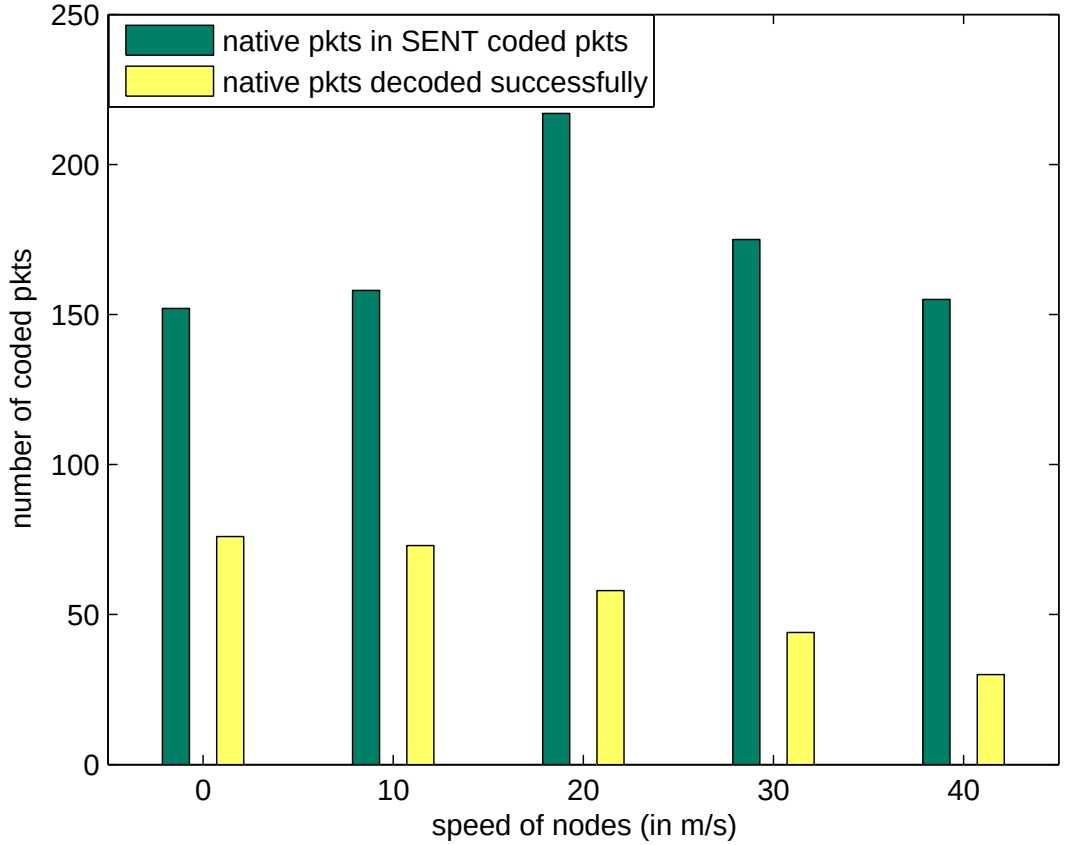

Figure 4.9: Impact of mobility on coding opportunities.

Fig. 4.9 shows that slow increase in mobility can actually lead to an increase in chances of coding. However, as mobility increases further, chances of coding decreases. This is due to frequent changes in network topology which makes it difficult for neighboring nodes to effectively exchange reception reports hence few chances of coding.

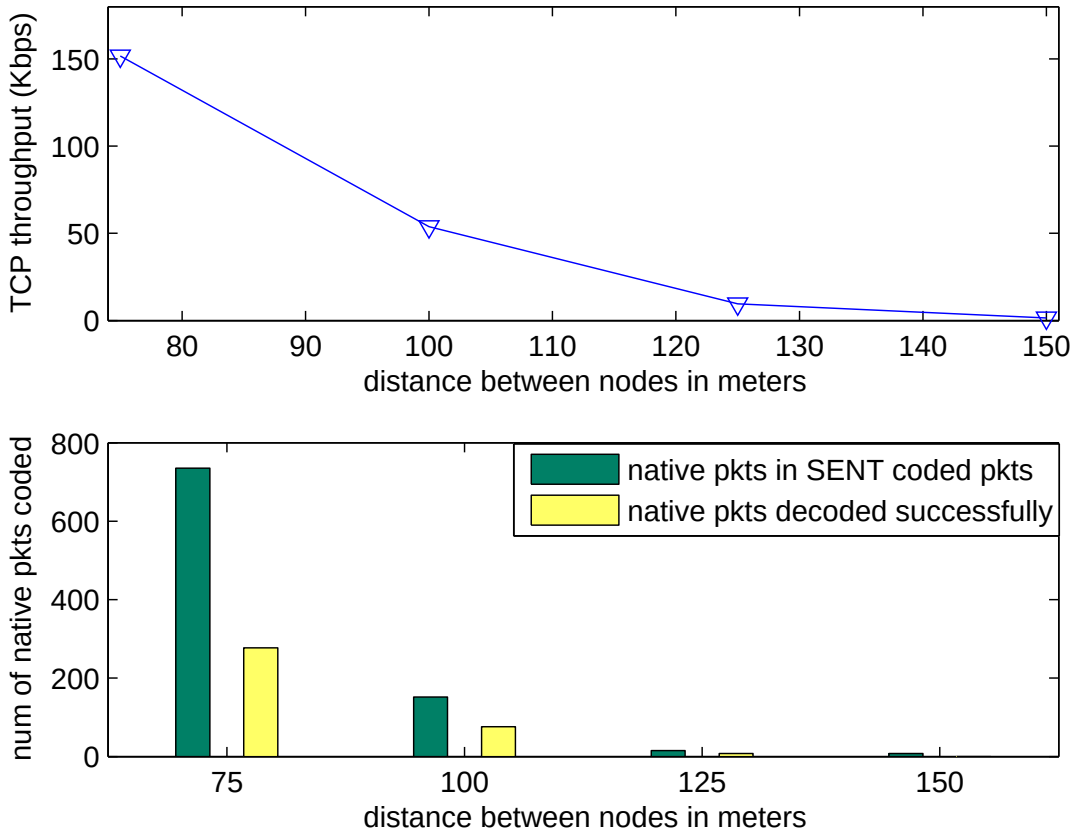

Figure 4.10: Distance versus throughput and coding opportunties.

When nodes are close together, channels are mostly in good quality. This contributes to network throughput improvement as shown in Fig. 4.10. Also it facilitates efficient exchange of reception reports thereby increasing chances of coding.

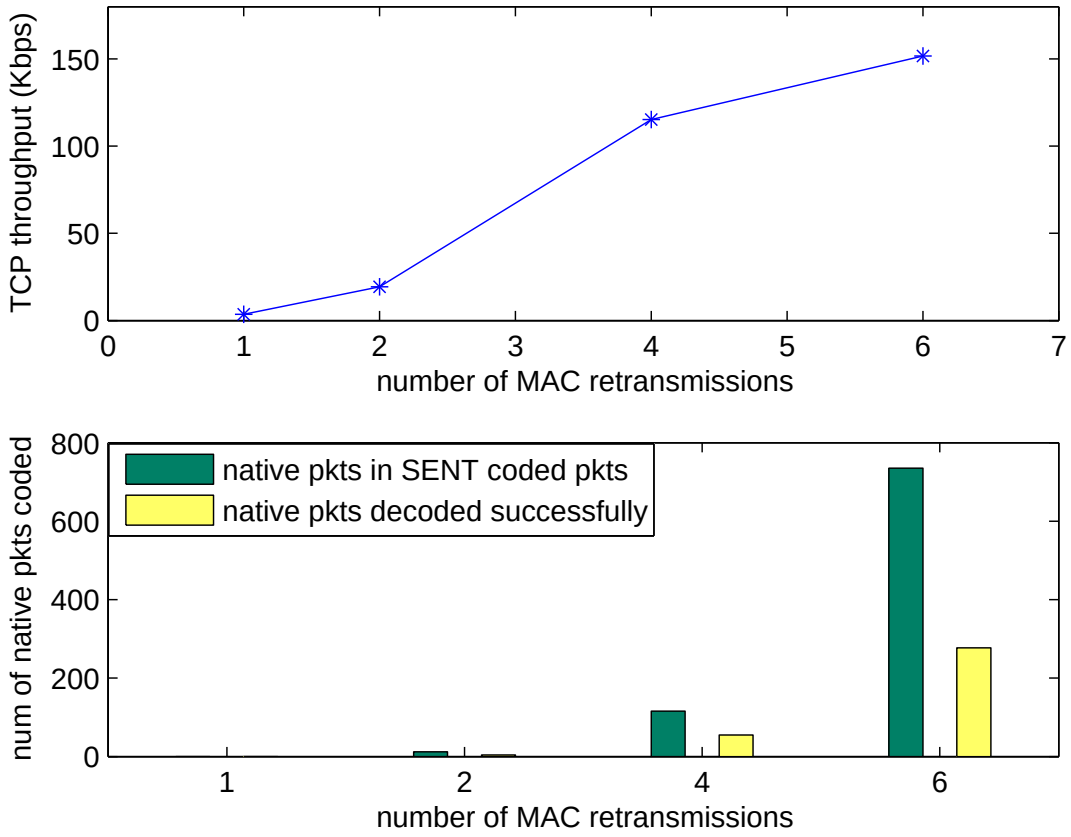

Figure 4.11: Impact of retransmission on throughput and coding.

As MAC retransmission increases, more coding opportunities are found as shown by Fig 4.11. However, there are fewer coded packets which are successfully decoded. This is due to the fact that in our scheme, when a retransmission is performed, the same coded packet is sent again without considering any possibility of coding new packets. It must be noted however that when opportunistic scheduling and TCP awareness is taken into consideration, only packets with good links are coded. Therefore, chances of successful decoding at the receiver are very high.

