This is to certify that the
dissertation entitled

SELF ORGANIZATION IN MEDIUM ACCESS CONTROL
FOR WIRELESS AD HOC AND SENSOR NETWORKS

presented by

FAN YU

has been accepted towards fulfillment
of the requirements for the

Ph.D. degree in Electrical Engineering

Major Professor's Signature

10/29/08

Date

SELF ORGANIZATION IN MEDIUM ACCESS CONTROL FOR WIRELESS AD HOC AND SENSOR NETWORKS

By

Fan Yu

A DISSERTATION

Submitted to
Michigan State University
In partial fulfillment of the requirements
For the degree of

DOCTOR OF PHILOSOPHY

Electrical Engineering

2008

ABSTRACT

SELF ORGANIZATION IN MEDIUM ACCESS CONTROL
FOR WIRELESS AD HOC AND SENSOR NETWORKS

By

Fan Yu

The objective of this thesis is to investigate the role of protocol self organization at the Medium Access Control Layer in wireless ad hoc and sensor networks. Protocol self organization is defined as autonomous and self-calibrating communication state machines that can adjust their operating parameters and state machine logic as a reactive response to external operating environment such as network load, node crowding, topology changes due to mobility, energy depletion, infrastructure failure and communication errors. While the traditional wired networks continue to enjoy relatively stable operating environments, the emerging wireless and sensor networks suffer from the mentioned operating instabilities due to their ad hoc deployments, specialized applications and various resource constraints. The goal in this work is to utilize protocol reorganization as defined above in order to mitigate the effects of the mentioned instabilities. We propose a MAC self organization framework for both intra-MAC and inter-MAC as outlined below.

For intra-MAC self organization, we develop an In-band Self-Organized MAC (*ISOMAC*), for wireless sensor networks with arbitrary mesh topologies. The specific problems the mechanism attempts to solve are arbitrary ad hoc deployment, energy limitation, and network dynamics. The novelty of the proposed framework

lies in its in-band control mechanism. Both analytical and simulation models show that in addition to the reasonably fast reorganization convergence, the energy penalty of the in-band information is quite negligible.

A cross-layer approach, Minimized Slot Misordered Routing (MSMR) with *ISOMAC* style MAC, is also developed. The self organizing TDMA MAC schedule is used as input to the routing protocol to generate routing paths.

As for validation and extension of the *ISOMAC* framework, an application adaptation in vehicular wireless networks is investigated. An adapted ISOMAC, Vehicular Self-Organizing MAC (*VeSOMAC*), which is capable of inter-vehicle message delivery with short and deterministic delay bounds has been developed. Through ns2 simulation experiments, the efficiency of self organizing abilities of *VeSOMAC* is demonstrated in the presence of fast topology variation in vehicular networks.

Finally, we propose an inter-MAC self organization framework in the form of dynamic MAC protocol switching. The problem we address with this framework is to make a network adaptable with dynamic network loading conditions while providing acceptable user perceived performance. The key concept is that a network node can autonomously switch across different MAC layer protocols as a response to network traffic heterogeneity. Therefore, the reorganization here is accomplished across multiple protocols. Both theoretical and experimental analyses support that the dynamic MAC protocol switching logic enhances the network-wide throughput in the presence of traffic heterogeneity.

To My Families

For All Their

Love and Support

Acknowledgements

I would like to thank my advisor Dr. Subir Biswas for supporting me through my Ph.D. study. His patient guidance allowed me to flourish during my research. I would also like to thank my committee Dr. Michael Shanblatt, Dr. Jian Ren and Dr. Philip McKinley for their precious suggestions and advices.

Thanks must be given to my lab mates Tao Wu, Jayanthi Rao, Anthony Plummer, and Muhannad Quwaider for all of the helps, brainstorming and discussions. Last but not least, I would like give thanks to Changqing Chen, Juanita Dion and Lan Yang along with the rest of my friends for supporting me through this process.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Background

1.1. Wireless Ad Hoc Networks

Wireless ad hoc networks are formed by wireless communication links among multiple communication devices. Nodes in an ad hoc network can forward data for other nodes, and hence the decision on whom and where the forwarding data should go are dynamically optimized based on the network connectivity and traffic load. It is unlike the traditional networks in which the task of data forwarding is commonly performed by designated routers with customized hardware and software. It is also different from managed wireless networks where a special node called access point manages the communication among other nodes.

Minimal configuration and quick deployment are the two most important features of ad hoc networks. These features make these networks favorable for emergency and tactical situations such as battle-field surveillance and disaster recovery. The decentralized nature of most wireless ad hoc networks provides the adaptability of applications in circumstances in which a central entity is not available or cannot be relied on. And the scalability of the wireless ad hoc networks can be possibly improved in contrast with managed wireless networks, although the overall capacity is still limited, which has already been identified in [1] and [2] theoretically and practically .

1.2. Wireless Sensor Networks

Wireless sensor is a hardware device which is equipped with battery, sensor

module, micro-processor, memory, and RF receiver/transmitter. The advances in micro-electro-mechanical system (MEMS) technology and digital electronics have made it possible to integrate a wireless sensor node with low-cost, low-power tiny hardware. For example, a Berkeley and Crossbow mote [3, 4] has an 8-bit Atmel AT90LS8535 microprocessor as CPU running at 7.3728 MHz clock frequency, providing a data transmission rate of 19.2 Kbps by using the 916 MHz wireless frequency. Figure 1.1 shows some of the currently available sensor devices.

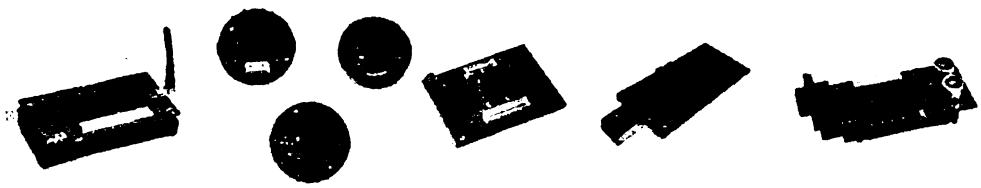


Figure 1.1: Sensor Nodes Mote2, Mote2dot, Imote, TelosB, Cricket

The concept of wireless sensor networks (WSN) is leveraged by these tiny sensor nodes, capable of sensing, processing, and communicating data. Many applications can be operated via WSN, such as environmental monitoring, target field imaging, intrusion detection, tactical surveillance, vehicle-to-vehicle management, seismic structure response, and ecosystems monitoring. A sensor network usually is composed of a large number of sensor nodes that are densely deployed either within the phenomenon or very close to it in a stationary manner. Most of the time it is not necessary to engineer or predetermine the exact geographic positions of sensor nodes, which makes it possible to arbitrarily deploy in some inaccessible terrains or in disaster relief environments. Hence self-organization capability of individual sensor nodes is essential for algorithms and protocols used in sensor networks. Collaboration between the sensor nodes is another distinct feature. Due to the limited

resources, each sensor node is expected to carry out few simple local processing and to transmit only the required and partially processed data to a "fusion" node through a low data rate flow. Node failure is also a problem that the sensor network protocols and algorithms have to face since sensor nodes are powered by battery and in most cases frequent replacements of batteries are not a possibility.

Following is a list of wireless sensor network specific issues [5-8]:

Application Specific: There are a large number of applications of the sensor networks such as environment data collection, public facility monitoring, vehicle tracking, battle-field surveillance and intrusion detection, which all benefit from combination of sensing, computing and communication technology. Different application scenarios diverge from each other (i.e., design requirements of a sensor network change with application), thus it is unlikely to find a general solution for all applications. For example, the challenging problem of low-latency precision tactical surveillance is different from that of a periodic weather monitoring task.

Ad Hoc Deployment: Often times, there is no infrastructure in the regions where sensor nodes are deployed. Under unplanned wireless sensor network deployment, it is the individual sensor node's responsibility to figure out the connectivity and data flow direction. This requires the sensor nodes to be self-organizing.

Energy Constraint: Different from ad hoc networks, sensor nodes in WSN are all powered by battery and the replacement is usually difficult. Thus energy consumption is a primary consideration for wireless sensor network protocol and algorithm design. Since communication is significantly more expensive than

computation [9], reducing energy dissipation used in communication becomes very important. Typically the non-essential energy consumptions are contributed from four sources.

- Protocol overhead: besides valid data certain control messages have to be exchanged to access communication resources;
- Message collision: unnecessary additional energy is wasted by retransmitting as well as receiving possible duplicate messages;
- Overhearing: sensor nodes spend energy in receiving messages for which they are not intended, because that message transmission usually is broadcast within WSN;
- Idle listening: when there is no centralized scheduler to manage sequence of transmission, sensor nodes have to keep the wireless radio interface up (power on) for possible message receptions. Even when the sensor nodes do not transmit or receive any messages, the consumed energy can still be prevalent. Researches [9, 10] have shown that idle listening actually accounts for most of the energy expenditure at low traffic situations, which is quite common for most applications in WSN.

In order to prolong network lifetime, different strategies should be incorporated at various network layers including hardware improvement at physical layer, energy efficient medium access control protocol, avoiding packet collision and idle listening, optimal routing path selection, and content filtering or error tolerance at application layer.

Scalability: For certain applications, the number of sensor nodes in WSN can be very large, in some cases up to hundreds or thousands. Protocols and algorithms have to scale to these large numbers and deal with issues such as protocol overhead, limited bandwidth, and energy expenditure. Furthermore, it is not possible to build a global addressing scheme for the deployment of such large number of sensor nodes as the overhead of ID maintenance can become very high. Thus, traditional IP-based protocols can not be directly applied to WSN.

Network Dynamics: Although wireless sensor networks are often considered as stationary, some applications may require one or several powerful mobile nodes to achieve better performance [11, 12]. Beside mobile nodes, nodes addition and nodes failure (possibly due to battery drain) can also invoke network connectivity changes. The potential dynamics in the networks strengthens the need for self organization at the MAC and the routing layer. The MAC and routing protocols which can smoothly respond to the dynamics are preferable

Traffic Patterns: Unlike communication in traditional ad hoc networks which is based mostly on point-to-point transmission, sensor network mainly uses a multi-point to point communication paradigm and almost all applications of sensor networks require the flow of sensed data from multiple sources to a particular base station. This, however, does not prevent the flow of data to be in other forms (e.g., multicast or peer to peer). The reason that all messages are sent to all the radio neighbors is because a single sensor node lack the global knowledge of the whole network. In addition, the traffic in the sensor networks can be continuous,

event-driven, query-driven, or hybrid, based on specific applications.

Quality of Service: Providing bounded message latency and high delivery ratio may be crucial in certain time-constraint applications such as intrusion detection and disaster warning. However, in some other applications like environmental monitoring it is enough to have only occasional packet delivery but with long network lifetime requirements. Especially when battery power is below normal operating threshold, quality of service has to be compromised for lifetime.

Location Awareness: Location awareness in sensor nodes is important since data collection is normally based on the location of data and event generation. In a WSN often it is not feasible to use Global Positioning System (GPS) hardware in cost and energy constrained nodes. Methods based on triangulation [13], for example, allow sensor nodes to approximate their positions using radio strength from a few nodes with known coordinates. It is found in [13] that algorithms based on triangulation or multilateration can work quite well under conditions where only very few nodes know their positions a priori.

1.3. Applications of Wireless Sensor Networks

Sensor networks are initially motivated by military applications such as intrusion detection, battle-field surveillance and enemy tracking. Recent research shows a rapid increase on the civilian applications like environment data collection, health monitoring, and disaster relief.

1.3.1 Military applications

Battlefield Surveillance: Important terrains, routes, and paths can be monitored by

deploying sensor nodes to the dedicated field. Sensor nodes can sense the activities or predefined critical events then send raw or post-processed information to the base station. Real-time situation requests can also be generated at the base station, and disseminated to requested areas.

Target Tracking: Collaborative mobile sensing can cooperate with navigation system. For example, networked tracking of adversarial mobile targets using unmanned ground and airborne sensors is an emerging application for the militaries' network-centric warfare capability. Target tracking is a prevalent military requirement across a wide spectrum of surveillance and reconnaissance scenarios including those in remote and urban battlefields, inside large buildings and public places such as airports, train stations, and inside underground tunnels.

Forces and Equipment Monitoring: All forces and equipment can be attached with sensor nodes. Then the attached sensor nodes can periodically report the status to the commanders or leaders. The report can be sent to a single sink or multiple sinks and forward to upper command hierarchy if necessary.

1.3.2 Environmental Data Collection Applications

Clickable Ecosystem: This system is developed in the cooperation of Dr. Stuart Gage from Department of Entomology and Dr. Subir Biswas from Department of Electrical and Computer Engineering at Michigan State University. Sensor nodes are placed at interested spots in the outdoor field. Each sensor node automatically records the acoustic signals, weather data and images of monitoring spot. After the recording, acoustic signals are sent back to a data server in the laboratory via

wireless through a single-hop or multi-hop network depending on location and availability. If individual sensor node is strong enough and there are not many obstacles in the sensing field, single-hop communication will be used to collect all sensing data. Otherwise, multi-hop communication will be used so that sensing field is covered without requiring a very powerful sink node. When an acoustic signal is received by the server, further analysis on biological indices are computed based on acoustic intensity in selected frequency bands. The final environmental acoustics are then stored in digital library, which can be accessed via regular internet network. Some sound tracks, sensing spot images and videos are available in [14].

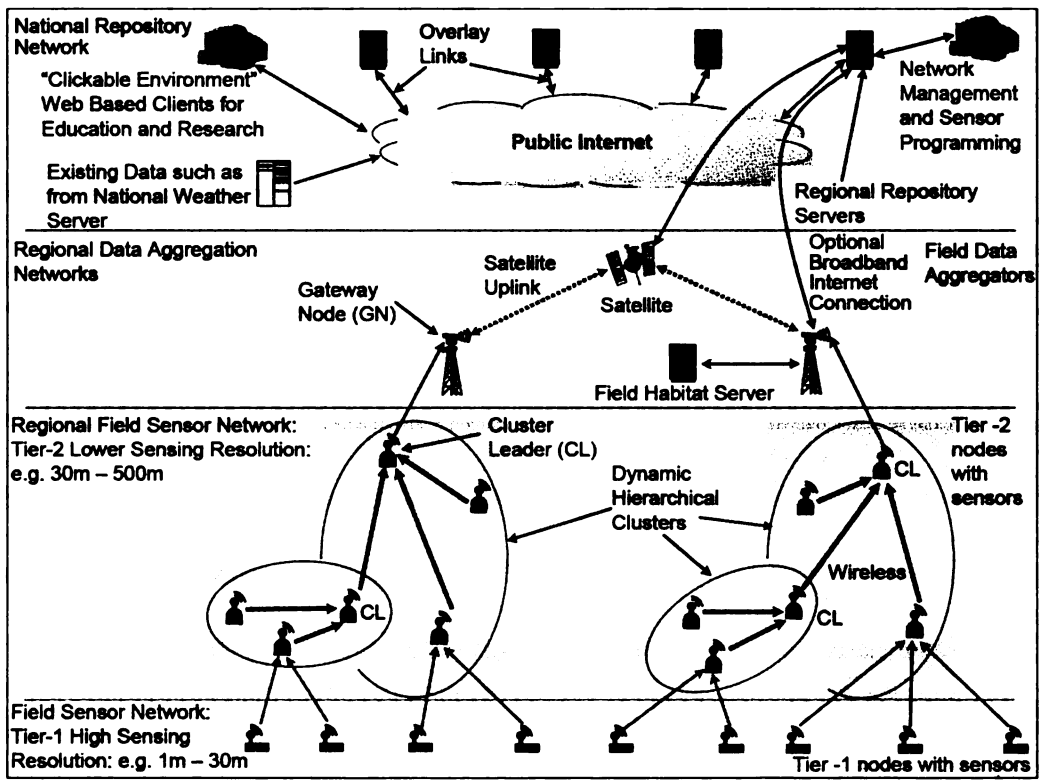


Figure 1.2: System architecture of clickable environment [14]

1.3.3 Health applications

Tracking and Monitoring Doctors and Patients inside a Hospital: A small and

light sensor can be attached to each patient in the hospital. The sensor is able to monitor the heart rate, blood pressure, body temperature, and to locate the patient. Sensor data is sent to a central monitoring center and processed so that all patients can get immediate response if any reading is abnormal. And sensor nodes can be attached to doctors to locate them for potential emergencies.

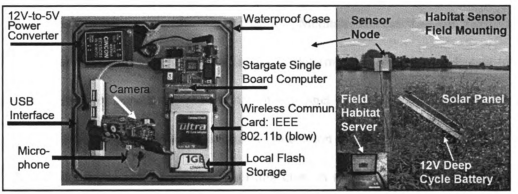


Figure 1.3: Clickable environment sensor platform [14]

1.3.4 Other Applications

Home Automation: Smart sensor nodes can be embedded in all home electronics, such as vacuum cleaners, micro-wave ovens, refrigerators, washing machines, TVs, air conditioners, lamps, and computers etc. All the electronics can interact with each other or with external network via internet or cellular network. End users are also able to control or manage home devices locally and remotely.

Vehicle Tracking and Detection: Sensors with GPS function can be built into cars. Location information of cars are recorded and sent back to base station for example city traffic management center. Individual car is tracked down and real-time traffic information can be derived from fusing all cars' information.

Chapter 2

Self Organizing Medium Access Control Protocols

2.1 Role of Medium Access Control in Wireless Networks

In wireless networks, packet collision is a common challenge [15-19]. Packet collisions usually result from two or more nodes sending data at the same time over the same transmission medium or channel. Medium Access Control (MAC) protocols have been developed to aid nodes to determine as to which node gets to use the channel when competitions for the channel exist. In the literature, this problem is known as multi-access channel allocation problem of the MAC layer, which belongs to a sub-layer of the data link layer in the seven-layer Open System Interconnect (OSI) network model. The addressing and channel access control mechanisms proposed in the literature make it possible for several nodes to communicate within a network.

2.2 Concept of MAC Self Organization

Generally speaking, protocol self organization means that elements in a protocol interact with each other to dynamically achieve a global protocol function or behavior [20, 21]. According to the definition above, the services provided by MAC self organization should be "self" adapted to environmental changes. More specifically, the concept of self organization includes the following four aspects.

(i) The protocol itself is unsupervised and distributed.

(ii) Protocol parameters can be dynamically changed by adaptation of network variables. The purpose of dynamic changes is to constantly improve and

maintain better protocol performance.

(iii) Protocol state machine can be also changed when changing just the protocol parameters may not be enough to cope with the changes in network environment.

(iv) A node can even change its operating protocol when the protocol state machine and parameter changes within a single protocol may not be enough to cope with the changes in network environment.

2.3 Needs for MAC Self Organization

MAC layer has a direct bearing on how data can be exchanged between two nodes along a routing path in the network. Because of the unique features we described in chapter 1, such as constraints on computational ability, storage and energy resources, MAC protocols in wireless Ad Hoc and sensor networks are quite different from traditional networks. MAC self organization addresses those constraints in the following ways:

(1) Ad hoc deployment: In Ad Hoc and sensor networks nodes are deployed in an unplanned manner and with typically no centralized entity. The MAC protocol itself at each node has to find out the connectivity and neighborhood information so that all the nodes can access the shared medium or channel to create a basic communication infrastructure. The MAC self organization addresses this requirement because of its autonomous operation feature. And if any unplanned redeployment happens, it could adjust to the dynamics accordingly.

(2) Energy limitation: As mentioned before, energy is a scarce resource in wireless

sensor networks. In addition to building a basic communication link, MAC protocol in wireless sensor network has to be energy efficient by providing a smart sleep/wake-up mechanism so that it ensures sensor nodes to sleep as long as possible to save energy consumption and to extend network lifetime. The smart sleep/wake-up schedule has to be obtained in a distributed manner, and the changes caused by energy change need to be addressed. MAC self organization is suitable for these operations due to its distributed manner and dynamic adjustment features.

(3) Network dynamics: There are three major aspects of network dynamics. They are topology dynamics, load/traffic dynamics, and communication dynamics.

- Topology dynamics often include node failures, link failures, and node mobility. Sensor nodes are more prone to failure in contrast to traditional networks due to limited resources. Thus new sensor nodes can be deployed for failure recovery or redundancy. Also a certain number of nodes in a network can be mobile for enhancing the network operations, and any node mobility adds to the network dynamics.
- Load dynamics is usually spatial and temporal. It is typically introduced by non-uniform deployment or emergence of certain events.
- Communication dynamics can be generated by outside radio interference, radio medium fluctuation, and radio strength change due to fading.

The advantage of MAC self organization is that it copes with these various dynamics by changing protocol parameters, state machine, or even the protocol itself, as outlined in Section 2.2.

2.4 Research Issues Addressed in this Thesis

A number of protocol self organization issues related to the MAC layer in wireless ad hoc and sensor networks is investigated in this dissertation. A summary of the investigated issues and their relations are outlined in Figure 2.1.

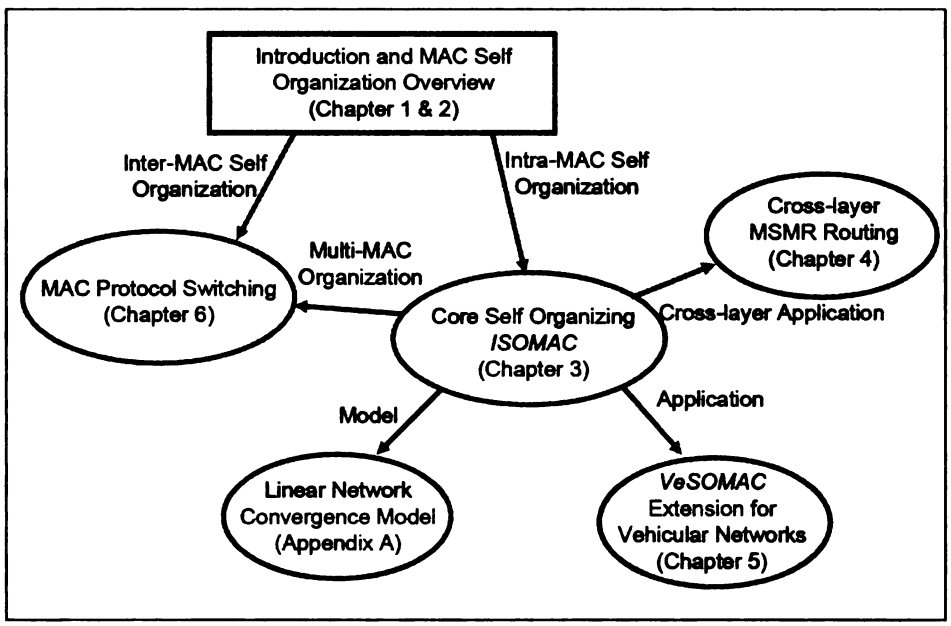


Figure 2.1: Pictorial summary of the investigated issues in this thesis proposal

2.4.1 Design of Self Organizing MAC

2.4.1.1 Motivation and Related Work

The sensor nodes are often critically constrained by their operating energy availability. This makes energy management as one of the key challenges in designing such systems. As a result, wireless embedded networking is often perceived as a design problem to deal with energy-bandwidth and energy-delay tradeoff, as opposed to the classical delay-bandwidth tradeoff in conventional networking systems.

The protocol Sensor-MAC (SMAC) [9] attempted to introduce interface wake-sleep cycles in the presence of random channel access. T-MAC [22] enhances the performance of SMAC by making the wake periods adaptive. Both of them suffer from the inherent energy inefficiency of contention based protocols. In TRAMA [23] nodes maintain a network-wide common frame which is organized into alternating contention-based "random access" and contention-free TDMA allocation "scheduled access" regions. Control packets of TRAMA are still susceptible to collisions. TDMA-W [24] is another distributed out-of-band TDMA protocol. But the adaptation of network dynamics is not handled well in TDMA-W.

2.4.1.2 Proposed Work

We wanted to address the intra-MAC self organization problems with energy limitation, network dynamics, and ad hoc deployment from the list in Section 2.2. Thus an In-band Self-Organized MAC (ISOMAC) protocol is proposed. ISOMAC is a distributed TDMA typed MAC protocol. By avoiding explicit timing information exchange, ISOMAC can work without network-wide time synchronization which can be prohibitive for severely cost-constrained sensor nodes. Efficient energy expenditure is achieved by employing a partial node wake-up and header-only transmission strategy based on the instantaneous nodal data rate. The slot-cluster effect, caused by in-band bitmap constraints, enables ISOMAC to offer better spatial channel reuse compared to traditional distributed TDMA protocols. The detailed ISOMAC protocol will be introduced in chapter 3.

2.4.2 Cross-layer Design

In traditional network design, the seven-layer open systems interconnect (OSI) [25] model divides the overall networking service into seven layers and defines a hierarchy to be provided at each layer. The central idea of cross-layer design is to optimize the control and exchange of information over two or more layers to achieve significant performance improvements by exploiting the interactions between various protocol layers. There are two primary reasons that favor the cross-layer design. To begin with, wireless links create new problems for layered protocol design. One classic example is TCP which considers packet error as an indication of congestion. Additionally, cross-layer design could provide more opportunistic communication than single layer design [26-33]. In [28] the authors divide cross-layer design into four categories.

(1) Creation of new interfaces: The created new interface will be used to share information between layers. This includes sharing lower layer(s) information upward towards higher layer(s), higher layer(s) providing parameters downward towards lower layer(s), and an iterative loop between two layers.

(2) Merging of adjacent layers: This manifestation provides service from a new super layer that is the union of constituent layers.

(3) Design coupling without new interfaces: Two or more layers can be coupled together without creating extra interfaces for the purpose of information sharing.

(4) Vertical calibration across layers: The performance at high layer can be a function of the parameters at all layers below it.

2.4.2.1 Motivation and Related Work

While solving the energy problem at the MAC layer, the TDMA mechanism can give rise to increased end-to-end routing layer delay. This can happen when the TDMA slots of sensor nodes on a route are not time-ordered in the same sequence as the nodes appearing on the route. The problem is that if we have a TDMA type medium access control for delay sensitive and event based sensor network application, what is the best routing scheme.

DMAC [34] tries to allocate MAC communication sequence based on routing path along collecting tree in the sensor networks. However, DMAC only assumes unidirectional data flow towards the root in the presence of preset routes. In [35], the protocol proposes to use a link scheduling algorithm to find the minimum-delay schedule for given slot lengths for all the links. The ignorance of congestion at higher level nodes near sink is the primary disadvantage.

2.4.2.2 Proposed Work

We take a different approach to address the delay problem by computing minimum delay routes based on a given TDMA allocation. We try to find the delay-optimized routing based on pre-set TDMA slot allocation. Our proposed Minimized Slot Misordered Routing (MSMR) addresses the problem of end-to-end delay mitigation in sensor network with TDMA MAC. Delay reduction in MSMR is accomplished by computing least cost routes with a link cost formulation based on the degree of misordering of the TDMA slots of nodes across a link. Chapter 4 describes the details of MSMR.

2.4.3 Application of *ISOMAC* Protocol for Vehicular Networks

2.4.3.1 Motivation and Related Work

MAC self organization is important not only in sensor applications but also in other applications such as the ad hoc vehicular networks. Different features of the application impose different challenges on MAC self organization. Unlike energy being the primary issue in sensor networks, message (especially safety related) latency and delivery ratio become the first priority in vehicular networks.

A number of variations of CSMA/CA and 802.11 have been implemented in [36-38]. The unbounded delay caused by the fundamental random access mechanism is an issue for such protocols. The protocols in [39, 40] propose schedule-based TDMA mechanisms, all with a large upfront reconfiguration delays. The protocol LCA [41] proposes a scheduled TDMA scheduling mapping with vehicles' instantaneous geographical location. But the problem of complete pre-mapping of geographical locations and dimensioning of optimal cell size are non-trivial.

2.4.3.2 Proposed Work

A multi-hop delivery delay minimized MAC self organization protocol for the vehicular networks has been adapted from *ISOMAC* in chapter 5. The adaptation addresses the problems of bounded delivery latency and small reconfiguration delay. The adapted Vehicular Self-Organizing MAC (*VeSOMAC*) is designed to be vehicle location and movement aware so that the MAC TDMA slots in a vehicle platoon can be time ordered based on the vehicles' relative locations. Chapter 5 provides detailed information of *VeSOMAC*.

2.4.4 MAC Protocol Switching

2.4.4.1 Motivation and Related Work

Previously we look into the MAC self organization problem in the context of intra-MAC operations. Different types of MAC protocols have merits in varying network scenarios. For instances contention based protocols are generally suitable for bursty and low congestion scenarios, whereas schedule based protocols overwhelm in multiple concurrent traffic scenarios. Multiple different MAC protocols could coexist within the same network, especially a network with varying traffic dynamics. Instead of MAC self organization problem in intra-MAC protocol, coexistence of multiple MAC protocols raises new challenges for inter-MAC protocols.

Protocols like Funneling-MAC [42] try to deal with non-uniform network dynamics by a single hybrid MAC protocol. Although problems are relieved to some extent, these solutions are only suitable for applications with special traffic characteristics or assumptions with powerful nodes' coordination. The traffic heterogeneity is not fully explored to address the problem of maximizing the network-wide throughput.

2.4.4.2 Proposed Work

We propose to handle the MAC self organization challenges from coexistence of multiple MAC protocol in network by the concept of "MAC switching". This is used to deal with the ad hoc deployment and network dynamics problems. Instead of trying to give a generic solution, we elaborate the MAC switching concept between two candidate MAC protocols, TDMA and CSMA/CA. The modifications of

transmission rules in both protocols and the general switching logic have been developed. Chapter 6 provides details.

2.5 Summary

We first describe the role of Medium Access Control in wireless networks. The concept of the MAC self organization is then elaborated. Inspired from three distinct constraints of the WSN we further discuss the needs of the MAC self organization. The development of the topics addressed in this thesis is finally presented. A brief motivation and related work, followed by a summary of the proposed work, are provided for each of the four major topics in this thesis.

Chapter 3

Towards In-Band Self Organization in Energy-Efficient MAC Protocols for Sensor Networks

3.1 Introduction

3.1.1 Motivation

Because of their low cost, tiny form factor and remote unattended deployments, embedded sensor nodes are often critically constrained by their operating energy availability. This makes energy management one of the key challenges in designing such systems. As a result, embedded wireless networking in WSN is perceived as a design problem to deal with energy-bandwidth and energy-delay tradeoff, as opposed to the classical delay-bandwidth tradeoff in conventional networking systems. While energy syntaxes are being developed by the research community at all the protocol layers, energy-efficient Medium Access Control (MAC) remains a key design challenge. In this chapter, we present an intra-MAC self organization mechanism to mainly address the problems of energy, ad hoc deployment, and network dynamics.

3.1.2 Related Work

A common design approach for the existing energy-aware sensor MAC protocols [9, 22-24, 34, 43, 44] is to reduce *idle energy* consumption [45] by introducing network interface sleep. This is typically achieved by introducing a notion of sleep-awake cycle, in which a node sends and receives data during the wake periods, and conserves energy by switching off its interface during the sleep periods. The

smaller the wake-sleep duty cycle, the higher the savings. The goal is to operate in the smallest possible duty cycle while being able to support the application traffic loading. For sensor MAC protocols, it is often necessary to trade MAC delivery delay, effective throughput and node fairness [46] for energy efficiency.

Sleep-wake cycles are typically not feasible for traditional contention-based protocols such as ALOHA [47], CSMA [48], and 802.11 (DCF mode) [49]. This is primarily because these random access protocols are fundamentally asynchronous in not having timing coordination between the senders and their receivers. As a result, a node is needed to be always awake for possible packet receptions from its neighbors. This results in the energy inefficiency for random access protocols.

In a hybrid design, the protocol Sensor-MAC (SMAC) [9] attempts to introduce wake-sleep cycles in the presence of random channel access. During the wake periods nodes execute an 802.11-like contention based MAC protocol, and during the sleep periods all nodes turn their interfaces off. Sensor nodes form virtual clusters so that all nodes within the same virtual cluster maintain the same wake-sleep schedule. By decreasing the wake-sleep duty cycle, SMAC can trade latency and effective channel capacity for energy efficiency. The main concern with SMAC is that since its basic medium access mechanism is contention-based, the protocol is still susceptible to collisions and its resulting energy inefficiency during waking periods. Also, the protocol assumes static traffic loading with predetermined duty cycles. While the latter issue is addressed in the protocol T-MAC [22] by making the waking periods adaptive, it still suffers from the inherent energy

inefficiency of contention based protocols, packet collisions.

This inefficiency can be avoided in Time Division Multiple Access (TDMA) based protocols. In TDMA protocols such as HiperLan-II [50], since all transmissions within a MAC frame are pre-scheduled, it is possible for a node to sleep when it is not expected to transmit or receive packets. Although it solves the energy problem, an implementation issue with HiperLan-II of its dependency on a centralized base station for TDMA scheduling still exists. For ad hoc deployed sensor networks, such centralized infrastructure is usually not feasible, and thus the design goal should be to develop a TDMA style protocol with distributed MAC slot scheduling algorithms.

The protocol TRAMA [23] implements such a distributed TDMA allocation among time synchronized nodes. In TRAMA, nodes maintain a network wide common frame which is organized into alternating contention-based "random access" and contention-free "scheduled access" regions. Data transmission is performed in "scheduled access" slots and neighbor information exchange is performed in "random access" slots. After the neighbor information (up to two-hops) is collected during the contention-based phase, the nodes execute a distributed election method to decide the transmission schedules within the local neighborhood. TRAMA clearly achieves the goal of distributed TDMA for improved energy efficiency without a centralized coordinating base station. The following operational shortcomings of TRAMA have been identified. First, global TDMA frames require network wide time synchronization, which can be a severely restrictive process [51,

52] for cost-constrained sensor nodes, especially in very large networks with potentially thousands of nodes. Limited energy, bandwidth, hardware and unstable links in multi-hop sensor networks with unpredictable delay characteristics make network wide time synchronization particularly difficult. Second, although the data communication in TRAMA is contention-free, the control information exchange is still susceptible to collisions which can limit the energy-gain of the protocol to some extent. Finally, a new node can join the network only during the random access period. Thus, for keeping the node join-latency small, the random-scheduled duty cycle will have to be appropriately dimensioned based on the rate of change in network topology. In spite of these operational difficulties, TRAMA provides a useful TDMA framework with distributed MAC scheduling.

TDMA-W [24] is another distributed MAC scheduling protocol without using contention based control like TRAMA. Nodes in TDMA-W send control packets during scheduled data slots, and that is how the neighbors' allocation information is disseminated. Based on its neighbors' allocation information, a node is able to select a collision-free transmission slot, which will be used for subsequent data as well as control packet transmissions.

Although TDMA-W has a steady state allocation target very similar to our proposed *ISOMAC*, the former has the following shortcomings. First, TDMA-W, as presented in [24], requires absolute slot identifications to be exchanged through the control packets. Unlike *ISOMAC*, that makes TDMA-W dependent on global framing and time synchronization. Second, TDMA-W incurs a high energy cost as

follows. In each frame, a node must remain awake during its wake-up slot even if it is not transmitting or receiving during that frame. This implies that a node must expend one full data slot worth of energy in each frame irrespective of its communication activities. Third, in TDMA-W there is no syntax for a node to inform its neighbors about its own "wake-up slot". In the absence of this information, although the protocol will work when all network nodes are added/activated simultaneously, it will not work in more realistic incremental node addition scenarios. Whereas, *ISOMAC* addresses the shortcomings of TDMA-W.

3.1.3 Proposed *ISOMAC* Protocol

We present a distributed TDMA scheduling protocol *ISOMAC* (In-band Self-Organized MAC), which achieves similar goals as in TRAMA, but without its restrictions stated above. The key idea in *ISOMAC* is to use an in-band control mechanism for MAC self-organization. Instead of sending explicit control packets as in TRAMA, nodes use a bitmap vector in data packet headers for exchanging slot occupancy information among the neighbors. In the header of each outgoing packet, the transmitting node inserts a bitmap vector which represents the relative TDMA slot timing of all its 1-hop neighbors with respect to the transmitting node's own slot location. As a newly joined node receives data packets from its neighbors, it gradually learns about the TDMA slot locations of all its neighbors and the neighbors' neighbors (i.e. 2-hop neighbors). Based on this allocation information of up to two-hop neighbors, the new node is able to select a collision-free transmission slot. In ISOMAC, this is how an in-band control mechanism is used to avoid the

contention based out-of-band control, as used in TRAMA.

In addition to its energy efficiency, the primary features of *ISOMAC* are as follows.

1. MAC self-organization is achieved using a novel in-band technique. Since there is no contention based operation in the protocol, it is virtually collision free at steady states.
2. *ISOMAC* does not depend on network time synchronization. Although it can be implemented with time synchronization, experimental results indicate that the performance improvement due to synchronization is quite modest.
3. The protocol is fair. A guaranteed data rate is allocated to each node, which is allowed to use a part or the entire allocated rate depending on its individual traffic load
4. *ISOMAC* is insensitive to moderate channel errors. This is notable since in-band protocols are generally known to perform poorly in erroneous channel conditions.
5. *ISOMAC*'s in-band bitmap vector design intrinsically provides higher spatial reuse compared to other distributed TDMA protocols.
6. In *ISOMAC*, information about the location of wake-up slot is implicit and therefore, unlike in TDMA-W, it can handle both simultaneous and incremental node additions.

3.2 Network, Traffic and Operation Model

ISOMAC is designed for arbitrary sensor mesh topology using a single channel

with spatial reuse. Nodes are static and the links are bidirectional. Upon deployment, a node executes the in-band *ISOMAC* protocol for choosing a collision free transmission slot. Nodes automatically adjust their allocation to accommodate topology changes caused by node arrivals and departures.

ISOMAC is routing protocol agnostic and it can support point-to-point, point-to-multipoint [53] and multipoint-to-point sensor [54] traffic. In *ISOMAC*, a node is allowed to transmit data at any rate up to λ_{alloc}, which is the maximum allowable rate per node. A split-slot interface sleep technique is introduced to ensure that a node can adjust its sleep-wake duty cycle to optimize its energy expenditure according to its own and the immediate neighbors' data transmission rates.

The notion of frame in *ISOMAC* is completely local to individual nodes (same size for all nodes). Timing information exchange between nodes is relative, thus the protocol can work without network time synchronization. If the nodes, however, are time synchronized, *ISOMAC* can be implemented with global frame structures. In either case, all sensor nodes in the network are required to run at the same clock frequency, with limited degree of allowable clock drifts.

3.3 ISOMAC Protocol Details

3.3.1 Frame and Slot Structure

Packets are assumed to be of constant length and correspond to the Tx-slot[1] duration τ. As shown in Figure 3.1, each Tx-slot is split into a header sub-slot (T_{header}) and a relatively much larger data sub-slot (T_{data}). A frame is of duration

[1] From this point, unless otherwise specified, the term *slot* will be used to refer to a *transmission slot*.

T_{frame}, and it defines the minimum periodicity of transmission slots from any node. Therefore, the allocated rate to a node can be written as $\lambda_{alloc} = 1/T_{frame}$ packets per unit time. Although one Tx-slot is periodically allocated to each node during a frame, a node can choose to skip transmissions in the absence of data packets in its transmission buffer. Such data skipping are marked as "Header-only Tx" in Figure 3.1.

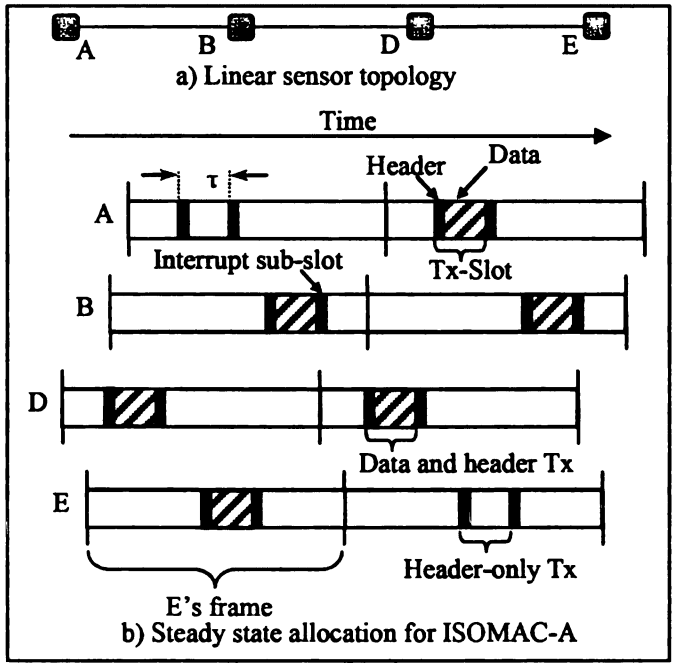


Figure 3.1 Slot allocation in ISOMAC

For each node, immediately after its transmission slot, there is an *interrupt sub-slot*. This receive-mode sub-slot can overlap with any of this node's 1-hop neighbors' transmission slots. As a result, the interrupt sub-slots do not affect the available data rate, as it is intended only for receiving. Also, this sub-slot is typically much smaller than the data sub-slot duration. In our design, $T_{\text{int}errupt}$ has been set to be equal to the header duration T_{header}. The purpose of this sub-slot will be

explained later when the protocol details are elaborated. Note that all the defined sub-slot types include necessary guard times for modem preambles and Tx-Rx switch over latencies, when applicable.

3.3.2 Asynchronous *ISOMAC* (*ISOMAC*-A)

3.3.2.1 Steady State Allocation and Transmission

Slot allocation in *ISOMAC* needs to satisfy the following timing constraint.

Timing Constraint: At steady state, no two one-hop or two-hop neighbors' transmission slots can completely or partially overlap. Overlaps between one-hop neighbors cause direct collisions and between two-hop neighbors cause hidden collisions [55].

A valid allocation schedule for nodes A, B, D and E in the linear network topology of Figure 3.1:a, is shown in Figure 3.1:b. Since the shown allocation is for the asynchronous version of *ISOMAC*, nodes are depicted to maintain their own frame boundaries. Note that except the node pair (A, E) all other nodes are within up to two-hop distance of each other. That is why in the allocation schedule, all nodes have completely non-overlapping Tx-slots except those of nodes A and E. Such overlapping slots provide spatial channel reuse in ISOMAC.

Although in each frame a full Tx-slot is allocated to each node, it may transmit both header and data, or only header. In other words, even if there is no data to be sent, a header is always transmitted in each frame from each node. Packet headers are used by *ISOMAC* protocol for sending allocation information using an in-band bitmap vector as explained below.

3.3.2.2 In-band Header Bitmap

Relative timing information about Tx-slot allocation is exchanged using a Bitmap Vector (BV) in each packet header. Bits in the BV from a node represent how its one-hop neighbors are occupying Tx-slots in the immediate temporal vicinity of the node's own Tx-slot. The concept is explained in Figure 3.2, in which the top segment illustrates how a node P is allocated a Tx-slot within its own TDMA frame. The middle row depicts the Tx-slots occupied by all of P's one-hop neighbors. Note that although these neighbors' slots are shown in the timing context of P's frame, all four neighbors maintain their own frames which are asynchronous to each other.

The bottom row shows the bitmap vector that node P inserts in the header of each of its outgoing data packets. Middle of the bitmap represents the timing of P's own slot. In this example, the bitmap vector is 4-bit long and each bit represents the occupancy status of two slots in the vicinity of P's own Tx-slot. For example, the '1' in "+1" location indicates that two slots immediately following P's slot are already fully or partially occupied. Similarly, a '0' in the "-1" location indicates that node P perceives both the slots before its own slot to be free.

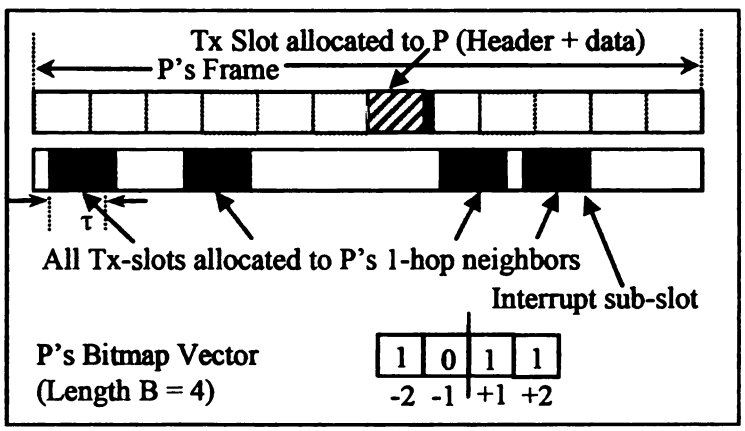


Figure 3.2: Concept of in-band header bitmap

The bitmap vector length is typically much smaller than the total number of Tx-slots in a frame, and therefore it can convey the occupancy information only about limited number Tx-slots around the temporal vicinity of the transmitting node's own slot. In the example in Figure 3.2, the frame size is 12, whereas the bitmap vector length is only 4, which can convey the occupancy information about 8 slots. Observe that with a bitmap size B = 4, P is unable to represent the occupancy information about one of its neighbors' (the one in extreme left) slot.

Using this header bitmap, a transmitting node continually informs all its 1-hop neighbors about Tx-slots that are occupied by the transmitting node's 1-hop neighbors. By listening to the bitmaps in all received packets, a node can figure out the relative slot locations of all its 1-hop and 2-hop neighbors. In turn, with this information, the receiver node can choose a collision-free Tx-slot which is non-overlapping with the slots of its 1-hop and the 2-hop neighbors.