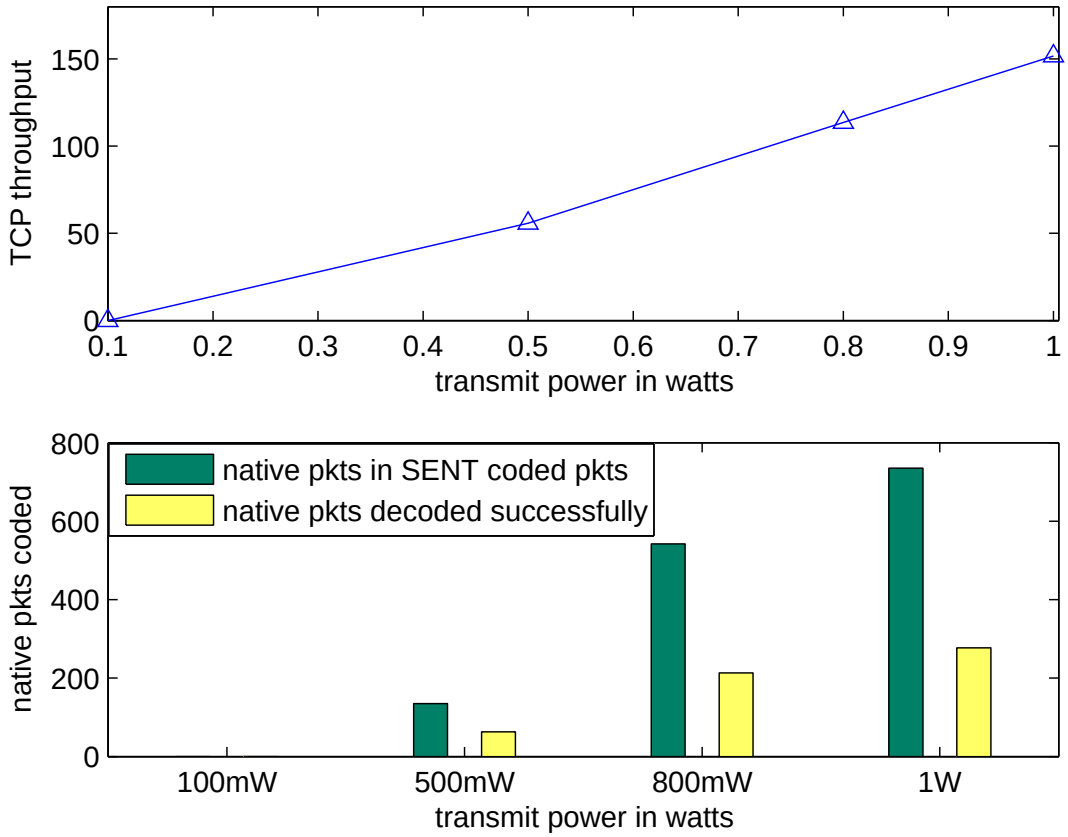

Figure 4.12: Transmit power on coding opportunities.

As transmit power increases, more coding opportunities are found because of enlarged communication range for each node.

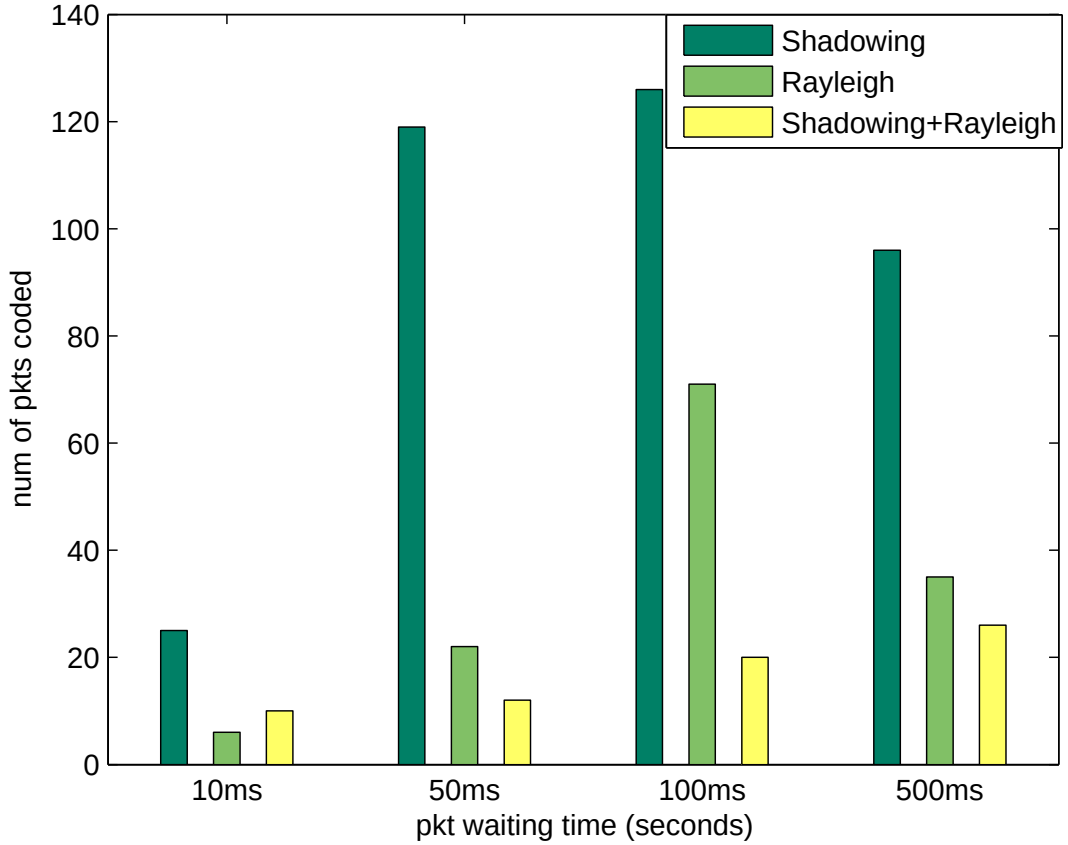

Figure 4.13: Channel models on coding opportunities.

Generation of coding opportunities in wireless mobile ad hoc network depends on several factors and one of them is the environment itself. If the environment is favourable, coding opportunties are generated in abundance. However, when environment is less favourable, few coding opportunities are available. Fig. 4.13 demonstrates that an environment in which fading occurs slowly (that is Shadowing model), coding opportunities are in abundance. This is because, in this type of environment, channel behavior is mostly stationary over the entire frame duration. Therefore, more packets are successfully received hence high chance of coding. However, in a fast fading environment such as high mobility environment, channel behavior is not

stationary over the entire frame duration. It changes frequently and if channel estimation techniques are not employed at the receiver for successful packet detection, network performance deteriorates and inevitably, coding opportunities decreases as shown in Fig. 4.13. An environment which combines shadowing and rayleigh fading exhibits the worst performance.

4.6 Summary

This chapter presented a detailed description of the main scheme proposed by this thesis which is TCP-Aware network coding with opportunistic scheduling in wireless mobile ad hoc networks. Other schemes which were previously proposed in the literature were also discussed and used for performance comparison. Simulation results showed that when the traditional network coding is upgraded to TCP-Aware network coding and combined with opportunistic scheduling in wireless mobile ad hoc network, high performance benefits are achieved.

Chapter 5

Adaptive Control of Packet Overhead in Network Coding

Existing network coding schemes such as COPE [6] require exchange of information among neighboring nodes in order to correctly encode and decode data packets [29]. If more overheard packets from the network are captured and kept in a packetPool resulting in more information exchange among nodes, then more coding opportunities are more likely to be found. However the big question is, would this approach necessarily lead to high network throughput?

Based on this question, this chapter presents the impact of this approach on throughput. From the findings, it then proposes a better scheme called adaptive-W scheme whose objective is to adaptively control the waiting time of overheard packets that are stored in the packetPool to achieve tradeoff of throughput and overhead. For comparison purposes, fixed-W scheme previously proposed in the literature by Katti *et al.* [6] is also presented.

To investigate the impact of overheard packets in throughput, a simulation was carried out in network simulator-2.33. Note that in this simulation, traditional network coding which is the foundation upon which our scheme is implemented, was

considered. A shadowing channel model with the following parameters was used: path loss exponent of 2.8, shadowing standard deviation of $6dB$, reference distance of $1m$. Packet error rate (PER) was set at 0.01. Transmit power was $1watt$ and MAC retransmissions was 6. Each simulation took 10 seconds and a total of 50 simulations for each case were carried out. 16 TCP sessions were generated in the network with 25 nodes located in fixed locations representing a square Bravais lattice in a two dimensional plane. This locations were not randomly generated.

From Fig. 5.1, it can be seen that when overheard packets from the network spend longer time in packetPool, the overall throughput of the network decreases. This is due to the fact that when packets spend more time in the packetPool, the number of packets in the packetPool increases. As a result, long reception reports are generated which lead to high packet overhead hence low network throughput. This is also supported by Fig. 5.2 which shows that the probability that TCP throughput will be more than 80Kbps for the scheme with packet waiting time of $10ms$ is 0.32 where as the probability is 0.2 for the scheme with packet waiting time of $100ms$. This shows a significant difference of 0.12 in probability in favour of the former.

Fig. 5.3 illustrates that when packet waiting time increases, more coding opportunities are generated. As a result of this, one would expect network throughput to increase in Fig. 5.1 and Fig. 5.2. However, it turns out that in this scenario, packet overhead generated is so high that it dorminates the outcome of throughput as compared to coding opportunities.

Note that coding opportunities found in Fig. 5.3 do not increase forever. They start decreasing in number at some point. This is due to high packet overhead

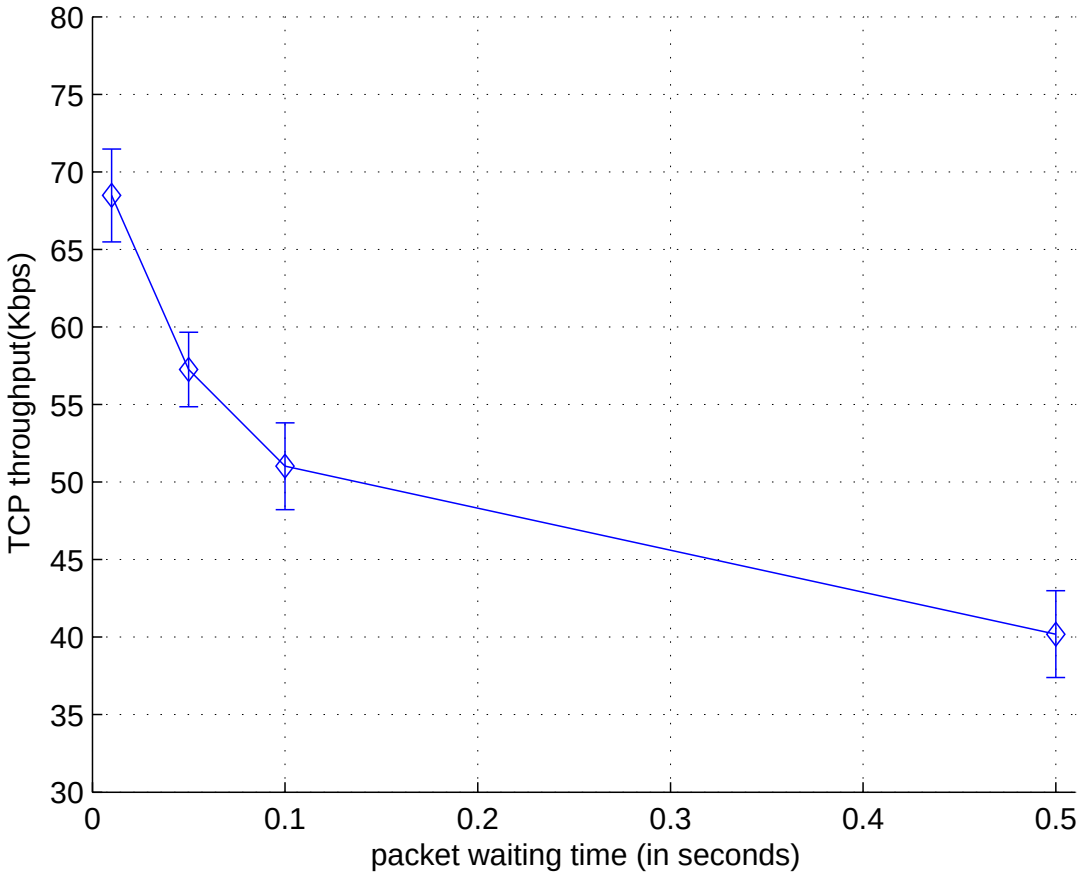

Figure 5.1: TCP throughput versus packet waiting time.

which consumes lot of network resources such as bandwidth leading to missed coding opportunities.

The analysis above shows that there is need for a better scheme. Therefore, this thesis presents a new scheme called adaptive-W scheme which adaptively controls packet waiting time in packetPool to achieve tradeoff of throughput and overhead.

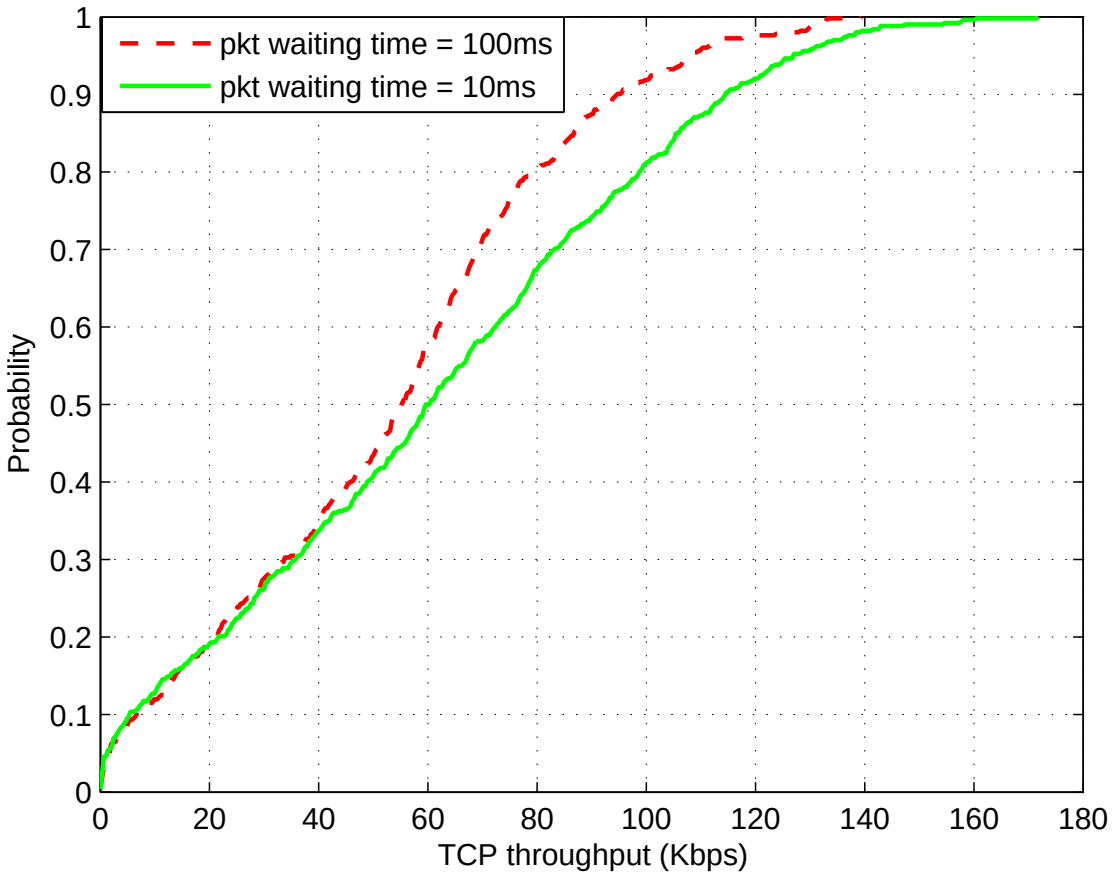

Figure 5.2: Cdf graphs for different packet waiting times.

5.1 Numerical Analysis

Through numerical analysis and results, this thesis proposes adaptive-W scheme. Specifically, consider (5.1) below;

$$F_e = 1 - (1 - BER)^{L_{fr}}, \tag{5.1}$$

where F_e is the frame error rate, BER is the bit error rate and L_{fr} is the length of the frame. Let l_c be the variable length (in bits) of reception report in XOR header such that $L_{fr} = L_d + l_c$ where L_d (in bits) is the data portion of the frame. Consider two reception reports (l_{ci} and l_{cj}) with two different lengths such that $l_{ci} < l_{cj}$ so

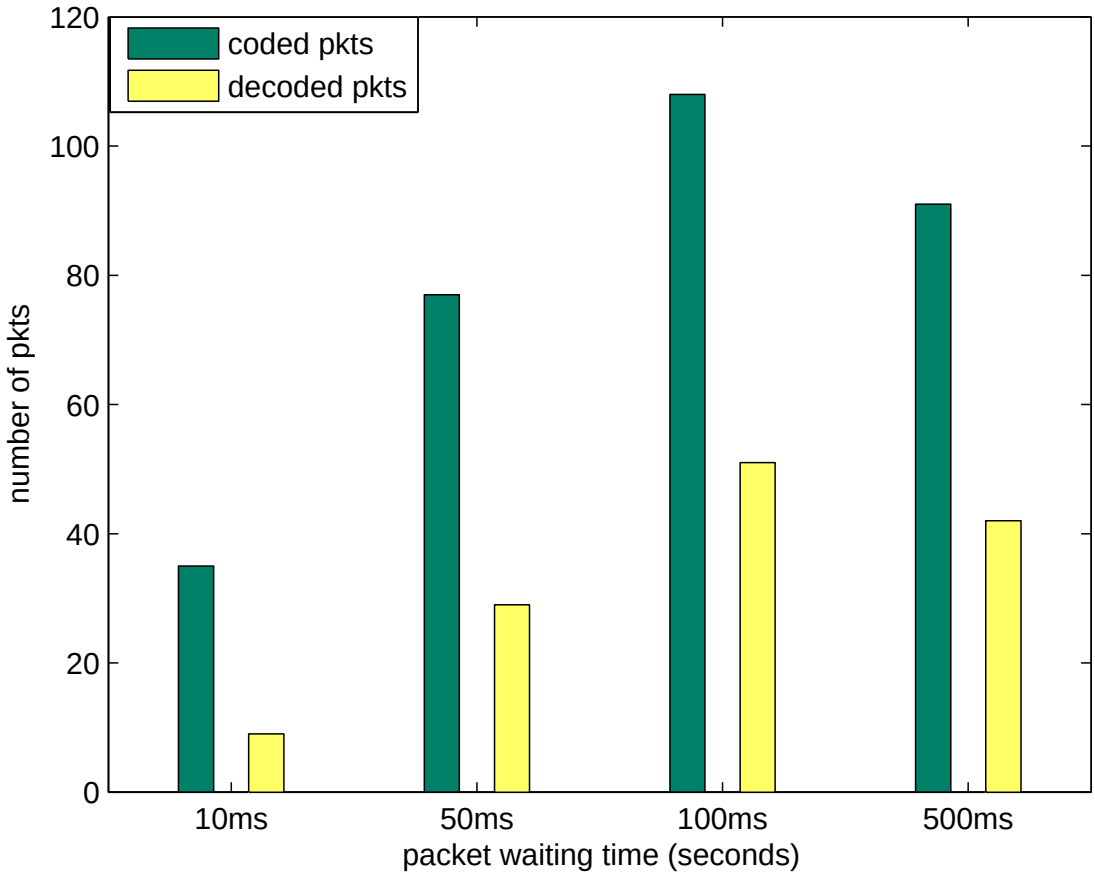

Figure 5.3: Coding opportunities versus packet waiting time.

that $\Delta F_e = F_{ej} - F_{ei}$. But $F_{ei} = 1 - (1 - BER)^{L_{fri}}$ and $F_{ej} = 1 - (1 - BER)^{L_{ej}}$. Therefore, we have

$$\Delta F_e = (1 - BER)^{L_d}\{(1 - BER)^{l_{ci}} - (1 - BER)^{l_{cj}}\} \tag{5.2}$$

From (5.2), $|1 - BER| < 1$. Therefore, for $\Delta F_e \geq 0$, $l_{ci} \leq l_{cj}$. Set $l_{ci} = 0$ as reference length of reception report in XOR header for analysis so that we have

$$\Delta F_e = (1 - BER)^{L_d}\{1 - (1 - BER)^{l_{cj}}\} \tag{5.3}$$

l_{cj} in (5.3) represents overhead introduced by reception report in XOR header. The more time packets spend in the packetPool, the longer the XOR header and the

packet itself. Let W be the average time spent by each packet in packetPool and λ be the average arrival rate of packets in packetPool. Also, let N be average number of packets in the packetPool so that $N = \lambda W$ according to Little's theorem. As shown previously, the XOR header has a byte reserved for number of reports in the reception report (see Fig. 3.8). Also, 5 bytes are reserved for the information of each packet being reported. Therefore, $l_{cj} = 8+40N$ in bits. Let $W \in \{10ms, 50ms, 100ms, 500ms\}$ and consider $L_d = 160$ bytes, $BER = 5.69917364\text{x}10^{-4}$ which corresponds to packet error rate, $PER = 0.01, N_{retrans} = 6, N_{fr} = 1$. If $W = 10ms$ and $\lambda = 10pkts/sec$, then $N = \lambda W = 0.1$ giving $\Delta F_e = 3.28644\text{x}10^{-3}$ which corresponds to 0.63% overhead as shown in Fig. 5.4. Note that ΔF_e indicates by how much F_e increases as frame size increases due to reception report overhead. For example, in Fig. 5.4, if packet's waiting time is $0.2sec$ and arrival rate of packets is 50pkts/sec, then the frame error rate increases by 20%.