The primary advantage here is that all timing information exchanged is relative to the slot location of the transmitting node. This relative timing allows *ISOMAC* to be implemented with or without network time synchronization. Though in the asynchronous version, the bitmap efficiency is reduced by the overhead that one bit here represents the occupancy information of two slots instead of one. This is because a neighbor's Tx-slot can partially occupy two slots in the absence of TDMA frame synchronization. In the worst case, this effect may actually render half of the system capacity unusable. In the synchronous case, however, this overhead does not exist. This indicates that in the asynchronous *ISOMAC*-A, the capacity efficiency is

traded for more distributed deployments without the need for time synchronization across the network.

Since the bitmap length is kept smaller than the frame slot count, the *ISOMAC* allocation needs to satisfy the following *Bitmap Constraint*: *If two nodes i and j are one-hop neighbors then i's chosen slot should be able to be represented within the bitmap vector of j. The same is applicable for node j's slot. Since each bit corresponds to two slots, this constraint means that the slots of nodes i and j can not be more than B slots apart, where B is the bitmap length.*

For a node, the bitmap constraint is satisfied when it is able to find a '1' corresponding to its own time slot in the bitmaps of all its 1-hop neighbors. In the allocation shown in Figure 3.2, the separation between P's Tx-slot and one of its neighbors' Tx-slots (extreme left) is more than B (which is 4) slots. This indicates that the shown allocation does not satisfy the bitmap constraint, and therefore is not stable. The bitmap constraint is particularly important because a large B incurs higher capacity overhead, and therefore the goal of the protocol would be to use the smallest possible B, which will allow the protocol to converge within acceptable time duration.

3.3.2.3 Transmission Slot Feasibility

For a new node that is entering the network, a feasible transmission slot is one that satisfies the timing and the bitmap constraints as defined in Sections 3.3.2.1 and 3.3.2.2.

Feasible Slots: From the bitmap standpoint, a feasible slot should be within a time

region represented by shared '0's in the bitmaps transmitted by all the neighbors of the new node.

Consider the topology in Figure 3.3, in which a new node R joins in between two unconnected nodes P and Q. Bitmaps (with length 4) from P and Q are shown in Figure 3.3:a as received by the new node R. The shared '0's in the bitmaps of P and Q indicate a feasible time region for node R to choose a Tx-slot from. The feasible region is indicated in the figure.

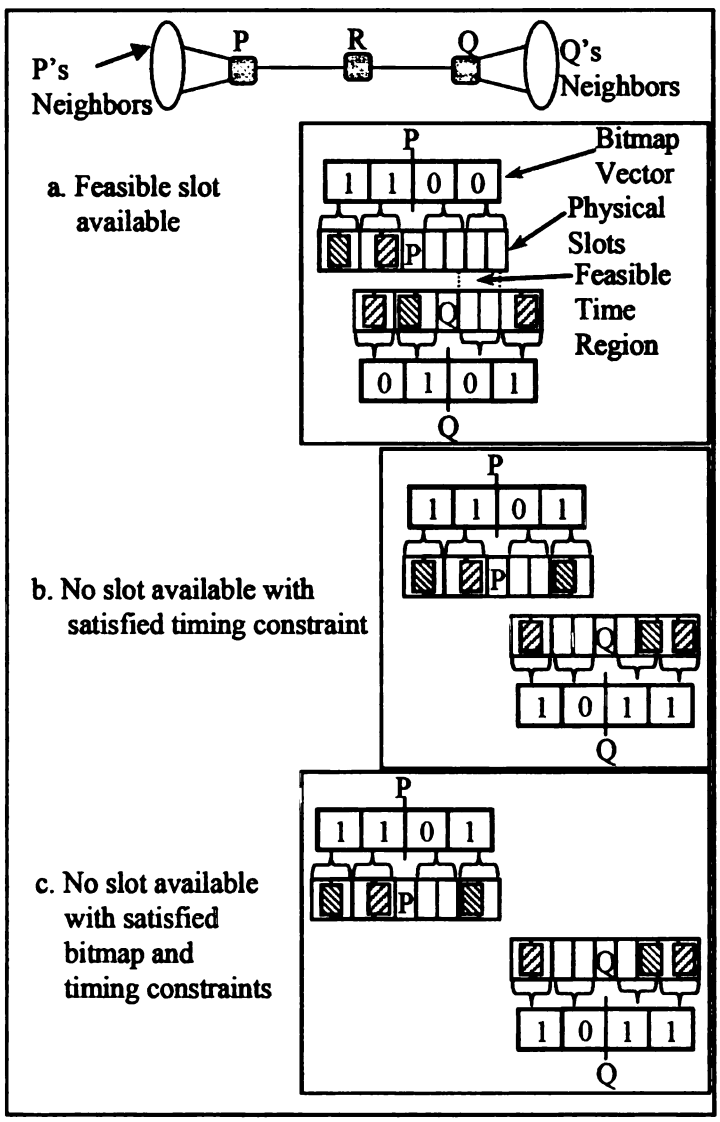


Figure 3.3: Slot selection feasibility

Proof of Feasibility: Since a shared '0' bit indicates that the corresponding time region is not used by any of P's and Q's 1-hop neighbors, it is guaranteed to be hidden collision free from all of R's 2-hop neighbors. Also, since the chosen slot is within the bitmap of all R's 1-hop neighbors (P and Q in this case), it is guaranteed not to be used by any of those 1-hop neighbors. These two clauses together satisfy the *ISOMAC* protocol constraints for the chosen slot.

Since a slot (chosen by R) within that feasible time region is within the bitmap of both of R's neighbors, the temporal separation between R's slot and any of its neighbor's slots is guaranteed to be less than the bitmap vector length, which is 4 here. This satisfies the bitmap constraint. Since both timing and bitmap constraints are satisfied, the chosen slot can be declared as feasible.

For the bitmap vectors in the scenario in Figure 3.3:b, since there are no shared '0's, no collision-free time region will be available for R to choose a slot from. If a slot is chosen by R from the time region indicated by a '0' in P's bitmap then it would collide with a 1-hop neighbor of Q. Therefore a hidden collision cannot be avoided because of the violated timing constraint. Another possible situation is depicted in Figure 3.3:c, in which the bitmap constraint is violated. As a result, although there exist collision-free slots at '0' regions, no feasible slots will be available for the new node R in scenarios pictured in Figure 3.3:c.

3.3.2.4 Sleep-wake Schedule

In a steady state, a node remains awake only during the following time segments:

1. During the transmission slot header duration (T_{header}), and data duration

(T_{data}) only if data needs to be transmitted. A *data-present* bit in the header is set if data will be transmitted following the header in a given Tx-slot.

2. During the header duration (T_{header}) of all its one-hop neighbors' slots. Locations of those Tx-slots are learned simply by listening to the transmissions from all neighbors. While receiving a header from a neighbor, if the *data-present* bit is found to be set, the receiving node will remain awake. Otherwise, it goes to sleep after the header is received.
3. During its *Interrupt sub-slot* immediately after the Tx-slot. During this time, a node waits in receive mode for notification interrupts from any newly joined neighbor nodes.

In transient state, when a node receives a notification interrupt from one of its newly joined one-hop neighbors, it remains awake for W number of frames (described in section 3.3.2.9) in order to learn about the location of the new node's self-chosen transmission slot.

3.3.2.5 Protocol Overview

The key strategy is that if a newly joined node does not immediately find a feasible Tx-slot, it chooses a non-feasible slot defined as follows. The slot is chosen right in the middle of the slots of its 1-hop neighbor pair which has the largest slot distance among all possible neighbor pairs. Upon choosing a slot, the new node starts transmitting periodically once per frame[2]. This action may first cause (the middle point may satisfy few neighbors' constraints, but not all of them) all its 1-hop

---

[2] In ISOMAC-A, upon entering the network, a node arbitrarily picks its own frame starting point.

neighbors to be forced out of their *Stable* (see Section 3.3.2.9) allocation state. But then an iterative slot movement is used for all the nodes in the neighborhood for incrementally attaining steady allocation state for all the neighbors including the perturbing new node. During this iteration, each node attempts to move its slot in the middle of all its neighbors' slot locations.

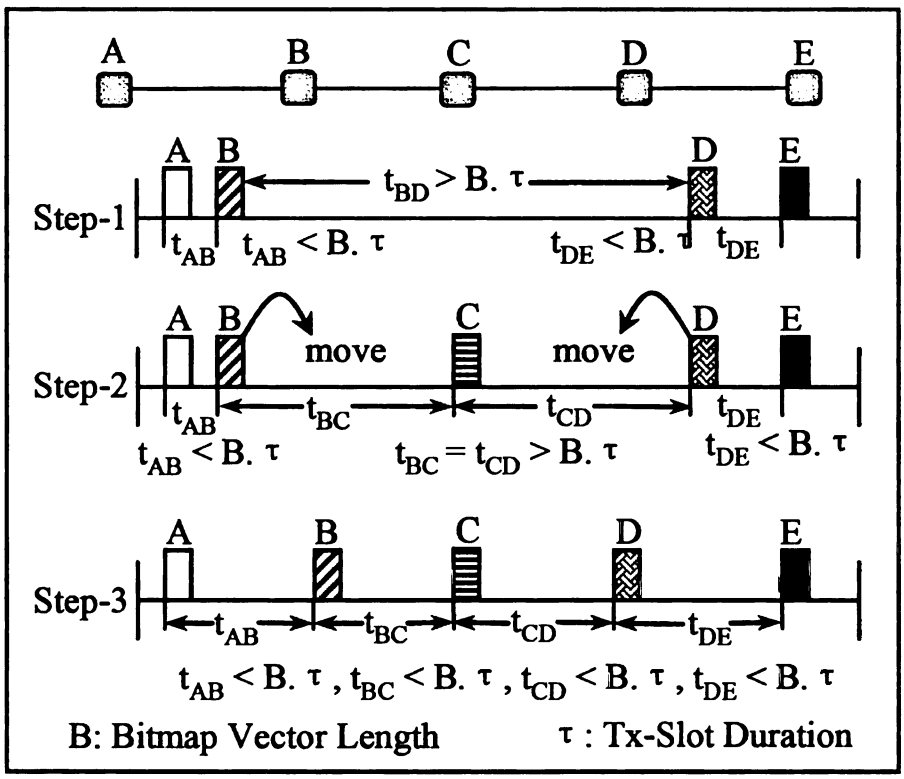


Figure 3.4: Iterative slot movement for convergence

Consider the 5-node topology in Figure 3.4, in which the step-1 of the allocation depicts Tx-slots chosen by nodes A, B, D and E before node C enters the network. With the allocation in step-1, t_{AB}, the minimum temporal distance between A's and B's slots, is smaller than the quantity $B \times \tau$, and therefore it satisfies the bitmap constraint as stated in Section 3.3.2.2. Since their slots do not collide with any of their 1-hop and 2-hop neighbors, the timing constraint is also satisfied. As a result,

nodes A and B are in a stable allocation state. The same applies to nodes D and E. To summarize, at Step-1 both the sub-networks A-B and C-D are at a steady allocation state.

Upon entering the network, node C learns about the slot locations of its 1-hop neighbors B and D from their periodic transmissions, and the 2-hop neighbors (A and E) through B's and D's bitmaps. Since the value of t_{BD} is larger than $2 \times B \times \tau$, C cannot find a slot that will satisfy the bitmap constraint of being able to be representing both B's and D's slots in C's own bitmap vector. As shown in step-2, unable to find a feasible slot, C chooses a collision free but non bitmap-constraint compliant slot that is right in the middle of B's and D's slot locations. Since t_{BC} and t_{CD} in step-2 are both larger than $B \times \tau$, now none of the nodes B, C and D can satisfy the bitmap constraint, thus their allocations become unstable. Nodes A and E however still remain stable.

Responding to its instability, in step-3 B moves its slot right in the middle of those of its 1-hop neighbors A and C. The goal of B is that by reducing the slot separation t_{BC} it will be able to satisfy the bitmap constraint and make its own allocation stable. Following the same strategy, node D moves its slot between those of its own 1-hop neighbors C and E. Node C does not move since it was already in the middle of its 1-hop neighbors B and D. After these moves, the slot separations between any node and all its 1-hop neighbors become smaller than $B \times \tau$, and therefore the allocation in Step-3 is considered to be stable because of its compliance with both the timing and bitmap constraints. In case the bitmap constraint was not satisfied for

all nodes in Step-3, the same process of slot movement would have to continue till the allocation stabilizes.

To summarize, the core idea is to let each node iteratively move its slot near the middle of its neighbors' slots until an allocation pattern which is stable for all nodes in the neighborhood emerges. The idea of *middle* is chosen because in the absence of time synchronization, the collective *middle* provides a common reference for all nodes in the neighborhood.

3.3.2.6 Logic for Slot Selection in *ISOMAC*-A

In a more general situation (compared to the one in Figure 3.4), when a node has a large number of neighbors, and no feasible slot is available, middle point computations and slot selections involve the following steps:

1. Compute the relative slot distances between all different neighbor pairs. For a given neighbor pair, their relative slot distance is computed by choosing the minimum slot separation between them,
2. Identify the middle point (T_{middle}) of slots of the neighbor pair that has the maximum relative distance, and
3. Choose a feasible slot randomly (uniform distribution) from the range T_{middle} $-B\tau$ to $T_{middle}+B\tau$, where B is the bitmap vector length. If no feasible slot is available in this range, in each subsequent frame the range is extended using a binary exponential strategy until a feasible slot is available.

Steps 1 and 2 take care of frame asynchrony, and step 3 minimizes collision probability by introducing stochasticity when two or more unstable nodes end up

computing the same middle point as their new slot choices. If a collision still happens, the mechanism for its detection and resolution is presented in Section 3.3.2.7.

Although in the example in Figure 3.4, a sequential approach is shown for explanation purpose, no such sequence of slot movement is enforced in *ISOMAC*. Nodes move asynchronously and independently and therefore multiple nodes can end up colliding with each other because of their slot movement to overlapping time regions. In the actual implementation, we implement a series of randomization both in terms of the time and target location of the slot movement for minimizing such collisions.

From the sleep-wake standpoint, every time a node moves its slot, it remains completely awake during W number of frames. This is to make sure that the node monitors the activity in the neighborhood for W frames based on which it will decide if its own allocation is still unstable or it has become stable. In our implementation typical value of W was in the range of 3 to 5.

3.3.2.7 Collision Handling

During transience, caused by a topology change, if nodes P and Q in Figure 3.3 choose overlapping transmission slots, a hidden collision will take place at node R. Since R is not able to listen to P's and Q's transmissions due to the collision, it will simply indicate those two overlapping slots to be empty ('0') in its own bitmap. Upon receiving R's bitmap, node P looks for its own location in R's bitmap to see if P's transmission was successfully heard by R or not. A '0' corresponding to P's

Tx-slot will indicate that there was a collision or packet corruption due to channel error. If the situation persists for a preset number of frames then node P will attempt to move its own Tx-slot to resolve the collision. Node Q will also behave similarly.

To summarize, a node is required to be able to see its own slot in the bitmaps of all its 1-hop neighbors. Otherwise, a collision or channel error is said to have occurred. This way, the bitmaps in *ISOMAC* are used as an implicit acknowledgement technique for detecting hidden collisions.

3.3.2.8 Role of Interrupt Sub-slot

When a new node joins a neighborhood and starts transmitting on a newly chosen Tx-slot, there is no way for the existing neighbors to know about this new node and its transmission slot. Without being able to get this information, all one-hop neighbors of the new node would sleep during its transmission slot and miss the data packets. Also, without knowing its existence, the neighbors will not be able to represent this new node's Tx-slot in their bitmap vector.

To address this, immediately following its transmission slot, a node remains awake for a small *interrupt-subslot* (of duration $T_{\text{int}\,errupt}$). Before the new node chooses a slot, it sends interrupt transmissions during the interrupt sub-slots of all its neighbors. Since all nodes remain awake during their individual interrupt sub-slots, they can receive this interrupt transmission and know that there is a new node in the vicinity. After a neighbor is interrupted, it remains awake for W frames to learn about the Tx-slot location of the newly joined neighbor.

3.3.2.9 Protocol State Machine

The state machine of the *ISOMAC* protocol is presented in Figure 3.5. Upon joining the network, a new node arbitrarily chooses its own frame boundaries and enters a *Listen* state that lasts for *W* frames to learn about the Tx-slot locations of all its 1-hop and 2-hop neighbors. Locations of 1-hop neighbors are learnt from their direct transmissions and the locations of 2-hop neighbors are learnt from the bitmap vectors transmitted by the 1-hop neighbors. After this initial listening duration, the new node chooses a feasible slot using the logic described in Section 3.3.2.6. At this point, it also interrupts all its 1-hop neighbors so that the neighbors wake up and remain awake for *W* frames to learn about the slot location of the new node. During the *Listen* state a new node needs to remain awake continuously.

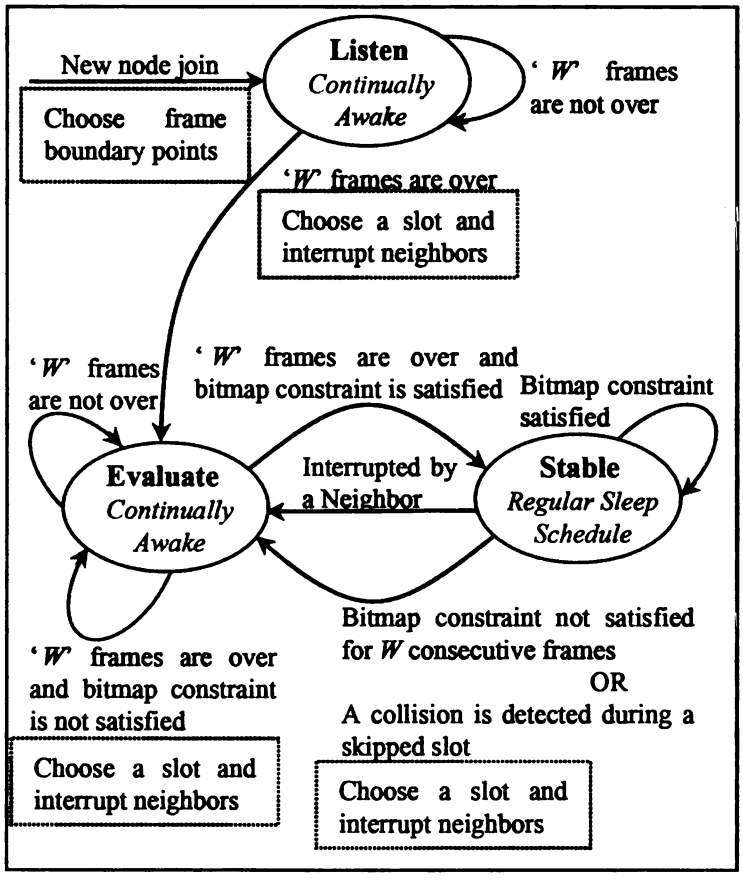


Figure 3.5: State machine for ISOMAC

After choosing its own slot, the new node enters an *Evaluate* state which lasts for W frames. The node expects that during these frames, allocation should stabilize in the neighborhood. After W frames are over, if the node finds that its bitmap constraint is satisfied then it simply gets to the *Stable* state and starts following the regular sleep-wake schedule. Otherwise, it assumes that its chosen slot could not reach a steady state. In that case, the node chooses a new feasible slot, interrupts all its neighbors and re-enters the *Evaluate* state. Since during the *Evaluate* state it is necessary that all 1-hop neighbors should be able to listen to the node, it remains awake during the entire state.

While in the *Stable* state, a node continuously monitors the bitmaps received from all its 1-hop neighbors to detect any allocation instability including hidden collisions. If instability is detected through unsatisfied bitmap constraint for W consecutive frames, the node chooses a new slot, interrupts its neighbors and transits to the *Evaluate* state, waiting to stabilize allocation in the neighborhood. Another reason for a node to transit from *Stable* to *Evaluate* state is when it is interrupted by one of its 1-hop neighbors. Such an event indicates that the interrupting node has chosen a new slot, either because it has joined the network or there is allocation instability caused by channel error or another newly joined node. Upon receiving an interruption, the node does not change its Tx-slot but it simply transits to the *Evaluate* state to monitor the transient activity before it can come back to the *Stable* state.

In the protocol state machine, both *Listen* and *Evaluate* are transient states and

they occur only during topology changes. In majority of the time a node remains in the *Stable* state with a very tight energy schedule as outlined in Section 3.3.2.4. As a result, although the energy expenditure is significantly more in the first two states, overall, *ISOMAC* is very energy efficient.

In *ISOMAC*, a special type of collision can happen in extremely rare situations when all nodes in the network happened to have entered the network exactly at the same time and they have chosen the same slot. In such a case, none of the nodes are able to detect the collision. To exit from this situation, a node is programmed to randomly skip one of its transmission slots once in a large number of frames, so that during the skipped slot it can listen if any of its neighbors is also transmitting during the same Tx-slot. If such a situation is detected, the node simply chooses a new feasible slot, interrupts its neighbors and transitions to the *Evaluate* state. Although extremely rare, this type of collision is slightly more frequent for the synchronous ISOMAC because of its common frame timing across the nodes.

3.3.3 Synchronous *ISOMAC* (*ISOMAC*-S)

Network time synchronization can be leveraged in *ISOMAC* by using the following two mechanisms. First, with a common notion of frame and synchronized slot boundaries across all nodes, the bitmaps can be used more efficiently so that a single bit is able to represent only a single Tx-slot, as opposed to two slots as in *ISOMAC*-A. As a result, a B-bit vector in ISOMAC-S can represent the slot locations of B neighbors as opposed 2B neighbors as in ISOMAC-A.

Second, time synchronization offers a significant simplification in the iterative

slot movement strategy compared to that in *ISOMAC*-A. In *ISOMAC*-A, each node attempts to move its Tx-slot towards the collective temporal midpoint of all its neighbors' slots. The midpoint was chosen because in the absence of common frames it provided a shared temporal target for all nodes in a neighborhood. In *ISOMAC*-S, since all nodes share the same frame, this midpoint approach is not needed. Instead, nodes can move their slots towards the beginning of the common frame, which now acts as the shared temporal target for all nodes in a neighborhood.

If a feasible slot is not found, the following steps are used in *ISOMAC*-S for new slot selection.

1. Identify the neighbor which has the earliest slot in the frame. Say the location of that slot is at time T_{early}.
2. Choose a slot randomly (uniform distribution) from the range T_{early}-$B\tau/2$ to T_{early}+$B\tau/2$, where B is the bitmap vector length. If the feasible slot is not found, like in *ISOMAC*-A, a binary exponential range increment is performed till a feasible slot is found. Note that the range should not be extended beyond the frame start boundary.

Just like in the asynchronous case (Section 3.3.2), these two steps are repeated iteratively for all nodes in the neighborhood till a collective steady state is attained for all the neighbors. Although this slot selection logic for *ISOMAC*-S is different from that of *ISOMAC*-A (see Section 3.3.2.6), its core protocol logic is still the same as that shown in Figure 3.5.

3.4 Model for Frame Size Dimensioning and Energy Consumption

The bounds of frame size F (in number of slots) can be computed using the following model. For fixed packet duration of τ seconds (header and data), the channel capacity can be expressed as $1/\tau$ packets per second (PPS). If M represents the maximum number of combined 1-hop and 2-hop neighbors, then the wireless bandwidth in a neighborhood is shared by (M+1) nodes. Therefore, the maximum data rate that can be allocated to each node is: $\lambda_{\max} = 1/(\tau(M+1))$.... (3.1).

Let the actual allocated data rate be λ_{alloc} PPS per node ($\max[\lambda_{alloc}] = \lambda_{\max}$) and the frame duration be T_{frame} seconds. For an allocation strategy of one slot per node per frame, $T_{frame} = 1/\lambda_{alloc}$ and since $T_{frame} = F \times \tau$, one can write $F = 1/(\tau \times \lambda_{alloc})$.... (3.2).

The lower bound of F is decided by the bitmap constraints. For *ISOMAC*-S, since a node's bitmap is required to represent the slots of all its N neighbors, the length of bitmap vector: $B \geq N$. Also, the number of slots represented by a bitmap cannot be larger than the frame size F. Therefore, $F \geq B \geq N$. To guarantee the timing constraint: $F \geq M+1$.... (3.3).

For ISOMAC-A, since each neighbor's slot can occupy at most two bits in the bitmap, $B \geq 2N$. Also, since each bit may correspond to at most two slot locations, $F \geq 2B$. Combing these two: $F \geq 4N$.... (3.4). Combining those two inequalities, the lower bound of frame size for asynchronous *ISOMAC* is: $F \geq \max(M+1, 4N)$.... (3.5). For the synchronous *ISOMAC*, the lower bound of F can be written as: $F \geq \max(M+1, N)$.... (3.6). The upper bound in Equation 3.2 and the lower bounds in Equations 3.3 and 3.4 define the feasible region for F to be chosen from.

Note that this feasible region is defined by the packet duration, allocated rate per node, and the network density in terms of the maximum number of combined 1-hop and 2-hop neighbors.

Like in any standard TDMA protocol, the average MAC delay is: $F \times \tau/2$ .... (3.7), which increases with F. As for energy consumption in ISOMAC-S, let T_{awake} (in sec.) be the total time that a node needs to be awake during a second for the following operations.

Header and Data Transmission: Uptime for such transmissions is $\lambda \times \tau$ sec, where the actual data rate is λ PPS.

Header and Data Reception: Since a node is required to receive data transmissions for all its N neighbors, the corresponding wake time is $\lambda \times \tau \times N$ sec.

Header-only Transmission: Since λ can be less than λ_{alloc}, the number of header-only transmissions in a second is $(\lambda_{alloc} - \lambda)$. Substituting $\lambda_{alloc} = 1/F\tau$ the uptime for header-only transmission is $(1/F\tau - \lambda)h\tau$ seconds, where h is the ratio T_{header}/τ .

Header-only Reception: To receive a header-only transmission, a node needs to be awake for header duration. Total uptime for such receptions from N neighbors is $(1/F\tau - \lambda)h\tau N$ seconds.

Interrupt Slot: Each interrupt slot lasts for $h \times \tau$ second, during which a node needs to remain awake. Since there is only one interrupt slot (immediately following its transmission slot; see Figure 3.1) per frame for it, and there are $1/F\tau$ number of frames per second, the node needs to remain awake for $(1/F\tau)h\tau$ seconds out of a full

second. This additional wake-time, however, is not necessary when the node's interrupt slot is overlapped with any one of its neighbors' transmission slots. In *ISOMAC*-S, The probability of that happening is N/B, since at steady state, all N neighbors should be represented within B slots around this node's own transmission slot. Therefore, the wake duration due to interrupt slots is $(1-N/B)\times(1/F\tau)\times h\tau$.

Adding all five components, in ISOMAC-S, a node needs to remain awake for:

$$T_{awake} = \lambda\tau(1+N)+\left(\frac{1}{F\tau}-\lambda\right)h\tau(1+N)+\frac{h}{F}\left(1-\frac{N}{B}\right) \quad (3.8)$$

seconds out of a full second. This quantity represents ISOMAC-S's energy expenditure for the duty cycle of a node's operation.

In case of *ISOMAC*-A, all the energy components except the one due to the Interrupt Slots are the same as those in *ISOMAC*-S. Since in *ISOMAC*-A, a B-bit bitmap vector from a node represents the slots of 2B neighbors, the probability of its Interrupt Slot being overlapped with at least one neighbor is $hN/2B$. When an overlap does not occur (probability:$1-hN/2B$), the wake time as in *ISOMAC*-S is h/F. When an overlap does occur, the wake time due to the Interrupt Slot may or may not be there depending on the relative slot timings of this node and its neighbors. So the average wake duration is $h/2F$. Combining these with the wake up components for Header and Data Transmission, Header-only Transmission, and Header-only Reception, the expression for T_{awake} in case of *ISOMAC*-A is:

$$T_{awake} = \lambda\tau(1+N)+\left(\frac{1}{F\tau}-\lambda\right)h\tau(1+N)+\left(1-\frac{hN}{2B}\right)\frac{h}{F}+\frac{hN}{2B}\times\frac{h}{2F} \quad (3.9)$$

Equations 3.6 and 3.7 demonstrate that the energy expenditure increases with

higher data rate, network density, and packet header duration. However, nodes are more energy efficient with larger frame size, which inflates the MAC delay (see Equation 3.5). This clearly indicates the delay-energy trade off in *ISOMAC*.

As for dimensioning the frame size F, the first step is to determine its feasible range using Equations 3.2, 3.3 and 3.4. The next step will be to choose an appropriate value within the feasible range for striking the right balance between the allowable MAC delay and acceptable energy efficiency for specific applications.

Note that the bitmap size is upper bounded by the frame size, which in turn is decided by the network density (a derivative of M). Therefore, successful dimensioning of these parameters depends on *a priori* knowledge about the maximum network degree of a target network. While this is a key assumption for *ISOMAC*, it is important to note that for any distributed TDMA protocol [23, 24] the frame size needs to be dimensioned based on the network density or degree.

3.5 Multi-slot *ISOMAC* for Variable Traffic

The *ISOMAC* protocol logic can be extended from its basic "one slot per frame" mode to more general traffic scenarios in which multiple slots may have to be assigned to individual nodes.

Following the basic *ISOMAC* process as described in Sections 3.3.2 and 3.3.3, each node first assigns itself a periodic *basic slot*. The *basic slot* is used for sending a bitmap with information about all the node's neighbors. Subsequently, depending on its traffic requirements, a node can grab periodic additional slots based on the timing information received via direct receptions and via the bitmaps received from the

basic slots of its neighbors. Transmissions on these additional slots do not include bitmaps because the *basic slots* are already used for this purpose. The interrupt process for the additional slots is the same as that for the basic slots.

When a node requires more bandwidth it may grab additional slots using the process described above. When the slot is no longer required, the node stops transmitting in the slot so that it becomes available for other nodes in the neighborhood to choose, when they require additional network bandwidth. Also, when a neighbor does not hear a transmission on a slot for W frames (see state machine in Figure 3.5) it stops waking up on that slot any more to conserve energy.

If the nodal degree is N, the maximum number of two-hop neighbors per node is N^N. If each node maintains S slots (including the *basic slot*), then there must be $S \times N^N$ number of slots in one frame. This expression indicates the capacity requirement of multi-slot *ISOMAC* protocol.

Multi-slot allocation is feasible for both *ISOMAC*-A and *ISOMAC*-S. Also, there is no restriction as all nodes should have the same number of additional slots. Each node must have a basic slot, and a number of optional additional slots based on its own capacity need and the availability in the local neighborhood.

Although the protocol can handle variable traffic, it is mandatory for a node to own a slot and send a header-only transmission even when there is a no packet to be transmitted. While the energy expense due to this header transmission can be reduced by using a small header size, the bandwidth penalty of owning the slot cannot be avoided. This is the basic cost associated with any in-band approach. The

advantage of this, however, is faster slot reorganization, especially in the presence of energy aware node sleeping which generally make a protocol less reactive.

3.6 Performance

ISOMAC has been simulated using a C-based event-driven sensor network simulator.

Network Size (number of nodes)	100
Network density (average number of neighbors)	5
Frame size (number of slots)	100
Bitmap size	24
Packet error rate	0%
State evaluation time W (frames)	3
Slot duration (millisecond)	5
Header duration (percentage of slot duration)	2.5%
Node deployment mode	Arbitary
Data rate (percentage of λ_{alloc})	10%
ISOMAC protocol version	ISOMAC-A

Table 3.1: Baseline system parameters in simulation

3.6.1 Experimental Parameters

We implement *ISOMAC* on 100 nodes that are randomly distributed within a rectangular sensor field. While keeping the sensor transmission range fixed at 40m, the sensor field dimensions are changed for altering the sensor density. Following the specification in [56], we choose a radio data rate of 25.6 Kbps and fixed packet duration of 5ms, which is also the TDMA slot duration. Unless stated otherwise, for all experiments, the frame duration is chosen to be 100 slots and a packet header is assumed to occupy 2.5% of a Tx-slot. As for data model, each sensor node is allowed to generate traffic at a maximum rate of one packet per frame. All reported

performance numbers are based on local MAC layer traffic. No end-to-end routing is considered. Unless stated otherwise, all results correspond to the asynchronous version of the *ISOMAC*.

3.6.2 Protocol Convergence

A network is considered to be in a stable state when all its nodes are in the *Stable* state as shown in Figure 3.5. The convergence time is defined as the interval from when a new node is added to a stable network to when the network again becomes stable.

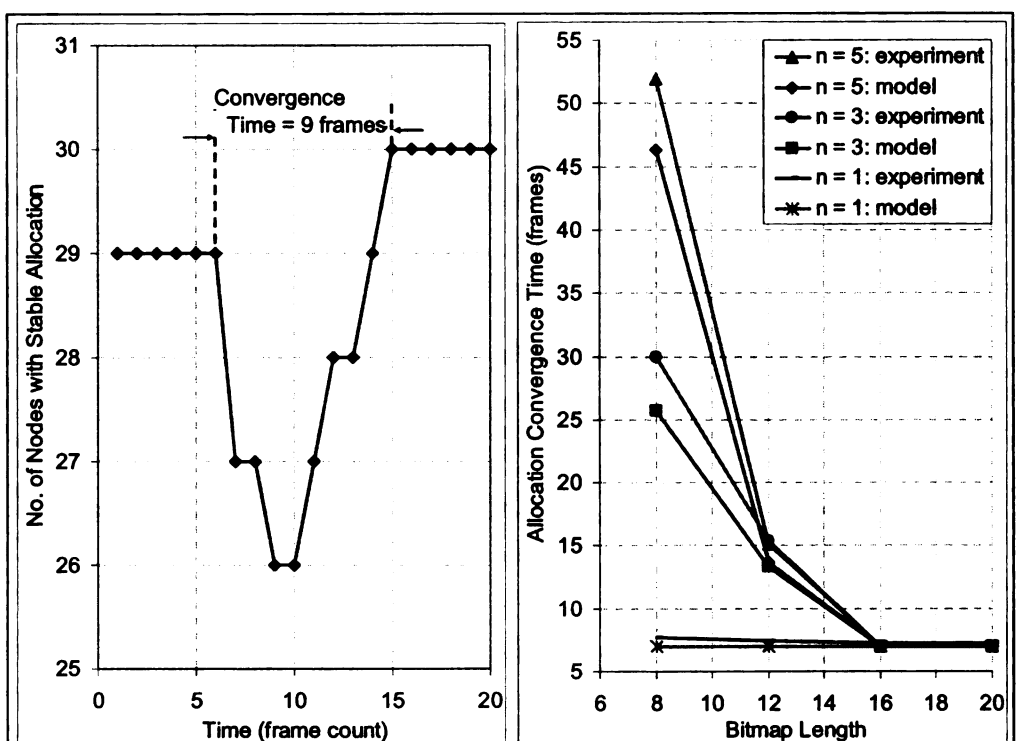


Figure 3.6: Convergence characteristics for ISOMAC-A

The concept is illustrated using the experimental results in Figure 3.6:a. It is shown that a new node is added in a stable network of 29 nodes at the beginning of

the 6th frame. Addition of this 30th node causes temporary instability by forcing three nodes out of their *ISOMAC Stable* states. But eventually, the network converges by the end of the 15th frame. Therefore, the convergence time is registered to be 9 frames.

To understand the convergence characteristics, we first experiment with a linear network as shown in Figure 3.4. It is assumed that before a new node is added, two disconnected sub networks, each with n nodes, are already in allocation steady sates. The timing separation between these two disconnected sub networks is 48 slots, and the inter-node timing separation within each sub network is 3 slots. In the scenario depicted in Figure 3.4, nodes A, B, D and E form two physically disconnected, yet stable allocation clusters (each with $n = 2$), before node C is added in such a way that it physically connects the two disconnected sub networks. As explained in the Figure, node C merges the two existing stable allocation clusters to form a large stable cluster with all *5* nodes. We conduct experiments with various existing cluster sizes (different values of n) and measure the protocol convergence time after adding the cluster-joining node in the middle.

Convergence in such linear networks for different bitmap length is reported in Figure 3.6:b. For each data point, the average convergence time is computed from 500 experiments as outlined above[3]. Experimental results are compared with those obtained from an analytical model presented in Appendix A. As expected, the

---

[3] Unless stated otherwise, all results reported in this section correspond to an average taken from 500 experiments.

convergence latency increases with network size since more *ISOMAC* iterations are needed to merge larger allocation clusters (see Figure 3.4 and its explanations in Section 3.3.2.5). Therefore, convergence in larger networks is slower in general.

Note that for larger networks (n = 3, 5), convergence is faster with larger bitmap vectors. This is because with larger bitmap vectors, the allocation is less restrictive due to the *bitmap constraint*, as explained in Section 3.3.2.2. As a result, fewer slot movements are necessary before network-wide allocation steady states are found. The result is faster convergence.

The final point observed from Figure 3.6:b, is that the experimental results match quite closely with those obtained from the analytical model. The slight differences at smaller bitmaps are due to a few restrictive assumptions in the model and will be explained in Appendix A.

Convergence of *ISOMAC*-A in a 100-node arbitrary mesh topology is depicted in Figure 3.7. In this experiment, 100 sensor nodes are sequentially added at arbitrary locations in a rectangular sensor field. An example convergence trace is shown in Figure 3.7:a, in which the convergence latency after each node addition is shown for the first 50 nodes. Results for the last 50 nodes were not shown for the sake of maintaining clarity in this graph.

The primary observation in Figure 3.7:a is that with increasing network size the convergence gets slower, and with larger bitmap vector length the convergence is faster. These indicate that the results form the model for linear networks, as shown in Figure 3.6:b, extend for more realistic arbitrary mesh networks. Since locations of

added nodes are arbitrary, for some nodes they can simply become a new sub network, or a part of only one existing sub network. After addition of such nodes, convergences are fast and are represented by small latencies mostly fewer than 5 frames. However, the large peaks in the convergence trace correspond to scenarios in which a newly added node is forced to merge two or more disconnected sub networks as described for a linear network in Section 3.3.2.5. Observe that with larger bitmap vectors, there are fewer such allocation latency peaks compared to that with shorter vectors.

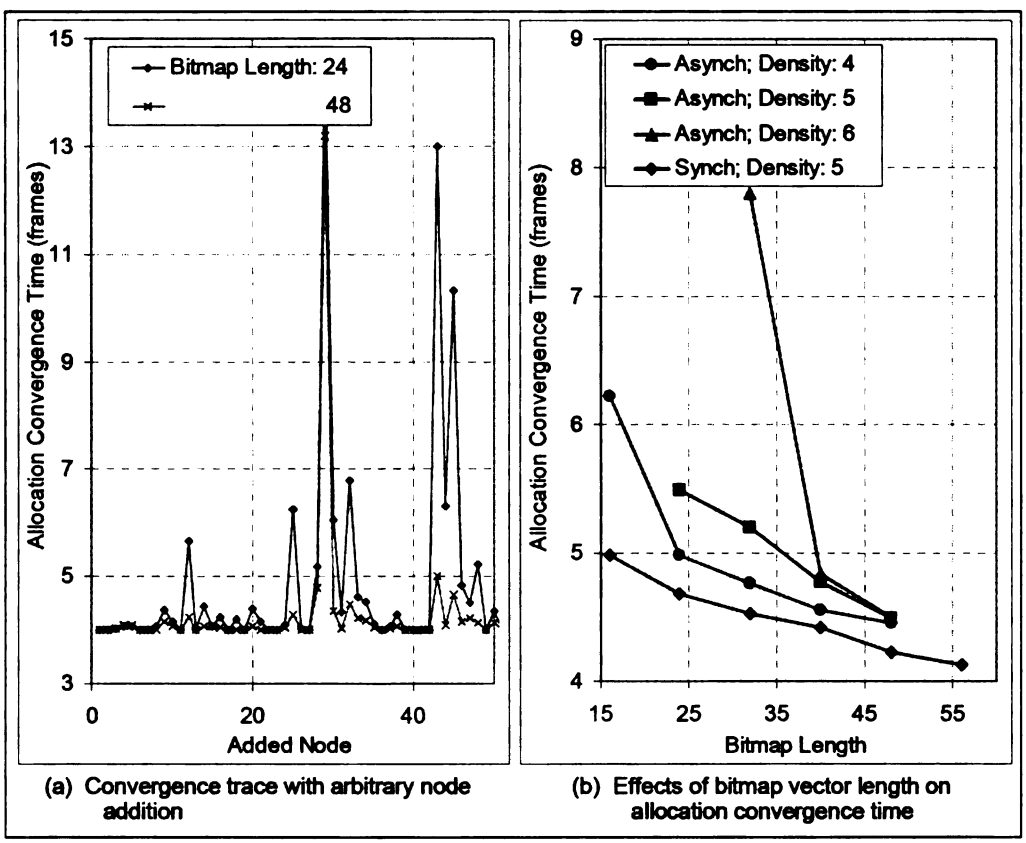


Figure 3.7: Convergence in arbitrary mesh networks

The effect of network density (average number of neighbors N) on convergence latency is shown in Figure 3.7:b. In this experiment, all 100 nodes are first allowed

to join the network and the allocation convergence latency is measured after each node is added. Finally, the convergence average for all 100 nodes was computed. These experiments were carried out for 500 times to compute the final average that is reported in Figure 3.7:b.

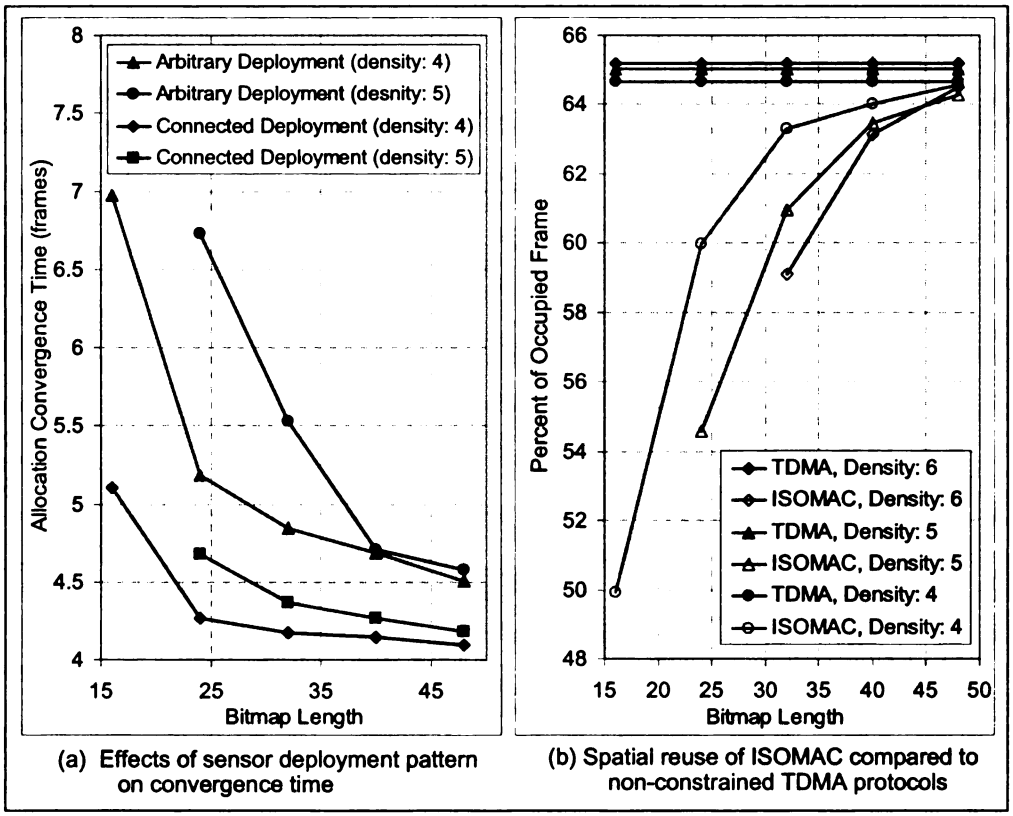
With higher density, since more iterations are needed to get stability in a neighborhood, the convergence latency tends to be larger. In fact, if the bitmap vector is not sufficiently long then a convergence may not be feasible. In these experiments with *ISOMAC*-A, no convergence was found when the bitmap length was 16 for network density 5, and when the bitmap lengths were 16 and 24 for network density 6. As shown in Figure 3.7:b, although it is evident that *ISOMAC*-S performs better than *ISOMAC*-A, the average gain is never more than just one frame duration. This implies that expensive network time synchronization can be avoided within a sensor network running *ISOMAC*.

3.6.3 Sensor Deployment

In a sensor field, nodes can be added either as an *Arbitrary Deployment* (as in Figure 3.7:a), or they can be added in a preplanned way so that no disconnected sub networks are formed during the entire course of deployment. The latter case, which we term as *Connected Deployment*, a newly added node is always placed in such a way so that it can be a part of an existing connected graph. This would correspond to a sensor deployment starting from one side of the sensor field and then gradually approaching to the other side of the field.

Convergence characteristics for both modes of deployments are reported in

Figure 3.8:a. For the *Connected Deployment*, since the newly added nodes are never forced to merge multiple existing allocation clusters, the convergence latency peaks (see Figure 3.7:a) never happen. This explains why the convergences in this deployment mode are always faster than those in *Arbitrary Deployment* mode. This trend holds across all bitmap lengths and network density values that we have experimented with. These results imply that whenever possible, a controlled sensor deployment is desirable from the convergence standpoint.