When a node gets an opportunity to transmit data packet, it goes through the entire set of packets (in packetPool) which were previously sent or overheard from the network and extract information for reception report to be included in the outgoing packet. Therefore if there is high network traffic, nodes will overhear lots of packet in a particular time interval and vice versa. Consequently, with high network traffic, lots of packet will be kept in packetPool which will result in long reception reports. This introduces additional F_e as shown in Fig. 5.4, which results in performance deterioration. Fig. 5.5 shows that if the average time spent by packets in packetPool, W, is as long as $0.5sec$ and the average packet arrival rate is $100pkts/sec$, the overhead introduced by reception report in a packet is as high as 160% (that is approximately

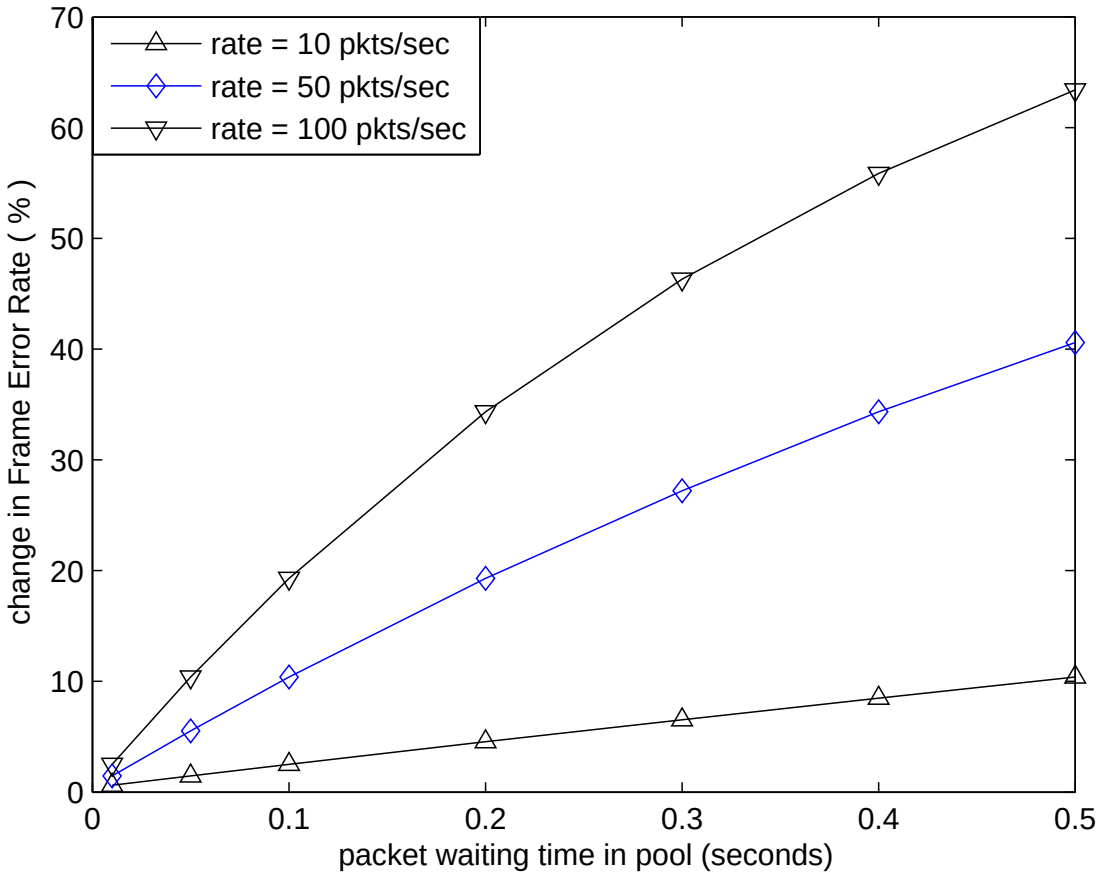

Figure 5.4: Impact of packet arrival rate on frame error rate.

50 packets in packetPool as shown in Fig. 5.6) which is more than doubling the packet length itself.

Although this presents more coding opportunities, it is an inefficient use of network/node resources because long reception reports consume lot of bandwidth, transmit power and node memory.

5.2 Fixed-W Scheme

In fixed-W scheme, W is kept as a constant indicating the amount of time packets should spend in packetPool regardless of network traffic level and required frame

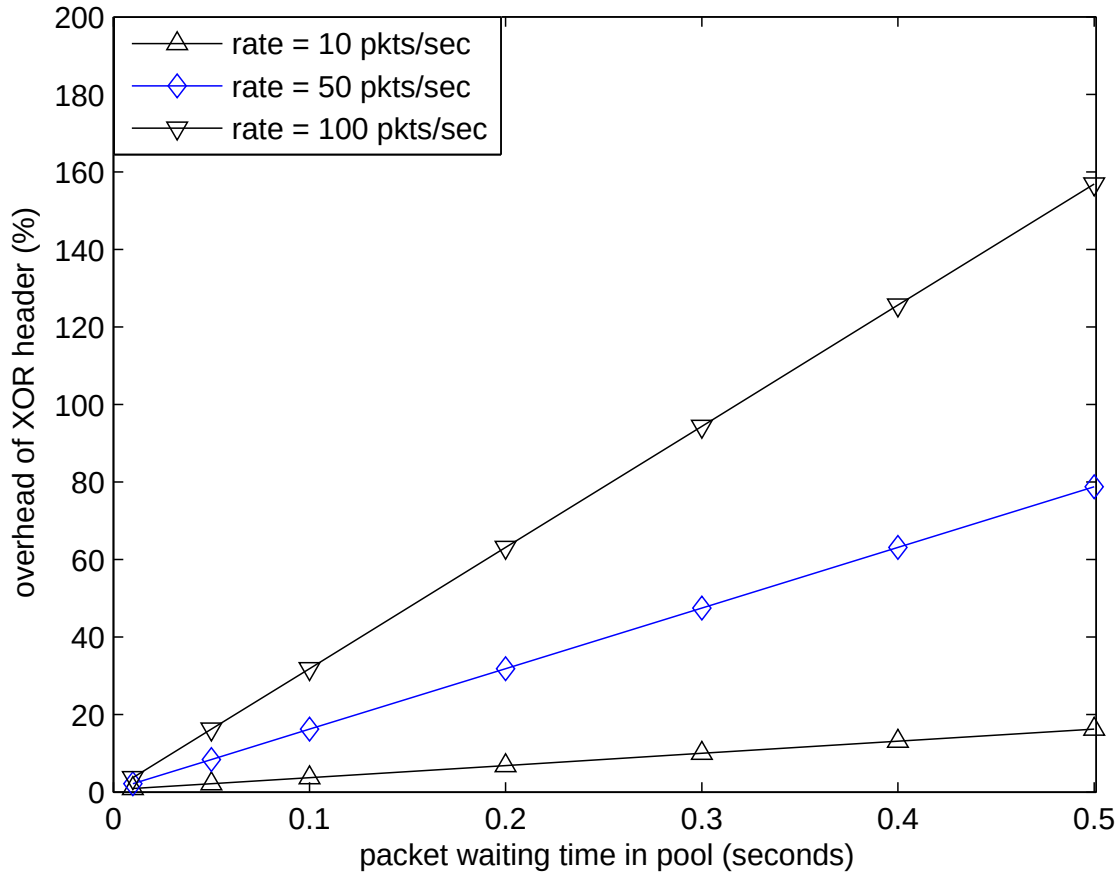

Figure 5.5: Effect of packet arrival rate in XOR header.

error rate. Therefore, λ is the only parameter that affects N. That is, if network traffic happens to be very high such that packets are overheard by a node in ubandance, then packetPool will have a large number of overheard packets and vice versa. Note that this scheme assumes that each node has an infinite amount of memory to accommodate all overheard packets which in practice is unrealistic.

5.3 Proposed Adaptive-W Scheme

This scheme adaptively controls the packet waiting time in packetPool given inter-arrival rate of overheard packets (λ) and required frame error rate, F_e. Specifically,

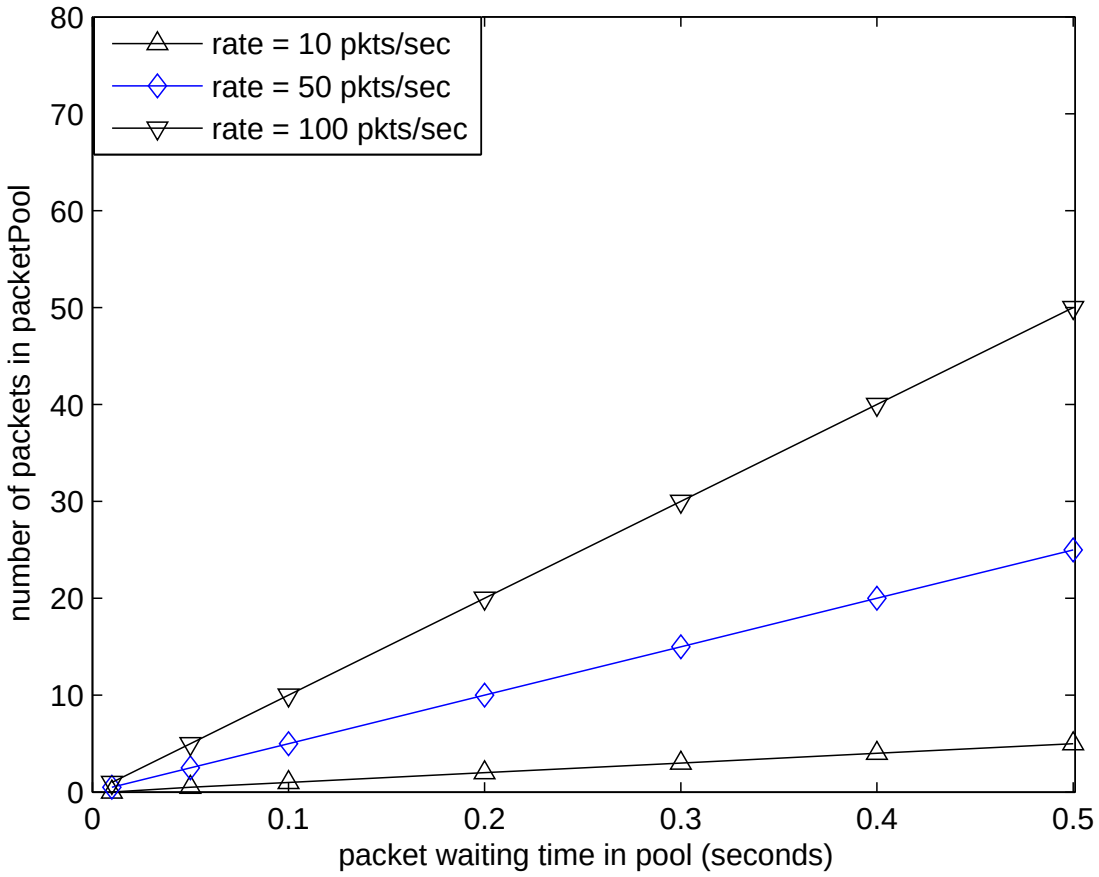

Figure 5.6: Effect of packet arrival rate on number of packets in packetPool.

for every ΔF_e that can be tolerated, there is a corresponding number of packets that can be stored in the packetPool. Let Ω represent the required number of packets that can be in the packetPool for a particular F_e. Whenever there is an overheard packet from the network or a packet that has just been sent by a node, a copy of it is added to the packetPool. For each addition of packet, the scheme ensures that there can never be more that Ω packets in the packetPool to satisfy the required F_e. The waiting time (W) for each packet in the packetPool is updated every $10ms$. If more packets are added within $10ms$, then packets in the packetPool will wait for shorter time before being discarded. However, if few packets are added within that particular

time interval, then packets in the packetPool will wait for longer time. The basic idea of this approach is to control packet overhead due to reception reports generated by using information of packets in the packetPool.

Note that λ is related to network traffic level and wireless channel conditions. That is, if network traffic increases, chances are that more packets will be overheard by nodes per unit time and vice versa. Also, if incoming channel quality is poor most of the time, more packets will be dropped and will never make their way into the packetPool.

W is time packets can spend in a packetPool such that $W \in \{t_1, t_2, \ldots, t_n\}$ where $t_1, t_2, \ldots, t_n \in \Re_+$ are optimized times packets. Specifically, consider Fig. 5.4. Suppose ΔF_e that can be tolerated for a specific frame error rate is 20%. Also, assume that the current average packet arrival rate, $\lambda = 50 pkts/sec$ so that $W = 0.2 sec$. This corresponds to $N = 10 pkts$ in packetPool illustrated in Fig. 5.6 and 30% overhead introduced by reception report shown in Fig. 5.5. Therefore, if network traffic level changes such that average packet arrival rate is now $\lambda = 100 pkts/sec$, then by employing the proposed scheme, W will dynamically be changed to $0.1 sec$. This will still maintain the required F_e. Also, it maintains the average number of packets in the packetPool, N, at 10 shown in Fig. 5.6 which means approximately same number of coding opportunities can still be generated. In addition, reception report overhead is kept at 30%. If later on, the average packet arrival rate reduces by half (that is, $50 pkts/sec$), then adaptive-W will be dynamically changed to $0.2 sec$ and still maintain the required F_e.

Note that by adaptively controlling packet overhead, loss rate of packets is min-

imized. Furthermore, it utilizes bandwidth, node memory and transmit power more efficiently than fixed-W scheme. Nevertheless, this scheme introduces additional system complexity because it has to collect information on traffic level from the network by estimating average inter-arrival rate of overheard packets. Based on this information and the required F_e, this scheme then dynamically chooses the best W from a set of optimized times, $t_1, t_2, \ldots, t_n$.

5.4 Simulation Results and Discussion

5.4.1 Simulation Models

The system model employed in this work is Shadowing channel model. Parameters used are as follows: path loss exponent of 2.8, shadowing standard deviation of $6dB$, reference distance of $1m$. Network Simulator-2.33 version was used to carry out all simulations. Three simulation scenarios were defined. These are: mobility, network traffic increase and network node increase. Note that the vertical lines on simulation graphs represent 95% confidence level (see 4.5.2).

5.4.2 Simulation Results

In network traffic scenario, the network had 25 nodes. There was 100 meter separation between nodes placed in fixed positions representing a square Bravais lattice in a two dimensional plane. Traffic was increased by 2 TCP sessions each time from 8 to 16 TCP sessions. Note that as network traffic increases, nodes overhear packets in abundance. Therefore, without adaptive control of overheard packets in packetPool, high packet overhead is generated as demonstrated by fixed-W scheme in Fig. 5.7.

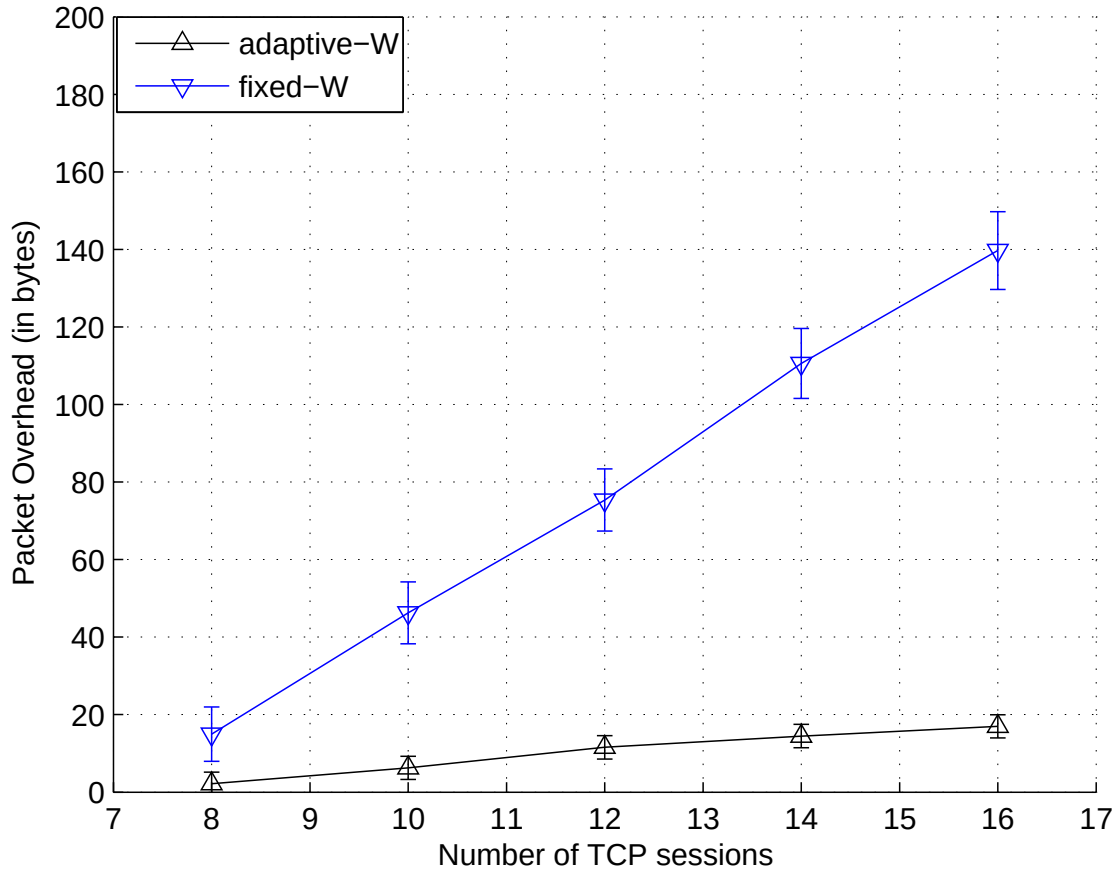

Figure 5.7: Impact of network traffic on packet overhead.

Fig. 5.8 shows that this adversely affects throughput because of inefficient bandwidth utilization. However, with adaptive control mechanism, packet overhead is kept low to meet the required F_e. Therefore, as shown in Fig. 5.8, this leads to better TCP throughput because of efficient bandwidth utilization. Also, less node memory and less transmission power are used.

In mobility scenario, initial locations, routes and final locations of nodes where predefined. Nodes were made to move towards and past each other. Initial locations of nodes had 100 meter separation distance between nodes. Traffic was fixed at 16 TCP sessions. Fig. 5.10 shows that when speed of nodes increase up to 30m/s, on average