(a) Effects of sensor deployment pattern on convergence time

(b) Spatial reuse of ISOMAC compared to non-constrained TDMA protocols

Figure 3.8: Effects of sensor deployment pattern and spatial reuse in ISOMAC

3.6.4 Spatial Channel Reuse

Distributed TDMA protocols are expected to achieve spatial channel reuse [55], which helps increasing the supported traffic for a given number of sensor nodes and channel bandwidth. We measure spatial reuse by computing the percentage of a

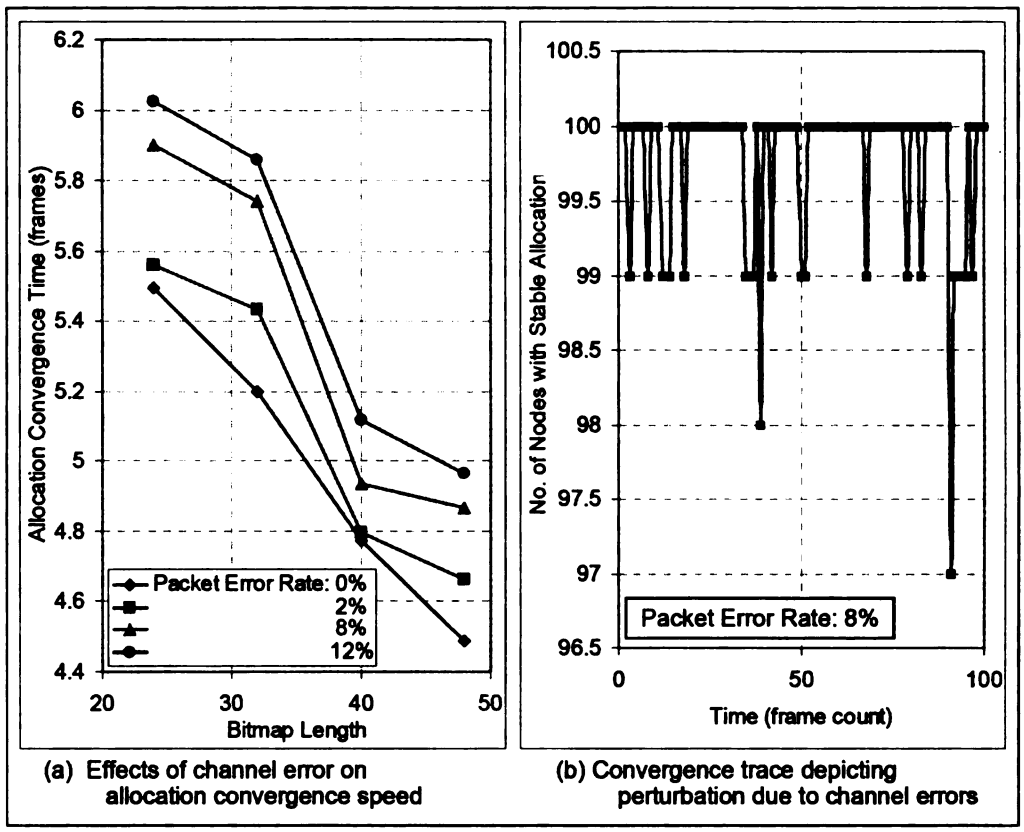
TDMA frame that is used by allocated Tx-slots. With 100 nodes in the field and our chosen frame size of 100 slots, if there was no spatial reuse then 100% of the frame will be occupied by the Tx-slots allocated to 100 nodes. With spatial reuse, however, because of overlapped Tx-slots, less than 100% of the frame will be occupied. Therefore, the lower the frame occupancy, the better the spatial reuse.

A comparison of percentage frame occupancy between *ISOMAC* and other distributed TDMA protocols such as TDMA-W [24] is presented in Figure 3.8:b. The main allocation difference between the two protocols is in the following. While slot allocations in *ISOMAC* and regular TDMA protocols are constrained by the *timing constraint* (see Section 3.3.2.1), *ISOMAC* allocations are additionally restricted by the *bitmap constraint*. As a result, the Tx-slots of neighbor nodes in ISOMAC tend to cluster together. This clustering gives rise to higher slot overlapping, which in turn delivers lower frame occupancy compared to regular TDMA cases.

Since with smaller bitmap vectors the degree of clustering is higher, the frame occupancy numbers are smaller. With our chosen frame duration of 100 slots, when the bitmap vector size is increased to 50, the cluster width for ISOMAC-A (asynchronous case) becomes 100, which is the entire frame. At this point, the effect of bitmap constraint vanishes, and therefore, *ISOMAC's* spatial reuse performance becomes the same as that of regular TDMA protocols.

To summarize, due to its *bitmap constraint*, *ISOMAC* provides inherently better spatial reuse compared to unconstrained distributed TDMA protocols such as TDMA-W. However, this improvement is traded for slower allocation convergence,

especially for smaller bitmap vectors.



(a) Effects of channel error on allocation convergence speed

(b) Convergence trace depicting perturbation due to channel errors

Figure 3.9: Effects of channel error on protocol convergence

3.6.5 Channel Errors

While a node is in the *Stable* state (see Figure 3.5), if it does not receive implicit in-band acknowledgements from all its neighbors for more than W frames, the node transits from *Stable* to *Evaluate* state. In the presence of channel errors, excessive packet header (and bitmap) corruption can cause protocol instability due to such undesirable transitions state transitions from *Stable* to *Evaluate*. This impacts both transient and steady state operations of a network.

The impact of channel errors on convergence for the 100-node network is reported in Figure 3.9:a. Observe that although the convergence is slower at higher channel

errors, the actual increase in latency is just less than one frame for all bitmap lengths and packet error rates up to 12%. This is due to the threshold parameter '*W*' used in the protocol state machine. As a baseline, we use *W* to be 3, which means at least three consecutive packet headers from a neighbor has to be corrupted before a node transits from *Stable* to *Evaluate* state. Larger *W* will make the performance more insensitive to the channel error, but at the expense of slower recovery times.

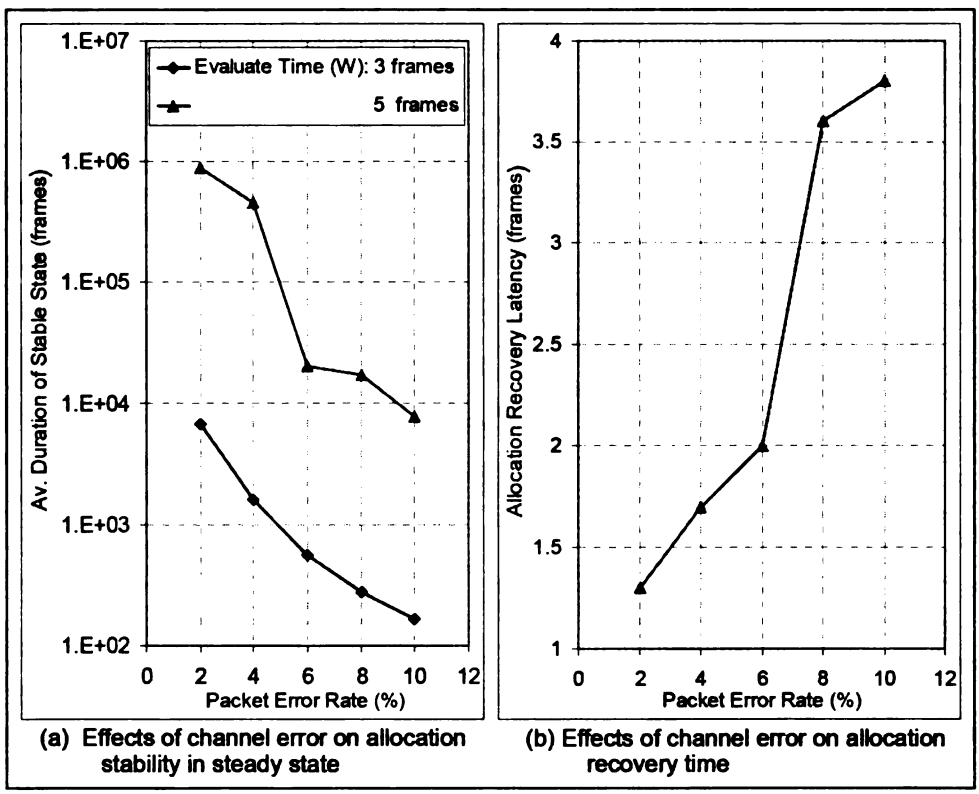
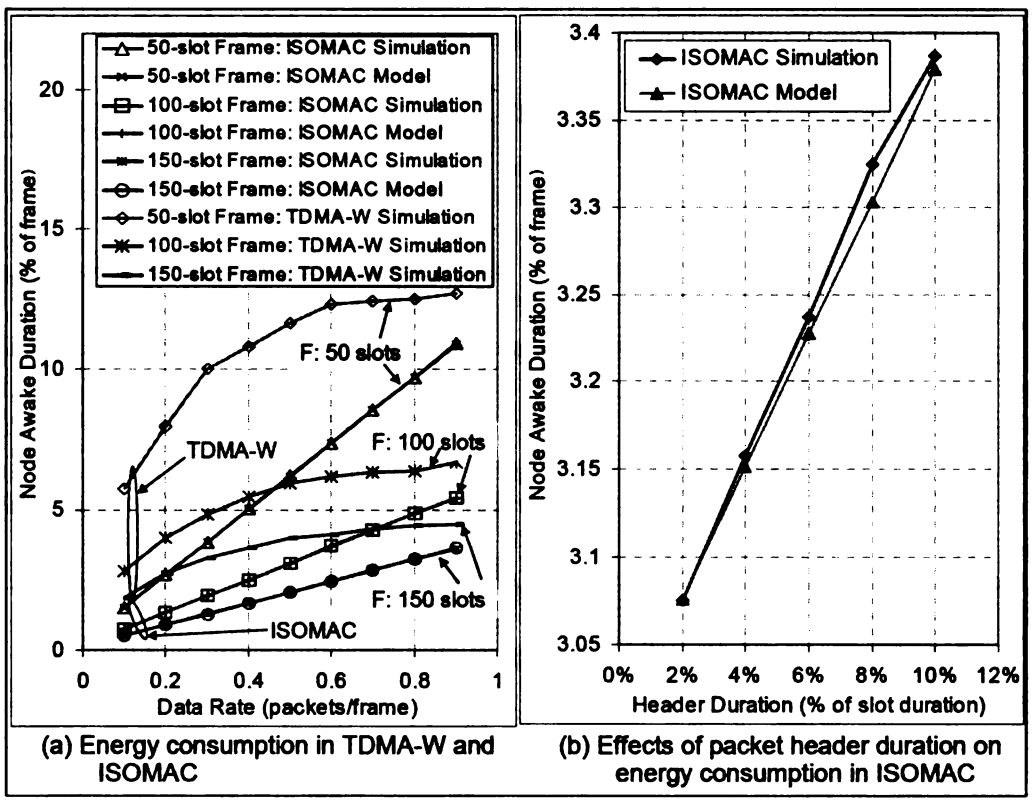


Figure 3.10: Stability and allocation recovery under varying packet error rates

The effects of channel error in network steady state are shown in Figure 3.9:b, which depicts the number of nodes in *Stable* state. These results are shown for a packet error rate of 8%.

The steady state performance from an individual node's perspective is depicted in Figure 3.10. The bitmap vector length for these experiments was chosen to be 24.

The average interval between two consecutive transitions from *Stable* to *Evaluate* is shown in Figure 3.10:a. For example, with the value of W chosen as 3, a node remains at the *Stable* state for an average duration of 4000 frames for a packet error rate of 4%. Also, as explained in the previous paragraph, a larger W helps a node to remain unperturbed for longer duration.



(a) Energy consumption in TDMA-W and ISOMAC

(b) Effects of packet header duration on energy consumption in ISOMAC

Figure 3.11: Energy performance of TDMA-W and ISOMAC

The non-linear effect in Figure 3.10:a can be explained as follows. With E as the packet error rate, the probability of a node to be forced out of the *Stable* state can be written as E^W, which represents the probability of corruption of W consecutive packets from any one of its neighbors. For N neighbors, the probability can be written as $N \times E^W$. This expression explains the non-linear degradation of steady state performance with packet error rate, as shown in Figure 3.10:a.

Once a node is forced out of stability due to channel errors, it waits for a transition back to the *Stable* state till it receives W consecutive successful headers from all its one-hop neighbors. We define this time as the allocation recovery latency, which is reported in Figure 3.10:b. As expected, the recovery latency is larger with higher channel errors. Observe that for packet error rates up to 10%, the recovery time is less than 4 frames' duration, which is somewhat smaller than the lowest convergence latency, as depicted in Figures 3.7 and 3.8. Based on the performance reported in Figures 3.9 and 3.10, it appears that although ISOMAC is an in-band control protocol it can tolerate moderate packet errors up to 10% without being excessively affected in terms of its convergence and recovery latencies.

3.6.6 Energy Consumption

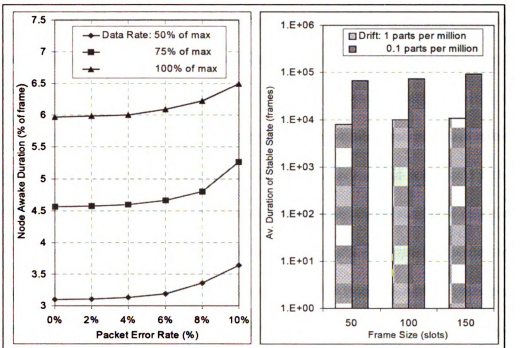
Steady state energy consumption is represented by the fraction of frame duration that a node's wireless interface is required to remain up. Experimental and analytical energy results (see Equation 3.7 in Section 3.4) for the asynchronous *ISOMAC* and TDMA-W [24] are presented in Figure 3.11. As shown in Figure 3.11:a, for *ISOMAC* with higher data rates, since more number of full packets are needed to be transmitted and received as opposed to header-only transactions, the overall energy consumption is larger. It also indicates that for a given data rate, if the frame size increases, a node will have to remain up less frequently, and therefore the energy expenditure reduces. Observe that the experimental and the model data for *ISOMAC* from Equation 3.7 are in complete agreement. As shown in Figure 3.11:a, TDMA-W has a relatively higher energy overhead, compared to *ISOMAC* – especially at lower

rates. In each frame, a node must remain awake during its wake-up slot even if it is not transmitting or receiving during that frame. This implies that a node must expend one full data slot worth of energy in each frame irrespective of its communication activities. This affects TDMA-W's energy efficiency more at lower rates, since more often a node remains awake during its wake-up slot without any Tx/Rx activities. With increasing data rates, the wake-up slots start getting utilized more often and that is why the energy expenditure for TDMA-W becomes saturated at higher rates.

At steady state, even in the absence of data to be transmitted, a node in *ISOMAC* requires to transmit a packet header with its bitmap during its allocated Tx-slot in each frame. While this is essential for all its neighbors to maintain their allocation steady states, these header transmissions consume energy that is not directly related to sensor data. Figure 3.11:b, characterizes the energy cost with varying packet header durations as a percentage of the total packet duration. The graph in Figure 3.11:b indicates that an increase in header size from 2% to 10% of the packet size, the resulting increase in node up time is less than half a percent. This linear change is also predicted in the model presented in Equation 3.7. As in Figure 3.11:a, the experimental and the model data are in excellent agreement.

As demonstrated by the results in Section 3.6.5, excessive channel errors in the in-band bitmap vectors cause temporary transitions from *Stable* to *Evaluate* state. Since a node remains continuously awake in the *Evaluate* state, these state transitions increase energy expenditure during the protocol steady state. This effect is demonstrated in the experimental results shown in Figure 3.12. Since a node has to

miss *W* (chosen to be 3) consecutive headers from any of its one-hop neighbors before it departs from the *Stable* state, at lower error rates such transitions do not happen very frequently. That is why the energy consumption is somewhat insensitive to packet error rates up to 6%. When it increases further, we see the increase in energy expenditure. From these results, it can be concluded that although noisy channels can worsen the energy efficiency of *ISOMAC*, the overall impact for packet error rates of up to 10% is quite within a tolerable range.



(a) Effects of channel error on energy consumption

(b) Effects of inter-node clock drift on allocation stability at steady state

Figure 3.12: Effects of channel error on energy, clock drift on allocation stability

3.6.7 Effects of Clock Drift

The effects of clock drift on *ISOMAC's* steady state performance is reported in Figure 3.12:b. We let each node choose a clock drift based on a given mean value and 10% variance. The drift direction for an individual node is randomly chosen,

either forward with time or backward. The results in Figure 3.12:b are reported as the average duration of stable state from an individual node's perspective. The variance and difference in drift direction cause relative clock drift, which affects the *ISOMAC*-A protocol as follows.

Since *ISOMAC*-A relies on local clock for slot and frame timing, the relative drift can move a node's transmission slot with respect to those of its neighbors. This can perturb allocation stability of *ISOMAC*-A by progressively violating the mapping between the relative slot locations and the information coded in the corresponding bitmap vectors. As a result, nodes can transition from *Stable* to *Evaluate* state, thus affecting the stability of the protocol.

Every time a node enters into the *Stable* state, the average duration it stays stable before a transition to *Evaluate* state due to clock drift is reported in Figure 3.12:b. The longest possible bitmap vector (frame duration in number of slots) was used for a network with density 4. As expected, with smaller drift (0.1 ppm as opposed to 1ppm) ISOMAC-A demonstrates better stability. For example, with 1 parts per million drift and frame size 150, a node remains in the *Stable* state for an average duration of 10765 frames. With 0.1 ppm drift, the number increases to nearly a million frames. It can be also observed that larger frames size helps improving the allocation stability to some extent. This is because with larger frame size, the average time gap between any two neighbors' slots is also larger, and larger gaps make the protocol less vulnerable to the undesired slot movement due to the relative clock drift. To summarize, the *ISOMAC*-A protocol can be significantly influenced

by the clock drift, although for *ISOMAC*-S (the synchronous version) this is less of a problem.

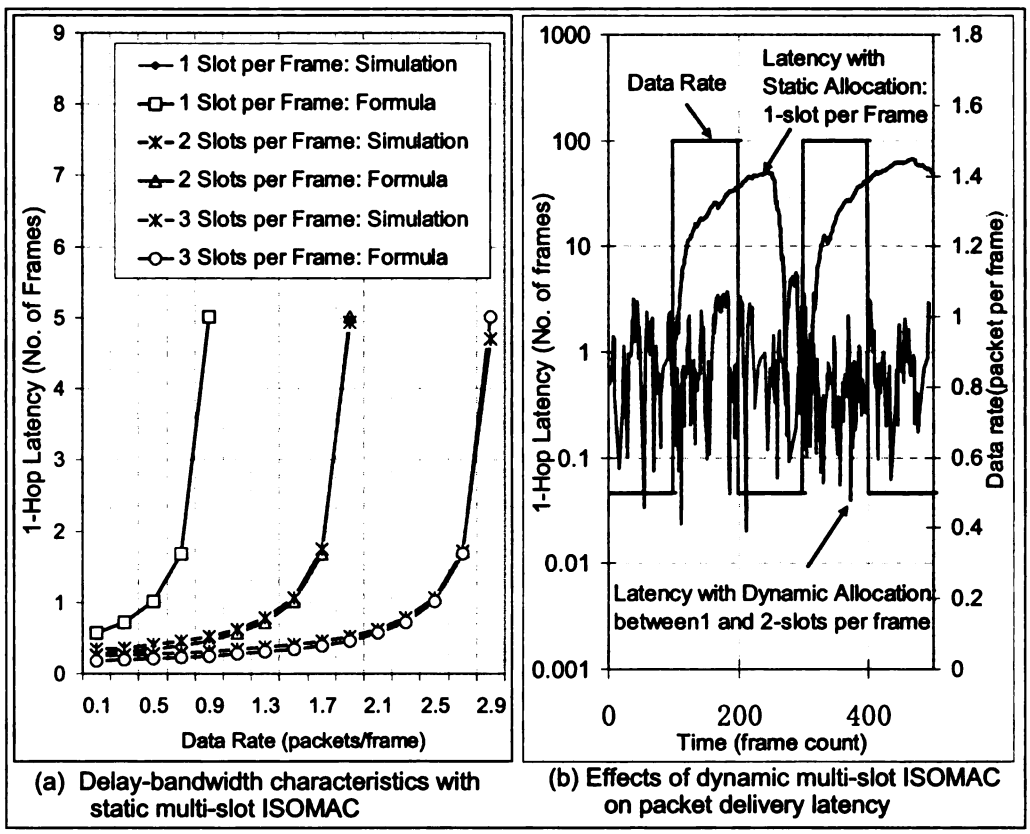


Figure 3.13: Effects of static and dynamic multi-slot allocation

3.6.8 Multi-Slot *ISOMAC*

Latency-bandwidth characteristics for *ISOMAC*-A with static multi-slot allocation (see Section 3.5) is reported in the graphs in Figure 3.13:a. During this experiment, all nodes are statically allocated 1, 2 or 3 slots per frame, including the *basic slot*. As expected, with increasing date rate the latency increases following the standard queuing pattern. Also, for the same data rate, the latency is smaller for scenarios in which higher numbers of slots are allocated per frame. This also indicates that statically allocating larger number of slots can enhance the sustainable

MAC data rate of the *ISOMAC* protocol. In addition to the simulation experiments, we have evaluated the latency numbers using the TDMA delay analysis mechanism presented in [57]. The latency figures are analytically computed to be $D = \frac{1}{2}F\tau + F\tau\frac{\lambda}{2(1-\lambda)} + \tau$, $D = \frac{1}{4}F\tau + \frac{F}{4}\tau\frac{\lambda}{2(1-\lambda/2)} + \tau$, and $D = \frac{1}{6}F\tau + \frac{F}{6}\tau\frac{\lambda}{3(1-\lambda/3)} + \tau$ for allocations of 1, 2, and 3 slots/node/frame respectively. Figure 3.13:a demonstrates that the simulation experiments produce results that are very close to the analytical results.

The working and the effects of dynamic multi-slot *ISOMAC*-A on latency are demonstrated in Figure 3.13:b. In this experiment, variable capacity needs are created by switching the data rate from a node between 0.5 packets/frame and 1.5 packets/frame after every 100 frames. With 1 slot/frame static allocation for the node, the latency changes from a moderate value for the lower data rate, to a dramatically increased value for higher rates. This is simply because the sustainable data rate for the 1 slot/frame allocation is much smaller than the required 1.5 packets/frame rate, and therefore heavy queuing develops. For the same variable data rate, a dynamic multi-slot allocation has also been experimented with. In this case the allocation for that node is dynamically switched between 1 slot/frame and 2 slots/frame based on the nodes instantaneous data rates. As evident from Figure 3.13:b, the latency in this dynamic case is much more contained compared to that in the static allocation case. It is always in the range of 0.1 to approximately 5 frames compared to the case of basic *ISOMAC* with static allocation.

3.7 Summary and Conclusions

We have presented a self-organizing MAC protocol for distributed sensor networks. In the proposed *ISOMAC* protocol, a fixed length bitmap vector is used in each packet header for exchanging relative slot timing information across immediate and up to 2-hop neighbors. It is shown that by avoiding explicit timing information exchange, *ISOMAC* can work without network-wide time synchronization which can be prohibitive for severely cost-constrained sensor nodes in very large networks. Through simulation experiments it was shown that *ISOMAC's* performance without time synchronization is just marginally worse than that with synchronization. It was also shown that a slot-clustering effect due to in-band bitmap constraints causes ISOMAC to offer better spatial channel reuse compared to traditional distributed TDMA protocols. Results demonstrate that with in-band bitmap vectors of moderate length, *ISOMAC* converges reasonably quickly - approximately within 4 to 8 TDMA frame duration. Also, if the bitmap is restricted within 10% of packet duration, the energy penalty of the in-band information is quite negligible. Due to the fact that in *ISOMAC* a node has to own a slot and send a header-only transmission even when there is a no packet to be transmitted, the protocol would perform better in relatively more uniform traffic scenarios with less such zero-traffic situations.

Chapter 4

Cross-layer Routing Protocols in the Presence of MAC Self Organization

4.1 Introduction

4.1.1 Current MAC Self Organization Strategies in WSN

Generally contention-based protocols are susceptible to collisions and its resulting energy inefficiency during the wake periods. Recent researches show that this inefficiency can be avoided in Time Division Multiple Access (TDMA) protocols. Since all transmissions within a TDMA MAC frame are pre-scheduled, it is possible for a node to sleep when it is not expected to transmit or receive packets. Such distributed TDMA approaches have been explored in protocols such as TRAMA [23] and TDMA-W [24]. These and other TDMA based sensor MAC protocols all attempt to achieve collision-free slot allocation such that nodes within up to 2-hops of each other cannot share overlapping transmission slots. Spatial reuse can happen beyond this 2-hop neighborhood. Differences among the existing sensor TDMA protocols [23, 24] exist in the control mechanisms for distributed slot scheduling. But once the slots are allocated, they behave as regular TDMA protocols and follow similar sleep-wake cycles for energy conservation.

4.1.2 End-to-End Delay due to TDMA Slot Misordering

While solving the energy saving problem at the MAC layer, the TDMA mechanism can give rise to increased end-to-end routing layer delay. This can happen when the TDMA slots of sensor nodes on a route are not time-ordered in the

same sequence as the one the nodes appear on the route.

As an example, consider a data flow from node *A* to node *D* on the linear topology in Figure 4.1:a. With the TDMA *Allocation-1* (see Figure 4.1:b), a packet from node *A* can be delivered to node *D* within a single TDMA frame duration. This is possible because with *Allocation-1* the TDMA slots on the route from *A* to *D* are time-ordered in the same sequence as the one the nodes appear on the route itself. As a result, when it is a node's turn to relay a packet, the packet is already available because it was transmitted by this node's upstream neighbor during an earlier slot within the same TDMA frame. For example, since node *B*'s TDMA slot is preceded by node *A*'s slot, node *B* is guaranteed to have received the packet from *A* before its own allocated slot starts. The same applies to the slots for node *C*. As a result, the maximum end-to-end delay in this case is bounded by the TDMA frame duration. Note that the location of *D*'s slot doesn't affect the delay in this case since *D* is the sink.

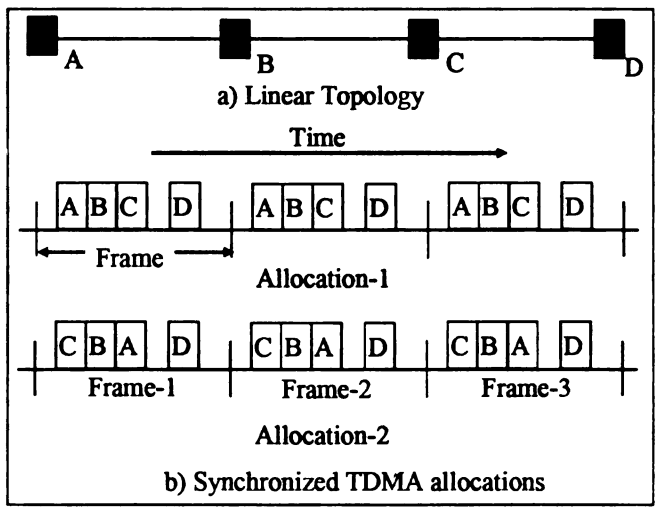


Figure 4.1: Effects of TDMA on end-to-end delay

Now consider *Allocation-2* in which the temporal ordering of the TDMA slots no

longer coincides with the sequence of nodes on the route from *A* to *D*. This slot misordering can introduce significant amount of end-to-end routing delay in the following manner. *A* will forward a packet to *B* during *A*'s slot in Frame-1. By the time *B* receives this packet, it is past *B*'s transmission slot in Frame-1, and therefore *B* has to wait till the next frame for transmission. After *B* sends the packet to *C* in Frame-2, the same misordering issue is faced by *C*. In the end, a packet from *A* to *D* will require three complete TDMA frames, as opposed to just one frame in the case of *Allocatioon-1*. In the worst case, when the N nodes on a route have the complete reverse ordering compared to the slot allocation ordering, the end-to-end delay will increase linearly to N-1 frame duration.

This problem of increased delay due to slot misordering can be exacerbated in sensor networks where large frame durations are usually chosen for maintaining low duty cycles for limiting energy consumptions. In TDMA-W [24], for example, the chosen frame duration is 1 second, which means that for a 50-hop route the worst case end-to-end delay due to slot misordering can be as large as 50 seconds.

In sensor networks with delay sensitive applications, such large end-to-end delays are highly undesirable. Especially for applications such as event monitoring for tactical surveillance, intrusion detection, industrial process monitoring, and security attack detection, such delays need to be mitigated.

The objective of this work is to introduce a Minimized Slot Misordered Routing (MSMR) technique to limit end-to-end application delays caused by the slot misordering problem. In MSMR, such delay reduction is achieved at the expense of

increased energy overhead in routing layer.

4.1.3 Related Work

In the protocol D-MAC [34], the delay issue discussed above is resolved by delay-optimized slot allocation based on pre-set routes. In [34] multipoint-to-point data model is assumed over a routing spanning tree. Once a routing tree is constructed, D-MAC allocates transmission slots to nodes based on their positions on the routing tree. Nodes at the lowest level are allocated the earliest transmission slots in a frame, and those at higher levels are allocated subsequent slots. This slot ordering ensures that as long as traffic flows from the lower to higher levels within the tree, all transaction within the tree can be performed within a single frame.

D-MAC suffers from the following shortcomings. First, the protocol is not able to operate in the absence of preset routes and therefore, unlike TRAMA [23] and TDMA-W [24] it is forced to assume that only one routing tree is allowed in the network. Second, D-MAC is not contention free like regular TDMA. With D-MAC, nodes at the same depth level of a routing tree share common transmission slots and therefore they are susceptible to collisions and the subsequent energy inefficiency. Third, D-MAC does not provide syntax for completely avoiding hidden-collisions. Finally, it assumes unidirectional data flow towards the root. This can be restrictive for many sensor applications where sensor programming commands need to be also routed downstream along the tree.

The other existing approach for mitigating end-to-end delay due to MAC slot

misordering is presented in [35]. This protocol proposes a cross-layer delay optimization scheme that works with variable length TDMA frame. The mechanism uses a link scheduling algorithm to find the minimum-delay schedule for given slot lengths for all the links. While this mechanism provides a cross-layer solution to the end-to-end delay problem, like D-MAC it also assumes unidirectional tree based data routing, which may not be practical for a number of sensor network applications. Also, the congestion at higher level nodes near the sink is not being considered in [35]. Moreover, the variable length TDMA frame may pose a disadvantage in terms of implementation complexity.

4.1.4 Proposed Solution

In this chapter we take a different cross-layer approach to address the delay problem by computing minimum delay routes based on a given TDMA allocation. Unlike D-MAC, which resolves the problem by delay-optimized slot allocation based on pre-set routes, we seeks an opposite point of view by finding the delay-optimized routing based on pre-set TDMA slot allocation. This is accomplished by computing least cost routes with a link cost formulation based on the degree of misordering of the TDMA slots of nodes across a link. The goal is that the routing algorithm will automatically choose a route with least amount of slot misordering on an end-to-end basis. Note that unlike D-MAC [34], which supports only multipoint-to-point data model, our proposed protocol is data model agnostic.

The primary advantage of this Minimized Slot Misordered Routing (MSMR) approach is that unlike in D-MAC, multiple routes can be simultaneously supported

in a network. Also, since no specific data models need to be assumed, both point-to-point and multipoint-to-point applications can be supported over MSMR. Moreover, the applications can be bidirectional.

Note that MSMR is a cross-layer technique but it does not depend on the specific TDMA allocation protocol at the MAC layer. Therefore, it is completely agnostic to how the TDMA scheduling is achieved, as long as the steady state TDMA scheduling is collision-free within 2-hop neighborhood.

4.2 Minimized Slot Misorderd Routing (MSMR)

4.2.1 TDMA Allocation Model

MSMR is designed for arbitrary sensor mesh topologies using a single radio channel with spatial reuse. Sensor nodes are assumed to be stationery and all links be bidirectional. Upon deployment, a node executes a distributed MAC scheduling protocol [23, 24] to be allocated a TDMA slot which is collision free from its all 1-hop and 2-hop neighbors. Frames across the network may or may not be time synchronized based on the availability of any network time synchronization mechanisms such as GPS.

An example of asynchronous slot allocation for the linear network in Figure 4.2:a is shown in Figure 4.2:b. Note that in this allocation, collisions are avoided by making all 1-hop and 2-hop neighbors' sending slots non-overlapping. However, spatial reuse is achieved by allowing 3-hop neighbors such as A and D to share overlapping transmission slots. Since this example allocation is asynchronous, although their frame durations are equal, the frame boundaries of individual nodes

are completely independent of each other. In spite of this frame asynchrony, the allocated slots satisfy the 2-hop non-overlapping rule [23]. After a TDMA slot is allocated, the allocation pattern remains unchanged till the network topology changes.

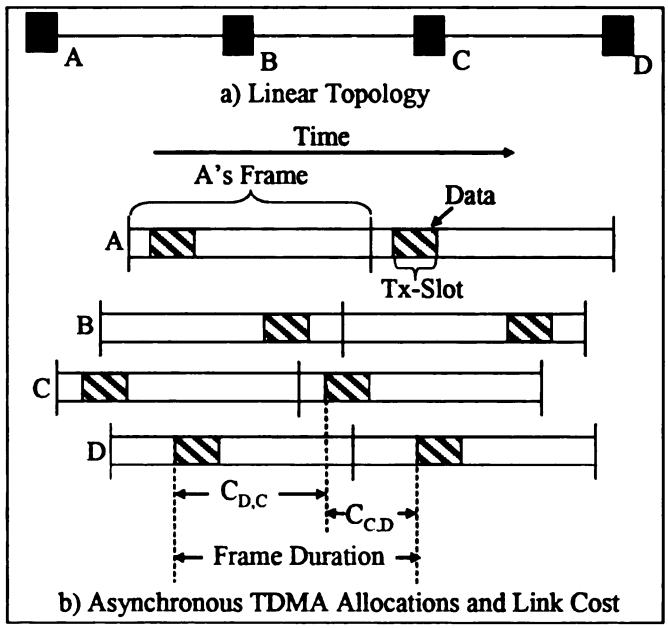


Figure 4.2: Example MAC allocation and link cost formulation

4.2.2 Link Cost Formulation

The objective of the Minimized Slot Misordered Routing (MSMR) is to avoid mismatch between the sequential ordering of the nodes in an end-to-end route and the temporal ordering of those nodes' TDMA allocation. We formulate a new link cost function to capture the degree of such mismatch by defining the cost of the link $L_{i,j}$ based on the relative slot timings for nodes i and node j.

The cost $C_{i,j}$ for link $L_{i,j}$ is defined as the time difference between node j's transmission slot and node i's transmission slot. Example of link costs for links $L_{C,D}$ and $L_{D,C}$ in Figure 4.2:a are shown in Figure 4.2:b. This formulation

always makes the sum of the cost of link j to i and link i to j the fixed frame duration. In other words, the condition $C_{i,j} + C_{j,i} = F$ is always true, where $C_{i,j}$ is the cost from node i to j, $C_{i,j}$ is the cost from node j to i and F is the frame duration, which remains the same for all nodes. Since the MAC slot of the sink node on a route does not contribute to the end-to-end delay, the cost of the link that contains the sink as the receiver is always set to be zero.

The underlying meaning of this cost formulation is that $C_{i,j}$ represents the time delay from the time a packet is received by node j from node i, to the time the packet is sent out by node j. The time when the packet is received by node j is decided by the location of the transmission slot of node i.

4.2.3 Baseline MSMR Route Computation

After the link costs are assigned, a minimum cost route computation algorithm such as Dijkstra's can be used for computing MSMR routes. Distributed minimum-cost routing algorithms such as Bellman-Ford can also be used for the route computation.

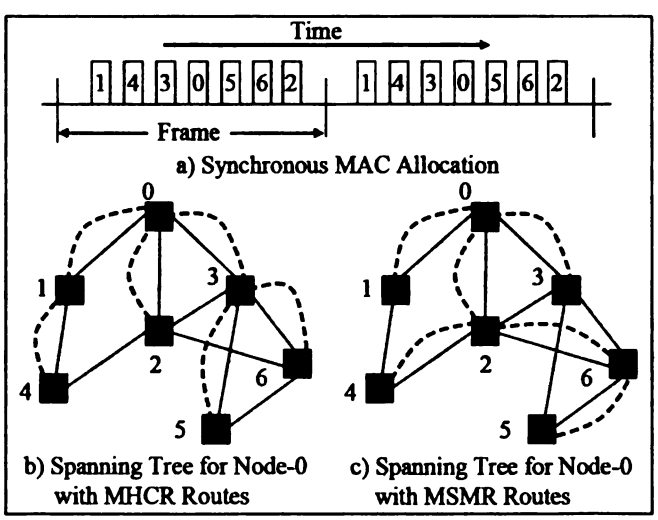


Figure 4.3: MSMR operation with synchronized TDMA

The process is further explained using an example of 6-node network as shown by the solid lines in Figure 4.3:b and 4.3:c. The synchronized MAC allocation pattern used for link cost computation is shown in Figure 4.3:a. Routes created from all nodes to node-0 using a Minimum Hop Count Routing (MHCR) is shown as a spanning tree represented by the dotted lines in Figure 4.3:b. Similarly, the spanning tree constructed using MSMR routing from all other nodes to node-0 is depicted in Figure 4.3:c.

Consider the route from node 5 to node 0, which is two hops in the case of MHCR routing, but three hops for the MSMR routing. This example indicates that sometimes it is desirable to take a longer route to minimize the end-to-end delay due to MAC slot misordering. This, however, also indicates that such delay reduction is achieved in MSMR at the expense of larger hop-counts which in turn would cause larger energy consumptions due to added transmissions. Therefore, it is fair to say that MSMR is expected to mitigate the end-to-end delay by trading overall energy efficiency.

4.2.4 Balanced MSMR to Mitigate Packet Queuing

So far, the end-to-end latency has been discussed only in terms of the delay due to MAC slot misordering. While this is reasonable at lower traffic volumes, delay due to packet queuing can dominate the overall end-to-end delay during congestions at higher traffic situations. This effect is expected to be prominent at nodes near the sink node (e.g. node-0 in Figure 4.3). Since all packets are destined towards a single sink node, these nearby nodes are more susceptible to congestion caused by traffic

aggregation. If the end-to-end cumulative queuing delay becomes dominating over the delay due to MAC slot misordering, then the effectiveness of MSMR on delay reduction is likely to diminish.

This problem can be partially mitigated by in-network data aggregation as proposed [58].While aggregation in sensor networks can be effective for monitoring applications with spatially correlated sensor data, it is less so for event monitoring applications as targeted for this work.

```
/* Balanced MSMR Sensor Routing Mechanism */
Use a TDMA MAC [2,5] for slot scheduling
For all network links L_{i,j} {
    Using the allocated MAC slots, compute link slot
    misordering cost C_{i,j} as in Sec. 2.2;
}
Create an ascending ordered list of all nodes based on
their MHCR hop counts to the sink node;
for (each node from the list, chosen sequentially){
    Compute the MSMR route from this node to the sink
    using the baseline MSMR described in Section 2.3;
    Load traffic on this route at a target event rate;
        Measure the queuing delay at each node on the
route;
    Adjust the cost of all links on the route as:
            C_{i,j} = C_{i,j} + Q_j, where Q_j is the
            measured queuing delay at node j
}
```

Figure 4.4: pseudo-code of Balanced MSMR logic

Therefore, instead of adopting data aggregation, we rely on traffic load balancing for reducing the effects of queuing. By reducing the dominance of the queuing delay on the overall end-to-end latency we expect to preserve the effectiveness of MSMR

for latency reduction.

To mitigate queuing, we adopt a heuristics based centralized *Balanced* MSMR mechanism. Although a distributed version is feasible, considering its additional control and subsequent energy overhead, initially we have chosen to use a centralized version in this paper. Moreover, as considered in protocols LEACH-C [59] and BCDCP [60], centralized routing is deemed appropriate for networks with static sensor nodes, as targeted in this paper. The Balanced MSMR routing logic is outlined in the following pseudo-code in Figure 4.4.

The heuristics behind the *Balanced MSMR* logic is to a) load the shorter routes with targeted traffic, b) compute the link costs as a combination of latency due to slot misordering and queuing, and finally c) compute longer routes based on this adjusted cost metric. Considering the queuing delay as a part of the link cost helps mitigating this by load distribution at the routing layer. Note that the routes constructed by the *Balanced MSMR* mechanism depend on a target event rate from each node. This is a reasonable assumption in application-specific sensor networks with known long-term event generation rates.

In the *Balanced MSMR* heuristics, the longer paths are constructed after the shorter ones for the following reason. According to the presented logic, the later a route is created, the more accurate it is because of the iterative adjustments of the link costs during the earlier route constructions. The objective is to make the longer routes more accurate so that the effects of inaccuracy in queuing latency estimation are minimized across the network. In order to attain that the longer routes are

constructed after the shorter ones.

4.3 Performance Characterization

4.3.1 Network and Data Model

MSMR has been simulated using a C-based event-driven and packet-level sensor network simulator, and its performance has been compared with the Minimum Hop Count Routing (MHCR). The reported results represent MAC and routing layer modeling without a detailed PHY layer model. Since we compare the performance of MSMR and MHCR at the routing layer, any PHY layer variation may change the absolute routing layer performance, but the comparative performance trends of the protocols reported here are expected to be valid.

A sensor network with nodes deployed uniformly within a circular sensor field has been simulated. While keeping the transmission range constant, we vary the sensor field size for experimenting with different network densities. Also, to keep the routing trees symmetric, the sink node (access point) is always placed at the center of the circular sensor filed.

For the MAC layer, the slot duration and the frame duration are chosen to be 5 ms and 500 ms respectively. As for the MAC, we have used the scheduling scheme of the TDMA-W [24] sensor MAC protocol. Note that the proposed MSMR mechanism is agnostic to the control/scheduling part of the underlying MAC protocol. It depends only on the resulting MAC schedule, and therefore can work with other TDMA MAC protocols [23] without any performance differences.

Event-based data is generated at a constant rate (events/sec/node) from all nodes

except the sink node. The data is contributed from independent sensing events which are assumed to be occurring at nodes with a Poisson distribution. The cost modeling is performed as explained in Section 4.2.2, and a centralized Dijkstra's minimum-cost algorithm is used for computing routes in both MSMR and MHCR. In the following results, each data point corresponds to the average of 500 unique simulation runs.

4.3.2 Effects of MSMR on Delay

MSMR has been evaluated in a 5000-node network in which each sensor node generates packets at a very low data rate corresponding to approximately 0.001 events per second. Initially, the data rate is deliberately kept very low in order to evaluate MSMR without packet queuing. Also, such low event rates are quite realistic for intrusion detection applications, where the events are rare but once such an event happens, it is required to be reported from the detecting sensor node to the sink node as quickly as possible.

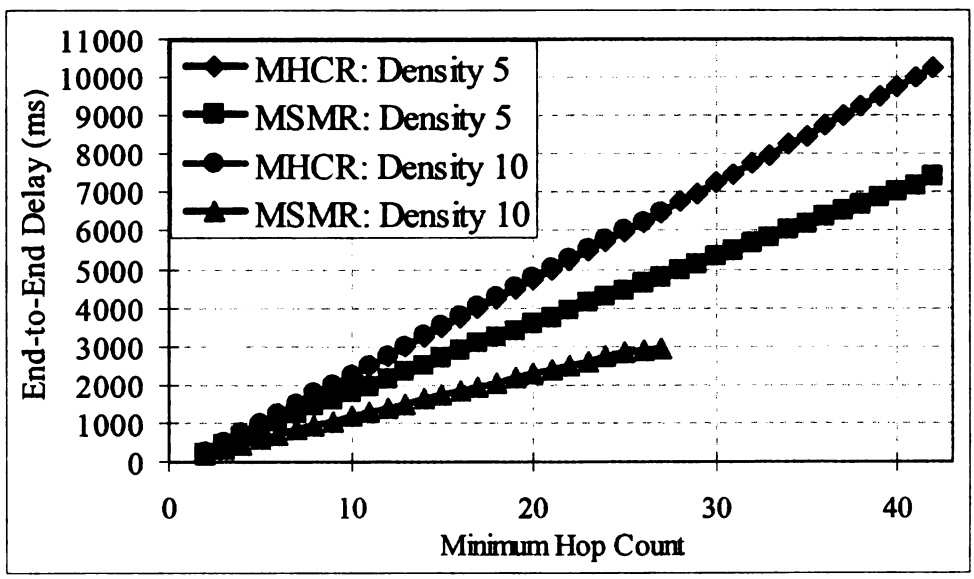


Figure 4.5: Effects of MSMR routing on delay

The experimental results for two different density values are presented as a function of hop-count in Figure 4.5. Density is defined as the average nodal degree. The hop-count in the x-axis represents the minimum possible hop-count from a sensor node to the sink node. From each sensor node, routes are computed to the sink using both MHCR and MSMR, and the corresponding end-to-end delays are measured through simulation. Afterwards, the delay values are plotted as a function of the minimum hop-count for all nodes in the network. The end-to-end delay is recorded as the time difference between the instant an event is detected by a sensor node and the time when it is reported to the sink node.

Note that for the minimum hop MHCR routing the end-to-end delay is insensitive to the network density, and therefore the graphs for MHCR with different density values are overlapping in Figure 4.5. Also, for both MHCR and MSMR, lower density requires longer hop-count routes.