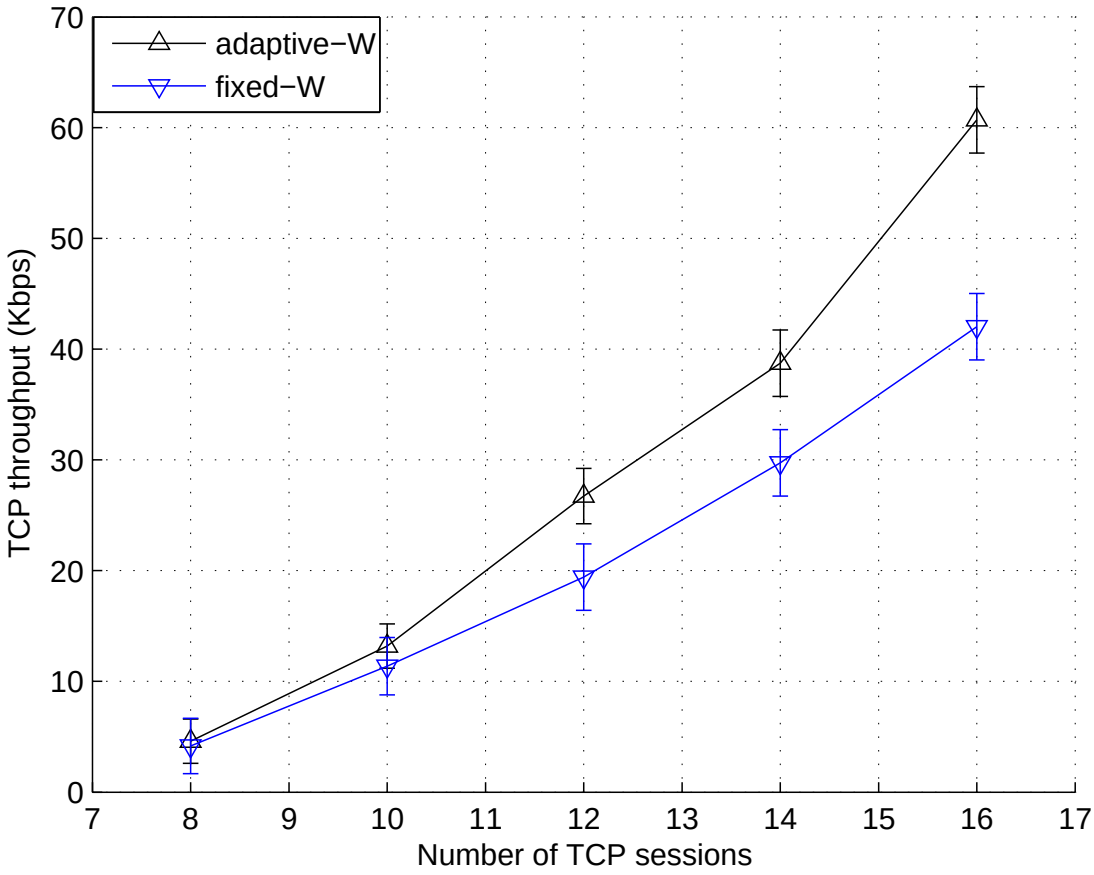

Figure 5.8: Impact of traffic on TCP throughput.

adaptive-W scheme exhibits approximately 15.5Kbits increase in TCP throughput for every meter travelled by each node in the network. Fixed-W scheme, on the other hand, shows only about 4.7Kbits increase. This is due to the fact that when nodes move towards each other at relatively low speed according to this scenario, they spend more time within each others communication range. Therefore, in adaptive-W scheme, the estimation made on inter-arrival rate of packets is relatively the same for most nodes. Consequently, given the required F_e, the average packet waiting time (W) computed is relatively the same for most nodes. This facilitates successful encoding and decoding of packets. On the other hand, low performance obtained

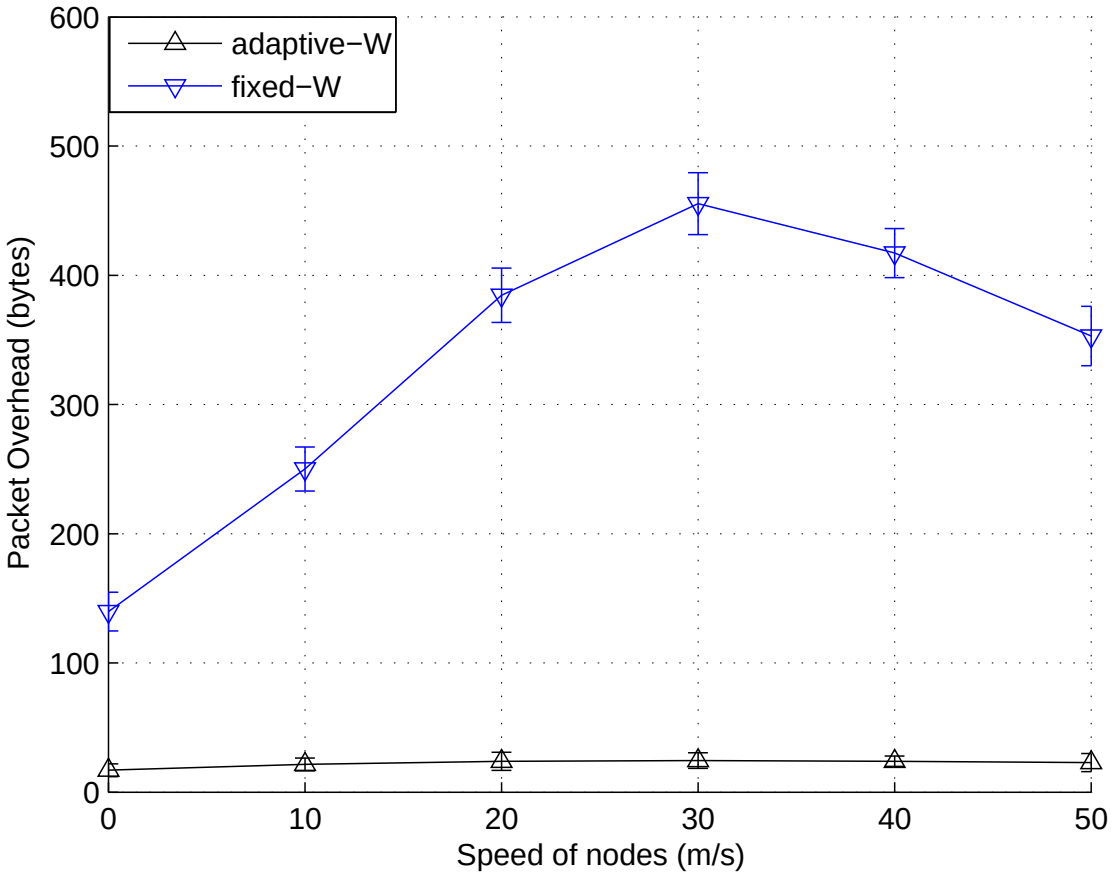

Figure 5.9: Impact of mobility on packet overhead.

by fixed-W scheme is due to packets that are delayed or lost because of high packet overhead as shown in Fig. 5.9. Note that for fixed-W scheme in Fig. 5.9, when network nodes are travelling at $30m/s$, approximately $450bytes$ of packet overhead is generated which corresponds to approximately 90 packets in packetPool. This is equivalent to packet overhead which is almost three times the size of a packet without reception reports included in its XOR header. As shown in Fig. 5.9, this does not only degrade network throughput but requires more memory at each node to accommodate packets overheard from the network. Also, more battery life is needed to transmit packets with high overhead.

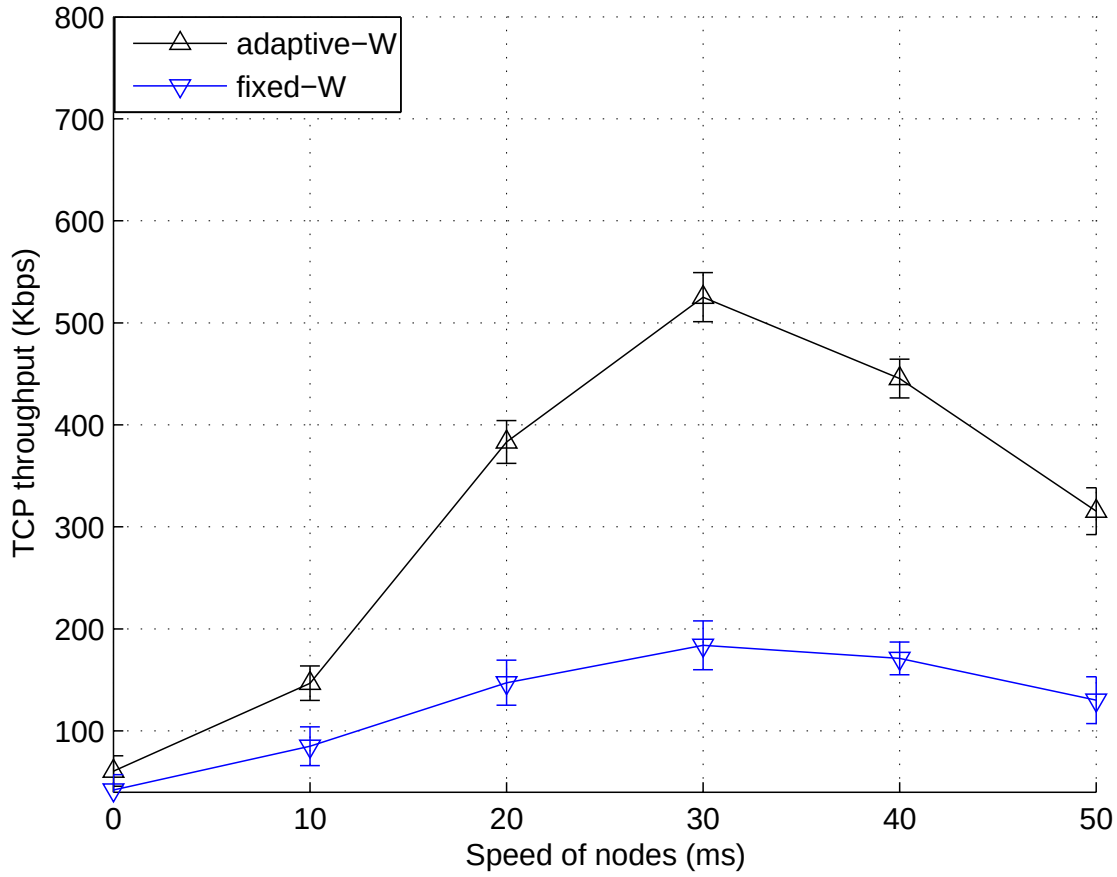

Figure 5.10: Impact of mobility on TCP throughput.