From Figure 4.5, it is evident that for a given network density, the MSMR routing delivers lower end-to-end delay compared to the min-hop MHCR routing at all the hop-count values. Data from nodes which are further from the sink benefits more from the MSMR routing. This monotonic dependency is due to the fact that end-to-end delay reduction by MSMR is a cumulative effect of MAC delay reduction over all the hops on a route.

Consider the results for density 10. For example, a packet traveling from a node with minimum hop-count 27 (from the sink) the end-to-end delay with the MHCR routing is 6.4 sec, whereas the delay with the MSMR routing is 2.9 sec. In other

words, by employing the MSMR technique, the reporting delay of an emergency event can be reduced by up to 3.5 seconds, which is a significant amount of time for possible remedial actions for an intrusion or disaster event. With a lower density of 5, similar trends are observed, although with a relatively lower delay reduction. This is because with lower density each node has fewer neighbors, and therefore MSMR has lesser route diversity leading to a smaller delay gain.

4.3.3 Energy-delay Tradeoff

As shown in Figure 4.3, the hop-counts of MSMR routes are always greater than or equal to those of the corresponding MHCR routes. Larger hop-counts translate to larger transmission energy expenditure for each packet transport to the sink, and therefore it can be said that MSMR obtains better delay performance at the expense of increased energy overhead. The measure for energy overhead is expressed in terms of Percent Hop Overhead $PHO = \frac{H_{MSMR} - H_{MHCR}}{H_{MHCR}} \times 100\%$, which is reported in Figure 4.6 as a function of hop-count. The x-axis represents the same quantity as in Figure 4.5.

The results in Figure 4.6, together with Figure 4.5, demonstrate how the energy-delay tradeoff plays out in the given context. As evidently seen from Figure 4.6, the increase in PHO with hop-counts is not monotonic. The overhead is less for packets from the nodes which has very small or very large minimum hop-counts from the sink. The reason for this overhead pattern is that nodes which are closest to the sink and farthest to the sink have less number of possible routes comparing to the nodes in the intermediate distances.

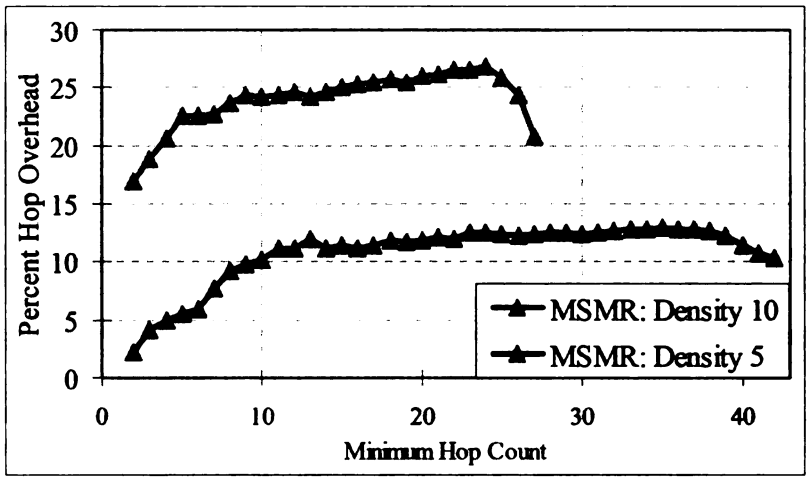


Figure 4.6: Effects of density and hop count on PHO

As for network density, higher densities bring in more delay reduction by MSMR (see Figure 4.5) due to increased routing diversity. As expected, the hop overhead is also larger for higher density. Observe this effect in Figure 4.6. As Experimental results in Figure 4.6 show that the PHO for density 10 is around two times of that for density 5.

4.3.4 Effects of Congestion and Packet Queuing

To study the effects of packet queuing as described in Section 4.2.4, we have experimented with MSMR routing in a 100-node network with varying event rates. Note that with 5ms slot duration and 500ms frame duration [24], the maximum possible event rate is 2 events/second/node. The results of packet queuing are reported in Figure 4.7.

Observe that for lower event rates (e.g. 0.05 events/second/node) MSMR delivers significantly better end-to-end delay compared to the minimum-hop MHCR routing. This result is consistent with what has been observed for a low event rate of 0.01 events/second/node in Figure 4.5. With increasing event rate, the delay performance

of MSMR starts deteriorating due to increased packet congestions. As reported in Figure 4.6, since MSMR uses longer routes than MHCR, it generates more overall traffic, and as a result is more susceptible to queuing than MHCR. In addition, links with smaller MAC slot misordering cost $C_{i,j}$ are found to be chosen in large number of MSMR routes, thus making those links highly congested. These explain why the MSMR delay shoots up at a relatively smaller event rate (i.e. 0.15 events/second/node) compared to the MHCR routing.

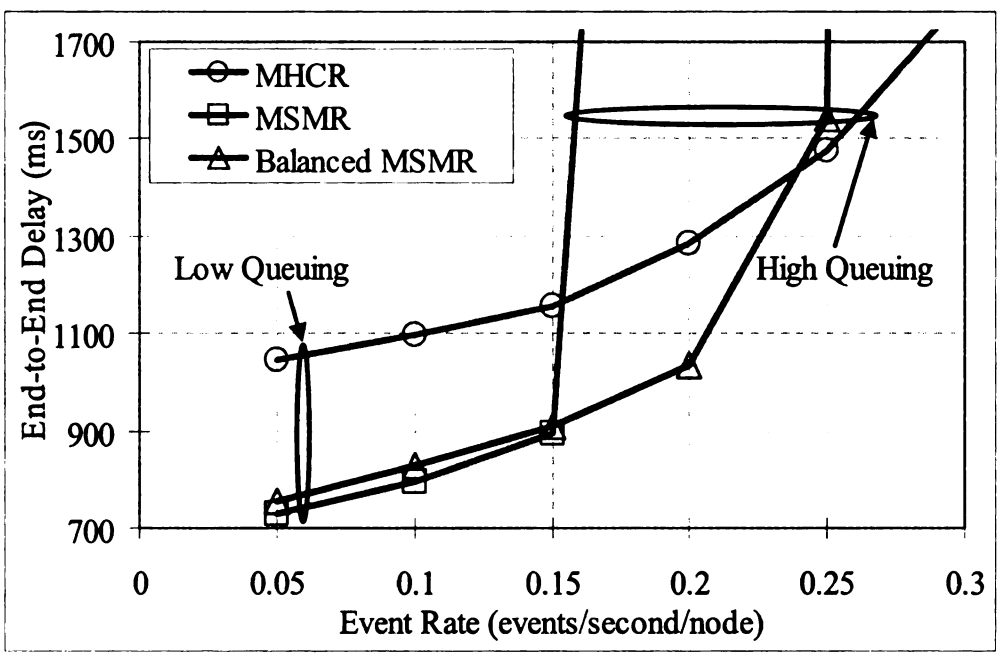


Figure 4.7: Packet queuing at higher sensor event rates

This result clearly demonstrates another tradeoff which states that MSMR routing can reduce the overall end-to-end delay only up to a maximum allowable event rate, which is smaller than the maximum allowable rate for MHCR.

4.3.5 Balanced MSMR for Mitigating Congestion

The effect of this *Balanced* MSMR on queuing is also shown in Figure 4.7. Note that at lower event rates, where the queuing is negligible, *Balanced* MSMR has

delay reduction abilities very similar to the plain MSMR. With increasing event rates, however, the load spreading in *Balanced* MSMR is able to defer the congestion load point compared to the plain MSMR. In this case, it is from 0.15 events/second/node to 0.25 events/second/node, which is almost a 66% improvement. Observe that for even higher loads, the delay for *Balanced* MSMR eventually exceeds that of MHCR, thus indicating that even with load balancing MSMR is more congestion prone than MHCR.

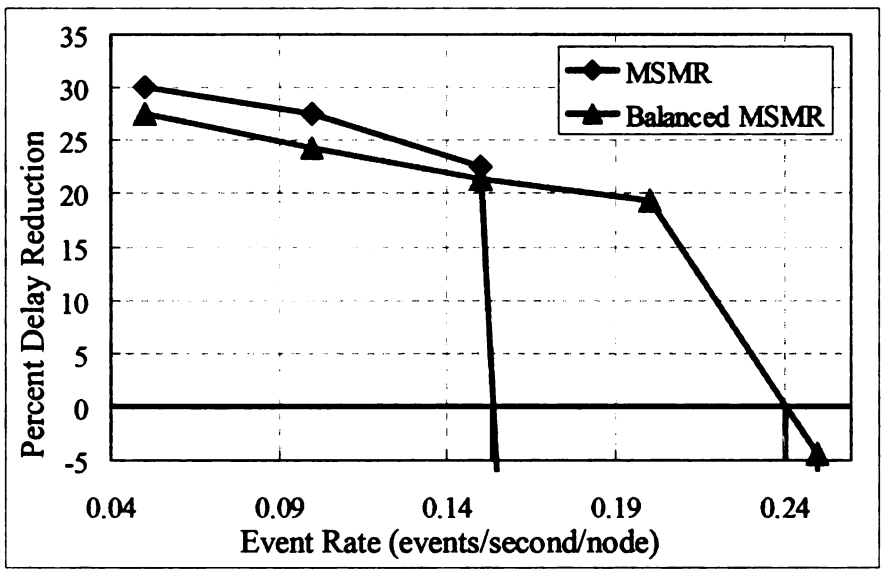


Figure 4.8: Increased allowable event rate with load balancing

The percent delay reductions with respect to MHCR routing for both balanced and unbalanced MSMR are shown in Figure 4.8. At lower event rates, MSMR slightly outperforms its balanced counterpart. Balancing however helps significantly in terms of sustaining the delay reduction property for higher event rates. In this case, while delay reduction for MSMR can be achieved up to 0.15 events/second/node, the same for *Balanced* MSMR is approximately 0.24 events/second/node. This is a gain of 60% in terms of maximum allowable event rate with sustainable delay reductions.

For very high rates, however, both MSMR and *Balanced* MSMR lose this property by actually performing worse than the MHCR routing.

4.4 Summary and Conclusions

A Minimized Slot Misordered Routing (MSMR) protocol has been proposed for end-to-end delay mitigation in sensor networks with TDMA MAC. Different from the existing D-MAC [34], which resolves the problem by delay-optimized slot allocation based on pre-set routes, MSMR seeks an opposite point of view by finding the delay-optimized routing based on pre-set TDMA slot allocation. MSMR has been targeted towards delay sensitive sensor network applications such as tactical surveillance, intrusion detection, and industrial process monitoring. Delay reduction in MSMR is accomplished by computing least cost routes with a link cost formulation based on the degree of misordering of the TDMA slots of nodes across a link. It has been shown that with realistic TDMA models and sensor event generation rates, in a 5000-node sensor network, MSMR can reduce the reporting delay of emergency events by up to 3.5 seconds, which is a significant amount of time for possible remedial actions. We have also shown that while controlling end-to-end delay due to slot misordering, at higher event rates MSMR can give rise to undesirable queuing delay as a result of congestions. We propose a centralized *Balanced* MSMR routing protocol that can strike a balance between queuing delay and slot misordering delay by adjusting link costs based on instantaneous node level congestions.

Future work on this topic includes developing a distributed Balanced MSMR

protocol. The goal is to avoid network wide information flooding and to let the sensor nodes compute the delay optimal routing locally.

Chapter 5

Application of *ISOMAC* for Vehicle-to-Vehicle Communications for Safety and Data Intensive Applications

5.1 Introduction

5.1.1 Application of *ISOMAC*

In Chapter 3, we provide a generic MAC self organization solution *ISOMAC* for wireless sensor networks. MAC self organization can play a crucial role in many other Ad Hoc networks such as battlefield organization, disaster recovery management, and vehicle communication. In this chapter, we adapt the concept of the proposed ISOMAC protocol in the context of vehicular networks.

5.1.2 Background and Motivation

The emerging Intelligent Transportation System (ITS) [61] architecture is currently being developed for enhancing vehicle safety, driving experience, fuel economy, and economic stimulation through safer and efficient transportation networks. A key component of the ITS architecture is vehicle-to-vehicle (V2V) and vehicle-to-roadside (V2R) communication for application data exchange between vehicles and road-side infrastructure.

ITS applications can be categorized broadly into two kinds: safety oriented and data intensive applications. Examples of the safety oriented applications include Cooperative Collision Avoidance (CCA), Cooperative Adaptive Cruise Control (CACC), Emergency Vehicle Preemption (EVP), and Intersection Collision Warning (ICW) systems. Notable examples of data intensive applications are vehicle based

probe data collection for traffic congestion management, drive-through payment such as the EZ Pass system, parking lot payment, enhanced route planning and guidance, and entertainment applications such as inter-vehicle gaming and music downloads [62]. From the networking standpoint, the safety oriented applications generally have much stringer performance requirements including lower and bounded delay [63], and better fairness. Whereas in data intensive applications, a higher achievable end-to-end throughput is favorable.

Dedicated Short Range Communication (DSRC) [62] is an emerging V2V and V2R communication standard, which is being developed for a FCC allocated 75 MHz wide spectrum segment at the 5.9 GHz band. Although IEEE 802.11p is currently recommended as the Medium Access Control (MAC) protocol within DSRC, like other random access protocols it does not work well at high traffic conditions. At high traffic loads, the packet delivery latency increases due to increased collisions, retransmissions, self-competition within a single data flow, and cross-competition among multiple data flows [64]. Also, because of the underlying random access, at higher loads the delivery latency becomes unpredictable, and therefore not bounded [64, 65]. These effects could also bring down the achievable end-to-end throughput for inter-vehicle data transfer through connections spanning across multiple vehicles.

This implies that for a delay-sensitive ITS safety application such as Cooperative Collision Avoidance (CCA) [66], 802.11 may not be able to provide the required small and bounded message delivery latency. This problem has been identified and

various solutions have been proposed in [63, 67]. It has been demonstrated that the stated latency problem is severer in the presence of vehicle crowding and broadcast storm in the events of road emergencies. The required end-to-end throughput is not provided in the data intensive applications by 802.11.

The above problems and requirements challenge the MAC self organization in the vehicular networks. To address those delay issues, which are far different from wireless sensor networks, we adapt the previous ISOMAC of wireless sensor networks and develop an alternative MAC layer architecture that can support ITS safety applications by providing bounded delivery latency over a DSRC network. The adapted TDMA protocol Vehicular Self-Organizing MAC (*VeSOMAC*) is self-configurable and fully distributed just like ISOMAC, and capable of low-latency data delivery with fast allocation convergence achieved by slot scheduling based on the relative location, speed and other context parameters of a vehicle platoon, and the application-specific data flow within the platoon.

5.1.3 Related Work

The desirable features [41] of a MAC protocol for V2V and V2R applications include a) small and bounded delay, b) inter-vehicle fairness, c) scalability with vehicle crowding, d) distributed operation, and e) the ability to cope with frequently changing network topology caused by vehicle movements.

The vehicular MAC protocols in the literature are again in two broad categories: contention-based and schedule-based. The contention based approaches have the advantage of not being sensitive to underlying mobility and topology changes. As a

result, unlike for the schedule-based protocols, vehicle movements do not impose any reconfiguration overhead due to the network topology changes. This is a significant advantage. The unbounded delay issue however, applies to all protocols in this category because of their underlying random access. A number of variations of CSMA/CA and 802.11 have been implemented in [36-38]. Although somewhat mitigated, the fundamental issue of unbounded delay still remains within this category of protocols.

Note, however, these contention based approaches are completely agnostic about the underlying mobility and topology changes. As a result, unlike the schedule-based protocols as explained later, the vehicle movements do not impose any reconfiguration overhead due to the topology changes. This is a major advantage of the contention based approaches in vehicular networks.

The protocols in [39, 40] and [23] propose schedule-based TDMA mechanisms, in which TDMA slots are self-selected by the nodes in a distributed manner. While providing bounded data plane latency, the contention based slot allocation process itself involves collisions, which are stochastically resolved. As a result, the slot reallocation due to topology change may often incur a large upfront delay before the application data can start or resume flowing. The protocol in [67] proposes to configure a token ring protocol so that the maximum delivery delay is always bounded by the round-trip token time. Although being bounded, the delay can be very large for large rings due to high vehicle crowding. This can be addressed by having multiple rings, but then the ring management with frequently changing

topology can cause huge upfront reconfiguration delays. MCS/CDMA [68] is another schedule based protocol that uses CDMA code scheduling. Each vehicle has to have parallel receiver matched filters corresponding to all the codes used. If the search for free code is done sequentially, it could take long time when the number of vehicles involved is large. This will add substantial reconfiguration latency in the event of a topology change. Also, when there are more vehicles than the number of codes, there could be a contention for free codes resulting in large delays in channel access. The protocol LCA [41] proposes a scheduled MAC, in which TDMA slots are allocated based on a vehicle's instantaneous geographical location, which is pre-allocated a TDMA slot. This mechanism offers bounded delay, and also there is no reconfiguration latency due to network topology changes. However, the system requires complete pre-mapping of geographical locations to TDMA slots, which may have practical limitations especially when the huge geographic coverage of a transportation system is considered. Also, dimensioning optimal cell size for varying application requirements and vehicle density is a non-trivial problem.

The following distributed TDMA protocols, which were proposed for wireless sensor networks, can also be considered in the context of vehicular networks. The protocols TDMA-W [24] and DRAND [69] both use an out of band handshake signaling mechanism for distributed TDMA slot allocation. Nodes send control packets during scheduled data slots, and that is how the neighbors' allocation information is disseminated. Based on its neighbors' allocation information, a node is able to select a collision-free transmission slot, which is used for subsequent data

as well as control packet transmissions. Both these protocols have been designed for networks with limited node mobility, since frequent topology changes may require substantial reconfiguration latency caused by the two-way handshake signaling needed after every topology change event.

The protocol ZMAC [70] offers a hybrid solution, in which although each node is allocated a dedicated TDMA slot, the nodes are allowed to contend for bandwidth using CSMA with a goal to improve the bandwidth usage. This protocol too, due to its out-of-band allocation signaling, may suffer from high slot reallocation latency after a network topology change. The protocol D-MAC [34] allocates node and route aware TDMA slots with a goal of minimizing the end-to-end application delay. The primary difficulty of applying D-MAC to the targeted vehicular applications is that the MAC protocol assumes preset routes, which are not feasible in most safety related applications for their latency constraints. Additionally, the protocol does not provide syntax for completely avoiding hidden-collisions, which can prove detrimental to safety critical vehicular applications.

From the existing literature we conclude that schedule based protocols are desirable for their bounded delays, which is a critical requirement for ITS safety applications which require fast reconfiguration and low message delivery latency. However, the primary researchable question still remains: how to cope with frequent topology changes by fast TDMA reconfiguration. This paper attempts to address this key question.

5.1.4 *VeSOMAC*: Vehicular Adaptation of *ISOMAC*

A distinctive feature of the adapted Vehicular Self-Organizing MAC (*VeSOMAC*) is its distributed design with fast schedule reconfiguration for coping with vehicular topology changes. Distributed design allows MAC allocation without having to depend on roadside infrastructure or virtual schedulers such as leader vehicles. This allocation autonomy, coupled with the bitmap based in-band signaling mechanism, allows *VeSOMAC* to perform fast slot reconfiguration after topology changes in dynamic scenarios such as highway platoon mergers, vehicle passing, and other urban traffic situations. Fast slot reconfiguration translates into low upfront latency (during a topology change) which was noted to be a serious issue for other deterministic protocols in [67] and [68].

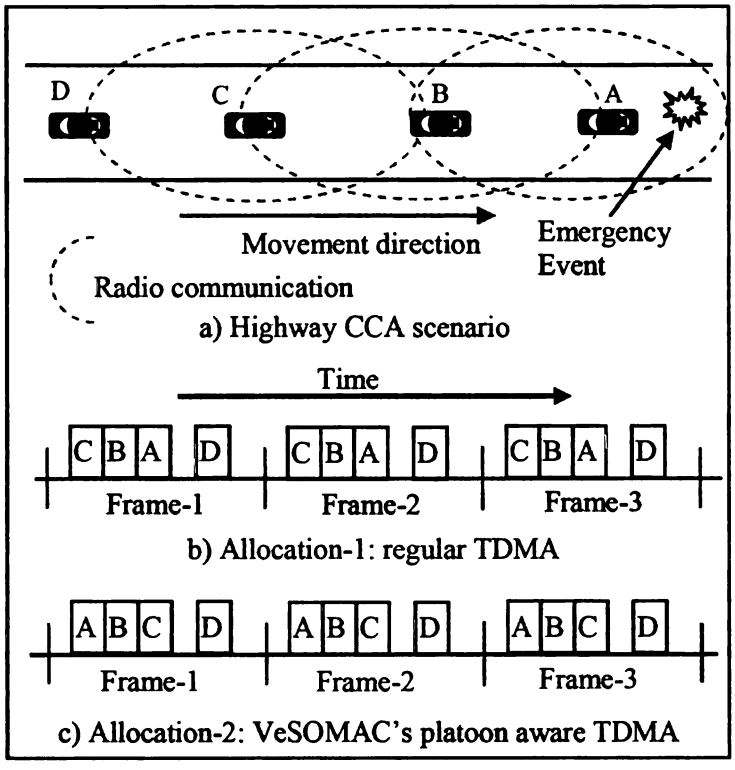


Figure 5.1: Location aware MAC allocation by synchronous *VeSOMAC*

VeSOMAC is adapted to be vehicle location and movement aware for application

specific delay provisioning. Consider the Cooperative Collision Avoidance (CCA) application as shown in Figure 5.1:a. After an emergency event (e.g. an accident) in front of the platoon *A-B-C-D*, the platoon head *A* periodically broadcasts warning messages instructing other vehicles to slow down for avoiding collisions[4] [66, 72]. Such warning messages are to be forwarded across the entire platoon with minimum possible delivery latency, by and to all vehicles. With an example TDMA allocation [39] with arbitrarily slot placement (Figure 5.1:b), it will take three TDMA frames before the message generated by vehicle *A* will be delivered to all vehicles in the platoon. However, with a possible *VeSOMAC* allocation, in which slots are allocated based on the vehicles' relative locations (see Figure 5.1:c), the delivery delay can be significantly reduced. In this example, all messages can be delivered within a single frame. This improvement can be much more pronounced for larger platoons. This way *VeSOMAC* can effectively enhance highway safety by leveraging its ability to allocate slots based on location, speed, and other vehicular contexts.

The adapted *VeSOMAC* inherently has all the advantages of *ISOMAC*. Furthermore, the avoidance of out-of-band explicit signaling makes *VeSOMAC* a faster reallocation, which is particularly suitable for vehicular applications with frequently changing network topologies. Finally, an option of sequential TDMA slot allocation with respect to vehicles' physical positions allows *VeSOMAC* to reduce message delivery delays for a number of highway safety applications such as Cooperative

---

[4] An example: on a 2004 foggy morning, more than 200 cars were involved in a multi-car pile-up on I-96 near Lansing MI [71].

Collision Avoidance.

5.2 *VeSOMAC* Protocol Adaptation Details

5.2.1 Adapted Frame and Slot Structures

The adapted *VeSOMAC* generally has the same frame and slot structure as *ISOMAC* like shown in Figure 5.2. Similarly to *ISOMAC*, in *VeSOMAC* it is mandatory for each vehicle to send a packet every frame, even if no application data is available.

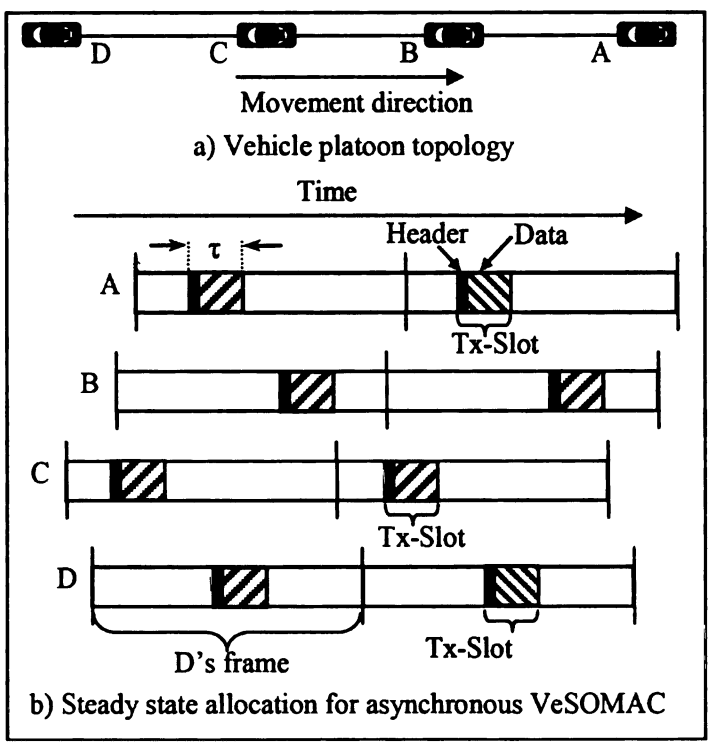


Figure 5.2: Slot structure and steady allocation in adapted asynchronous *VeSOMAC*

The only difference we make to the adapted *VeSOMAC* is that we remove the "interrupt" sub-slot slot from *ISOMAC* framework. The purpose of the interrupt slot in *ISOMAC* is to advertise existence of newly joined node to sleeping neighbors. As we stated before, energy is less a consideration in the vehicular networks. The

adapted *VeSOMAC* requires nodes to maintain awake all the time. Thus there is no chance of missing joining advertisement because of in sleeping mode. The structure of interrupt sub-slot consequently fades out from the adapted *VeSOMAC* logic.

VeSOMAC operates in both synchronous and asynchronous modes just like *ISOMAC*.

5.2.2 Adaptation of *ISOMAC* Protocol Logic for the proposed *VeSOMAC*

5.2.2.1 Timing and Bitmap Constraints Adaptation

The adapted *VeSOMAC* still follows the timing and bitmap constraints as defined in *ISOMAC*. The timing constraint is hold for any distributed TDMA MAC protocol. There is no change in the timing constraint at all in *VeSOMAC*. Spatial reuse in *VeSOMAC* is accomplished by allowing vehicles which are more than two hops away to share TDMA slots as usual.

Convergence results from *ISOMAC* show that larger bitmap vectors can attain faster convergence. The *VeSOMAC* logic potentially could completely remove the bitmap constraint by having the maximum B value (frame size). Even with the largest required bitmap vector, which is the frame duration in number of slots, the number of bits required is usually much smaller (around 5% [73, 74]) compared to the entire data packet, and therefore its capacity overhead can be considered negligible. In addition to the function of the "dummy packet" in *ISOMAC*, *VeSOMAC* obtains more responsive slot reorganization compared to the out-of-band mechanisms that rely on explicit reallocation signaling without contribution to any form of channel congestion.

5.2.2.2 Location Aware Slot Ordering for Delay Reduction

As shown in the *Allocation-2* in Figure 5.1:c, one way to achieve location aware delay reduction is to temporally order the slots in the same sequence as the vehicles appear in a platoon. A packet from the platoon-front *A* can now be delivered to the platoon-tail *D* within a single frame duration. This is because when it is a vehicle's turn to relay a packet, the packet is already available because it was transmitted by this vehicle's upstream neighbor during an earlier slot within the same frame. For example, since *B*'s slot is preceded by *A*'s slot, *B* is guaranteed to have received the packet from *A* before its own slot starts. The same applies to the slot for *C*. As a result, the maximum end-to-end delay in this case is bounded by the frame duration.

The absence of location-aware slot ordering introduces end-to-end delay in the following manner. According to the allocation in Figure 5.1:b, vehicle *A* will forward a packet to *B* during *A*'s slot in Frame-1. By the time *B* receives this packet, it is past *B*'s transmission slot in Frame-1, and therefore *B* has to wait till the next frame for transmission. After *B* sends the packet to *C* in Frame-2, the same misordering issue is faced by *C*. In the end, a packet from *A* to *D* will require three complete TDMA frames, as opposed to just one frame in the case of *Allocatioon-1*. In the worst case, when all *N* vehicles in a platoon have the complete reverse ordering, the end-to-end delay will be *N-1* frame durations. Therefore, in order to minimize end-to-end routing delay, *VeSOMAC* allocation needs the following constraint to be satisfied.

Ordering Constraint*: If two vehicles i and j are geographical neighbors and i's location is ahead of j in the platoon, then i's chosen slot should be earlier than j's*

slot in the time domain.

The *ordering constraint* is optional, and it is useful when the wireless messages flowing from the front to the tail of a platoon are more delay critical than the messages flowing in the reverse direction. While this is generally true for most ITS safety applications, a reverse requirement can be accommodated by adjusting the definition of the constraint itself. A disadvantage of the *ordering constraint* is that it can delay the self-configuration process of *VeSOMAC* by slowing down its convergence. More about convergence will be discussed later in Section 5.2.2.5.

5.2.2.3 Transmission Slot Feasibility Adaptation

Besides *timing*, *bitmap* constraints in ISOMAC, the feasible transmission slot of *VeSOMAC* has to satisfy the *ordering* constraints as defined in Section 5.2.2.2.

The definition of feasible slots now becomes: within the admissible region, a *feasible time region* for a vehicle is defined by the duration which is sooner (on the left on time axis) than the slots of all its rear neighbors, and later (on the right on the time axis) than the slots of all its front neighbors. Any slot chosen within the feasible region will satisfy all the protocol constraints.

Consider the *VeSOMAC* example in a highway platoon scenario in Figure 5.4. A new vehicle R joins in between two unconnected vehicles P and Q. Bitmaps (with length 4) from P and Q, as received by the new vehicle R, are shown in Figure 5.4:a. The shared '0's in the bitmaps of P and Q indicate an admissible time region for vehicle R. The feasible region is indicated in Figure 5.4:a.

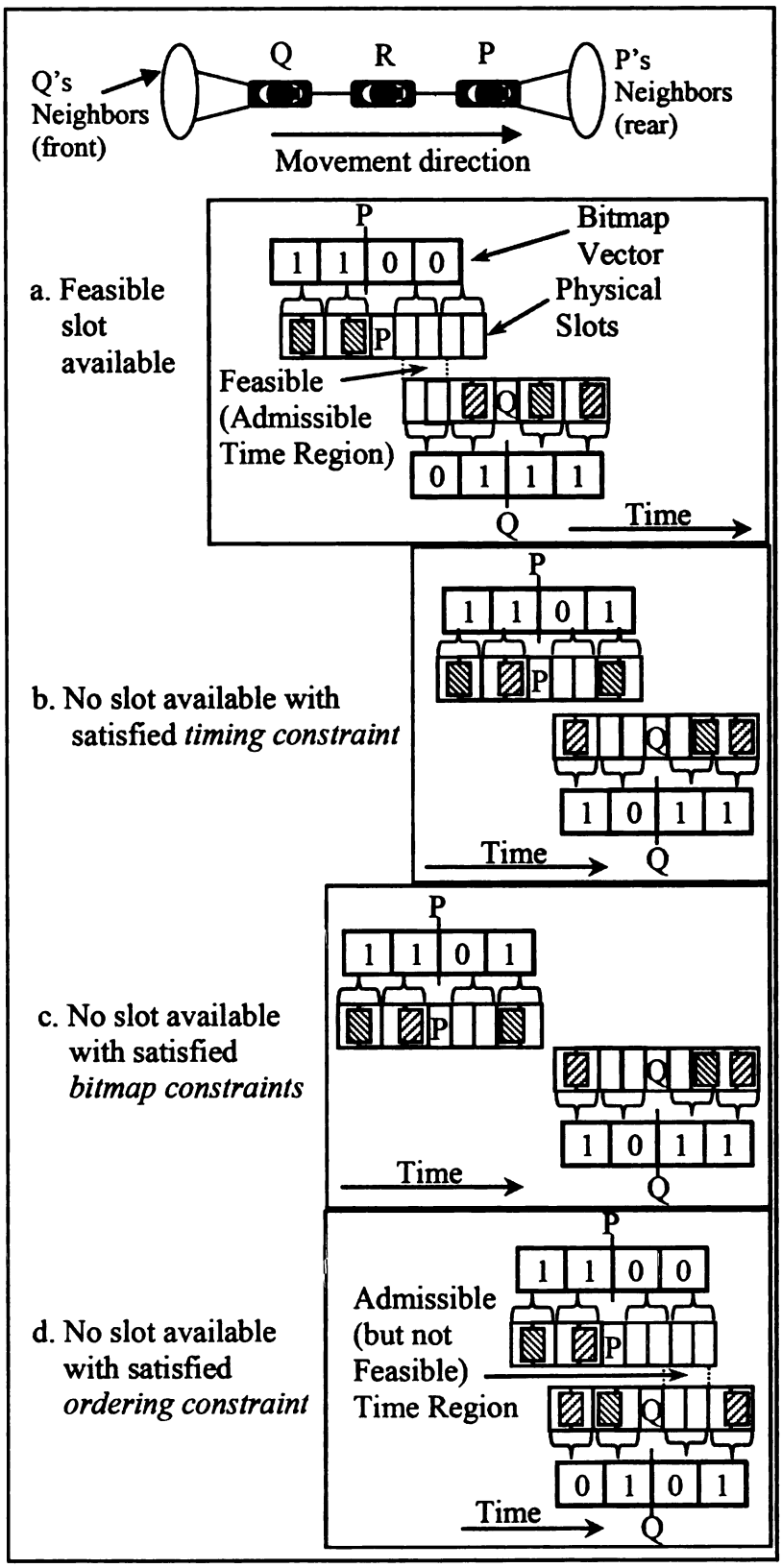


Figure 5.3: Adaptation of *ISOMAC* feasible slot scenarios in example *VeSOMAC*

Proof of Feasibility: Like in *ISOMAC* timing and bitmap constraint is guaranteed. Also, since the admissible region is both sooner than *R's* rear neighbors' (*Q* and its neighbors) slots and later than *R's* front neighbors' (*P* and its neighbors) slots, the *ordering constraint* is also satisfied. Therefore, in Fig. 4:a, the entire admissible region is also a feasible region.

Scenarios in Figure 5.3:b and c are same as *ISOMAC*. In Figure 5.3:d although an admissible region is found, a feasible region is not there because of a violation of the ordering constraint. This is because although the admissible region for *R* is later than vehicle *P*'s slot, a slot cannot be chosen from the admissible region so that it lies sooner than *Q*'s slot.

5.2.2.4 Adapted *VeSOMAC* Protocol Overview

Similar to *ISOMAC*, the key strategy is that if a newly joined vehicle cannot immediately identify a feasible time region, it first chooses a non-feasible slot that lies on the right side of the slot of the vehicle immediately ahead in the platoon. At this stage, the *ordering constraint* for the rear neighbor is not satisfied and the bitmap constraint may or may not be satisfied. In the following iterative slot movement each vehicle attempts to place its slot behind the slot of its immediate front neighbor.

Contrast to *ISOMAC*, both the initial slot selection and the iterative slot movement process require a vehicle to know its neighbors' location information. This is accomplished by including a vehicle's location in the header of each of its transmitted packets. This is a reasonable assumption for most of the ITS application

models [67, 75] with access to onboard GPS systems.

Consider the topology in Figure 5.4, in which the allocation step-1 depicts Tx-slots chosen by vehicles *A-E* before vehicle *C* enters the network. Here we assume the maximum bitmap length (frame length), thus eliminating the *bitmap constraint*. Therefore, we only consider the *timing* and the *ordering* constraints. With the allocation in step-1, vehicles *A*, *B*, *D*, and *E* each has a collision free slot that satisfies the *timing* constraint. Also vehicle pairs *A-B* and *D-E* individually satisfy the *ordering* constraint. This is because *A's* slot is ahead of *B's*, and *D's* is ahead of *E's*. As a result, at step-1 all vehicles are at a steady allocation state.

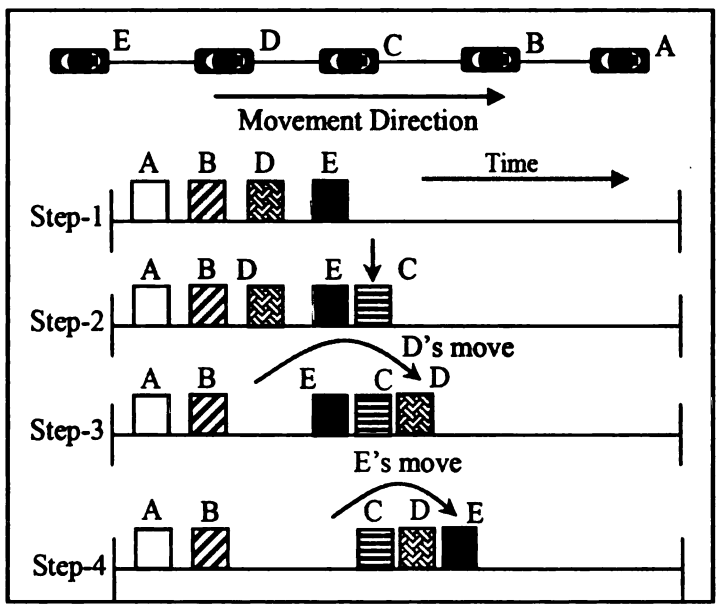


Figure 5.4: Iterative slot movements for allocation convergence

Upon entering the network, *C* learns about the slot locations of its 1-hop neighbors *B* and *D* from their periodic transmissions, and the 2-hop neighbors (*A* and *E*) through the bitmaps in *B's* and *D's* packet headers. Since the temporal gap between *B's* and *D's* slots is less than a single slot, *C* is not able to find a slot

satisfying the *ordering constraints* with respect to both *B* and *D*. As shown in step-2, unable to find a feasible slot, *C* chooses a collision free but non *ordering-constraint* compliant slot right after *E's* slot. With this, the *ordering constraint* with respect to *B* is satisfied but not with respect to *D*. Since at the end of step-2 the slot ordering of *C-D* is reverse to their physical ordering, now vehicles *D* and *C* cannot satisfy the *ordering* constraint, and therefore their allocations become unstable. Vehicles *A*, *B* and *E* however still remain stable.

Responding to its own instability, in step-3 *D* moves its slot after its front neighbor *C's* slot for satisfying the *ordering* constraint. Similarly in step-4, *E* moves its slot after *D's* slot. After these moves, the allocation slot ordering of the platoon becomes consistent with the vehicles' physical ordering. Finally, the allocation pattern after step-4 becomes compliant with the *timing*, *bitmap* and the *ordering* constraints, and therefore it is considered to be stable.

To summarize, the core idea of *VeSOMAC* logic adaptation is to let each vehicle iteratively move its slot following its immediate front neighbor's slot until a combined allocation pattern which is stable for all vehicles in the neighborhood emerges.

If the *ordering constraint* is removed, the iterations above become much simpler. In Figure 5.4, the allocation reconfiguration will now stop at step 2 after *C* chooses its slot. This indicates a much faster convergence (2 steps) compared to the one with *ordering constraint* scenario (4 steps) as explained above. Generally speaking, by removing the *ordering constraint* in *VeSOMAC*, it is possible to trade data plane

delay for faster protocol convergence during a topology change.

5.2.2.5 Generalized Slot Selection Logic of *VeSOMAC* Adaptation

After joining a network, if a vehicle finds no feasible slot as defined in Section 5.2.2.3, it picks a collision-free slot (satisfying *bitmap* and *timing*, but not the *ordering* constraint) that is chosen randomly (uniformly distributed) within $B.\tau$ duration after the slot of its immediately front vehicle. If no such collision-free slot is found, the same binary exponential logic in *ISOMAC* applies.

5.2.2.6 Inherent Collision Detection and Resolution Logic

Packet collisions in *VeSOMAC* are resolved using the exact logic used in *ISOMAC*. Note that the *VeSOMAC* logic can handle collisions in the presence of interference range as used in 802.11 or any kinds of packet collision caused by special vehicle network scenarios. Consider a situation with two vehicles *A* and *C*, which are more than two hops away and sharing a transmission slot. Another vehicle *B* is only *A's* 1-hop neighbor but within *C's* interference range. In this situation, *B* can not receive a message from *A*, since it will be corrupted due to the interference from vehicle *C*. According to *VeSOMAC/ISOMAC* logic, vehicle *B* in this situation will not acknowledge to *A's* transmission in its bitmap, therefore vehicle *A* will eventually move its time slot thus resolving the collision with *C*. This means that the timing constraint in *VeSOMAC/ISOMAC* for 2-hop communication range turns into an equivalent constraint for 2-hop interference range, when the latter is considered."

5.2.2.7 Protocol State Machine Adaptation

Adapted *VeSOMAC* protocol state machine with all three constraints is presented

in Figure 5.5, and the equivalent pseudo-code is depicted in Figure 5.6. In general *VeSOMAC* logic follows the same routine as *ISOMAC* logic. Only difference is the adapted *VeSOMAC* has three constraints as opposed to two constraints in *ISOMAC*.

Since the asynchronous *VeSOMAC* relies on local clock for slot and frame timing, the relative clock drift between vehicles can move a vehicle's transmission slot with respect to those of its neighbors. This can perturb allocation stability of *VeSOMAC* by progressively violating the mapping between the relative slot locations and the information coded in the corresponding bitmap vectors. As a result, nodes can transition from *Stable* to *Evaluate* state, thus affecting the stability of the protocol. With periodic time synchronization using onboard GPS, the relative drift is periodically reset, and therefore it is less of an issue for the synchronous *VeSOMAC*.

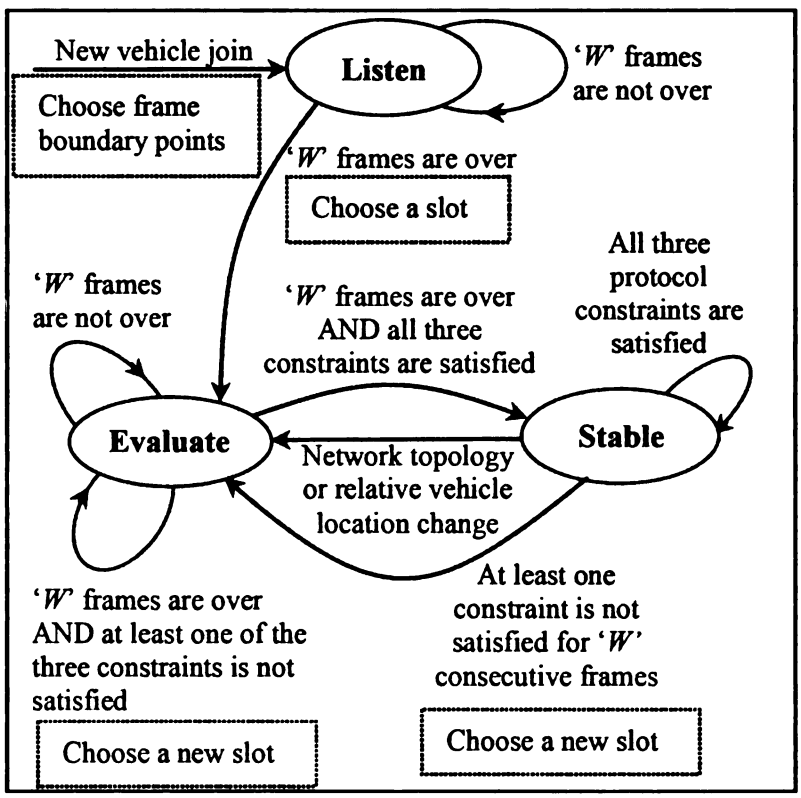


Figure 5.5: Adapted State machine for the *VeSOMAC* protocol with all constraints

```
/* VESOMAC Protocol Logic for Each Vehicle */
Arbitrarily choose the MAC frame boundary;
Listen:
    Listen to the existing network for W frames;
    // observe received transmissions to know the slot
    // location of all 1-hop neighbors; also observe the
    // bitmaps in the 1-hop neighbors' transmissions to
    // know the slot location of all 2-hop neighbors
    choose a slot using the logic described
    in Section 2.3.6;
Evaluate:
    Periodically transmit packets using the chosen slot
    and listen from neighbors for W frames;
    if (W is over && bitmap/timing/ordering constraints
        are all satisfied)
          goto Stable;
    else{
          Move the slot using the logic described in
          Section 2.3.5;
          goto Evaluate;
    }
Stable:
    Periodically transmit packets using the chosen slot;
    If (violation of at least one constraint persists
        for W frames){
          Move the slot using the logic described in
          Section 2.3.5;
          goto Evaluate;
    }
    If (new neighbor vehicle arrives or topology
        changes or packets get corrupted due to channel
        errors){/* do not change slot */
          goto Evaluate;
}
```

Figure 5.6: Pseudo code for the adapted *VeSOMAC* with all constraints

Due to the capacity scarcity created by inappropriately dimensioned frame size (see Section 3.4) when collisions are unavoidable in *VeSOMAC*, the optimality of allocation can be defined as how fairly the collisions are cycled among the participating vehicles. The objective is to prevent only a specific set of vehicles from suffering repeated slot collisions while the others enjoying the available bandwidth. According to the collision resolution logic in section 5.2.2.6, the implicit

acknowledgement mechanism in *VeSOMAC* forces the vehicles with colliding slots to move their slots so that the slot collisions are uniformly and randomly shifted around the vehicle in the neighborhood of limited capacity.

The model of frame size dimension in adapted *VeSOMAC* is same as in *ISOMAC*.