Note however that, when nodes move at speeds beyond 30m/s, TCP throughput deteriorates on average by about 10.5Kbits per meter travelled by each network node in adaptive-W scheme. This is attributed to frequent change in network topology which results in redistribution of network traffic flow. Therefore, there is huge difference in λ estimates for each network node as nodes move further away from each other. Due to different λ estimates for each node, the computed Ws by nodes vary which means that chances of some intended receivers failing to decode coded packets are very high as some nodes may have already discarded packets to be used in decoding. On the other hand, TCP throughput decreases by approximately 2.7Kbits per meter

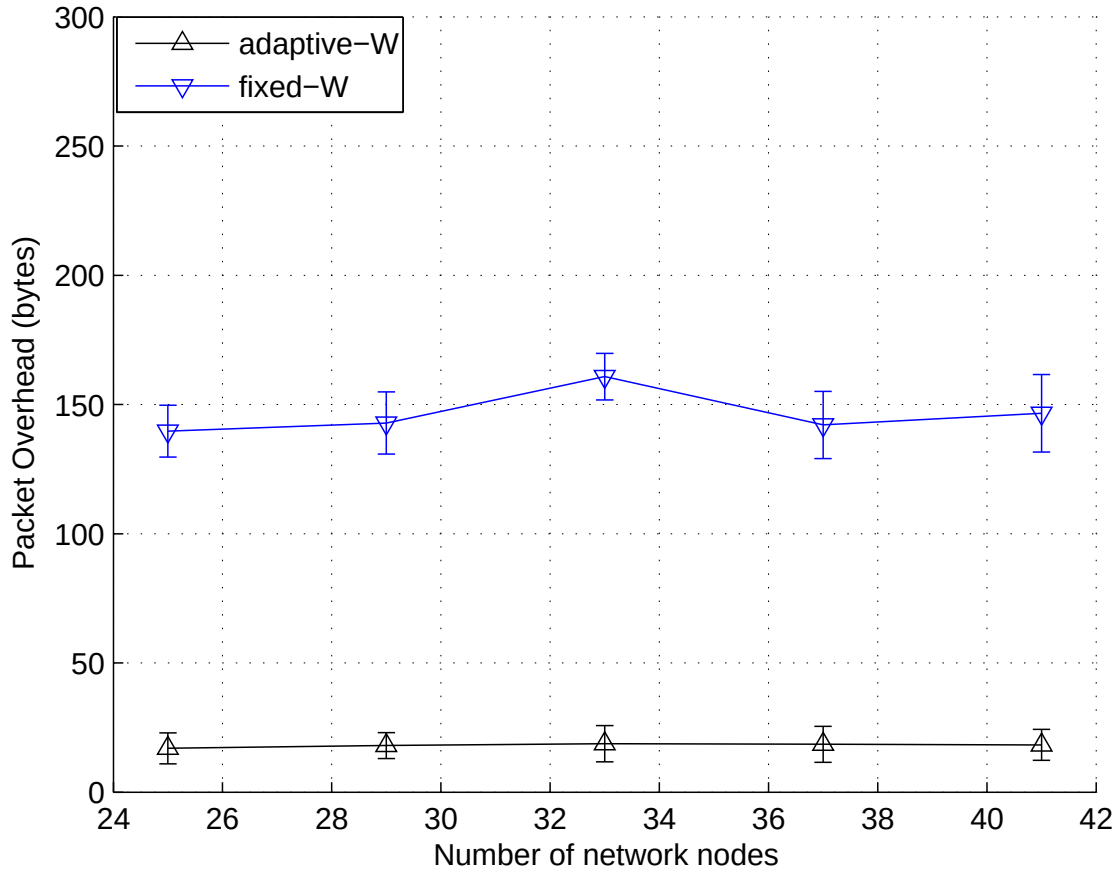

Figure 5.11: Impact of network nodes on packet overhead.

travelled by each node, which is less than the former. This is mainly due to overheard packets which spend long time in packetPool hence facilitates successful decoding of incoming coded packets. Despite this, adaptive-W scheme continues to perform far much better than fixed-W scheme. Take-away point from this analysis is that when nodes are crowded, packet overhead is the dominant factor in determining the overall TCP throughput but when nodes move away from each other, the dominant factor is how often can coded packets be successfully decoded at the receivers.

In network node scenario, nodes were increased by 4 each time from 25 to 41. New nodes were placed among existing nodes and 16 TCP sessions were generated. In this

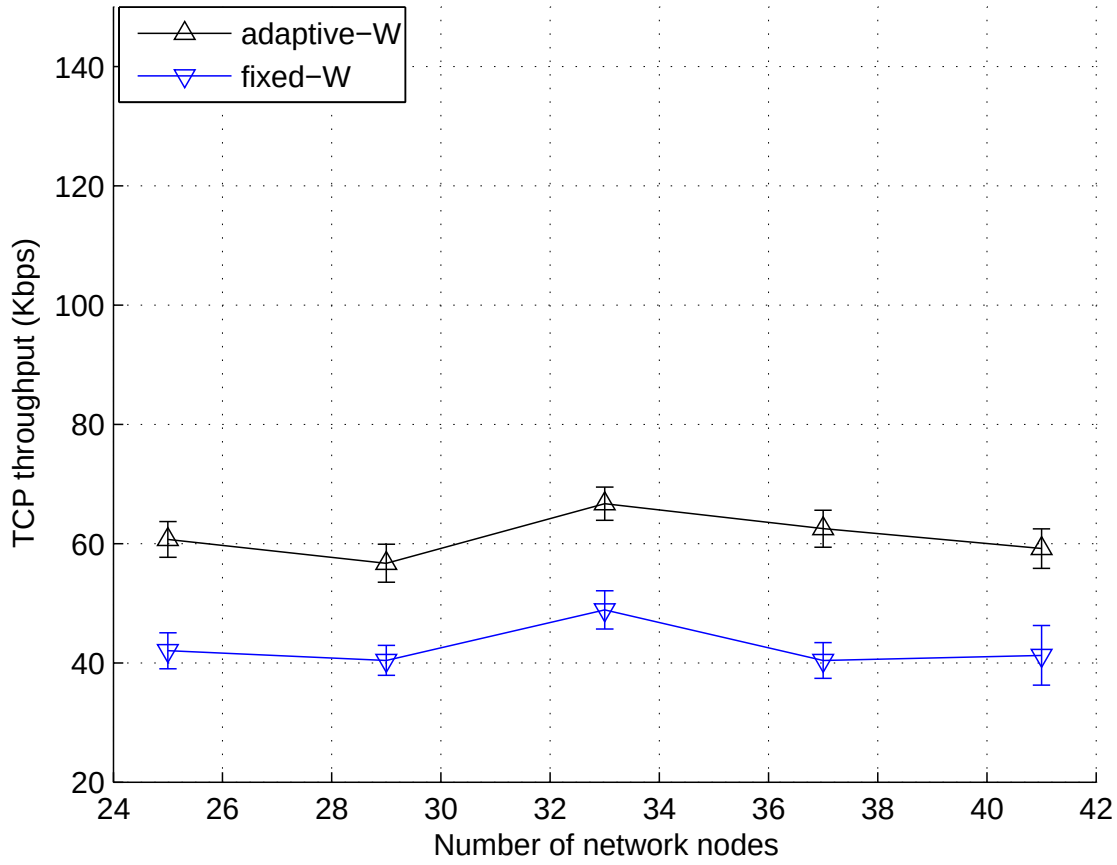

Figure 5.12: Effect of nodes on TCP throughput.

scenario there is no performance change when number of network nodes increases as shown in Fig. 5.12. However, adaptive-W scheme exhibits better performance than fixed-W scheme. The dorminant factor in overall network throughput is packet overhead. Fig. 5.11 shows packet overhead of approximately 145 bytes for fixed-W scheme which is equivalent to 29 packets in the packetPool. On the other hand, for adaptive-W scheme there is atmost about 20 bytes of packet overhead which is equivalent to 4 packets in packetPool.

Coding opportunities in adaptive-W scheme are limited as illustrated in Fig. 5.13 which corresponds to 25 network nodes in Fig. 5.12 and in Fig. 5.11. This is due to

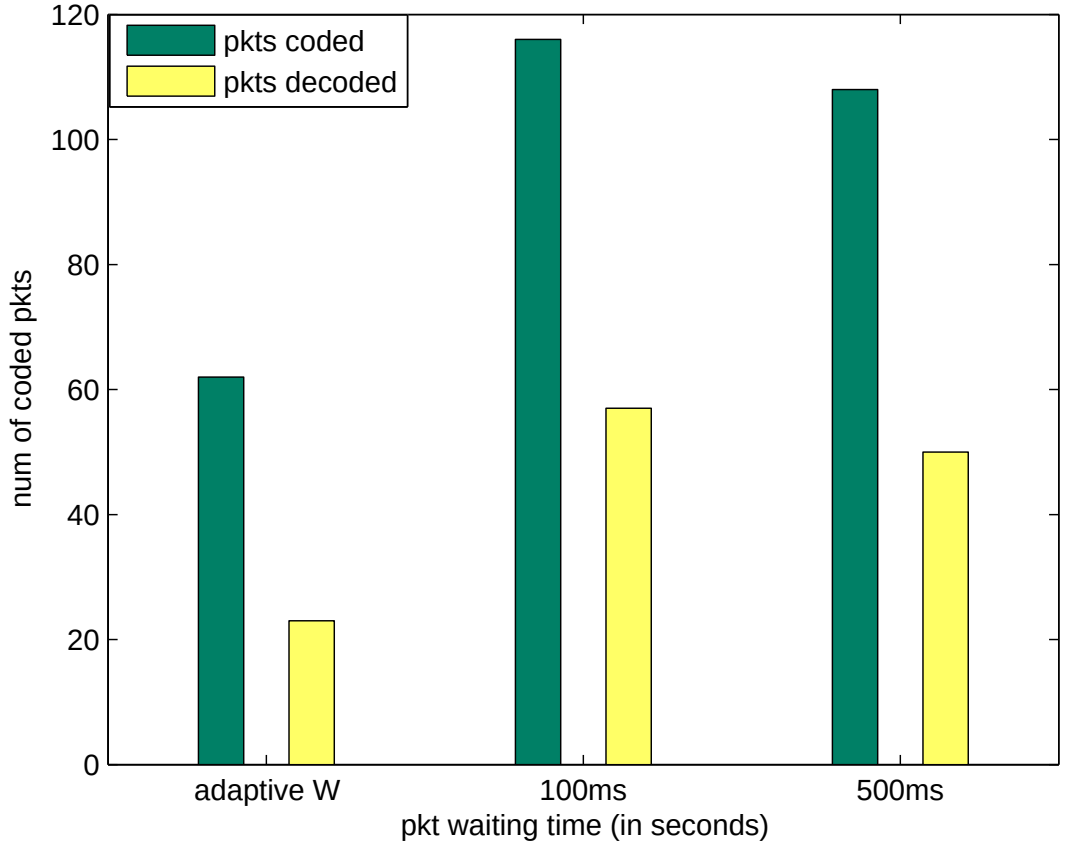

Figure 5.13: Coding opportunities versus packet waiting time.

the required frame error rate (F_e) adaptive-W scheme has to satisfy which specifies the number of packets that can be stored in packetPool at any time instant. As a results of this, few reception reports are shared among nodes and this limits coding opportunities. Furthermore, adaptive-W scheme has approximately 33% chance of successfully decoding coded packets where as fixed-W scheme has close to 50% chance of successfully decoding incoming coded packets. This is mainly due to network nodes which are far away from each other and thus experience different network traffic levels. As a result, there are different computed Ws in adaptive-W scheme by nodes. Therefore, some nodes end up discarding packets which could be useful in decoding

incoming coded packets. It has to be emphasized that part of the reason why both schemes exhibits low probability of successfully decoding incoming coded packets is that TCP and channel information were not taken into account.