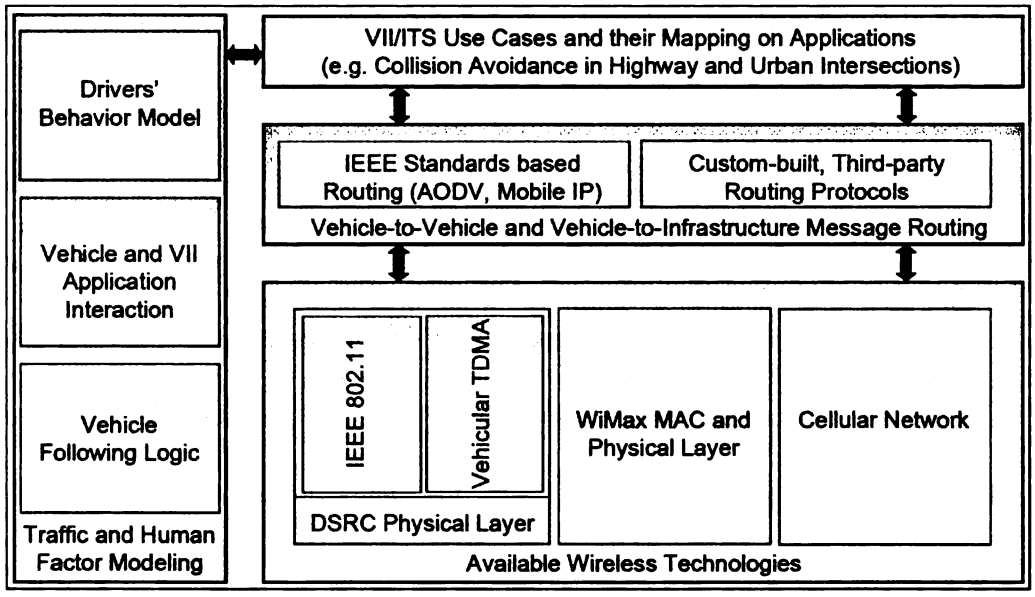


Figure 5.7: *VeNTSim*: ITS application and network evaluation/planning tool

5.3 Performance Evaluation

A hybrid simulation system for joint evaluation of different wireless network technologies, including their MAC and routing protocols, ITS applications, and vehicle following logic with drivers' behavior has been developed for evaluating the UICW and other applications under different MAC protocols. Figure 5.7 depicts the architectural components of our *VeNTSim* system which is designed to be open for incorporating the evolving DSRC and other radio technologies, ITS applications [62], and their required network protocols. Using *VeNTSim*, Both DSRC and non-DSRC radio technologies, including WiMax [62] and Cellular mechanisms [62] can be

architecturally evaluated for characterizing their impacts on heterogeneous ITS applications and use cases outlined by the research community and various standardization consortiums.

The networking functions in *VeNTSim* has been developed on top of ns-2 network simulator [76] by adding a vehicle mobility module that can react to the received wireless messages according to the modeled vehicle following logic with various drivers' reaction models. An ITS application modeling module capable of simulating a series of ITS applications such as cooperative collision control, cooperative cruise control and emergency vehicle preemption [62] has been also added. The synchronous version of *VeSOMAC* has been implemented at the ns-2 MAC layer, so that they can be compared with the 802.11 protocol running in the same radio environment. 802.11 is chosen for comparison is simply because it is the current DSRC recommending protocol. The architecture of *VeNTSim* is designed to be open for incorporating evolving DSRC and ITS applications [62], and their required network protocols.

5.3.1 *VeSOMAC* Protocol Convergence

After a network topology change, the convergence latency for *VeSOMAC* is defined as the time interval from when at least one vehicle becomes unstable to when all the vehicles become *Stable*, as defined in the state machine in Figure 5.5. The scenario shown in Figure 5.8 corresponds to an instability triggered by a vehicle passing 40 vehicles in a platoon. From the *VeSOMAC* standpoint, the platoon is stable till the 8^{th} frame, when the switching takes place. The vehicle passing causes

temporary instability by forcing at least two vehicles (not necessarily always the passing and the last-passed vehicles) out of their *VeSOMAC stable* states. But eventually, the network converges after 11 frames, which is 0.11 second for 10ms long frames.

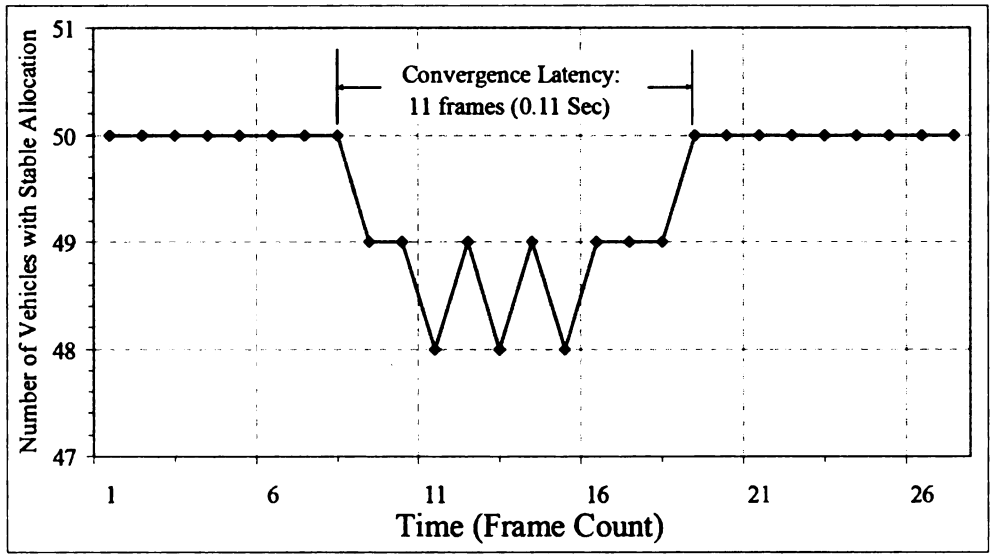


Figure 5.8: *VeSOMAC* convergence dynamics after a topology change

5.3.1.1 Platoon Merger

Figure 5.9 depicts convergence performance when two platoons, with individually stable *VeSOMAC* allocation, merge together. A bitmap size of 96 bits, which is closed to the frame size (100 slots), has been used for eliminating the *bitmap constraint* (see Section 3.3.2.2). Convergence during a platoon merger is needed because certain vehicles within the merging platoons may have had slots that can violate the *timing* and/or the *ordering constraints* after the merger. Therefore, post-merger slot movements are needed to resolve such situations. Observer that the convergence latency is always smaller without the *ordering constraints*, because convergence in this case is needed for fewer nodes which violate only the *timing*

constraint. Also, the convergence is faster when two platoons of 10 and 40 vehicles merge, compared to the 20 and 30 vehicles case. The results in Figure 5.9 demonstrate that during mergers of practical size highway platoons, the allocation convergence latency of *VeSOMAC* remains within only few hundreds of milliseconds, which are deemed acceptable for ITS related applications. It is also evident that the *ordering constraint* in *VeSOMAC* logic can be used for trading reduced data plane latency for slower protocol convergence.

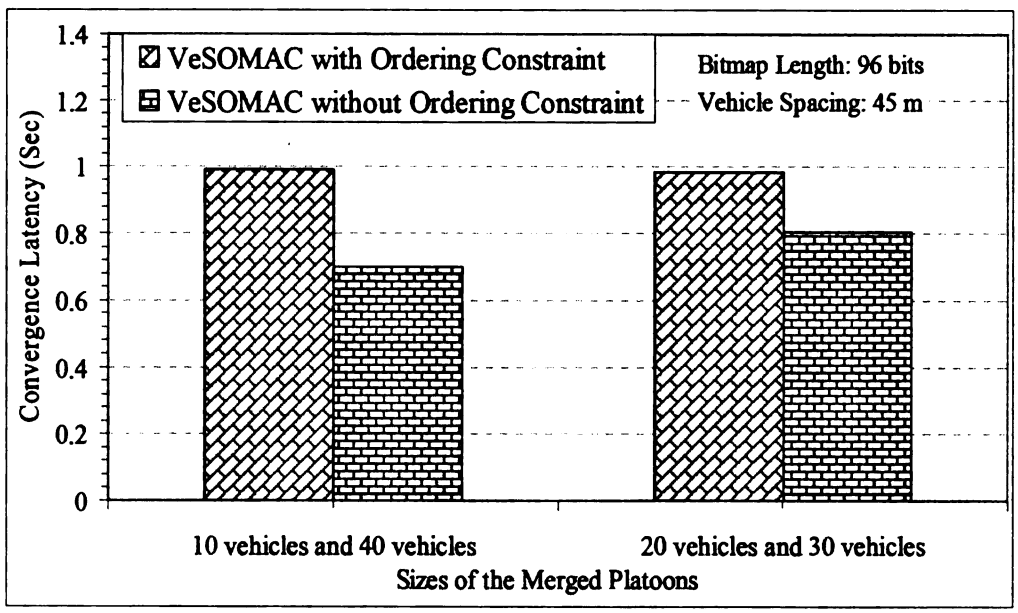


Figure 5.9: *VeSOMAC* convergence latency for platoon mergers

5.3.1.2 Intra-platoon Vehicle Passing

Figure 5.11 reports *VeSOMAC's* convergence performance when a vehicle passes few vehicles in front to get ahead in the platoon. Convergence after a passing is needed because a passing vehicle's slot can violate the *timing* and/or *ordering constraints* of the vehicles that it is passing and the ones in the neighborhood of its new location in the platoon. As expected, the convergence latency increases with

longer passing events because more vehicles' slots are prone to be violated in these cases. Also, the absence of the ordering constraint does expedite the convergence process. For all the experimented scenarios within a 50-vehicle platoon, the post-passing allocations have always converged within 100 ms, when the *ordering constraint* was not used. Notice that the convergence time of 40 vehicle-passing is the largest. The reason for is explained as followed. The slot distance between two immediate vehicles is 2.5 slots on average. Considering the frame size is 100 slots in our experiments. This means that on average every 40 vehicles, the slot position will be repeated. As a result, passing 40 vehicles has the largest probability to have a *VeSOMAC* slot collision. This explains why the convergence time of passing 40 vehicles is the largest. And the convergence would not increase as passing more number of vehicles.

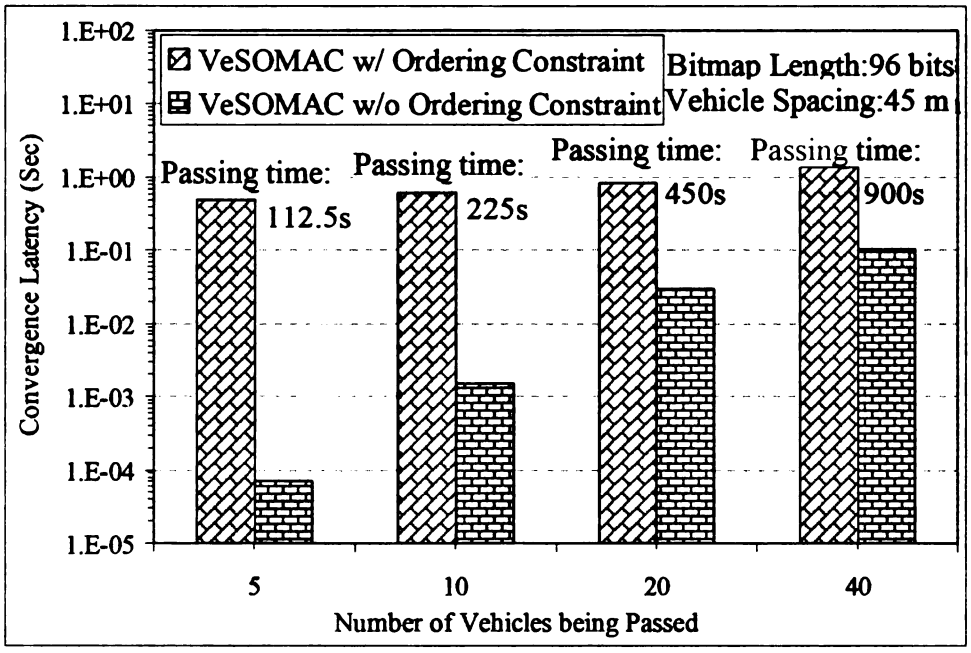


Figure 5.10: *VeSOMAC* convergence latency for intra-platoon vehicle passing

VeSOMAC has a speed-aware component which prevents a passing vehicle from

perturbing the allocation stability of the vehicles that it passes at a higher speed. The relative speed is inferred from the vehicle-location and timing information in the packet headers. A vehicle's *VeSOMAC* state machine does not react to the transmissions from neighbors with relative velocity larger than a pre-defined threshold. This way, the effects of passing shows up only after the passing vehicle settle down at a platoon location.

We have observed the convergence behavior with different size of the bitmap vectors. As expected, since with larger bitmap size the *bitmap constraint* becomes weaker, the convergence becomes faster. For example, in an experiment in which a vehicle passes five other vehicles, the convergence latency was registered to be 0.54 sec. and 0.49 sec. for bitmap size of 72 and 96 respectively. With *ordering constraint* turned on, the absolute values have dropped but they showed similar bitmap dependency. This indicates that whenever possible, the largest possible bitmap size should be used for the fastest protocol convergence.

Although the evaluation has been focused primarily on highway situations, the *VeSOMAC* architecture is fully extendible for other traffic and application scenarios including data streaming, internet services, vehicle-to-roadside and urban intersections, and bi-directional highway traffic.

5.3.2 Highway Scenario Performance

The experiments in this paper were carried out for the CCA application on a one-lane highway platoon scenario [72] with DSRC radio communications. V2V communication was leveraged for reducing chain vehicle crashes caused by

emergency events in front of moving highway platoon. First, an emergency event is simulated in front of a moving platoon. Upon detecting the event, the platoon-front vehicle rapidly decelerates (8 m/s^2) [77] and starts broadcasting periodic Wireless Collision Warning (WCW) packets with the format (*event-id, sequence-number*). While the *event-id* remains constant for an event, the *sequence-number* is incremented within subsequent WCW packets, transmitted once in every 100ms [37, 66]. Upon receiving a WCW packet for an event for the first time, each vehicle in the platoon starts decelerating (4 m/s^2) [78] in order to avoid any impending collision due to the emergency event. Also, it rebroadcasts the packet only when it receives it for the first time. Fast and reliable WCW message delivery across the platoon is expected to reduce the number of vehicles involved in a chain collision [66, 72].

The CCA application was simulated in the presence of background UDP traffic generated by data intensive ITS applications. Because of their non-deterministic message recipients, all CCA traffic is forwarded using MAC layer broadcasts coupled with multi-hop broadcast forwarding [66]. The data intensive traffic is unicast forwarded both at MAC and routing layers. Unless specified otherwise, we have always used the worst case *VeSOMAC* allocation in which the TDMA slot ordering is completely reverse to the vehicle ordering in a platoon. The purpose is to evaluate *VeSOMAC*'s worst case application impacts compared to 802.11. The baseline vehicle, network, and *VeSOMAC* parameters are summarized in Table 6.1. Each presented data point corresponds to the average from 500 independent simulation runs.

Vehicle Related	
Platoon Size	50 vehicles
Vehicle Speed	68 mph (30 m/sec)
Inter-vehicle Spacing	25m to 45m ≡ [0.8 sec to 1.5 sec]
Vehicle Length	4 m
Emergency Deceleration	8 m/s^2
Regular Deceleration	4 m/s^2
Drivers' Reaction Time	0.75 sec to 1.5 sec
Network Related	
Channel	DSRC 5.9 GHz band, 24Mbps
Radio Model	Two ray ground
Radio Range	300m
MAC Protocols	IEEE 802.11 and Worst case *VeSOMAC-Synchronous*
WCW Packet Size	300 bytes (0.1 ms)
WCW Message Period	100 ms
VeSOMAC Frame Size	100 packets (10 ms)
VeSOMAC Bitmap Size	96 Bits (very weak *bitmap constraint*)
VeSOMAC Evaluation Time	$W = 3$ frames

Table 5.1: Baseline experimental parameters

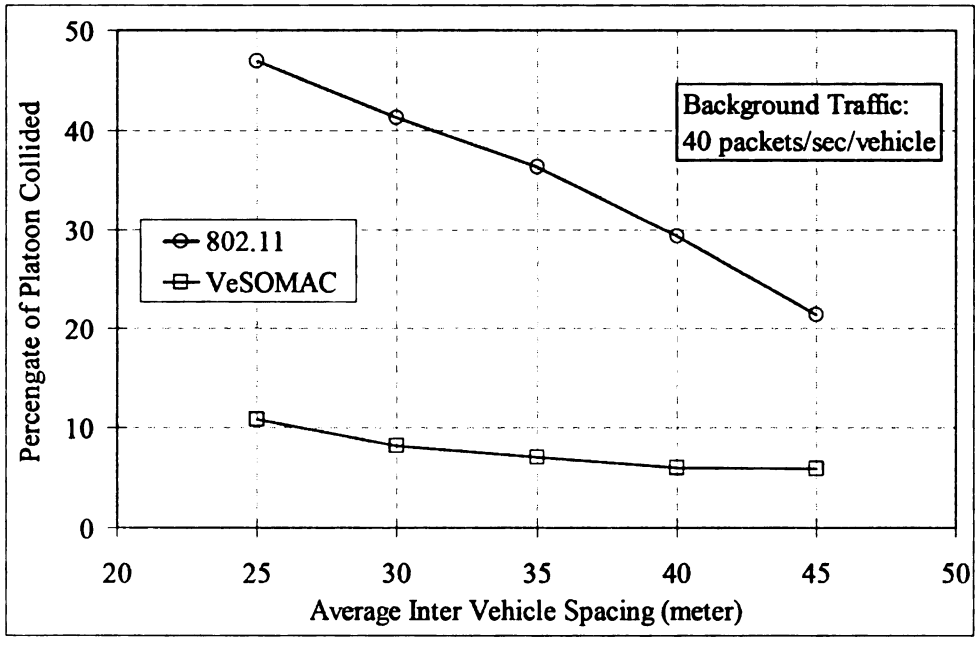


Figure 5.11: Efficiency of *VeSOMAC* for reducing vehicle crashes in CCA application

5.3.2.1 Highway Vehicle Crash Performance

Effects of Vehicle Spacing: The number of vehicles crashed as a percentage of a 50-vehicle platoon is plotted in Figure 5.11. With the CCA system turned off, if the vehicles decelerate based only on the tail brake light of the front cars, then for this entire range of vehicle spacing, all cars in the platoon were found to crash in a chain collision. However, as shown in Figure 5.11, by turning the CCA system on, with *VeSOMAC* as the MAC layer protocol, it was possible to bring the platoon collision down to 8%, when the vehicle spacing is nearly one second (i.e. 30 m). With 802.11, however, the vehicle crash probabilities are observed to be significantly higher, especially for closely following vehicles. As expected, fewer vehicles crash with increasing vehicle spacing. This is because with larger inter-vehicle space a vehicle gets a longer time cushion for safely decelerating to a stop before crashing into the vehicle in front.

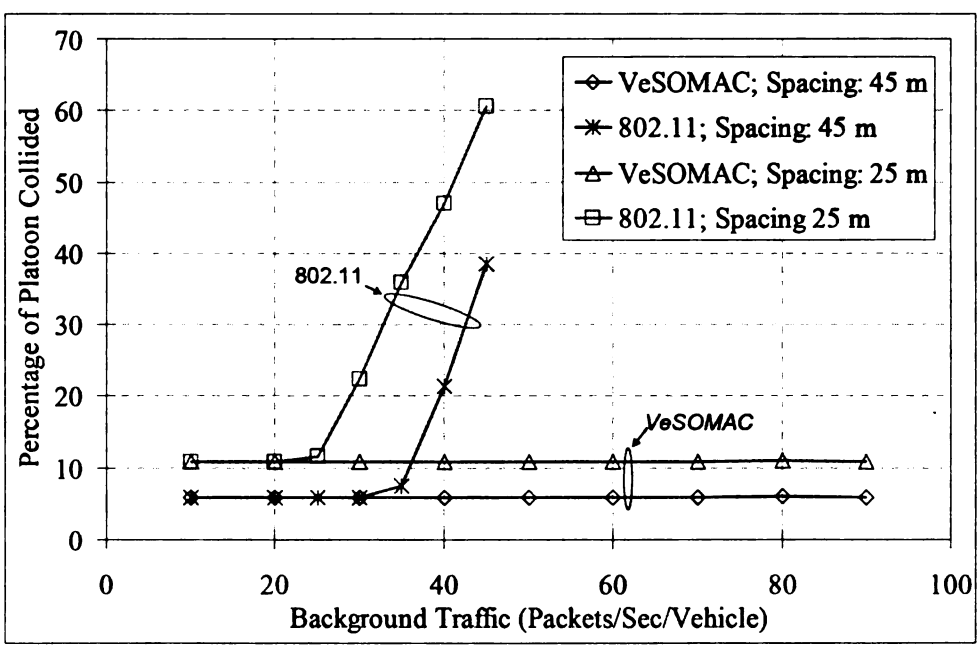


Figure 5.12: The effects of background non-safety traffic on avoiding vehicle crashes

Effects of Background Traffic: As shown in Figure 5.12, the performance advantage of *VeSOMAC* over 802.11 is maintained for a large range of background traffic load from the non-safety applications. Unlike 802.11, for which the MAC layer delivery delay goes up with the background traffic due to increased access contentions, the TDMA based *VeSOMAC*'s delay is virtually insensitive to the amount of background traffic. This is because of zero packet collisions, which explain why the vehicle crash count remains flat for *VeSOMAC* with increasing background traffic. For 802.11, on the other hand, the crash count increases linearly beyond background traffics of 25 ppsv and 35 ppsv for vehicle spacing of 25m and 45m respectively.

The cross-platoon message delivery latency for an example run of CCA with *VeSOMAC* is presented in the top graph of Figure 5.13. Latency is defined by the duration between when the emergency event occurs at the platoon front and when a corresponding WCW message is delivered to a vehicle. Relative stop distances between consecutive vehicles are reported in the middle graph. Since the vehicle length is assumed to be 4m, any relative stop distance of 4m or less corresponds to a crash. For vehicles avoiding a crash, the relative distance thus indicates the margin of safety provided by CCA. The bottom graph further reports the severity of vehicle crashes in terms of the relative speed between two crashing vehicles. Any relative speed which is greater than zero indicates a crash, and its magnitude indicates the crash severity.

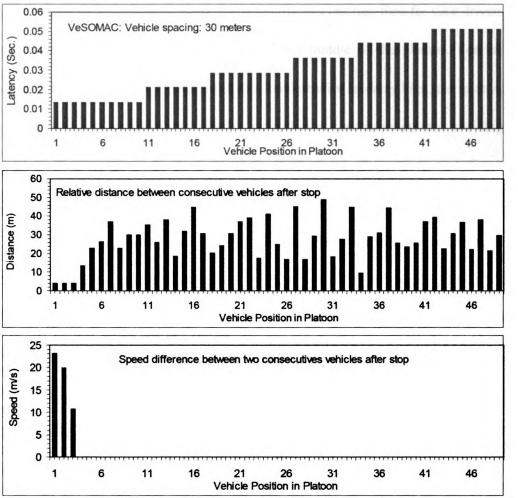


Figure 5.13: Latency and crash statistics for CCA with *VeSOMAC*

Results for an example CCA run with 802.11 MAC are reported in Figure 5.14. For *VeSOMAC*, since there are no packet collisions and the cross-platoon latencies are very small (up to only 51 ms), the only crashes are in the platoon front. These crashes are due to the lack of enough distance cushions and cannot be avoided with CCA even with zero message delivery latency. For 802.11, due to packet collisions and delay unpredictability, the WCW message latency increases significantly towards the rear of the platoon, where the broadcast traffic load increases progressively. This abrupt increase in latency from few milliseconds to few seconds

causes a cluster of vehicles to crash due to insufficient reaction time for their drivers to brake. This explains the chain crashes at the middle of the platoon. For the vehicles towards the rear, although the absolute latencies are very high, the relative latency is small. Therefore, all vehicles get sufficient time to react, thus avoiding crashes. This explains why there are no crashes towards the platoon end. Crash performance from these example CCA runs with *VeSOMAC* and 802.11 are consistent with the average crash results from 500 different experiments presented in Figure 5.11 and 5.12.

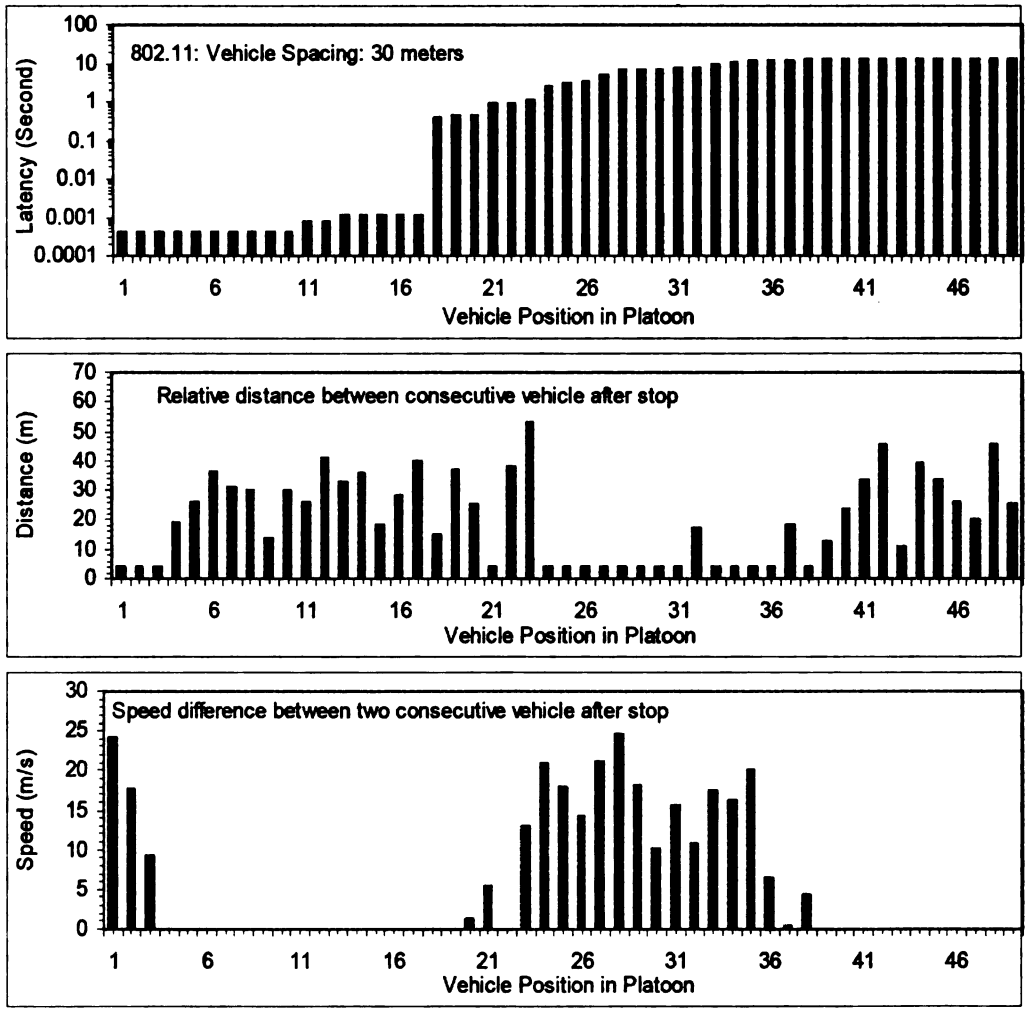


Figure 5.14: Latency and crash statistics for CCA with *802.11*

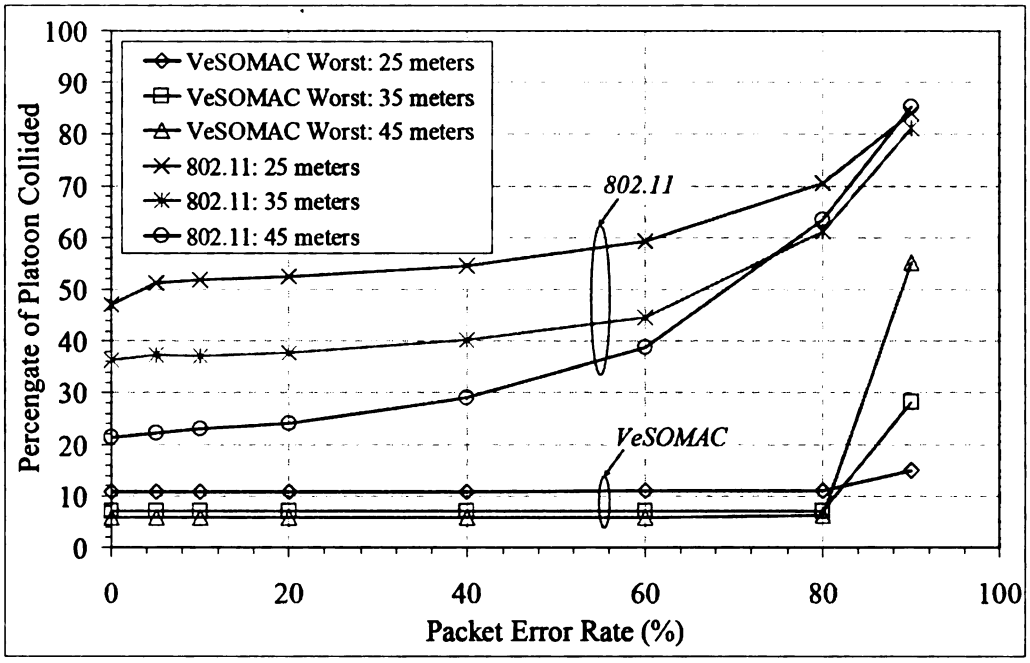


Figure 5.15: Effects of channel error on cooperative collision avoidance

Effects of Channel Errors: Crash performance with independent bit errors (no fading and burst errors were modeled) are reported in Figure 5.15. With increasing packet errors up to 80%, *VeSOMAC's crash* performance is insensitive to the errors, whereas for 802.11, the crashes steadily increase. For 802.11, with increasing channel error, latency for successful reception of a WCW message progressively increases towards the platoon end. This delay is compounded by 802.11's own contention related packet drops. As shown in Figure 5.14, higher latency translates into more vehicle crashes. With *VeSOMAC*, the latency was found to be low and steady for up to the 80% error mark. This low latency is due to the fact that in the absence of MAC collisions, a vehicle receives a specific WCW message several times - once from each of its neighbors. Therefore, in spite of certain number of packet losses due to channel errors, the vehicle is still able to receive at least one

copy of the message from an emergency event. *VeSOMAC's* low latency and the subsequent insensitivity to channel errors are because this reception redundancy. For error rates beyond 80%, however, the delay and the vehicle crash count shoot up. Packet drops beyond 80% is not practical, and they are shown only for understanding purposes.

Within the practical range of channel errors, the crash performance expectedly worsens with smaller inter-vehicle spacing for both the protocols. With very large packet rates though, there is an interesting trend reversal in which closely spaced vehicles actually suffer from less crashes. The reason for this turned out to be higher reception redundancy at lower inter-vehicle spacing.

Based on these results we conclude that the proposed *VeSOMAC* protocol offers significantly better CCA vehicle crash performance compared to the DSRC proposed 802.11 protocols. In fact, within an acceptable packet error of less than 10%, *VeSOMAC* shows no sensitivity to errors.

5.3.2.2 Highway Data Plane Network Performance

5.3.2.2.1 Cross-platoon Message Delivery

Delivery Delay: Experiments were carried out with both *VeSOMAC* and 802.11 protocols with varying levels of background unicast load generated using a Poisson traffic model. For a 50-node platoon[5], the multi-hop WCW message delivery delay to vehicle number 49 is depicted in Figure 5.16. In addition to the worst case

---

[5] The platoon-front vehicle that encounters the emergency event is number *0* and the vehicle at the platoon end is number *49*.

allocation (see Section 5.3), a version of *VeSOMAC* with the ordering constraint turned on has also been experimented with. With increasing background traffic, the cross-platoon latency goes up for all the protocols due to increased MAC and queuing delays. However, compared to *VeSOMAC*, the delay for 802.11 is much more sensitive to the background data rate. With 802.11, for traffic higher than approximately 33 ppsv, the latency goes beyond seconds. With both versions of *VeSOMAC* however, the delays are constrained within only tens of milliseconds even for traffic rates as high as 90 ppsv. This large comparative latency of 802.11 further explains its poor CCA crash performance as shown in Figure 5.11 and 5.12.

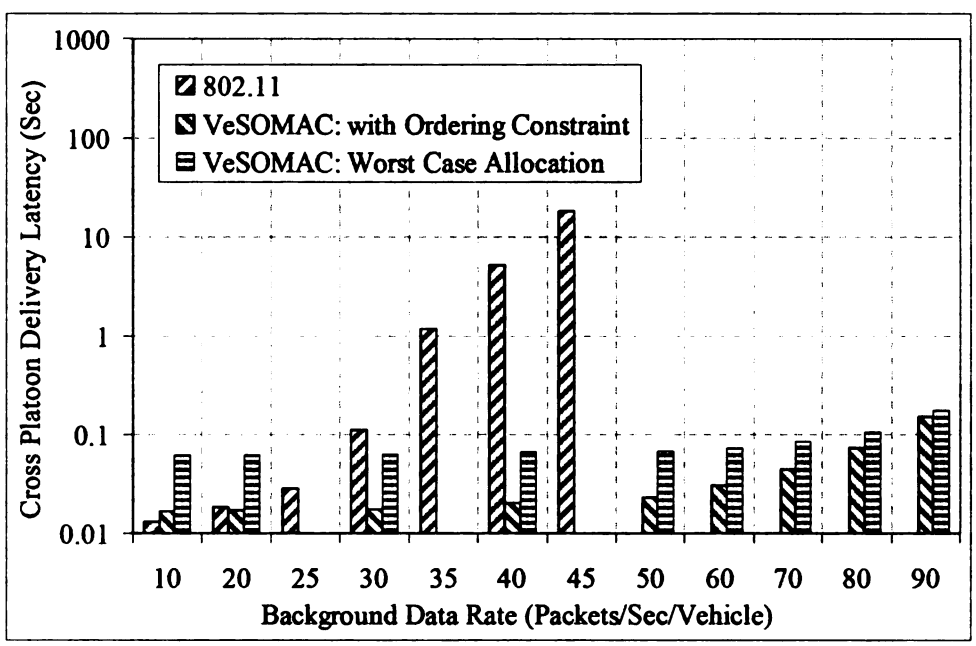


Figure 5.16: WCW message delivery latency across the entire platoon

At lower rates (i.e. between 10 to 20 ppsv), there is no queuing latency for both the protocols, and the MAC delay for *VeSOMAC* is actually larger than that of 802.11. This is because unlike in 802.11, for TDMA protocols a half-frame average MAC latency is there irrespective of the data rates. As the data rate increases, the

802.11 delay increases at a much faster rate due to MAC layer collisions, retransmissions and carrier-sensing delays. For *VeSOMAC* however, there are no such delays at the MAC layer at steady state. As a result, the queuing effects are also less severe. These account for the lower rate sensitivity for the *VeSOMAC* delay.

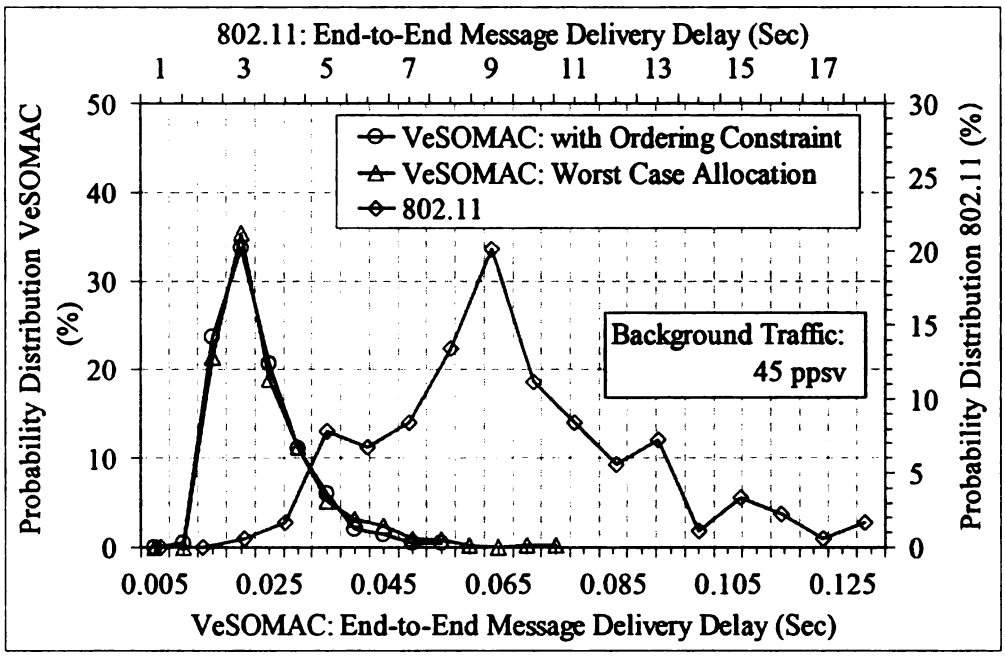


Figure 5.17: Distribution of cross-platoon delivery latency for 802.11 and *VeSOMAC*

Delay Distribution: Distributions from 500 experiments involving message deliveries to the platoon-end vehicle are plotted in Figure 5.17. For 802.11, the distribution has a much wider spread (tens of seconds) compared to both the versions of *VeSOMAC* (only tens of milliseconds). Unlike 802.11, since *VeSOMAC* does not have stochastic delay due to carrier sensing, collisions, and retransmissions, its latency is deterministic. This explains the low delay jitter for *VeSOMAC*, which not only contributes to its better CCA crash performance, but it can be also beneficial for inter-vehicle audio and video streaming applications [62].

Packet Drops: Drop statistics for the WCW safety messages across the platoon is

presented in Figure 5.18. Due to collisions, 802.11 is susceptible to frequent packet drops. For instance, with a background data rate of 45 ppsv, on an average an 802.11 based CCA application would loose the first WCW message by the time it reaches the 24th vehicle in the platoon. Meaning, if the platoon-front was not periodically sending the WCW messages, the vehicles beyond the 24th vehicle would not have received the message, thus suffering from the possibility of chain crashes. Similarly, the 2nd WCW message gets lost by the time it reaches the 32nd vehicle. However, because of zero collisions, *VeSOMAC* can deliver the very first WCW message to all vehicles in the platoon. The results from Figure 5.18 reinforce the CCA crash performance findings in Figure 5.11 and 5.12.

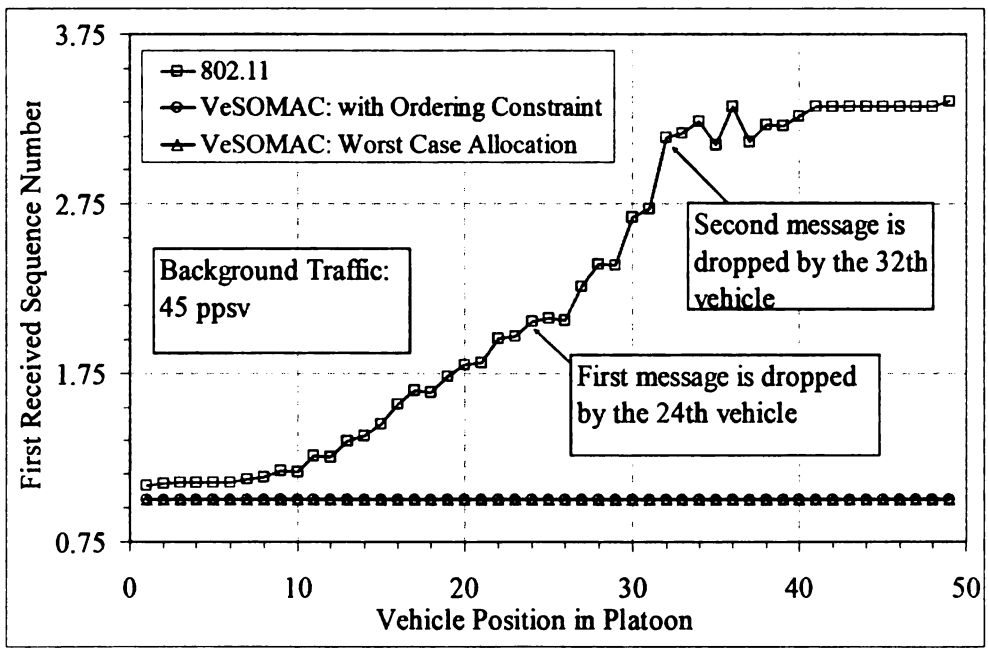


Figure 5.18: Message drop at different platoon locations

5.3.2.2.2 Benefits of Location-aware Slot Ordering

Cross-platoon latency performance for the worst, the average and the best case *VeSOMAC* are presented in Figure 5.19. The best and the average cases represent

VeSOMAC with *ordering constraint* turned on and off respectively. And the worst case represents a forced allocation in which the TDMA slot ordering is completely reverse to the vehicle ordering in a platoon. For a low background data rate of 10 packets/sec/vehicle (ppsv), *VeSOMAC* with *ordering constraint* is able to deliver a WCW message to the platoon-end vehicle approximately 21.5ms sooner than the average case (without *ordering constraint*), and approximately 45ms sooner than the worst case allocation. This indicates that the *ordering constraint* in the *VeSOMAC* logic (Section 5.2.2.7) can indeed achieve reduced multi-hop packet delivery latency through a location aware slot allocation.

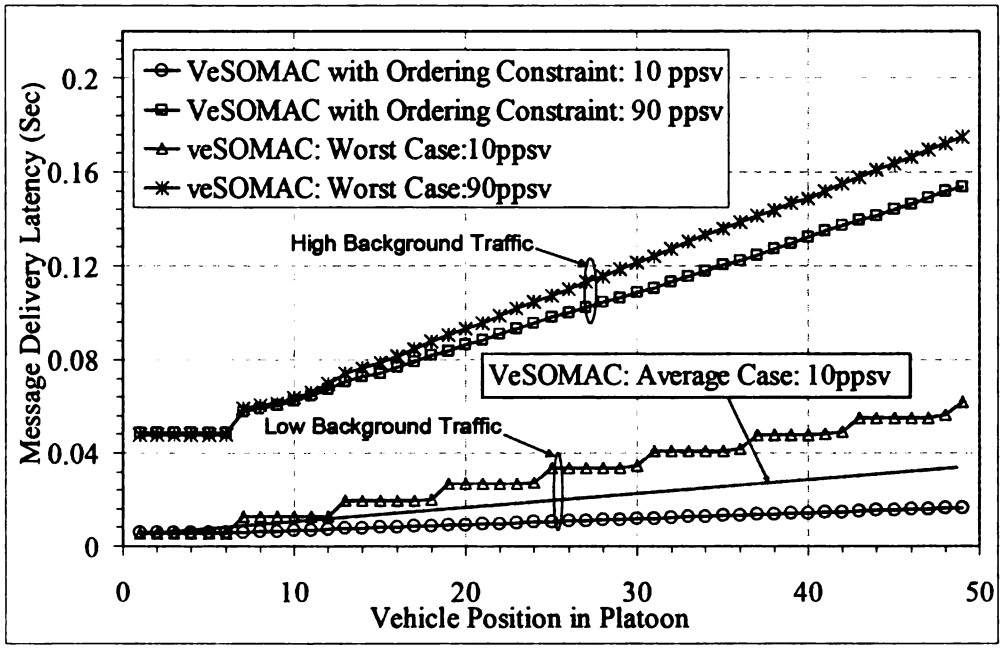


Figure 5.19: Effects of the *ordering constraint* on delay for *VeSOMAC*

However, for higher background data rates (e.g. 90 ppsv) the delay differential between with and without *ordering constraint* scenarios are less prominent. This is because packet queuing at higher rates introduces delay, which makes the cut-though forwarding with ordered TDMA as explained in Section 5.2.2.2 not possible. With

the present experimental setup, the benefits of *ordering constraint* were found to be preserved for rates up to approximately 46 ppsv.

5.3.2.3 Highway Scenario Performance during VeSOMAC Reorganization

Although the performance of CCA with TDMA at its steady state allocation is consistently better than that with 802.11 MAC, the majority of TDMA protocols (with a notable exception of the protocol LCA [41]) suffer from transient instability during slot reorganizations triggered by network topology changes due to vehicle movement. The results in this section report the impacts of such TDMA slot reorganizations on an ongoing CCA process.

Controlled vehicle movements in terms of vehicle passing within a 50 vehicle were created for triggering slot reorganization. The vehicle passing events were created in a way so that each such event is able to give rise to a temporary (before the reorganization process is able to converge) slot collision between a passing vehicle and the vehicles being passed. As a result of such collisions, the WCW message propagation within the platoon is blocked, thus causing potential vehicle crashes. Sequential vehicle passing events are designed in a way that each vehicle passing event is able to block the WCW message relay within the platoon for one WCW message cycle period, which is 100ms. In other words, with k consecutive passing events, the WCW messages generated in k number of consecutive WCW message cycle periods are blocked from propagation within the platoon.

Each vehicle passing event involves a simultaneous group passing movement of 6 adjacent vehicles and subsequent slot collisions between all those 6 vehicles and 6

other vehicles that are being passed. In total, there are 12 vehicles that are involved in one passing event. For the given radio range and inter-vehicle spacing as reported in Table 6.1, 6 represents the minimum group size that is required for completely blocking a WCW message due to temporary collisions caused by the TDMA slot reorganization process. The vehicle crash performance with different number of sequential passing events is shown in Figure 5.20.