5.5 Summary

This chapter presented the problem of fixing packet waiting time in packetPool in XOR network coding. It also analysed the disadvantages of that approach. To that end, it presented the new adaptive-W scheme whose objective is to adaptively control packet waiting time in packetPool in order to achieve tradeoff of network throughput and overhead. Simulation results showed significant performance improvement demonstrated by the new adaptive-W scheme as compared to fixed-W scheme.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

The problem stated in section 2.2 has been solved: as shown in chapters 3 and 4, an efficient TCP-Aware network coding with opportunistic scheduling, which is capable of enhancing TCP performance in wireless mobile ad hoc networks has been developed. In addition, adaptive-W scheme, whose purpose is to adaptively control packet waiting time in packetPool to achieve tradeoff of throughput and overhead, has been developed. Simulation results show that when traditional network coding is upgraded to TCP-Aware network coding and combined with opportunistic scheduling in wireless mobile ad hoc network, there is significant performance improvement as compared to other schemes in low and high mobility environment, when network traffic increases and when the number of network nodes increases. Simulation results also show that when adaptive-W scheme is considered, more performance improvement can be achieved.

6.2 Future Work

The future work include: 1) optimization of parameters indicating the state of TCP congestion window size; 2) to enhance the performance of TCP-Aware network coding

with opportunistic scheduling in high mobility environment in which steepest decline in TCP throughput was experienced as nodes started traveling at high speed; 3) to find mechanisms through which adaptive-W scheme can be enhanced more especially in high mobility environment which showed steepest decline in throughput; 4) studying routing layer and finding ways by which its performance can be enhanced.

Bibliography

[1] H. Yomo and P. Popovski, "Opportunistic Scheduling for Wireless Network Coding," in *Proc. IEEE ICC'07*, pp. 5610–5615, June 2007. [cited at p. vii, 2, 13, 14, 19, 22, 41]

[2] R. Ahlswede, N. Cai, S.-Y. Li, and R. Yeung, "Network Information Flow," *IEEE Trans. Inform. Theory*, vol. 46, pp. 1204–1216, Jul. 2000. [cited at p. 1]

[3] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi, "On MAC Scheduling and Packet Combination Strategies for Practical Random Network Coding," in *Proc. IEEE ICC'07*, pp. 3582–3589, June 2007. [cited at p. 1, 2]

[4] J. Jin and B. Li, "Adaptive Random Network Coding in WiMAX," in *Proc. IEEE ICC'08*, pp. 2576–2580, May 2008. [cited at p. 1, 2, 13, 16]

[5] D. Katabi, S. Katti, W. Hu, H. Rahul, and M. Medard, "On Practical Network Coding for Wireless Environments," in *Proc. Int'l Zurich Seminar on Communications*, pp. 84–85, 2006. [cited at p. 1, 13, 14]

[6] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, and J. Crowcroft, "XORs in the Air: Practical Wireless Network Coding," *IEEE/ACM Trans. Netw.*, vol. 16, pp. 497–510, June 2008. [cited at p. 1, 17, 23, 30, 37, 63]

[7] D. Nguyen, T. Tran, T. Nguyen, and B. Bose, "Hybrid ARQ-Random Network Coding For Wireless Media Streaming," in *Proc. ICCE'08*, pp. 115–120, June 2008. [cited at p. 1]

[8] L. Scalia, F. Soldo, and M. Gerla, "PiggyCode: A MAC Layer Network Coding Scheme to Improve TCP Performance Over Wireless Networks," in *Proc. IEEE GLOBECOM '07*, pp. 3672–3677, Nov. 2007. [cited at p. 1]

[9] M. Wang and B. Li, "Lava: A reality check of network coding in peer-to-peer live streaming," in *Proc. IEEE INFOCOM'07*, pp. 1082–1090, May 2007. [cited at p. 1, 2, 13]

[10] A. Campo and A. Grant, "Robustness of Random Network Coding to Interfering Sources," in *Proc. 7th Australian Communications Theory Workshop*, pp. 120–124, Feb. 2006. [cited at p. 2, 13]

[11] K. Li and X. Wang, "Cross-Layer Design of Wireless Mesh Networks with Network Coding," *IEEE Trans. Mobile Comput.*, vol. 7, pp. 1363–1373, Nov. 2008. [cited at p. 2]

[12] K. Lu, S. Fu, and Y. Qian, "Capacity of Random Wireless Networks: Impact of Physical-Layer Network Coding," in *Proc. IEEE ICC'08*, pp. 3903–3907, May 2008. [cited at p. 2]

[13] H.-M. Zimmermann and Y.-C. Liang, "Physical Layer Network Coding for Uni-Cast Applications," in *Proc. IEEE VTC'08S*, pp. 2291–2295, May 2008. [cited at p. 2, 12]

[14] D. Koutsonikolas, Y. C. Hu, and C.-C. Wang, "An Empirical Study Of Performance Benefits Of Network Coding In Multihop Wireless Networks," in *Proc. IEEE INFOCOM'09.*, 2009. [cited at p. 5, 15]

[15] H. Balakrishnan, V. Padmanabhan, S. Seshan, and R. Katz, "A Comparison Of Mechanisms For Improving TCP Performance Over Wireless Links," *IEEE/ACM Trans. Netw.*, vol. 5, pp. 756–769, Dec 1997. [cited at p. 7, 33, 34]

[16] W. Jun, F. Xi, C. N. Quan, J. Min, and Z. Ping, "A Cross-Layer Wireless TCP Enhancement Scheme In OFDM Network," in *Proc. IEEE Region 10 Conference*, pp. 1–4, Nov. 2006. [cited at p. 7, 34]

[17] T. Hasegawa, T. Hasegawa, and M. Lagreze, "A Mechanism for TCP Performance Enhancement over Asymmetrical Environment," in *Proc. IEEE ISCC03*, 2003. [cited at p. 7]

[18] M. Ghaderi, A. Sridharan, H. Zang, D. Towsley, and R. Cruz, "TCP-Aware Channel Allocation in CDMA Networks," *IEEE/ACM Trans. Netw.*, vol. 8, pp. 14–28, Jan. 2009. [cited at p. 7, 34, 44]

[19] S. Zhang, S. C. Liew, and L. Lu, "Physical Layer Network Coding Schemes Over Finite And Infinite Fields," in *Proc. IEEE GLOBECOM'08*, pp. 1–6, 30 2008-Dec. 4 2008. [cited at p. 11]

[20] F. Xue and S. Sandhu, "PHY-Layer Network Coding For Broadcast Channel With Side Information," in *Proc. IEEE ITW '07*, pp. 108–113, Sept. 2007. [cited at p. 12]

[21] J. K. Sundararajan, D. Shah, M. Medard, M. Mitzenmacher, and J. Barros, "Network Coding Meets TCP," in *Proc. IEEE INFOCOM'09*, April 2009. [cited at p. 12, 18]

[22] Z. Lin and B. Vucetic, "Power And Rate Adaptation For Wireless Network Coding With Opportunistic Scheduling ," *IEEE ISIT'08*, pp. 21–25, July 2008. [cited at p. 14, 19]

[23] Y. Sagduyu and A. Ephremides, "Cross-Layer Design For Distributed MAC And Network Coding In Wireless Ad Hoc Networks," in *Proc. IEEE ISIT'05*, pp. 1863 – 1867, Sept 2005. [cited at p. 14]

[24] B. Scheuermann and et al, "Near-Optimal Coordinated Coding In Wireless Multihop Networks," in *Proc. of ACM CoNEXT'07*, 2007. [cited at p. 14]

[25] P. Chaporkar and A. Proutiere, "Adaptive Network Coding And Scheduling For Maximizing Throughput In Wireless Networks," in *Proc. of ACM MOBICOM'07*, 2007. [cited at p. 14]

[26] P. Chou, Y. Wu, and K. Jain, "Practical Network Coding," *Allerton Conference on Communication, Control and Computing*, 2003. [cited at p. 15]

[27] M. Halloush and H. Radha, "Network Coding With Multi-Generation Mixing," in *Proc. CISS'08*, (Piscataway, NJ, USA), pp. 515 – 20, 2008. network coding;multigeneration mixing;data propagation;data recovery;. [cited at p. 15]

[28] S. W. Kim, "Concatenated Random Parity Forwarding in Large-Scale Multi-Hop Relay Networks," in *Proc. IEEE MILCOM'07.*, pp. 1–7, Oct. 2007. [cited at p. 16]

[29] R. Prasad, H. Wu, D. Perkins, and N.-F. Tzeng, "Local Topology Assisted XOR Coding in Wireless Mesh Networks," in *Proc. ICDCS '08*, pp. 156–161, June 2008. [cited at p. 16, 17, 63]

[30] X. Wang, G. B. Giannakis, and A. G. Marques, "A Unified Approach to QoS-Guaranteed Scheduling for Channel-Adaptive Wireless Networks," *Proc. IEEE'07*, vol. 95, pp. 2410–2431, Dec. 2007. [cited at p. 31]

[31] K. Daoud and B. Sayadi, "HAD: A Novel Function for TCP Seamless Mobility in Heterogeneous Access Networks," in *Proc. IEEE VTC'07F*, pp. 1451–1455, 30 2007-Oct. 3 2007. [cited at p. 33]

[32] Y. Wu, Z. Niu, and J. Zheng, "A Network-Based Solution For TCP In Wireless Systems With Opportunistic Scheduling," in *Proc. IEEE PIMRC'04*, 2004. [cited at p. 34, 43]

[33] K. Igarashi and K. Yamazaki, " Flight Size Auto Tuning for Broadband Wireless Networks," *submitted to IWCMC 2009 Next Generation Mobile Networks Symposium*, 2009. [cited at p. 45]