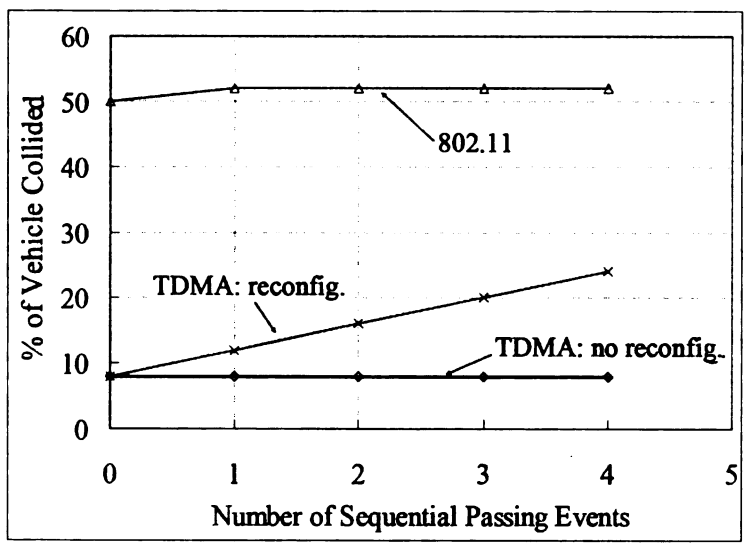


Figure 5.20: Impacts of TDMA slot reorganization during a UICW process

From Figure 5.20, it can be seen that since each passing event involves 12 vehicles, there are at most 4 sequential passing events possible within a platoon of 50 vehicles. Each passing event causes the WCW message propagation to be delayed by a complete WCW cycle period of 100ms, which is sufficient for two more vehicles in the platoon to crash. This explains as to why the percentage numbers for the crashed vehicles in Figure 5.20 increases linearly with the number of sequential passing events. Note, however, that for this experiment with 50 vehicles on the West-to-East street, the maximum percentage of platoon collided for TDMA is only

24%, which is less than half of that with the 802.11 MAC, which is 52%. The graph for TDMA without MAC reallocation depicts the baseline CCA performance in the absence of vehicle passing or other similar events that can trigger slot reorganizations.

Note that although not very practical, a large passing vehicle group size of six is chosen just to ensure guaranteed blockage of the WCW message. This serves a stress test for the TDMA protocol in the context of the targeted CCA application. In order to study the impacts of more practical group sizes, experiments were carried out for groups of 1 through 5 vehicles. Since the WCW messages are not able to be completely blocked in these situations, the vehicle crash count of CCA did not get any worse than the baseline performance with steady state TDMA, as reported in Figure 5.20. For the crashed vehicles, the collision speed in the experiments was found to be increased by at most 0.04 meters/sec. From the results in this section, it can be concluded that although the slot reorganization in TDMA due to network topology changes does degrade the CCA application performance temporarily, in an absolute sense it still remains better than that with the 802.11 MAC protocol.

5.3.3 Urban Traffic Intersection Scenario

5.3.3.1 Urban Intersection Collision Warning (UICW)

The Urban Intersection Collision Warning (UICW) application is designed for Cooperative Collision Avoidance (CCA) [62, 66] in urban traffic intersections using vehicle-to-vehicle DSRC communications. A representative UICW execution is described in Figure 5.21, which depicts a situation when the South-to-North traffic

light is red and the West-to-East light is green. Vehicle crashes in such a situation can occur if a bad driver on the South-to-North street runs the red light and collides with the cross vehicles in the West-to-East street. Once a single vehicle on the West-to-East street crashes with the bad driver's vehicle, a chain crash may also occur due to insufficient reaction time and stop distance available to the drivers of the West-to-East street.

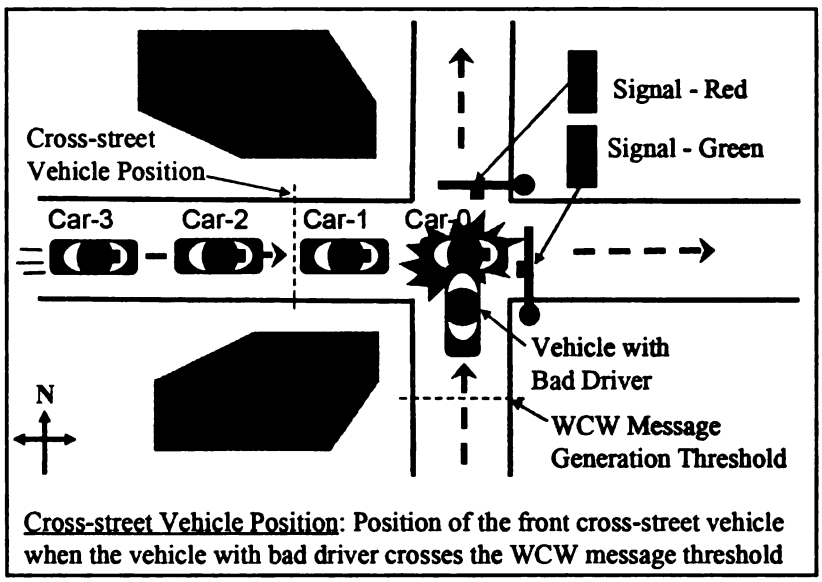


Figure 5.21: Cooperative Collision Avoidance in an urban intersection scenario

However, when the UICW application is turned on, such crashes can potentially be avoided or mitigated using the following mechanism. The DSRC onboard unit in the bad driver's vehicle first detects a problem situation from the vehicle's speed and location with respect to the intersection point. If a problem situation is detected so that the vehicle reaches a threshold distance from the intersection (referred to as the *WCW Message Generation Threshold* in the Figure 5.21) at a high speed while the traffic light is red, the onboard unit starts broadcasting periodic Wireless Collision Warning (WCW) packets, one in each message cycle period. A tuple containing the

identification of the originating vehicle and a monotonically increasing sequence number is used in the WCW messages for their unique identification. Upon receiving a WCW packet for the first time, each vehicle on the West-to-East street starts decelerating after a driver's reaction time, in order to avoid any impending collision due to the event. Also, the vehicle on the cross-street rebroadcasts the WCW packet when received for the first time. Note that the *Cross-street Vehicle Position* in Figure 5.21 represents the position of the front cross-street vehicle when the bad driver crosses the *WCW Message Generation Threshold* point. Although the WCW message is generated by the bad driver's vehicle in the above explanation, it can also be generated by a roadside infrastructure by measuring the speed of the violating vehicle at the *generation threshold* point, and by noting the status of the traffic light. Once the WCW message is generated by the roadside infrastructure, the same logic involving the WCW message interpretation applies.

Depending on the network protocol efficiency, vehicle speed, and the inter-vehicle spacing, a significant number of the cross-street vehicles can be saved using the UICW application as outlined above. From the network standpoint, the delivery delay and reliability of the WCW messages from the bad driver's vehicle to the cross-street vehicles play a crucial role in reducing the number of vehicles involved in a chain crash.

5.3.3.2 UICW Operational Details

Figure 5.22 illustrates the dynamics of a chain crash after the front car (Car-0) on the West-to-East street collides with the vehicle with bad driver on the

South-to-North street. For the sake of clarity, the dynamics of only three vehicles are presented in Figure 5.22. The y-axis in Figure 5.22 represents the vehicles' positions in terms of the distance from the street intersection point, as a function of time. As shown in the figure, the driver in Car-1 starts decelerating when he or she sees the tail brake light of Car-0, and the driver in Car-2 and Car-3 do so when they see the brake lights of the vehicles ahead. Note that a vehicle starts decelerating after a driver's reaction delay following when the vehicle ahead applies its brake. In Figure 5.22, with a finite driver's reaction time, Car-0 gets hit by Car-1 at the intersection point. Subsequently, Car-1 is hit by Car-2, and Car-2 is hit by Car-3. This example shows when the drivers react solely on the visual information (tail brake light), how all the vehicles on the West-to-East street can end up in a chain collision.

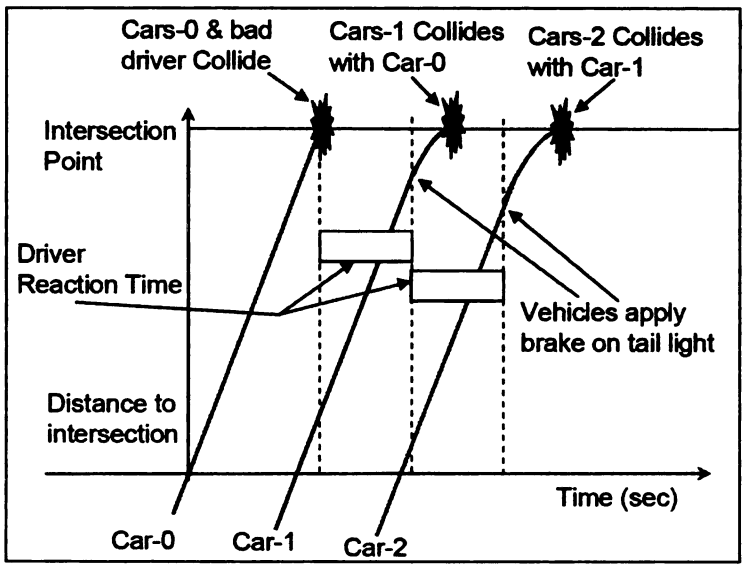


Figure 5.22: Dynamics of a cross-street chain collision without UICW

For the same scenario, the usefulness of UICW application is illustrated in Figure 5.23. With UICW [62] turned on, all West-to-East vehicles apply their respective brakes and start decelerating after a combined delay of Wireless Collision Warning

(WCW) message delivery and a driver's reaction time following the generation of the WCW message by the vehicle of the bad driver. In the depicted instance in Figure 5.23, for the given WCW delivery latency and the driver's reaction time, although the driver in Car-0 is able to apply its brake, due to insufficiently available stop distance the vehicle is not able to avoid a collision with the vehicle in the South-to-North direction.

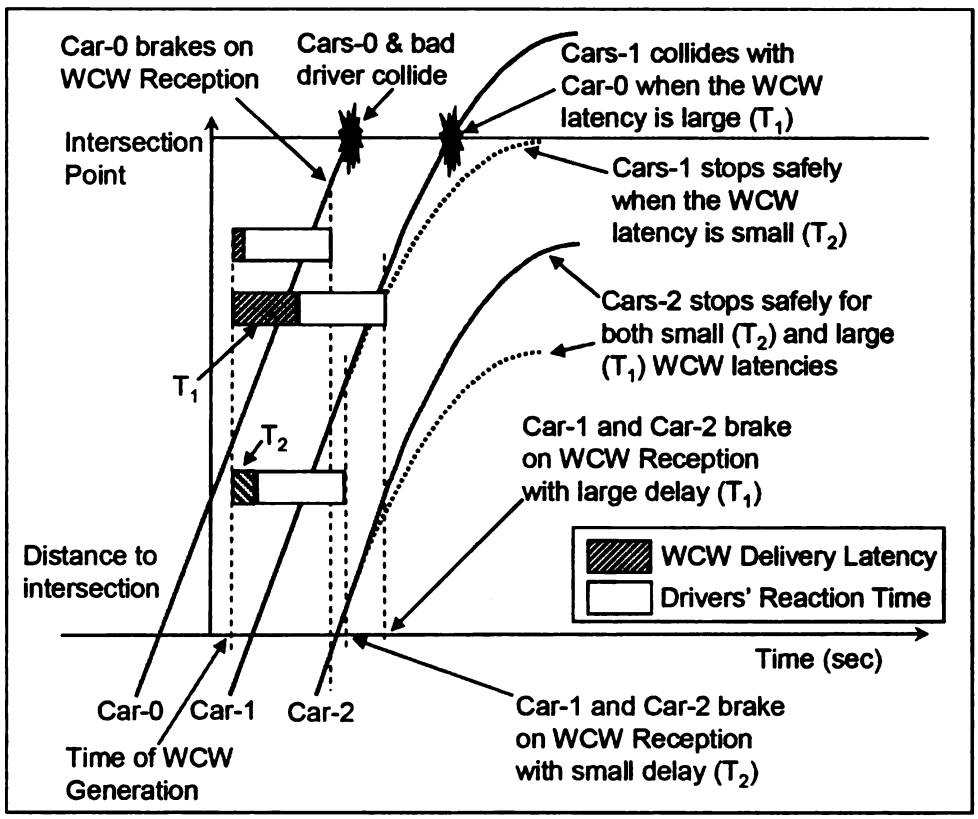


Figure 5.23: Leveraging UICW for reducing intersection vehicle crashes

For Car-1 and Car-2, two scenarios are explored: with a large WCW message delivery delay T_1, and a small delivery delay T_2. For the same driver's reaction time, while the smaller WCW latency can save Car-1, the larger latency cannot. For Car-2, however, because of its sufficiently available stop distance a crash can be avoided even with the large WCW delivery latency. The scenarios in Figure 5.22 and

5.23 illustrate that: a) the UICW application can indeed reduce vehicle crashes when the vehicles in the West-to-East street rely on wireless warning messages, and b) low message delivery latency is a key to the overall success of this ITS application. This also reinforces the need for a MAC layer protocol with low and deterministic delivery latency. The core logic for WCW message generation and interpretation are shown as the pseudo-code in Figure 5.24.

```
/* WCW msg Generation Logic at the vehicle with bad driver*/
do{
    keep checking the vehicle speed,
      distance to the intersection,
      and traffic signal status;
    if(this vehicle will be unable to stop before intersection
       && the traffic signal is currently red
       && distance to intersection <= threshold distance){
         /* the vehicle is exepected to run a red light */
         Start originating WCW msg. periodically;
    }
}
```

```
/* Message Interpretation Logic at the West-to-East vehicles */
/* A WCW Message arrives at a West-to-East vehicle*/
check the message origantor and sequence number;
if(new msg. origantor){
    store new msg. origantor id and seq. #;
    rebroadcast the msg.;
}
else{
    if(seq. # is new){// newly updated msg.
        store new seq. #;
        rebroadcast the msg.;
    }
    else{// received old msg.
        drop the msg.;
    }
}
```

Figure 5.24: Pseudo-code for the WCW generation and interpretation in UICW

5.3.3.3 Multi-slot Message Broadcast with TDMA MACs

With TDMA MAC, when the vehicle with bad driver in the UIWC application generates its WCW message, there is a possibility that this message will collide due to a MAC slot overlapping between the generating vehicle and at least another vehicle within its wireless range on the West-to-East street. Such collisions can occur before a TDMA slot reallocation can take place as a response to the network topology change caused by the vehicular movements. From an UIWC crash avoidance standpoint, these collisions can prove fatal and need to be avoided.

We introduce a *multi-slot MAC broadcast* mechanism in which the WCW generating vehicle sends the message on multiple TDMA slots in a frame, in addition to on its own allocated slot. This way, if the generating vehicle's slot does collide with that of a cross-street vehicle, the redundancy in the *multi-slot MAC broadcast* will improve the chance of a collision free transmission of the WCW message. The redundant slots are chosen randomly within the TDMA frame during the successive WCW message transmission to avoid any persistent collisions, thus further improving the chance of successful delivery of a WCW message. For a UICW scenario with vehicle density D (average number of 1-hop radio neighbor of a vehicle) and frame size F (number of TDMA slots per frame), if the *multi-slot MAC broadcast* redundancy is n (the number of TDMA slots used for a multi-slot MAC broadcast), then the probability of a collision free WCW transmission can be written as: $1-(D/F)^n$. Note that the *multi-slot MAC broadcast* is needed only for the UICW safety application and is not enabled for data intensive non-safety applications.

5.3.3.4 Urban Traffic Intersection Experimental Setup

For the evaluation in this paper, we have used DSRC as the radio technology with a custom-built message forwarding for UICW, and DSDV [79] routing for non-safety data intensive applications. The vehicle following logic in UICW comprises of the intersection traffic rules, and the drivers' behavior is modeled in terms of the reaction time with different ranges and distributions. A synchronous version of TDMA (*VeSOMAC* [73, 74]) has been implemented at the ns-2 MAC layer so that its impacts on the chosen applications can be compared with the current DSRC recommended 802.11 running in the same radio environment.

Vehicle and Scenario Related	
Platoon Size	50 vehicles
Vehicle Speed	45 mph (20 m/sec)
Inter-vehicle Spacing	9m to 15m ≡ [0.45 sec to 0.75 sec]
Vehicle Length	4 m
Emergency Deceleration	8 m/s^2
Regular Deceleration	4 m/s^2
Drivers' Reaction Time	Fast [0.5 - 1.0 sec], Slow [0.75 - 1.5 sec]
WCW Msg. Generation Threshold Dist.	50 m
Cross Street Vehicle Position	20 m to 50 m
Network Related	
Channel	DSRC 5.9 GHz band, 24Mbps
Radio Model	Two ray ground
Radio Range	100m
MAC Protocols	IEEE 802.11 & *Synchronous VeSOMAC*
WCW Packet Size	300 bytes (0.1 ms)
WCW Message Period	100 ms
TDMA Frame Size	100 packets (10 ms)
Channel Packet Error Rate	5%

Table 5.2: Baseline experimental parameters for the UICW application simulation

The UICW scenario was simulated in the presence of background UDP traffic generated by non-safety ITS applications. Due to their non-deterministic message

recipients, all UICW traffic is forwarded using MAC layer broadcasts and multi-hop broadcast forwarding [66]. The non-safety background traffic is unicast forwarded both at MAC and routing layers. The baseline simulation parameters are summarized in Table 6.2. Each presented data point corresponds to the average from 500 independent simulation runs.

5.3.3.5 Urban Traffic Intersection Vehicle Crash Performance

The percentages of 26 vehicles (25 cross-street and one bad driver) that crash during the simulated incident are reported in Figure 5.25. When the UICW is turned off, the drivers respond only to visual information, and a large number of cross street-vehicles crash. It was observed that one vehicle first collides with the bad driver's vehicle, and then the vehicles behind engages in a chain collision. With no UICW, at 45 mph vehicle speed, even with a large vehicle spacing of 15 meters, almost 75% of the cross-street vehicles crash due to the red light running of the bad driver.

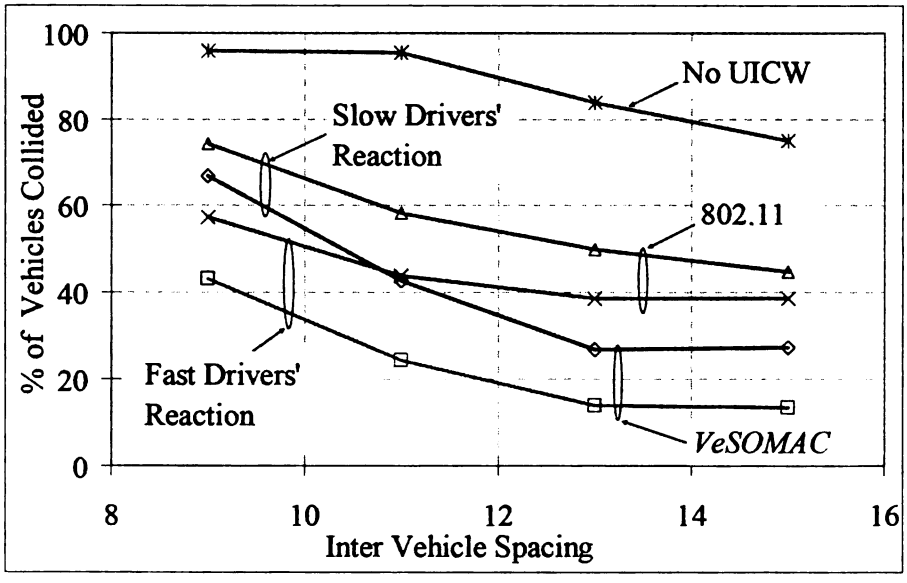


Figure 5.25: Vehicle crash performance with 802.11 and *VeSOMAC*

5.3.3.6 Impacts of MAC Protocols

When UICW is turned on, with *VeSOMAC* as the MAC protocol, it was possible to bring the crashes down to nearly 13% (3 vehicles), which is with 15m spacing and fast drivers' reactions (0.5-1 second). With 802.11, the vehicle crash probabilities are observed to be significantly higher; i.e. 39% for 15m spacing and fast drivers' reactions. As expected, fewer vehicles crash with increasing vehicle spacing. This is because with larger inter-vehicle space a vehicle gets a longer time cushion for safely stopping before crashing into the vehicle in front. Also, a fast drivers' reaction time helps to prevent the collision as shown in Figure 5.25. The reason that *VeSOMAC* has much smaller collision number can be explained from Figures 5.26 and 5.27 as followed.

The cross-platoon WCW delivery latency for an example run of UICW with *VeSOMAC* is presented in the top graph of Figure 5.26. This latency is defined by the duration between when the bad driver's vehicle generates the first WCW message after crossing the threshold point (see Figure 5.21) and when it is delivered to a vehicle. Relative stop distances between consecutive vehicles are reported in the middle. With a vehicle length of 4m, any relative distance of 4m or less corresponds to a crash. For vehicles avoiding a crash, the relative distance thus indicates the margin of safety provided by UICW.

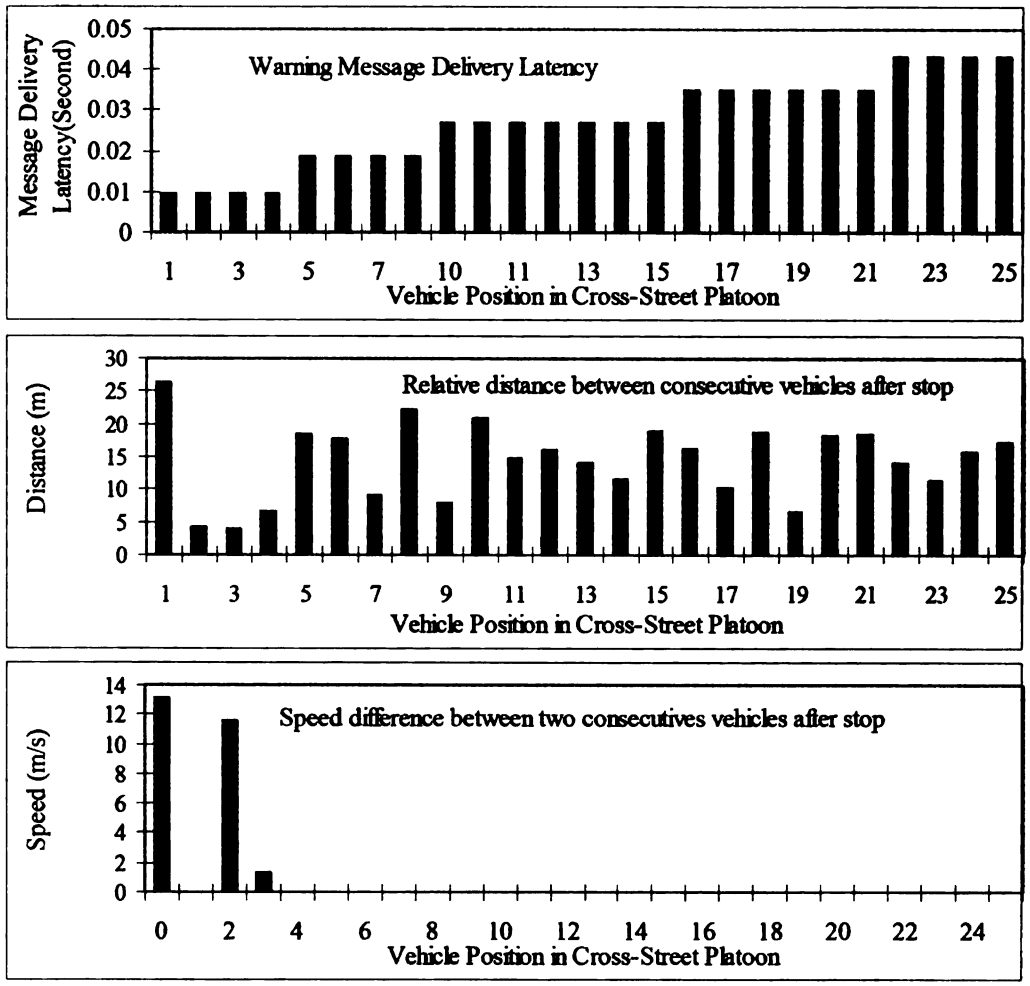


Figure 5.26: Latency and crash statistics for CCA with *VeSOMAC*

The bottom graph reports the severity of crashes in terms of the relative speed between two crashing vehicles. Relative speeds greater than zero indicates a crash and its severity.

Similar results for an example UICW run with 802.11 MAC are reported in Figure 5.27. For *VeSOMAC*, since there are no packet collisions and the cross-platoon latencies are very small (up to only 43 ms compared to seconds in 802.11), the crashes involve only the front of the cross-street vehicles. For 802.11, due to packet collisions the WCW latency increases significantly towards the rear of the cross street vehicles. This increase in latency causes a cluster of vehicles to crash due to

insufficient reaction time. This explains the chain crashes at the middle of the platoon starting from vehicle 7 in the Figure 5.27. Crash performance from these example UICW runs with *VeSOMAC* and 802.11 are consistent with the average crash results from 500 different experiments presented in Figure 5.25.

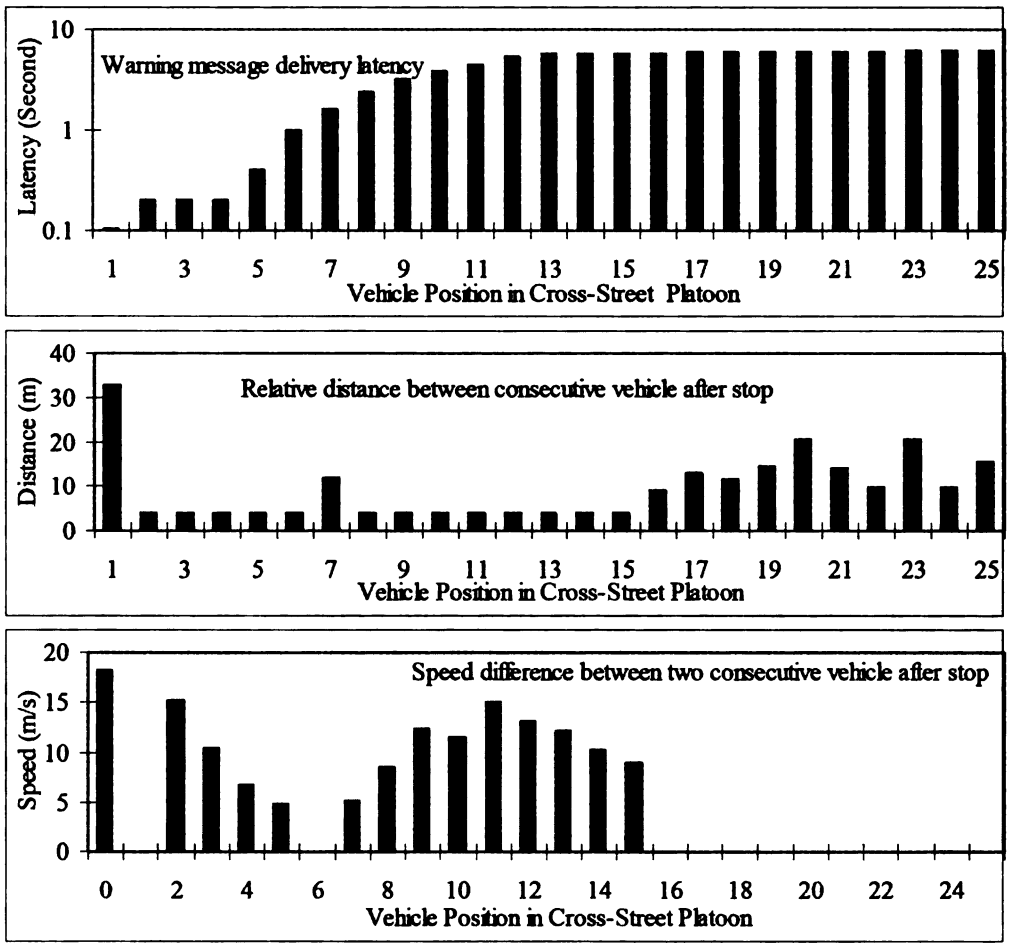


Figure 5.27: Latency and crash statistics for UICW with *802.11*

Based on these results we conclude that the proposed *VeSOMAC* protocol offers significantly better UICW vehicle crash performance compared to the DSRC-proposed 802.11.

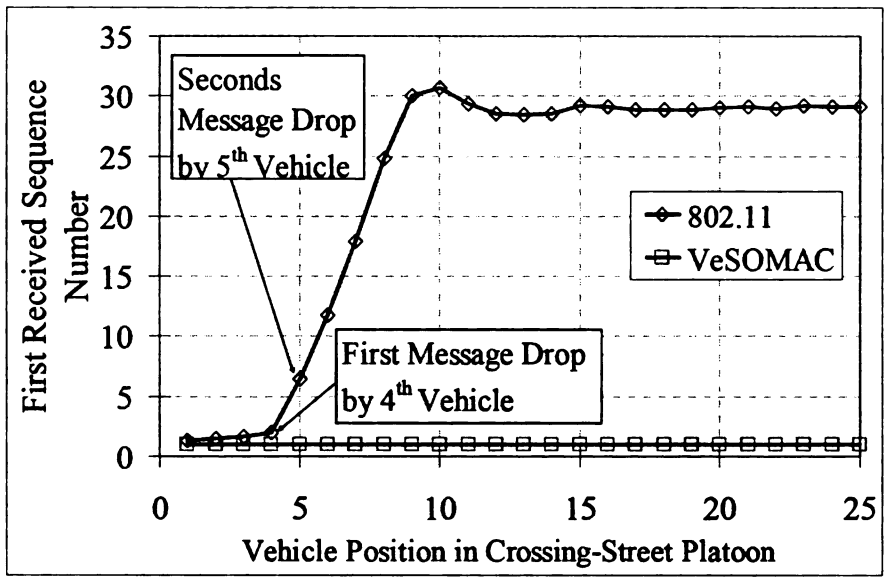


Figure 5.28: Message drop at different platoon locations

Packet drop statistics across the cross-street vehicles is presented in Figure 5.28. Due to collisions, 802.11 is susceptible to frequent packet drops. For instance, with a background data rate of 40 packets per second per vehicle (ppsv), on an average the 802.11 based UICW application would loose the first WCW message by the time it reaches the 4th cross-street vehicle. Meaning, if the message was not periodically broadcast by the bad driver's vehicle, the vehicles beyond the 4th vehicle would not have received the message, thus suffering from the possibility of chain crashes. Similarly, the 2nd WCW message gets lost by the time it reaches the 5th vehicle. However, because of zero collisions, *VeSOMAC* can deliver the very first WCW message to all cross-street vehicles. These drop results reinforce the UICW crash performance findings in Figure 5.25.

5.3.3.7 Impacts of Vehicle Count and Speed

Crash performance with varying vehicle count and speed is reported in Figure 5.29. As expected, with higher vehicle speeds more vehicle crash for the without

UICW as well as with UICW with 802.11 and *VeSOMAC* scenarios. This is because for a given vehicle spacing, at higher speeds, the vehicles get lower stop distances to avoid a crash.

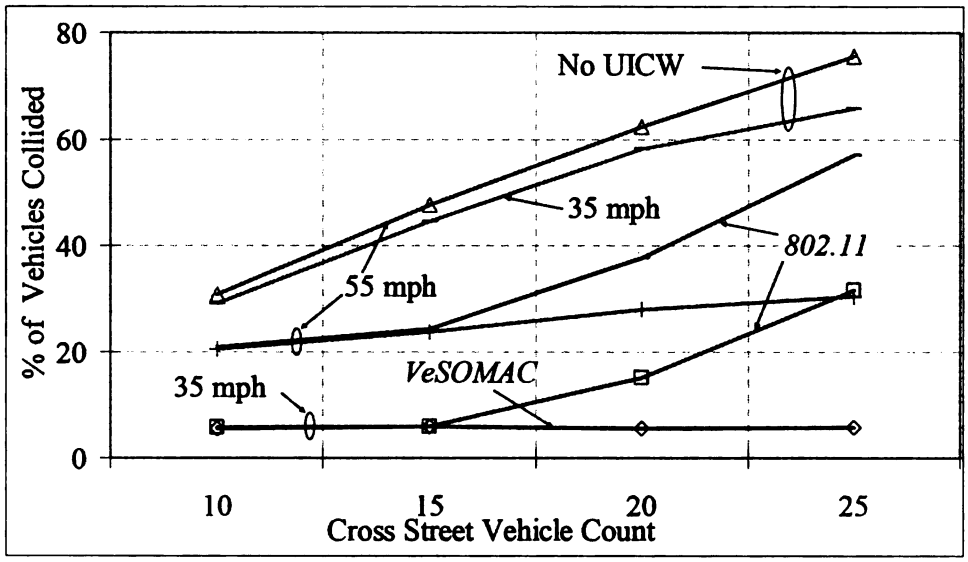


Figure 5.29: Crash performance with varying speed and vehicle count

Unlike the crash results in Figure 5.25 (for 25 vehicles), with fewer cross-street vehicles 802.11 performs as well as *VeSOMAC*. This is because the delay and drops for 802.11 are small (see Figure 5.27 and 5.28) and comparable to that of *VeSOMAC* for the front cross-street vehicles. With larger number of vehicles, however, the latency is larger – which explains more crashes for 802.11.

5.3.4 Inter-vehicle Data Transfer Applications Performance

The experiments in this paper were carried out for inter-vehicle data and file transfer applications on a one-lane highway platoon scenario [66] with DSRC radio communications. We have evaluated both DSRC-recommended 802.11 and the proposed *VeSOMAC* protocols with a channel rate of 24 Mbps. Two-ray ground propagation model was used with a radio transmission range of 300m. Packet and

frame size for *VeSOMAC* were chosen to be 300 bytes and 100 packets (10ms) respectively.

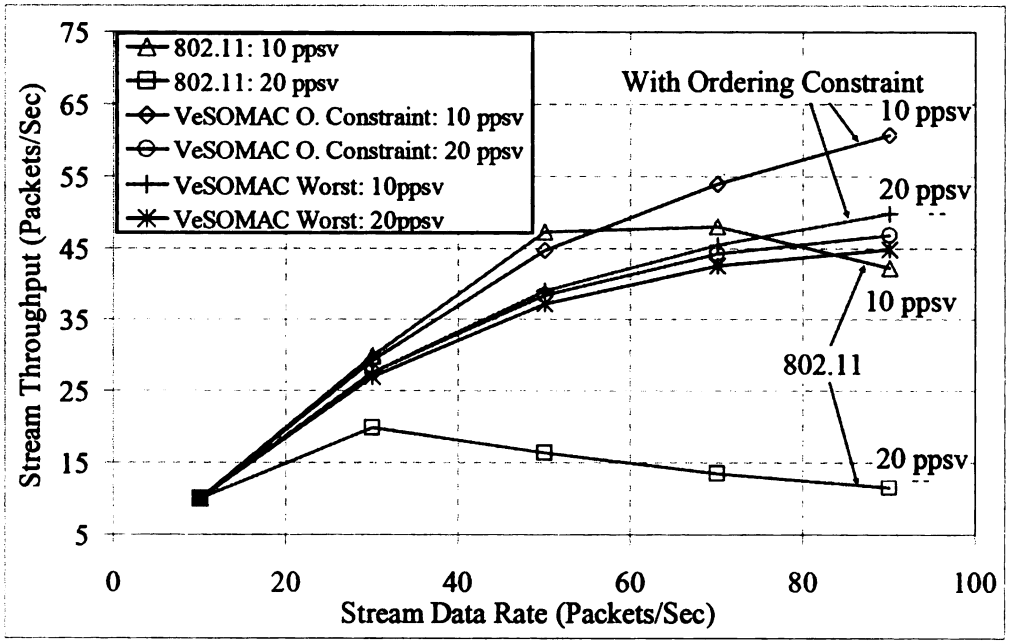


Figure 5.30: Throughput performance of non-safety data streams

5.3.4.1 Performance of Non-safety UDP Applications

Throughput of cross-platoon UDP data streams, representing applications such as inter-vehicle file, music, video, and traffic information downloads, are reported in Figure 5.30. With varying background traffic, the data rate of a UDP stream from the platoon front to the platoon end was changed to determine the maximum sustainable stream throughput. With low background traffic of 10 ppsv, a stream using 802.11 can sustain a throughput of up to approximately 47 packets/sec, beyond which it starts degrading due to packet collisions. With higher background traffic of 20 ppsv, its performance suffers drastically with a sustainable stream throughput of nearly 20 pps.

However, for *VeSOMAC* with *ordering constraint* the throughputs are much higher.

At 10 ppsv, *VeSOMAC* with *ordering constraint* is able to attain 27% higher throughput compared to the 802.11 (60 pps over 47 pps). Zero collisions, bounded delay, and location aware slot ordering in *VeSOMAC* account for this performance superiority. Without the *ordering constraint, VeSOMAC* can still offer better stream throughput than 802.11, especially at higher background rates. These results indicate the suitability of *VeSOMAC* for non-safety oriented vehicular data applications.

5.3.4.2 Performance of Non-safety FTP Applications

Figure 5.30 reports the FTP completion time for a 1 Mbyte data file transferred across vehicles which are multi-hop apart within a highway platoon of 50 vehicles. The reported results indicate the average FTP completion time computed over 30 and 10 simultaneous file transfer sessions active across different vehicle pairs within the platoon. In each experiment, a given number of connections with a specified connection length have been established with randomly chosen source and destination vehicles within the 50-vehicle highway platoon. Each point on the graph represents the average duration computed from 200 such independent experimental runs.

FTP has been run over TCP over 802.11 or *VeSOMAC*, and the TCP connections were set up using DSDV MANET [79] routing protocol. Note that such FTP based data transfer corresponds to a large number of data intensive applications such as maps, navigation and location based service, and music download [62].

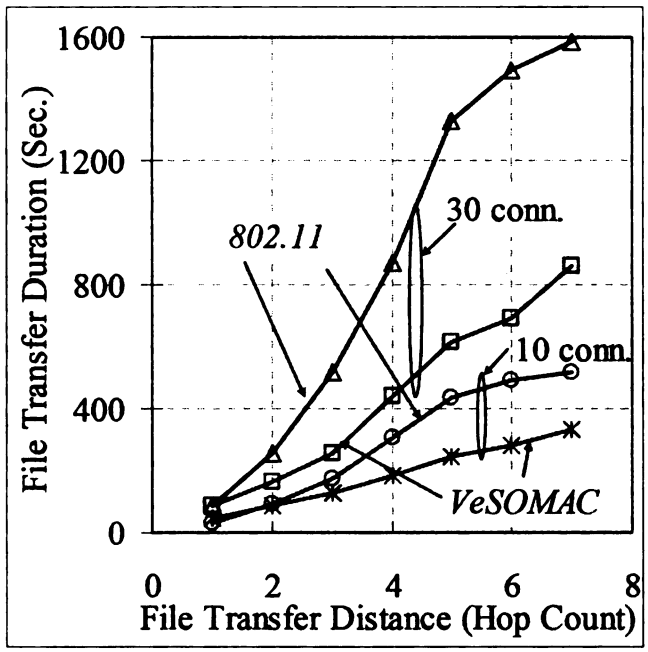


Figure 5.31: FTP duration with varying hop counts

The graphs in Figure 5.31 demonstrate that in majority of the situations, the file transfers are faster with the *VeSOMAC* compared to with 802.11, especially for scenarios when the involved vehicles are located farther apart within the platoon. The reason for this is better TCP performance with *VeSOMAC* compared to with 802.11 as reported in Figure 5.31. Note that the only scenario in which 802.11 performs better is when files are transferred across 1-hop, and with lesser number (10 in this case) simultaneously active connections within the platoon.

Figure 5.32:a reports the corresponding TCP throughputs using 802.11 and *VeSOMAC*. Observe that with longer connections the TCP throughput per connection reduces for both the MAC protocols due to increased user data traffic in the platoon. However, the proposed *VeSOMAC* enables higher TCP throughput compared to 802.11 for all scenarios except for 1-hop connections under small connection counts (10 in this case). For example, when the connections are 2-hop long and there are 10

connections present in the platoon, *VeSOMAC* can enable a per connection TCP throughput of 51 Kbps, which is a 66% improvement over the 802.11 enabled throughput of 30 Kbps. As explained later, the MAC layer packet drops of 802.11 in all scenarios except for the 1-hop connections under small connection counts turn out to be the reason [64] for the low TCP throughput.

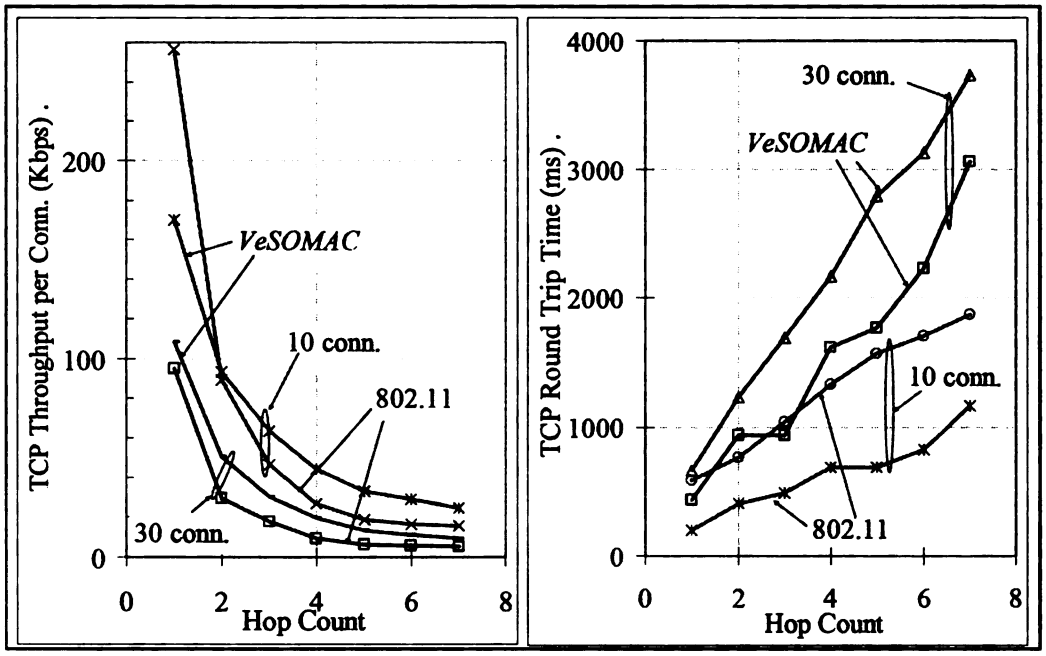


Figure 5.32: TCP Performance: a) throughput, b) round trip time

Generally higher TCP throughput using *VeSOMAC* can be explained through the TCP congestion window evolution as reported in Figure 5.33:a and 5.33:b (for a 5-hop connection with 30 simultaneously active connections). Observe that due to frequent MAC layer packet drops, the congestion window for the 802.11 case is reset much more frequently compared to the *VeSOMAC* case, in which there are no MAC drops. TCP segment drops due to 802.11 drops in this experiment are reported in Figure 5.34:a. The infrequent window resets for *VeSOMAC* are due to out-of-sequence packet delivery in the events of rerouting caused by the routing

protocol DSDV's periodic route refresh operations. Zero packet drops for the *VeSOMAC* allows the congestion window to grow higher, which in turn yields better TCP throughput.

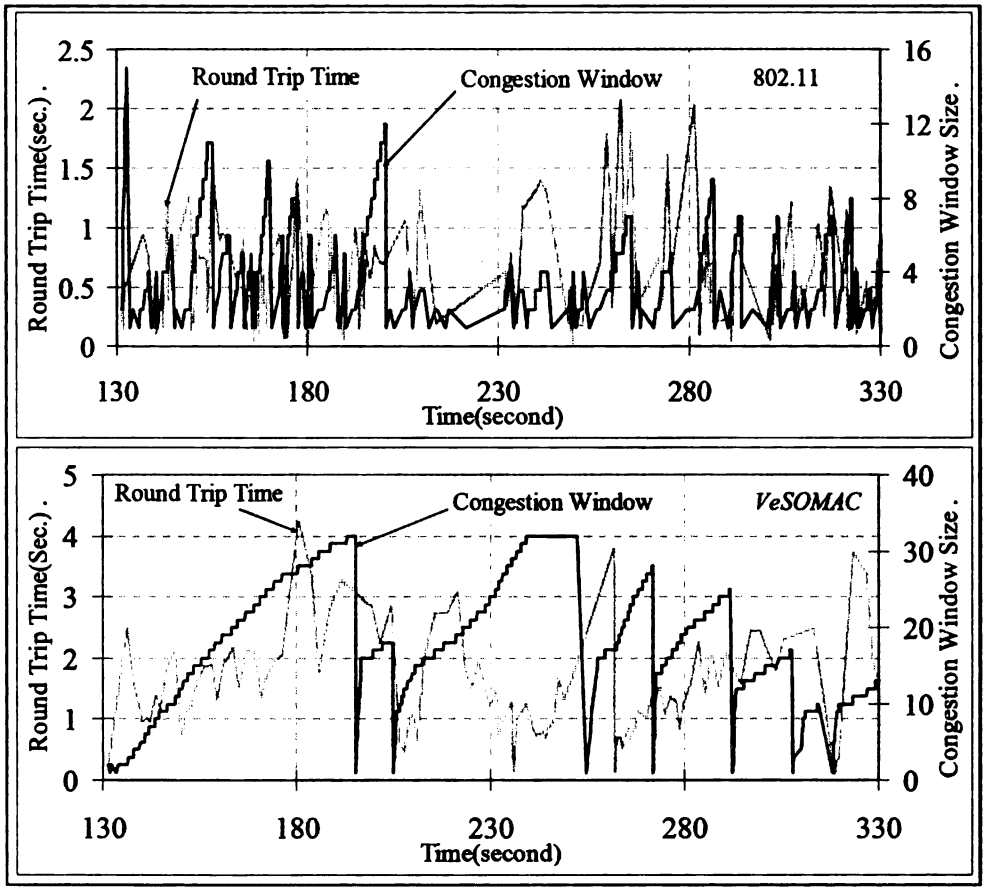


Figure 5.33: Dynamics of TCP : a) TCP-over-802.11, b) TCP-over-*VeSOMAC*

Note, however, from Figure 5.32:b and 6.33 that in spite of its better throughput, the TCP round trip delay for the *VeSOMAC* case is actually higher than that with 802.11. This is due to the higher load for *VeSOMAC* caused by larger TCP congestion windows. Larger TCP load in the *VeSOMAC* case is evident from the higher sent segment sequence numbers compared to the 802.11 case as shown in Figure 5.34:a. Higher TCP load forces *VeSOMAC* to operate at a higher MAC load point, which results into larger MAC queuing and subsequent TCP round trip time.

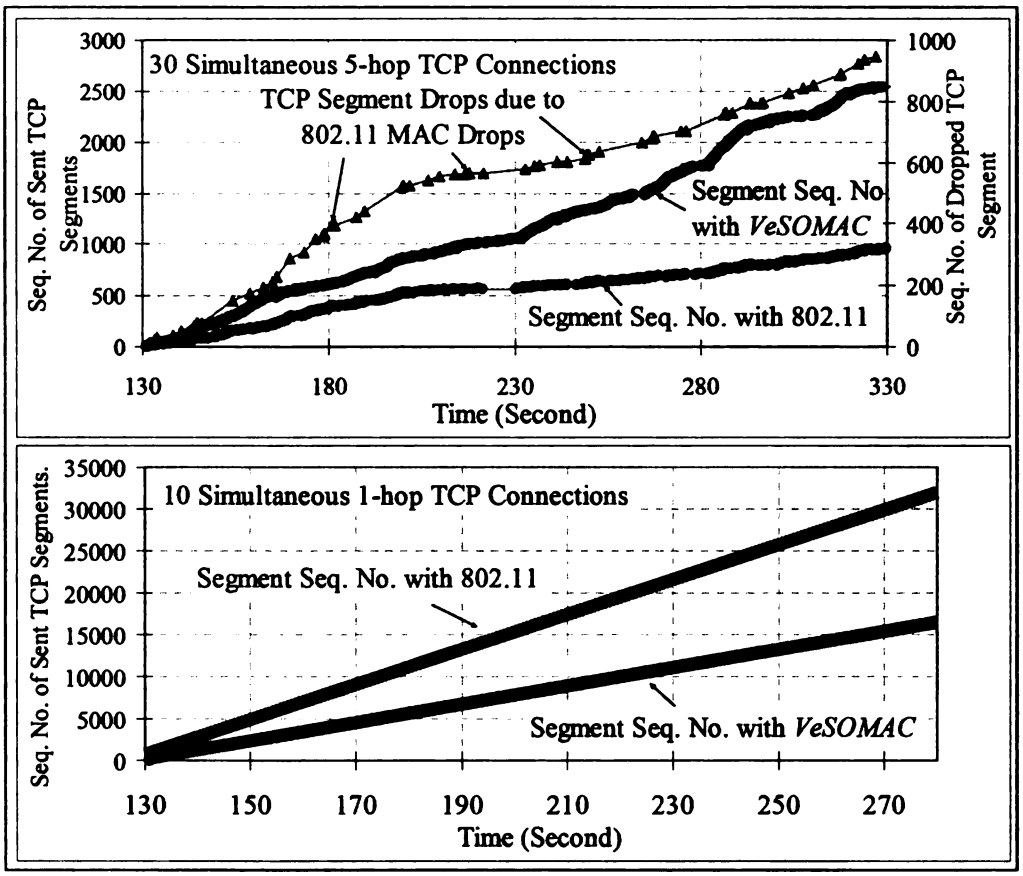


Figure 5.34: Dynamics of TCP segment generation:

a) 30 5-hop connections b) 10 1-hop connections

TCP loading for 1-hop connections under small connection counts (10 in this case) are reported in Figure 5.34:b. Unlike in Figure 5.34:a, 802.11 in this case enables higher TCP loading compared to the *VeSOMAC*. Lack of self-competition [64] for the 1-hop connections and mild inter-connection competition due to fewer connections help significantly reduce the MAC layer drops for the 802.11. This explains 802.11's better TCP and file transfer performance for 1-hop connections under small connection counts as shown in Figure 5.31 and 5.32.

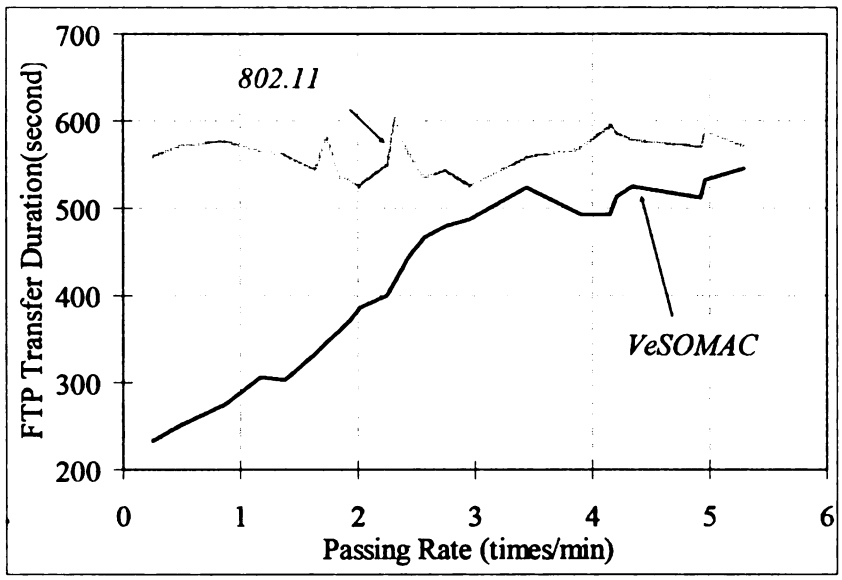


Figure 5.35: FTP duration with varying vehicle switching rates

5.3.4.3 Inter-vehicle Data Transfer Performance during *VeSOMAC* Reorganization

We also conduct experiments when there is a *VeSOMAC* slot allocation reconfiguration. A 50-vehicle platoon is used for these experiments and there are 10 concurrent 7-hop away TCP connections. We have some vehicles to pass certain number of vehicles, which is an ordinary traffic event in practical scenarios. Notice that all the experiments in the following are controlled in a way that the routing path of those 10 TCP connections will not be affected by the passing events. This is to say that the vehicles with passing events will not be on the route of any TCP connection. This minimizes the effect of the routing layer on the performance. Also, after each vehicle passing events, we force the passing vehicle to have a slot collision with one of its new neighbor which is the intermediate vehicle in the TCP connection route path. The *VeSOMAC* logic will then be used to choose a slot collision allocation after the passing events.

Figure 5.35 reports the FTP transfer duration of 1 Mbytes data changes over different vehicle passing rate in one of FTP connections. The FTP transfer duration of 802.11 is not prone to change with changing vehicle passing rate because of the inherit 802.11's topology change unawareness property. On average it takes around 550 seconds for 802.11 MAC to finish this FTP service. On contrast VeSOMAC has a much smaller FTP transfer duration as shown in Figure 5.31 in the stable state. Although the FTP transfer duration of *VeSOMAC* has a clear rise with increasing the vehicle passing rate, it reaches around 550 seconds when the passing rate is around 5.3 times per minute. *VeSOMAC* has still smaller FTP transfer duration than 802.11 does.

In the Figure 5.36:a, it shows the TCP congestion window evolution when there are two vehicle passing events, which all cause a slot collision on the routing path. The two passing events happen at time 190.0 and 260.0 second respectively. As we can see from the figure, there are TCP congestion window drops at both time stamps. The TCP segments loss during the temporary slot allocation configuration process leads to the TCP congestion window drops. After the *VeSOMAC* configuration, TCP congestion window then gradually increases to the maximum value. When there is no such vehicle passing event, as we can see the congestion window will maintain the maximum value after it increase to the maximum value. With 802.11 MAC, the congestion window evolutions with or without passing events are not much different. Similar to Figure 5.33, VeSOMAC is still able to reach much larger TCP congestion window size than 802.11 even there are some slot allocation reconfigurations.

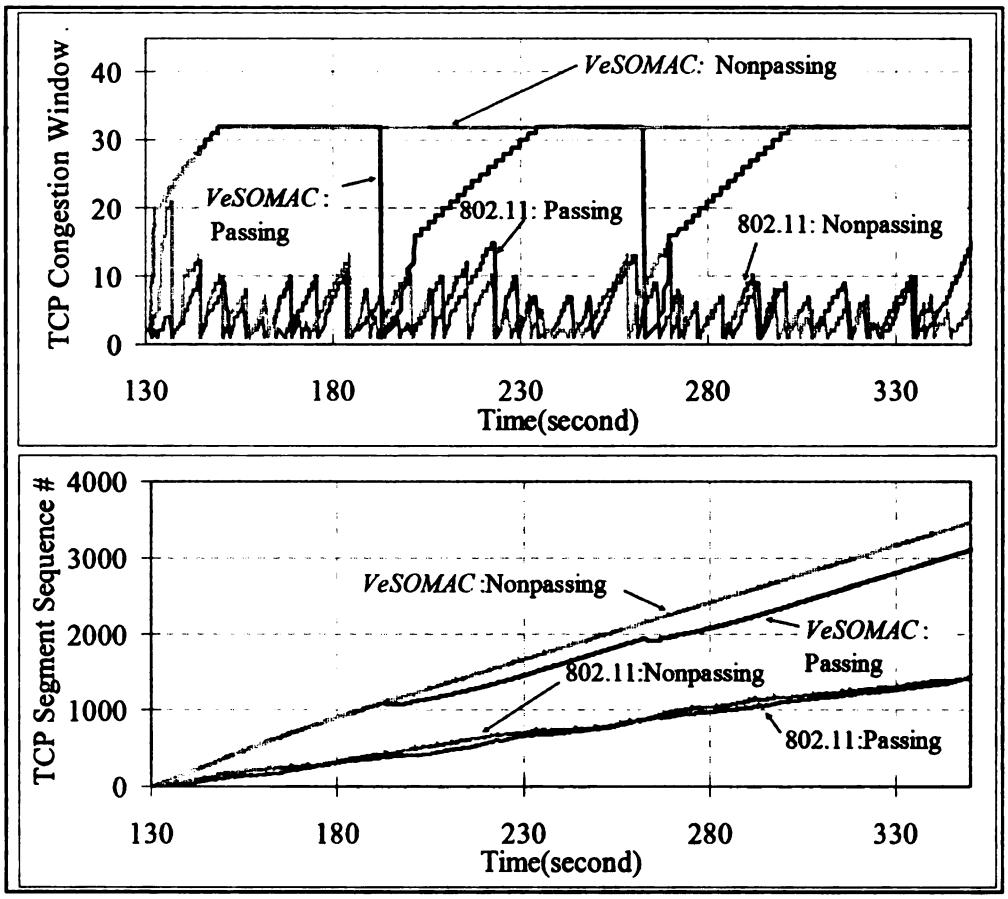


Figure 5.36: TCP Dynamics with passing events: a) congestion window, b) segments generation

Figure 5.36:b shows how the TCP segments generation pattern at the TCP source in the same TCP connection as we analysis in Figure 5.36:a. As we can see that two passing events cause some short period retransmissions. These retransmissions are the results of TCP segments loss during the short VeSOMAC reconfiguration period. The segments generation rate recovers quickly upon finishing slot reallocation. As we found before in the stable case, VeSOMAC is still able to send much more number of TCP segments than 802.11.

The dynamic of TCP throughput of the same connection is reported in Figure 5.37. Figure 5.37:a is the throughput dynamics with 802.11 MAC. As we mentioned

before, there is not much difference between with and without passing events. With VeSOMAC MAC, those two passing events will also temporarily affect the TCP throughput. As shown in Figure 5.37:b, there are two drops corresponding to two passing events at time 190.0 and 260.0 seconds. Whereas there is no such drop when no such passing events. Even with two drops, the average throughput of VeSOMAC is still around two times larger than 802.11.

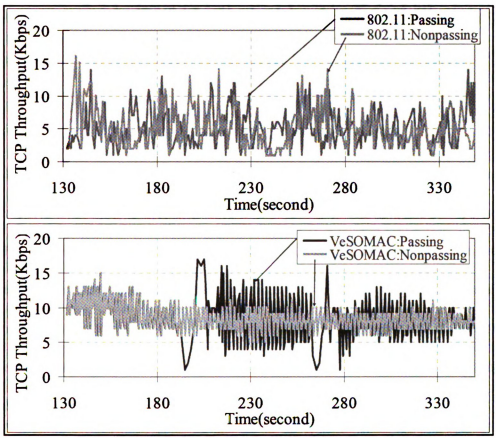


Figure 5.37: Dynamics of TCP throughput: a)TCP-over-802.11, b) TCP-over-*VeSOMAC*

Same as the results given before, the TCP round trip time is reported in Figure 5.38. *VeSOMAC* is having larger round trop time than 802.11 because of working in a higher traffic load.

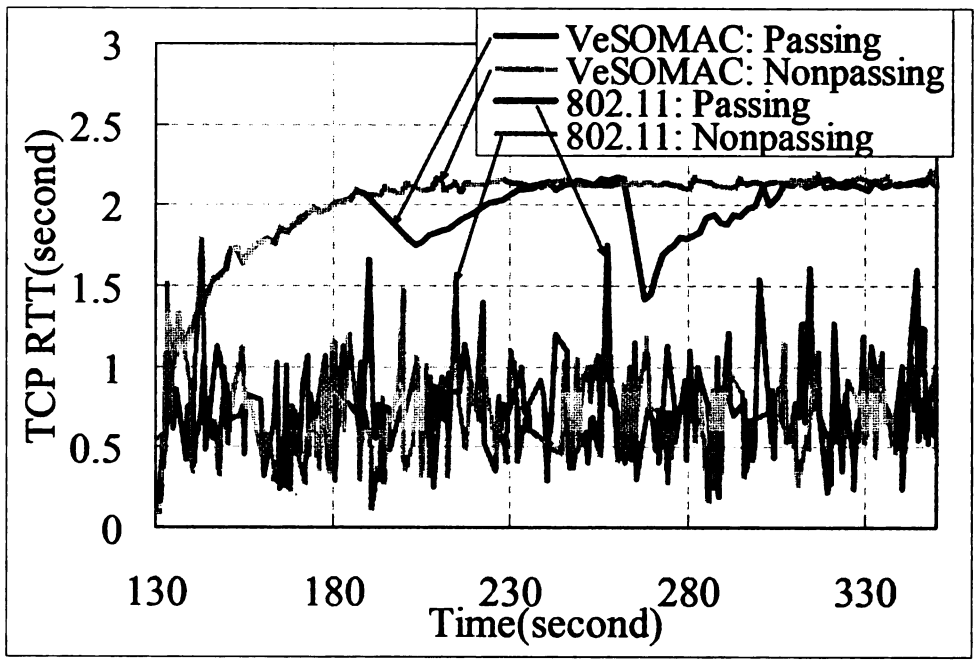


Figure 5.38: Dynamics of TCP segment generation

In the Table 6.3, we give some time constants of TCP, routing protocol and proposed *VeSOMAC* protocol. As we can see, if the vehicle platoon topology changes cause the routing and TCP protocol to respond, it may take 45 seconds to rebuild the TCP connections. This time constant is same even using mobility transparent 802.11 MAC protocol. Comparing to this time constant, *VeSOMAC* reconfiguration time is just 0.27% of it. This means *VeSOMAC* would not cause obvious delay overhead when there is a routing path reset-up.

TCP retransmission timeout (seconds)	DSDV routing entry timeout (seconds)	*VeSOMAC* reconfiguration time (seconds)
6	45	0.12

Table 5.3: Time constant of protocols

5.4 Summary and Conclusions

We have adapted *ISOMAC* logic to a Vehicular Self-Organizing MAC (*VeSOMAC*) protocol for distributed TDMA allocation in vehicle-to-vehicle wireless networks. In additional to *ISOMAC*, *VeSOMAC* is designed to be vehicle location and movement aware. Simulation results demonstrate that unlike the 802.11 style contention based protocols, *VeSOMAC* can offer better vehicle safety and data transfer throughput through smaller and bounded packet latency. It has been also shown that the protocol convergence during topology changes is fast including platoon mergers and vehicle passing.

Chapter 6

Dynamic MAC Protocol Switching for Performance Adaptation with Traffic Heterogeneity

6.1 Introduction

6.1.1 Background

Wireless sensor networks are initially motivated by military applications such as intrusion detection, battle-field surveillance and enemy tracking. Recent research shows a rapid increase in the civilian applications like environment data collection, health monitoring, and disaster relief. Although there have been significant recent improvements, maximizing network throughput at the MAC layer in a multi-mission network is still a key design challenge.

The traffic heterogeneity in a multi-mission network can be considered in three categories:

- Variance of data rate: Applications of wireless sensor networks can range from very low data rates such as intrusion detection events, to very high data rate video surveillance. In [80], the author has shown that the data rates in future wireless sensor networks can vary from several bits or bytes per second up to 10 to 100 Mbps. This means that there can be over 100 times spread of working data rates within the same network. Recent applications such as video sensor networks indicate that gradually the share of high data rate applications are expected to become similar to the share of traditional low data rate applications.

- Variance in number of active transmitters: The number of active transmitters can be different for different applications. Applications like environment monitoring, typically involve a large number of deployed nodes sending data simultaneously. Whereas in applications like intrusion detection, only nodes in the vicinity of an intrusion event transmit at a time. This heterogeneity in number of transmitters is unavoidable for multi-mission applications and networks.
- Spatial variance: It is unlikely that in a very large network all sensors will experience the same traffic profile. Instead, multiple and possibly distinct traffic profiles are expected in different parts of a network..

The problem we address in this chapter is: how to develop a MAC layer that can maximize network throughput in the presence of the above traffic and network heterogeneity.

6.1.2 Related work

The MAC layer approaches for wireless sensor networks can be divided into three broad categories: contention-based [9, 22, 48, 49, 81, 82], schedule-based [23, 24, 44, 50, 69, 83], and hybrid [42].

CSMA/CA [48] is an example of the contention based approach. If the channel is sensed busy before any transmission, nodes defer the current transmission for a random back-off period. The RTS/CTS handshake mechanism is used to prevent the exposed and hidden terminal problems. CSMA/CA has the advantages of simplicity, flexibility and robustness. No infrastructure support and clock synchronization is

needed for CSMA/CA protocol to work. The most prominent feature of CSMA/CA is that a node can access all available bandwidth in its neighborhood when needed. However, the medium access collisions and the corresponding unbounded delay is a problem for CSMA/CA and all other protocols in this category, mainly because of their underlying random access.

Comparing with contention based protocols, schedule based protocols like TDMA consists of establishing transmission schedules in a way that no collisions occur, and efficient spatial reuse of the available bandwidth can be achieved. A number of TDMA protocols for sensor networks are proposed in [23, 24, 44, 50, 69, 83]. Despite all the known drawbacks which have been pointed out in the literature, TDMA protocols allocate fixed and guaranteed bandwidth for all nodes in the network. The spatial reuse feature provides efficient bandwidth usage when all nodes in a network transmit at the same time. The collision free feature in the steady state makes sure that all the bandwidth is used for data transmissions rather than packet collisions.

In a hybrid design [42], the authors proposed a protocol called Funneling-MAC. The Funneling-MAC is designed to address the funneling effect near base stations in a sensor network. The funneling effect means that events or data generated in the sensor field travel hop-by-hop in a multipoint-to-point pattern toward one or more sink points. It assumes that the sink nodes are more energy-rich than other sensor nodes. A sink node triggers localized TDMA by a beacon broadcasting process. Nodes that are able to receive the beacon are said to be within an *intensity area*. The

boundary of the *intensity area* is regulated by the transmission power of the beacons from the sink. Through a dynamic depth-tuning algorithm, the throughput can be maximized and the packet loss rate is minimized at the sink point. A sink constantly monitors the traffic that arrives at the sink on a per-aggregated-path manner, calculates the favorable TDMA schedule based on the monitored traffic for all paths, and distributes the schedule by broadcasting a schedule packet at the same transmission power used for beaconing. Nodes in the *intensity area* run an alternate CSMA/TDMA frame, called superframe. The duty cycle of CSMA and TDMA is decided by the schedule computed by the sink. Nodes outside the *intensity area* still run CSMA. They are aware of the localized TDMA schedule by overhearing meta-schedule advertisements containing TDMA scheduling information, which are sent out by nodes in *intensity area* in the first event data packet toward the sink every beacon interval. The problem of how to achieve maximum throughput is alleviated to some extent, but the Funneling-MAC scheme is suitable only for multipoint-to-point applications and not for peer-to-peer traffic.

6.1.3 Proposed Dynamic MAC Protocol Switching

In this chapter, we present a distributed Dynamic MAC Protocol Switching framework, which achieves the goal of maximizing network throughput as in Funneling-MAC but without its restrictions. The key idea behind the proposed Dynamic MAC Protocol Switching is to use neighborhood traffic information to decide the most appropriate MAC protocol under current traffic situation for each active node. Instead of monitoring incoming traffic constantly only by the sink node,

all the nodes monitor the local traffic, compute the appropriate MAC protocol to run, and make decision of protocol switching within their neighborhoods. In fact, the presence of a sink node is not necessary for the proposed mechanism to work. The primary contributions of this chapter are listed as follows:

1. Propose a novel dynamic MAC protocol switching paradigm to address the problem of how to achieve the maximum throughput in a network with traffic heterogeneity.
2. Specify two examples of MAC protocols, CSMA/CA and TDMA, and describe the extensions of both protocols so as to use for co-existence and dynamic MAC protocol switching. The corresponding revision of transmission rules is also stated.
3. Develop the protocol switching criteria, supported with a mathematic model.
4. Experimentally validate the concept of dynamic MAC protocol switching.

The structure of this chapter is as follows. In Section 6.2 we explain the features of protocols supporting MAC switching. Two examples are used in this section to elaborate the concept. The detailed modifications and transmission rules of protocol revision are presented. The general MAC protocol switching logic is described in Section 6.3, including description and model of switching criteria, main switching logic and response to topology changes. An experimental evaluation is shown in Section 6.4 to support the proposed paradigm. Finally Section 6.5 states the conclusions and ongoing work.

6.2 Protocols Supporting MAC Switching

6.2.1 Generalized Concept

To generalize, there could be multiple MAC protocols running at different node clusters (see Figure 6.1) with heterogeneous traffic profiles. Conceptually, a typical example of a real-life situation is as follows. The home wireless router communicates with the outside internet through CSMA/CD wired MAC protocol, and at the same time uses 802.11b/g wireless MAC protocol to communicate with home computers. Therefore, the home wireless router operates in a Multi-MAC Operating Mode (MMOM) node. In the context of wireless sensor networks, the MMOM can have three differences from this example. First, the different MAC protocols all have to be wireless. Second, sensor nodes, unlike wireless router in the example, usually are not equipped with multiple interfaces. Various MAC protocols run at individual sensor node have to use the same wireless communication interface. Third, at MMOM nodes, different MAC protocols have to coexist. MMOM node has to bridge different MAC protocols through the same wireless radio interface. If different MAC protocols could not "talk" with each other, the network can be partitioned.

Most of the currently used MAC protocols, however, are not able to communicate with each other. Often times, researchers focus on proposing a single MAC protocol to solve a series of problems. For instance, contention based protocols are proposed to provide satisfactory performance for moderate traffic scenarios, whereas schedule based protocols are proposed to provide performance guarantee in high or congested traffic scenarios. This results in certain modifications of the current protocols so that

they can communicate with each other without changing their individual primary advantages.

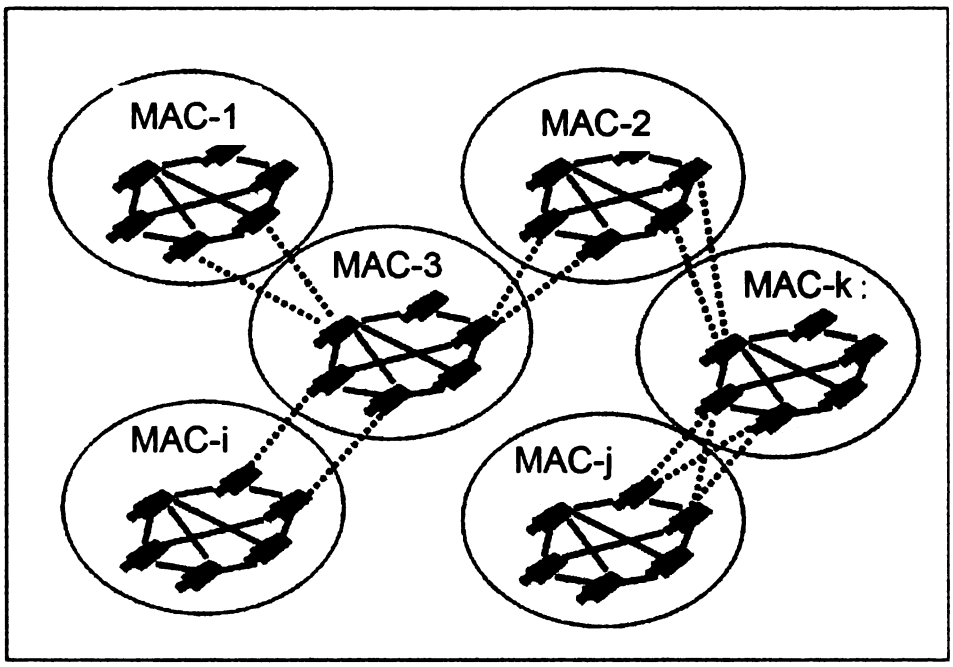


Figure 6.1: MAC variations within networks

6.2.2 Specific Mechanisms for MAC Protocol Switching

In this chapter, coexistence of two MAC protocols, namely a random access CSMA/CA and a schedule based TDMA, are studied. CSMA/CA and TDMA are chosen because they serve as good representations of contention based and schedule based protocols respectively. Current MAC mechanism, both random access [9, 22, 48, 49, 81, 82] and schedule based [23, 24, 44, 50, 69, 83], are not able to communicate to each other because of different "languages" they speak. In order to have two different protocols communicate with each other, smart adaptations of transmission rules to CSMA/CA and TDMA are required. With such adaptations, two protocols are then able to coexist at MMOM nodes. Furthermore, the coexistence also aids the capability of protocol switching. In this section, we

concentrate on the adaptations of transmission rules of CSMA/CA and TDMA to cope with the coexistence problem. The capability of protocol switching will be elaborated in Section 6.3.

6.2.2.1 Adaptations of CSMA/CA for TDMA Coexistence

In general, the basic transmission rules in CSMA/CA are not changed. A successful transmission between two nodes is still through a control message handshake. If a node wants to transmit, it first listens to the channel for a predetermined amount of time so as to check any ongoing activity in the channel. If the channel is "busy", the node will defer its own transmission. If the channel is "idle", the node then sends a Request-to-Send (RTS) message, and a Clear-to-Send (CTS) message will be sent back by the intended receiver. The handshake of RTS and CTS will alert all the nodes within the range of the sender and/or the receiver to maintain silence for the duration of the data packet transmission. Upon receiving the CTS message, the sender finally sends the data message to the receiver. The Acknowledgement (ACK) is not utilized in our CSMA/CA logic, which means that there is no link layer. The other features of CSMA/CA such as exponential back-off and so on are still applied here.

TDMA is well known for its collision-free transmission in steady state. We still want to maintain the collision-free feature of TDMA. Thus, the key idea of the adaptation of the CSMA/CA is to guarantee the collision-free transmission of TDMA nodes in a CSMA/CA nodes neighborhood. The pseudo code for the adapted CSMA/CA logic is shown in Figure 6.2. The new adaptations lie in two categories:

at sender side and at receiver side.

- Sender side: The sender is able to estimate the duration of a successful transmission in terms of RTS, CTS, the data message, and the inter frame space, before the RTS transmission. If it knows the TDMA slots of all its neighbors, then the sender can determine whether the following data transmission will overlap with the neighbors' TDMA slots or not. If there is at least one anticipated overlapping, the sender simply defers transmission till the end of all its neighbors' TDMA slots. Otherwise, the data transmission could carry on as usual.

```
/* Adapted CSMA/CA Logic */
Sender Side:
    Estimate the duration of a successful transmission;
    if ( any overlapping with TDMA neighbors' slots)
        defer transmission till end of neighbors' TDMA slot;
    else // no overlapping within knowledge of sender side
        start RTS transmission;
Receiver Side:
    Upon receiving RTS, check the duration of the intended
    message transmission;
    if ( any overlapping with TDMA neighbors' slots)
        send NTS message back to the sender;
    else // no overlapping within knowledge of receiver side
        start CTS transmission;
```

Figure 6.2: Pseudo code for adapted CSMA/CA logic

- Receiver side. Upon receiving an RTS message, the intended receiver is able to know the duration of the following data message by looking at the RTS NAV duration field of the RTS message. If within the knowledge of the receiver, there are TDMA slots of neighbors overlapping with the following data message, the receiver will send a Not-to-Send (NTS) message, containing the duration of the overlapping period, back to the

sender. Upon receiving the NTS message, the sender will then know when to restart the current data transmission to avoid the neighbors' TDMA slots.

6.2.2.2 Adaptations of TDMA for CSMA/CA Coexistence

Unlike CSMA/CA, TDMA requires more adaptations even for its basic transmission rules. The frame and slot structure of regular TDMA is maintained as usual. Every node has one slot per frame to transmit. The frame is periodic, so the transmission of each node is periodic with periodicity of frame duration. Instead of transmitting data directly within the slot in regular TDMA, we impose RTS/CTS handshake mechanism inside each TDMA slot. Before transmitting the data message, the two way RTS/CTS handshake is used. The RTS/CTS handshake mechanism is different from the one in CSMA/CA in a way that there is no need to sense the channel before transmitting the RTS message. With the new RTS/CTS mechanism inside each slot, TDMA is now able to "talk" to CSMA/CA. Note that even when two nodes use TDMA MAC to communicate, they still use RTS/CTS mechanism inside their respective slots. The pseudo code for the adapted TDMA logic is shown in Figure 6.3.

```
/* Adapted TDMA Logic */
Sender Side:
    if ( in sender's own TDMA slot)
        start RTS transmission and wait for CTS;
Receiver Side:
    if ( receiving RTS)
        send CTS message and wait for data message;
```

Figure 6.3: Pseudo code for adapted TDMA logic

6.2.3 Detailed Transmission Rules

6.2.3.1 Transmission Among Nodes with Same MAC Protocol

Nodes with same MAC protocol refer to nodes within two hop neighborhood which are all running the same MAC protocol. For CSMA/CA, nodes will act like running regular CSMA/CA. There is no need to consider TDMA slots because there are no TDMA MAC neighbors within two hop neighborhood. The modified CSMA/CA performs the same as the regular CSMA/CA. For nodes with TDMA MAC, the RTS/CTS mechanism is still used within the TDMA slot. The throughput performance of modified TDMA is expected to be the same as regular TDMA.

6.2.3.2 Transmissions From TDMA to CSMA/CA Nodes

The transmission from a TDMA node to a CSMA/CA node is schematically elaborated by an example shown in Figure 6.4.

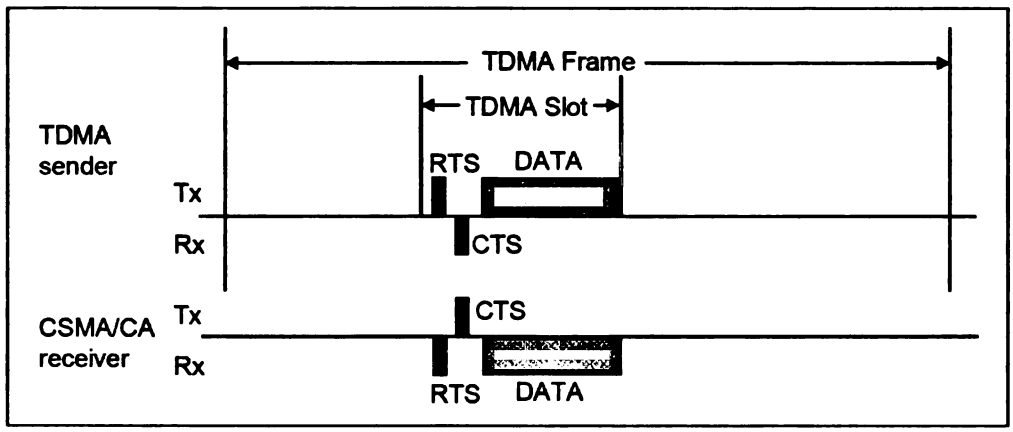


Figure 6.4: Transmission from TDMA node to CSMA/CA node

When a node with TDMA wants to transmit a message to a neighbor which is running CSMA/CA, RTS message will first be sent out in the beginning of the transmission slot. When the CSMA/CA neighbor receives the RTS from the TDMA node, a CTS message will be sent in response. After receiving the CTS message

from the CSMA/CA neighbor, data message transmission follows in the remaining slot time. Since the RTS and CTS message are generally very short compared to the actual data message, the transmission time spent on RTS/CTS handshake should be relatively small. The RTS/CTS exchange at the beginning of each slot of the TDMA nodes is regarded as overhead.

Another issue here is the relationship between the slot time and data message transmission time. If not carefully dimensioned, it is possible that the data message transmission will not be finished before the slot boundary ends. The transmission started in the previous slot could potentially cross over to the next slot, which may cause collision in the following slot. However, the slot size should be large enough to handle at least one message transmission, and we will make assumption that data transmission will not cross the slot boundary. The last thing to note is that TDMA nodes will only transmit messages in their assigned TDMA slot.

6.2.3.3 Transmissions From CSMA/CA to TDMA Nodes

The transmission rules used when a CSMA/CA node transmits to a TDMA neighbor node is more complex than the previous two cases. Ideally, assuming the carrier sense is successful, a CSMA/CA node should be able to transmit when there is a message to be sent without the notion of any slot constraints. However, in this design, when CSMA/CA node transmits to a TDMA neighbor, transmission should not start without any notion of timing. Without such constraints, there is a chance that the transmission for a CSMA/CA node may collide with the transmission of a TDMA node in its assigned slot time.

The collision at the CSMA/CA node is less of a problem because the retransmission mechanism will handle the collision anyway after timeout or back-off. The collision at the TDMA node is a relatively severe problem compared with at the CSMA/CA node. The transmission of TDMA nodes is scheduled once every frame. If there is a collision, transmission has to be deferred until the next frame, which could greatly increase the delivery latency. Thus the collision at the CSMA/CA node is much cheaper than at the TDMA node. This motivates us to give higher priority to TDMA nodes when CSMA/CA nodes try to access the medium at the same time.

The priority is achieved in the following two ways:

(1) CSMA/CA packet transmission should not cross the slot boundary to interfere potential TDMA neighbors' (up to 2-hop) transmission in the next slot. The owner of the next slot can be the neighbor of either the sender, which is a CSMA/CA node in this case, or the receiver, which is a TDMA node. To make sure both cases are handled, the existing message duration in the control message (RTS/CTS) is extended. The logic of the control message duration extension is explained in the following three scenarios.

First Scenario: at the CSMA/CA side, if message transmission including RTS/CTS duration starting from current time is not able to finish message transmission before entering a TDMA neighbor's slot, the CSMA/CA node should defer currently intended message transmission after the TDMA neighbor's slot.

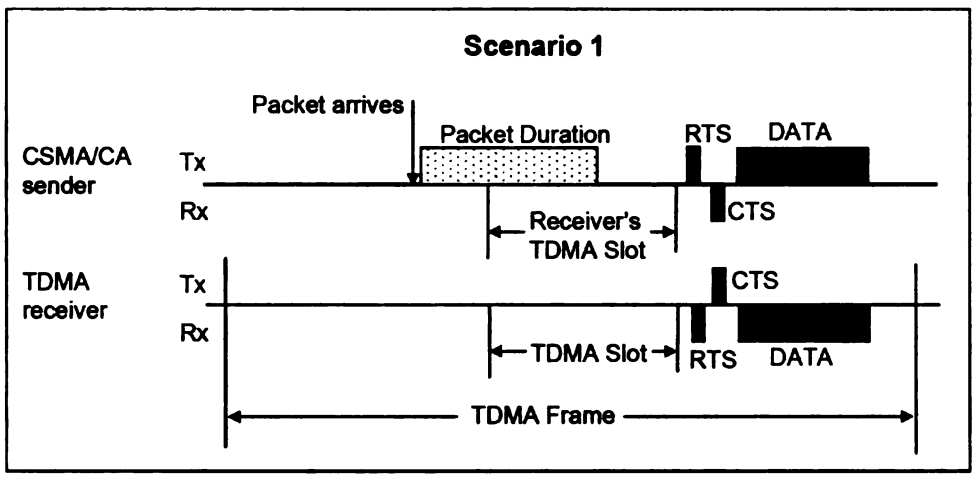


Figure 6.5: Transmission from CSMA/CA node to TDMA node scenario 1

Second Scenario: CSMA/CA node is free to transmit an RTS due to know the message overlapping period. After the RTS reception, if the TDMA node finds out that the interval between current time and the earliest next TDMA neighbor's slot is smaller than the RTS message duration, CTS will not be sent back to the CSMA/CA node. Instead, an NTS (Not-to-Send) message will be sent back to the CSMA/CA node.

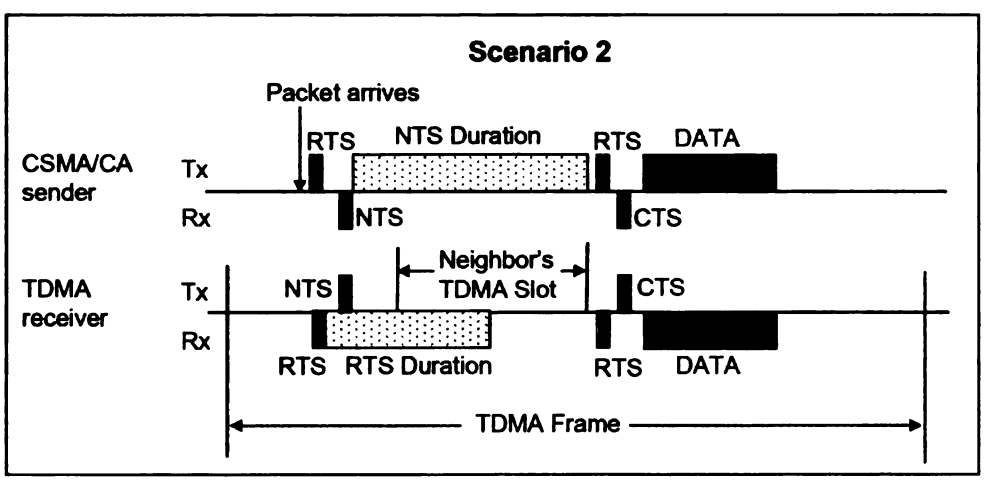


Figure 6.6: Transmission from CSMA/CA node to TDMA node scenario 2

Third Scenario: The TDMA node does not find any overlapping RTS duration and neighbors' TDMA slots. A CTS will be sent back to CSMA/CA node in the

RTS/CTS handshake. Upon CTS reception, data message could be finished if no violation of scenario one. In this case at the sender, there is no clear NTS message reception to ask for deferring. CSMA/CA node still checks the validity of the first scenario when having CTS. This redundancy is to ensure the priority even in the presence of network topology dynamics. The diagrams of all three scenarios are shown in Figure 6.5, 6.6, and 6.7 respectively.

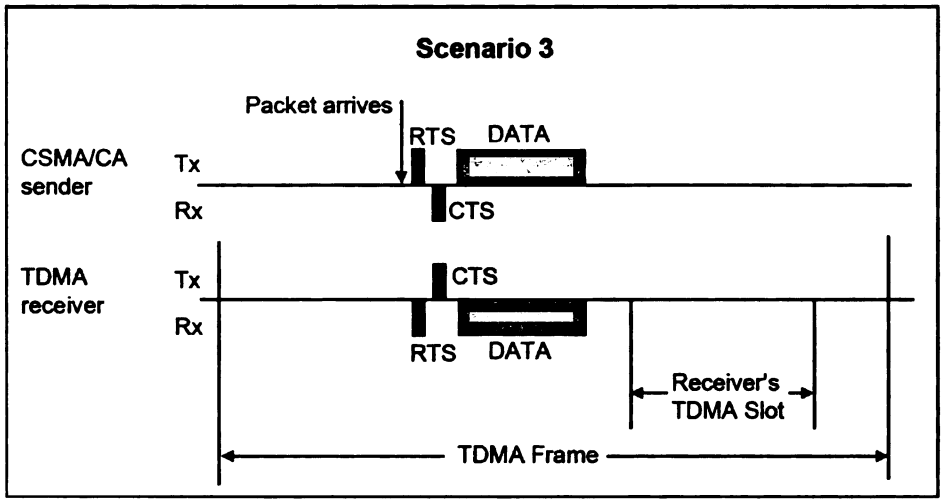


Figure 6.7: Transmission from CSMA/CA node to TDMA node scenario 3

(2) Usage of a TDMA neighbor's slots by a CSMA/CA node is allowed only when the TDMA neighbors do not have anything to send. This is accomplished by having a randomly distributed RTS deferring time for the CSMA/CA nodes when current time is in their TDMA neighbors' slots.

6.3 MAC Protocol Switching Logic

In this section we discuss the detailed design of the proposed protocol switching logic, and issues related to the involving criteria and thresholds.

6.3.1 Switching Criteria

The goal of protocol switching is to be able to provide the best network-wide

throughput irrespective of the heterogeneous traffic profile at different parts of a network. Thus, the criteria for protocol switching have to be decided by the network-wide throughput. If we consider the generic queuing model of MAC protocols, the network-wide throughput includes two distinct types, unsaturated and saturated throughput. For example, for a network with n active nodes and each node is modeled as a simple M/M/1 queue, the unsaturated network-wide throughput is $n \cdot \lambda$, and the saturated network-wide throughput is $n \cdot \mu$, where λ is the data rate at each node and μ is the queue service rate at each node.

The above observations lead to the following conclusions about the network-wide throughput in a fully-connected 1-hop network for both unsaturated and saturated cases. The network-wide throughput of TDMA, as shown in [84], is of the form:

$$S_{TDMA} = f(\lambda, n, F), \tag{6.1}$$

where λ is the generated data rate at each node, n is the number of active nodes, and F is the TDMA frame duration which is fixed for a given network. The performance analysis of CSMA/CA presented in [85-90] shows that the network-wide throughput of CSMA/CA is of the form:

$$S_{CSMA/CA} = g(\lambda, n, W_{\min}, m), \tag{6.2}$$

where λ is the generated data rate at each node, n is the number of active nodes (contenders), $W_{\min}$ is the minimum contention window size, and m is the

maximum back-off stage[6].

The throughput of a given network running protocol switching is:

$$S = S_{TDMA} + S_{CSMA/CA} = f(\lambda, n_{TDMA}, F) + g(\lambda, n_{CSMA/CA}, W_{\min}, m), \qquad (6.3)$$

where n_{TDMA} and $n_{CSMA/CA}$ are the number of nodes running TDMA and CSMA/CA MAC respectively. The combination of n_{TDMA} and $n_{CSMA/CA}$ is the number of active nodes n in whole network. Considering the situation that some of the nodes may not actively send messages all the time, the number of active nodes is not necessarily equal to the network size. Given a fixed network, F, m and $W_{\min}$ are all constants for TDMA and CSMA/CA protocols. Thus, the throughput is decided based on the λ and n values from Equation 6.3. Consequently, if we want to maximize the throughput, λ (generated data rate at each node) and n (number of active nodes) should be the criteria to decide protocol switching.

- Data rate (λ): Generally speaking, larger data rate results in larger throughput. The maximum service rate of TDMA is bounded by the TDMA frame duration, and can be reported as $\mu_{MAX-TDMA}$, which is fixed for a given network. From a single node's perspective, when the generated data rate λ is equal to or less than $\mu_{MAX-TDMA}$, theoretically, the achievable throughput is equal to λ while running TDMA MAC. On the other hand, the "maximum" achievable throughput with CSMA/CA MAC is equal to or

---

[6] In CSMA/CA MAC, a back-off time is uniformly chosen in the range (0, W-1) at each message transmission. The parameter W is called contention window. At the first transmission attempt, w is set to a value $W_{\min}$, and can go up to $W_{\max} = 2^m W_{\min}$ after m unsuccessful transmission.

less than λ, depending on the possible message collisions. To summarize, when the data rate λ is larger than $\mu_{MAX-TDMA}$, the throughput is equal to $\mu_{MAX-TDMA}$ with TDMA, and the throughput can be larger than $\mu_{MAX-TDMA}$ with CSMA/CA.

- Number of active nodes (n): Generally speaking, more active nodes lead to larger throughput. Assume when there is only one node in the network sending at a data rate λ, the throughput of that single active node is $S_{\text{sin}\,gle}$. When there is n active nodes, the network-wide throughput is $n \cdot S_{\text{sin}\,gle-TDMA}$ for TDMA and $n \cdot S_{\text{sin}\,gle-CSMA/CA}$ for CSMA/CA. The difference here is that $S_{\text{sin}\,gle-TDMA} = S_{\text{sin}\,gle}$, but $S_{\text{sin}\,gle-CSMA/CA} \leq S_{\text{sin}\,gle}$. The reason is that for TDMA there are no collisions, thus putting more number of nodes will not affect the throughput of individual nodes. Whereas for CSMA/CA, a contention based protocol, putting more active nodes means introducing more contenders to the same channel. From a single node's perspective, the throughput goes down with increasing number of active nodes.

From the above analysis, in order to boost fidelity of the network, a node should use TDMA MAC when the traffic and node density (λ and n) in its neighborhood is favorable for TDMA and should use CSMA/CA when the parameters λ and n are favorable for CSMA/CA.

6.3.2 Model for Switching Criteria

This section solves the problem as to when nodes should switch their MAC

protocols. From the analysis in Section 6.3.1, the decision of protocol switching is based on the data rate λ and the number of active nodes (contenders) n in the neighborhood. If $\lambda \leq \mu_{MAX-TDMA}$, it was shown in Section 6.3.1 that using TDMA MAC protocol gives better throughput. Therefore the number of contenders (n) in the neighborhood does not have to be considered for this data rate range. As the data rate becomes larger than $\mu_{MAX-TDMA}$, we should consider the number of active nodes n. Increasing n will at the same time shrink the $S_{single-CSMA/CA}$. If we also model CSMA/CA MAC as a queue, the critical point is when the data rate λ is equal to service rate of CSMA/CA ($\mu_{CSMA/CA}$), where $\lambda > \mu_{MAX-TDMA}$. After this critical point, even if we further increase λ or n, the throughput of CSMA/CA will not go up any more. This ceiling effect is because that at this time the throughput has already saturated. The throughput of CSMA/CA is unsaturated before the critical point ($\mu_{CSMA/CA}$).

```
/* MAC Protocol Switching Logic decisions with λ and n */
if ( λ <= μ_MAX-TDMA )
        run TDMA MAC;
else if ( λ <= μ_CSMA/CA and μ_CSMA/CA > μ_MAX-TDMA )
    // λ > μ_MAX-TDMA
    // and for moderate n, μ_CSMA/CA > μ_MAX-TDMA
        run CSMA/CA MAC;
else if ( μ_CSMA/CA <= μ_MAX-TDMA )
        run TDMA MAC;      // large n reduces μ_CSMA/CA
```

Figure 6.8: Pseudo code for protocol switching decision with λ and n

To summarize, the protocol switching decisions are as follows:

- When the data rate λ is not greater than $\mu_{MAX-TDMA}$, nodes should run TDMA;

- When λ is larger than $\mu_{MAX-TDMA}$, nodes should run CSMA/CA till $\lambda = \mu_{CSMA/CA}$;
- Nodes should run TDMA when $\mu_{CSMA/CA} \leq \mu_{TDMA}$ with large n. The corresponding pseudo code is shown in Figure 6.8.

Data rate is usually considered as an application layer parameter. If the decision of MAC protocol switching is based on the data rate, it requires a cross-layer design so that the upper application layer can pass the data rate parameter down to the MAC layer. Another alternative is to try to find a MAC layer substitution parameter for the data rate. The queue utilization, which is of the form $\rho = \lambda / \mu$, is a good candidate for the MAC layer substitution parameter for the data rate.

Considering the utilization ρ the corresponding protocol switching decision become:

- Nodes should run TDMA if $\rho_{TDMA} < 1$, or $\rho_{CSMA/CA} < \frac{\mu_{MAX-TDMA}}{\mu_{CSMA/CA}}$, where $\mu_{MAX-TDMA} < \mu_{CSMA/CA}$.
- Nodes should run CSMA/CA when $\rho_{TDMA} = 1$ and $\rho_{CSMA/CA} > \frac{\mu_{MAX-TDMA}}{\mu_{CSMA/CA}}$, where $\mu_{MAX-TDMA} < \mu_{CSMA/CA}$.
- Nodes should run TDMA when $\rho_{CSMA/CA} = 1$ (i.e. $\mu_{CSMA/CA} \leq \mu_{MAX-TDMA}$).

```
/* MAC Protocol Switching Logic decisions with ρ and n */

if ( ρ TDMA<1 or
   ρCSMA/CA<( μMAX-TDMA/μCSMA/CA ) with μMAX-TDMA< μCSMA/CA )
        run TDMA MAC;
else if ( ρ TDMA=1 and
    ρ CSMA/CA>( μMAX-TDMA/μCSMA/CA ) with μMAX-TDMA< μCSMA/CA)
        run CSMA/CA MAC;
else if ( ρ CSMA/CA=1)
        run TDMA MAC;  // large n makes μCSMA/CA<= μMAX-TDMA
```

Figure 6.9: Pseudo code for protocol switching decision with ρ and n

The numerical value of above decisions can be derived from the theoretical analysis in [85-88] by solving the following series of eight nonlinear equations:

$$\frac{1}{b_{0,0}} = \begin{cases} \dfrac{2pW[1-(2p)^m](1-p)+p(1-2p)(1-p^m)}{2(1-2p)(1-p)} + \dfrac{(2-p_b)(W+1)}{2} + \dfrac{1-p_b}{p_a}, m \le m' \\ \dfrac{2pW[1-(2p)^{m'}](1-p)+p(1-2p)(1-p^{m'})}{2(1-2p)(1-p)} + \dfrac{(2-p_b)(W+1)}{2} + \dfrac{1-p_b}{p_a} \\ \quad + \dfrac{p^{m'+1}(W_{\max}+1)(1-p^{m-m'})}{2(1-p)}, m > m' \end{cases} \tag{6.4}$$

$$\tau = \sum_{i=0}^{m} b_{i,0} = \frac{1-p^{m+1}}{1-p} b_{0,0} \tag{6.5}$$

$$p = 1-(1-\tau)^{N-1} \tag{6.6}$$

$$1/\mu = \frac{1-p-p(2p)^m}{1-2p}\frac{W}{2}\sigma + t_x + t_c\frac{p}{1-p} + \rho(N-1)(t_x + t_c\frac{p}{1-p}) \tag{6.7}$$

$$p_a = 1-e^{-\lambda\sigma} \tag{6.8}$$

$$p_b = \rho = \frac{\lambda}{\mu} \tag{6.9}$$

$$t_x = RTS + SIFS + CTS + SIFS + Header + DATA + DIFS \tag{6.10}$$

$$t_c = RTS + DISF \tag{6.11}$$

where m is the maximum number of retransmissions, m' is the maximum number of

back-off stage, σ is the slot unit duration of CSMA/CA, W is the minimum congestion window size, N is the number of contenders, λ is the data rate and μ is the service rate.

Equation 6.4 is the expression of $b_{i,k}$, the steady-state probability of the active state *(i,k)* when $i=0$ and $k=0$, where i is for the number of retransmissions and k is for the contention window size. Equation 6.5 expresses the probability τ that a node transmits in a randomly chosen time slot. Having the transmission probability τ, a collision happens if at least two nodes transmit at the same time. This gives the collision probability p in Equation 6.6. Equation 6.7 is obtained from [91] by having the collision probability p in Equation 6.6. Equation 6.8 is the expression of the probability that there is at least one frame arriving during the unit time slot σ. Equation 6.9 shows the probability that a node is in a busy period or equivalently. Both Equation 6.8 and 6.9 are written based on the queuing theorem in [92]. It is shown in Equation 6.10 for the average time that the channel is sensed busy due to as successful transmission. Equation 6.11 gives the average time that the channel is sensed busy due to a collision.

6.3.3 Mechanics of Protocol Switching

The general protocol switching logic is shown in Figure 6.10 in the form of a pseudo code. There are three steps which a newly joined node needs to follow for protocol switching logic.

(1) Upon joining the network, a new node first relies on a TDMA logic to choose a TDMA slot for future use. This is just an upfront one time process. At this

stage, the new node should not transmit any valid data messages. At the same time, the new node starts to build the neighbor table, which contains up to 2-hop neighbors' information including the moving average of their queue utilization and number of contenders. The complete information of the moving average of queue utilization and number of contenders is maintained by the new node.

(2) New node can then start with any type of MAC protocols to transmit valid data messages. There is no difference between starting with TDMA and CSMA/CA protocol. In our experiments, we start with CSMA/CA. The pseudo code is also shown with CSMA/CA as the starting MAC protocol.

(3) After starting with CSMA/CA protocol, the new node enters CSMA/CA MAC state which lasts for W message transmissions. During the W message transmissions, the new node will periodically check all its 2-hop neighbors' information about their queue utilizations and numbers of contenders. When W message transmissions are over, if the node finds that the protocol switching decisions are satisfied by at least $D\%$ of the total number of active 2-hop neighbors, it then simply goes to the TDMA MAC state. Otherwise, it assumes that in current network traffic and density situation it is not necessary to switch the protocol. In this case, the node maintains the current CSMA/CA protocol unchanged and starts to check the next W message transmissions.

```
/* MAC Protocol Switching Logic for Each Node */
Initial State:
    Choose a TDMA slot using any TDMA MAC logic;
    Start to maintain a moving average of local queue
    utilization;
    Start to maintain 2-hop neighbor table about
    neighbors' protocol type, queue utilization and their
    number of contenders;
    Start to run CSMA/CA MAC protocol;
CSMA/CA MAC State:
    Periodically check 2-hop neighbors' queue utilization
    & number of contenders for W message transmissions;
    Count the number of 2-hop neighbors, which satisfy
    the protocol switching decisions in Section 6.3.2
    if (W is over && count > D% of total number of 2-hop
        neighbors)
        goto TDMA MAC State;
TDMA MAC State:
    Periodically check 2-hop neighbors' queue utilization
    & number of contenders for W message transmissions;
    Count the number of 2-hop neighbors, which satisfy
    the protocol switching decisions in Section 6.3.2;
    if (W is over && count > D% of total number of 2-hop
        neighbors)
        goto CSMA/CA MAC State;
    If (no more traffic for T seconds)
        goto CSMA/CA MAC State;
```

Figure 6.10: Pseudo code of MAC protocol switching logic

While in the TDMA MAC state, the node still periodically checks 2-hop neighbors' information to see if they satisfy the protocol switching decisions. Like in CSMA/CA MAC state, if there is at least D% out of total number of 2-hop neighbors satisfy the protocol switching decisions, the node then goes to CSMA/CA MAC state. In addition to that, in our experiments we also define that if a node does not have traffic for T seconds, it switches back to CSMA/CA MAC protocol because of low λ.

In the general protocol switching logic, the Initial state is a transient state that only occurs during initial node joining. The transition between CSMA/CA and

TDMA will only happen when there is a trigger because of network traffic heterogeneity or variation as we described in Section 6.3.1.

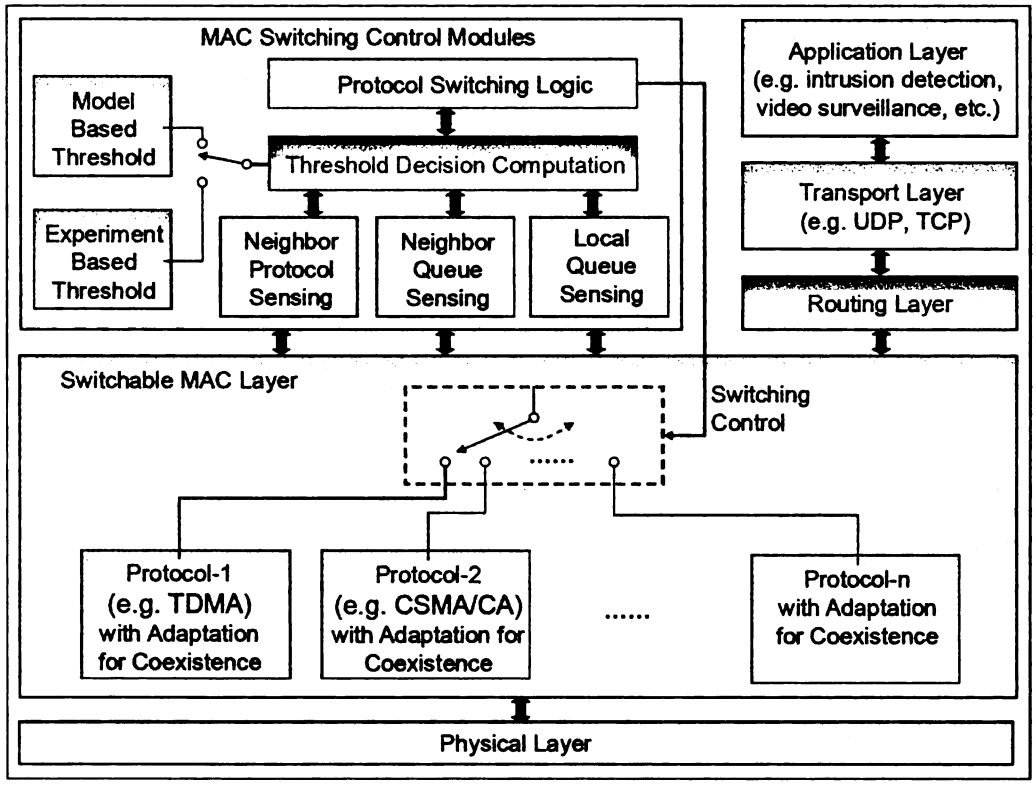


Figure 6.11: Adapted network stack with dynamic MAC protocol switching logic

6.3.4 Response to Network Topology Variations

Network topology variations can occur due to node joining or node leaving a network.

6.3.4.1 Node Joining

When a new node attempts to join an existing network, it listens to the information exchange within its neighborhood. The 2-hop neighbors' information can be inferred from the information exchanges of 1-hop neighbors like the regular TDMA does. If all the neighbors are running TDMA MAC, the new node can simply choose a slot like the ISOMAC logic [73, 74] or any other TDMA logic. However,

when few neighbors may run CSMA/CA, which means that the neighbors may not send anything over a long period. The silent period of the CSMA/CA neighbors could be larger than the listen period of the new node. The new node has to send an explicit notification packet to announce its existence, and then gets acknowledge packets from all its neighbors which run CSMA/CA MAC. The acknowledge packet is not MAC layer ACK message in CSMA/CA. It is the control message which belongs to the protocol switching logic. The neighbors' acknowledge packets contain the information about their assigned TDMA slots. The slot for the new node then can be decided based on all neighbors' slots. After choosing the slot, the new node starts to run CSMA/CA MAC as the protocol switching logic described in Section 6.3.1.

6.3.4.2 Node Leaving

The logic to deal with node leaving is much simpler compared with the one for node joining. Only the neighbor table update is needed within the neighborhood. No additional messages need to be sent. And the transmission rules based on neighbor table will also be adjusted by reclaiming the slot occupied by according to the leaving node.

6.4 Experimental Evaluation

We implemented our protocol switching logic on 100 nodes within NS2 simulator [76]. We choose a radio data rate of 2 Mbps and fixed packet duration of 2 ms, which is also the TDMA slot duration. All considered traffic is 1-hop. All reported performance numbers are based on 1-hop MAC layer traffic. Unless stated otherwise, we fixed the TDMA frame size to 20 slots for all experiments. The

numbers of active nodes and data generation rate have been varied. Before running protocol switching logic, we run *ISOMAC* (see Chapter 3) to get TDMA slot assignment. The other baseline parameters are shown in Table 7.1. The evaluation time (W) and percentage of decision count (D) are the parameters used in Figure 6.10.

TDMA MAC Related	
Frame Size (number of slots)	**20**
TDMA Slot Duration	**2ms**
CSMA/CA MAC Related	
Minimum Congestion Window Size	**32**
Maximum Number of Back-off Stage	**5**
CSMA/CA Slot size	**20us**
Maximum Number of Retransmission	**7**
SIFS Duration	**10 us**
DIFS Duration	**30 us**
Protocol Switching Related	
Evaluation Time	**W=3**
Percentage of Decision Count	**D=55%**

Table 6.1: Baseline parameters of experiment

6.4.1 Validation of Protocol Switching Decision Threshold

6.4.1.1 Protocol Switching Decision Threshold

In Section 6.3.2, we develop a mathematical model to estimate the protocol switching decision threshold. In this section, we illustrate the difference between the threshold obtained from the model and the threshold obtained through experimentation. The experiments reported in this section are for a 20-node fully connected network. All 20 nodes are 1-hop neighbors to each others. For TDMA

MAC protocol, the frame size is set to 20 slots. Here we only show the decision threshold for CSMA/CA, since the decision threshold for TDMA is just the queue utilization value equal to "1" as discussed in Section 6.3.1.

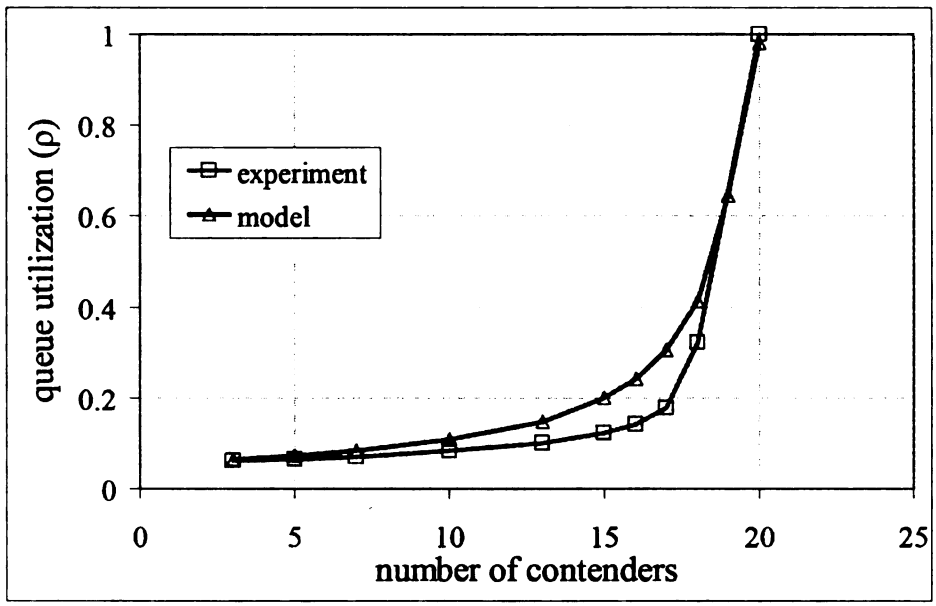


Figure 6.12: Protocol switching decision threshold for CSMA/CA

Figure 6.12 shows the protocol switching decision threshold for CSMA/CA MAC. The y-axis is the queue utilization (ρ), which is used in pseudo code in Figure 6.9 in Section 6.3.2 for switching decision threshold. The curve for model is derived by solving the series of nonlinear equations stated in Section 6.3.3. Because the TDMA frame size is 20 slots and TDMA slot duration is 2 ms, the maximum TDMA service rate $\mu_{MAX-TDMA}$ is 25 pps (packets/second). Therefore, the data rate λ used for CSMA/CA is the same as $\mu_{MAX-TDMA}$, which is 25 pps. The curve for the experiment is obtained by running simulation for this 20-node network with TDMA and CSMA/CA MAC individually. We manually find the points where CSMA/CA is achieving better throughput than TDMA with various number of contenders. The

corresponding queue utilization of those points are recorded and shown in Figure 6.12. As can be seen the queue utilization rises in both cases as the number of contenders increases. This is because the service rate for contention based protocol CSMA/CA ($\mu_{CSMA/CA}$) drops when there are more contenders transmitting at the same data rate (λ). This was discussed and explained in Section 6.3.1 and 6.3.2. The difference between the lines for the model and for the experiment initially becomes large with larger number of contenders, and finally shrinks when both lines meet at the ceiling value of "1". The initially increasing difference between the model and the experiment is caused due to the assumptions made by the model. As stated in [85, 87] the model assumes the collision probability p, and the queue busy probability p_b are independent of the number of contenders. However, as the number of contenders increases, which means that real traffic approaches to saturated scenarios from unsaturated scenarios, this assumption may not hold true. Finally, the shrinking difference is because model and experiment queue utilization are both bounded by the ceiling value of "1". When the model reaches "1", it can not grow larger any more while the experiment keeps approaching "1".

Figure 6.13 shows the last scenario that is discussed in Section 6.3.1, which is the saturated service rate for both MAC protocols. From this point onward, the "exp" in the legend of the graph means the results obtained from experiments, "model" means the results from solving equations in Section 6.3.2, "mesh" means the average results for three different arbitrary mesh multi-hop networks, and "1-hop" means the results from fully connected 1-hop networks. Here, the saturated service rates of both MAC

protocols are in fact the maximum throughputs of both MAC protocols from pervious discussion in Section 6.3.

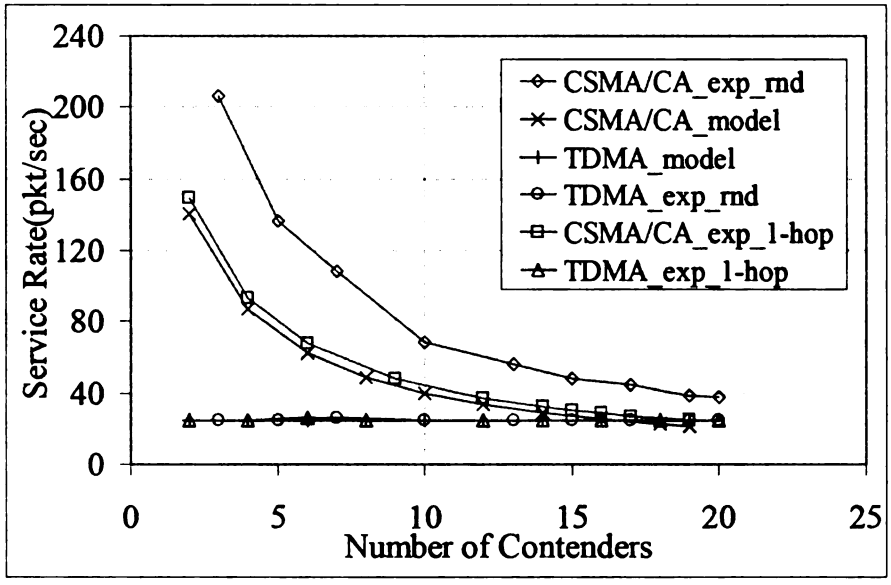


Figure 6.13: Impacts of number of contenders on maximum service rate

From Figure 6.13, the first observation is that when the traffic density (number of contenders) becomes larger than certain value (17 contenders in this case) the saturated service rate of TDMA is larger than that of CSMA/CA. Second, the experimental result in the 1-hop case closely matches the theoretical model result. Third, the experimental result of a 100-node arbitrary mesh network has a larger difference from the model result than the one of 1-hop case. This difference comes from the irregularity of the arbitrary mesh topology. The arbitrariness of node placement causes the number of contenders to be not uniformly distributed across the network. Also, nodes at the edge of the network have a much smaller node degree than nodes in the center. These two factors bring some benefits to CSMA/CA.

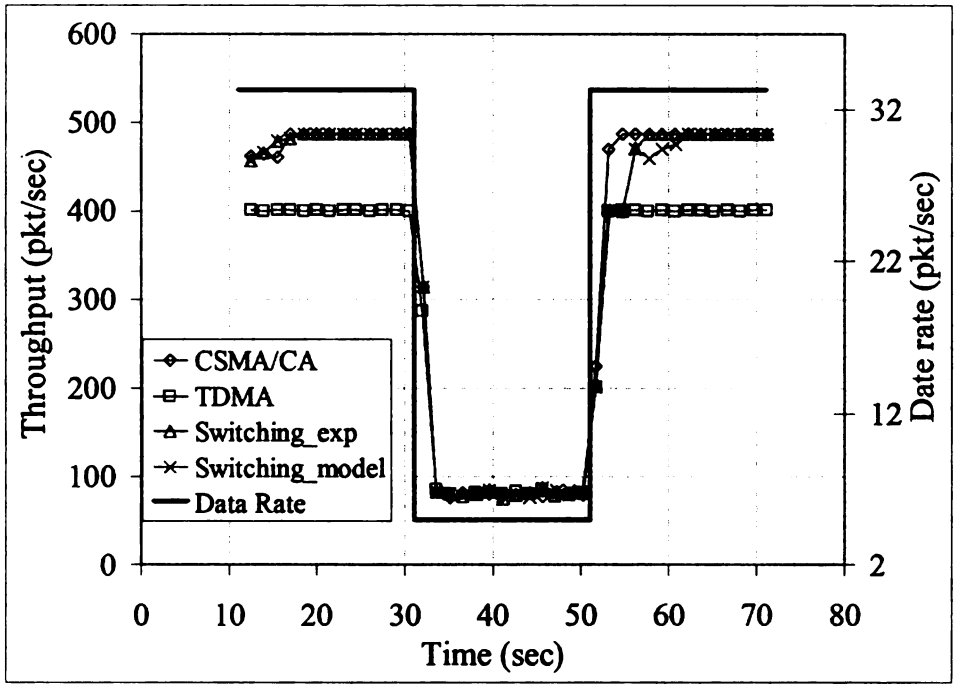


Figure 6.14: Impacts of protocol switching decision thresholds on fully connected network

6.4.1.2 Impacts of Switching Decision Threshold on Fully Connected 1-hop Networks

We use two different protocol switching decision thresholds to evaluate the performance of real-time protocol switching in this section. In Section 6.4.1.1, we see that for the 20-node fully connected network, the largest difference of decision threshold between model and experimental data is when the number of contenders is equal to 16. We thus conduct an experiment in the 20-node fully connected network having 16 active nodes sending 1-hop traffic with variable data rate. Figure 6.14 shows the real-time throughput performance of MAC protocol switching over the 20-node fully connected network. We start our traffic at time 11.0 second with a data rate of 33.3 packets/second, change the data rate to 5 packets/second at time 31.0 second, and finally change back the data rate to 33.3 packets/second.

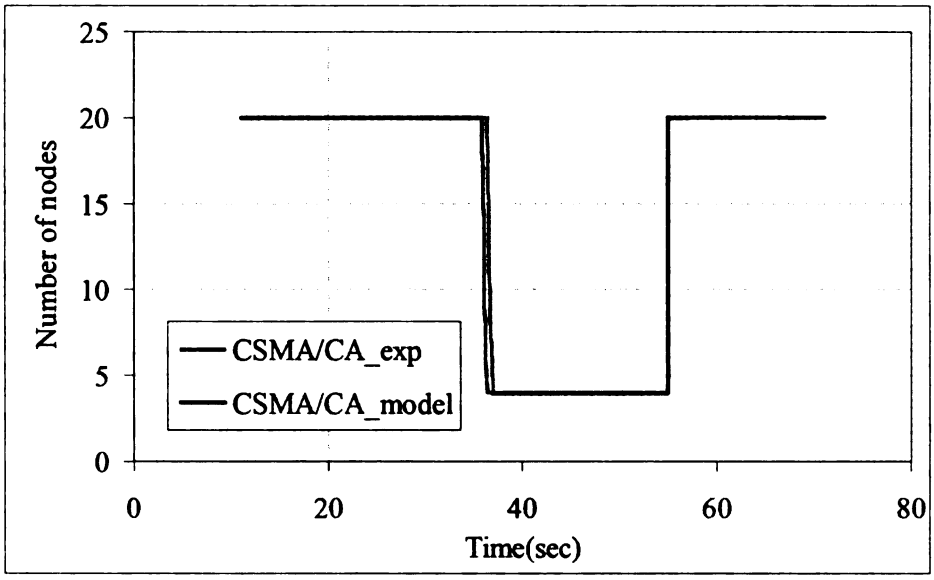


Figure 6.15: MAC protocol dynamics on fully connected networks

As we can see in Figure 6.14, at high rates if running TDMA, the throughput will be limited by the frame size. But CSMA/CA is able to achieve higher throughput than TDMA at high rates. At a lower rate, TDMA is expected to perform better than CSMA/CA because of zero collisions. But the data rate is so low that the benefit of TDMA is not obvious at lower data rates. However, using protocol switching, the real-time throughput is generally able to track the upper envelop of both CSMA/CA and TDMA protocol at both high and low rate scenarios. At high data rates, nodes choose CSMA/CA to obtain larger throughput, and TDMA is chosen at low rates. Note that there is a slight difference in throughput performance when switching logic is obtained based on thresholds from the experiments and from the model.

The difference is further elaborated by using Figure 6.15. Figure 6.15 shows the CSMA/CA protocol dynamics for all the 20 nodes. The y-axis shows the number of nodes running CSMA/CA MAC at any given time. The number of nodes running TDMA MAC can be computed by subtracting the CSMA/CA node count from the

total number of network nodes. As we can see in Figure 6.15, due to the difference in Figure 6.12, the experimental and the theoretical decision thresholds result MAC switching at different times. For example, using model based decision threshold, nodes start to switch to TDMA MAC about 0.5 seconds earlier than using experimental decision threshold. But this half seconds difference is small enough to not to make a difference in the throughput figure as shown in Figure 6.14. Especially when we consider the long term impact of protocol switching, the effect of different decision thresholds can be considered negligible.

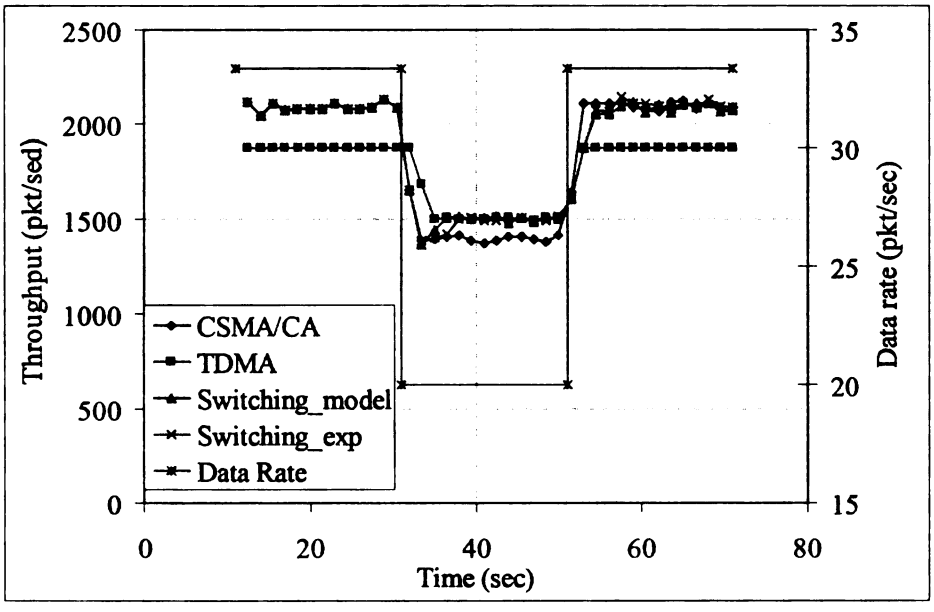


Figure 6.16: Impacts of switching decision thresholds on arbitrary mesh network

6.4.1.3 Impacts of Switching Decision Threshold on Multi-hop Mesh Networks

Experiments are also conducted on a 100-node network with arbitrary mesh topology to evaluate the switching logic with decision thresholds obtained from the model and from experiments. We still maintain the number of contenders within 2-hop neighborhood to be 16. Data rate varies from 33.3 packets/second to 20

packets/second at time 31.0 second and back to 33.3 packets/second at time 51.0 second respectively. Similar to the results shown in Section 6.4.1.2 for the fully connected network, the real-time throughput with protocol switching is able to track the upper throughput envelope for the mesh network scenario as shown in Figure 6.16. At high rates, nodes switch their MAC to CSMA/CA to achieve larger throughput, and change to TDMA to maximize the throughput at low rates.

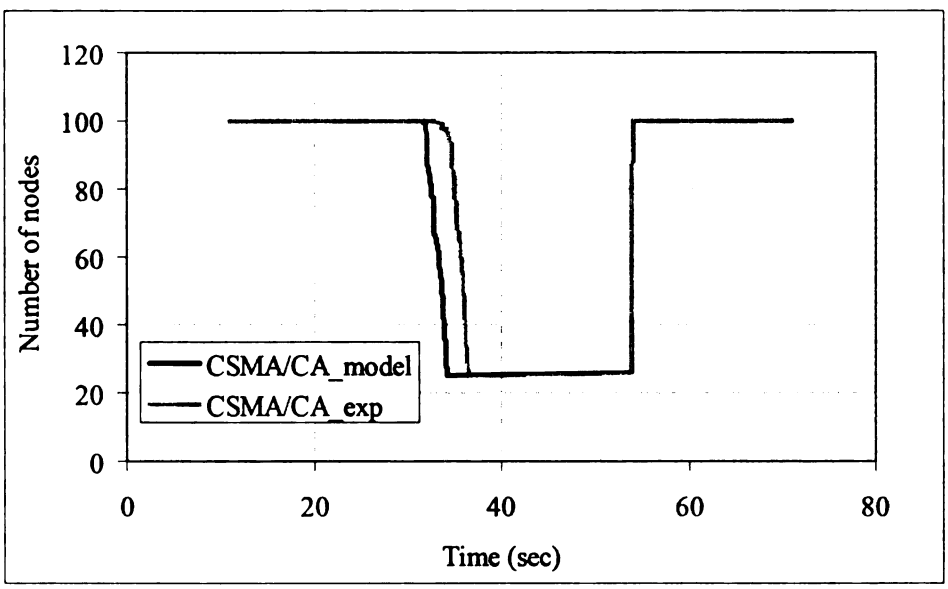


Figure 6.17: MAC protocol dynamics on arbitrary mesh network

In Figure 6.17, we can see that nodes using the experimental threshold are switching their protocol around 2 seconds later than when using model threshold. This explains the approximate 2-second delayed tracking of larger throughput for experiment threshold compared to the model threshold in Figure 6.16. The difference between model and experiment decision threshold is limited only at the point of protocol switching. The latency effect of 2 seconds on upper throughput envelope tracking is negligible compared with the long term benefit of protocol switching.

This is particularly true with the assumption that the switching is relatively infrequent compared to the usually long operating life of practical networks.

6.4.2 General Network Experiments

In this section, all the experiments are conducted in a general arbitrary mesh topology. All the protocol switching results are based on the experimental decision threshold.

6.4.2.1 Impacts of Data Rate

As stated in Section 6.3.1, data rate variance is one of the parameters that affects the MAC throughput. The data rate variance in wireless sensor networks is caused by several factors, such as mission objective changes or detection of special events. In our 100-node arbitrary mesh network, all nodes using the same data rate send to one of their 1-hop neighbors. The data rate varies from 100 packets/second to 20 packets/second at time 21.0 second and reversely at time 31.0 second.

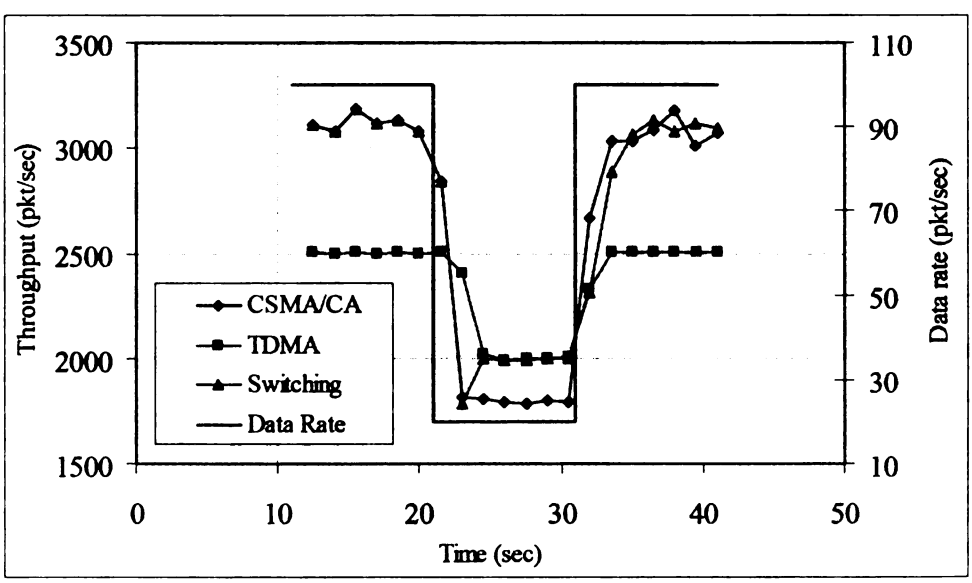


Figure 6.18: Real-time throughput with varying data rate

The real-time throughput values are shown in Figure 6.18. As we can see that for

TDMA, the throughput is limited by the maximum TDMA service rate at high data rates, whereas CSMA/CA is able to provide relatively larger throughput. In a low rate scenario, TDMA provides better throughput due to zero message collisions. Using the protocol switching logic, nodes are able to track CSMA/CA throughput at the high rates and turn to TDMA when the rates are less than the maximum TDMA service rate ($\mu_{MAX-TDMA}$).

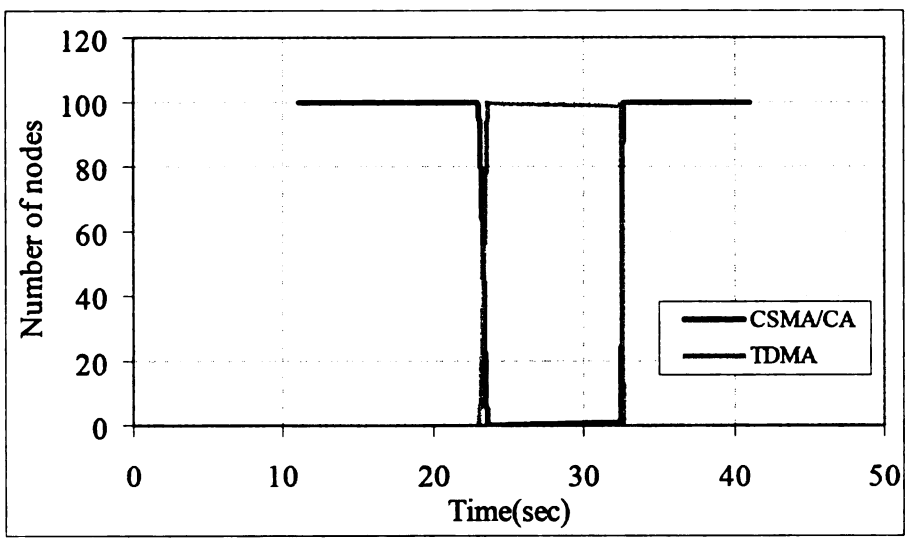


Figure 6.19: MAC protocol dynamics with varying data rate

The dynamic behavior of MAC switching is shown in Figure 6.19. As the data rate changes from high to low, the number of nodes running CSMA/CA reduces and the number of nodes running TDMA rises. Based on the switching decision thresholds represented in Figure 6.12, nodes decide to switch to different protocols based on the queue utilization, which is the threshold we use for switching. When data rate changes, the queue utilization does not change immediately. The queue behavior usually has a lag between rate changes and queue build-up, which is known as the time for the queue to reach the steady state in standard queuing theory [93]. This

explains why usually there is a few seconds delay between the time data rate change and the nodes switch their protocols.

6.4.2.2 Impacts of Number of Contender Nodes

Besides the data rate, the number of contenders is also a factor that affects the network throughput performance. An experiment is conducted to study the impacts of variable number of contenders on the performance of protocol switching. As shown in Figure 6.20, the number of active nodes starts from 100, changes to 25 at time 21.0 second, and turns back to 100 at time 31.0 second. When there are only 25 active nodes, they are randomly selected out of all 100 nodes. The number of contenders in a neighborhood is accordingly changed by varying the number of active nodes. All active nodes send data to all 1-hop neighbors at rate of 35 packets/second.

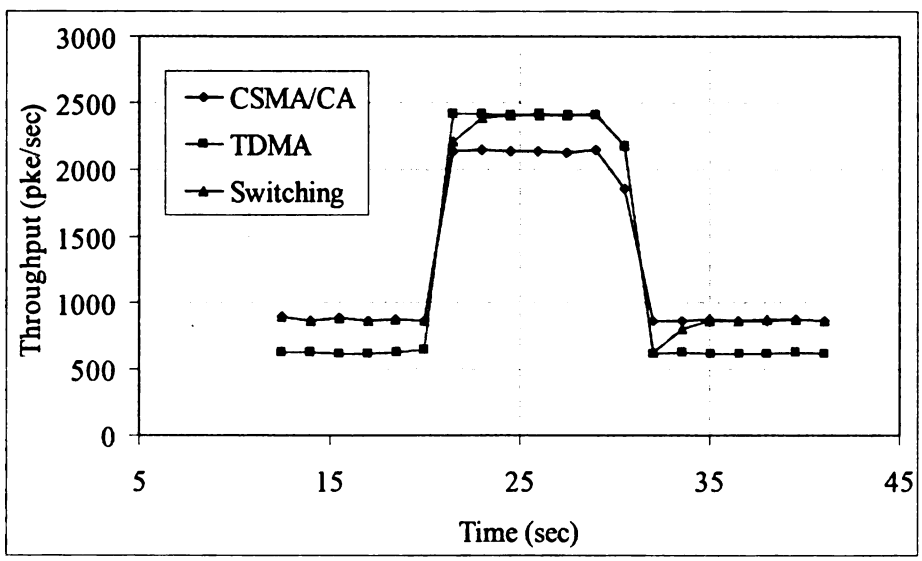


Figure 6.20: Real-time throughput with varying number of contenders

According to the graphs in Figure 6.20, CSMA/CA performs better when there are less number of contenders because of its larger service rate in the absence of

collisions as we describe in Section 6.3.1. When the number of contenders increases, the contention brings down the service rate of CSMA/CA. Observe that the protocol switching logic is able to track the throughput upper envelope of both the protocols, providing larger throughput in a manner that is independent of the number of contenders.

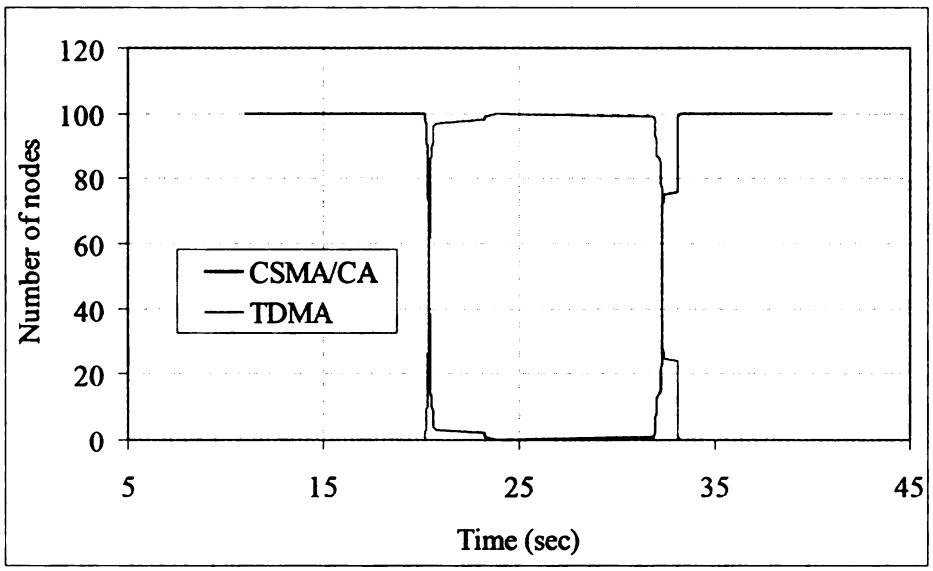


Figure 6.21: MAC protocol dynamics with varying number of contenders

The dynamics of protocol distribution is shown in Figure 6.21. Like the results shown before, the switching of the protocol does not happen at the same time for all the active nodes. There is an approximate two seconds delay for all the active nodes before they make the decision for switching.

6.4.2.3 Impacts of Spatially Clustered Traffic Profiles

In the previous experiments, the switching logic causes all the active nodes in a network to switch their protocols to the same type. The scenarios describe in this section, active nodes in a network can run different MAC protocols and switch to different protocols when there is spatial traffic heterogeneity. The experiment here is

conducted in a rectangular field with an arbitrary mesh network topology consisting of 400 active nodes. The varying traffic profile is shown in Figure 6.22. The traffic distribution is changed from Profile-(1) to Profile-(2) at time 31.0 second. In high traffic density region, a node sends to all its 1-hop neighbors each with a data rate of 3 packets/second. Nodes in the medium traffic density region send to only one 1-hop neighbor at a data rate of 140 packets/second. Nodes in the low traffic density region send to only one 1-hop neighbor at a rate of 12.5 packets/second.

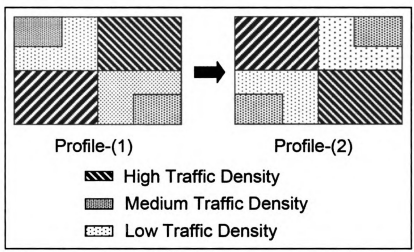


Figure 6.22: Pictorial diagram of spatial traffic variation

A node in the high traffic density region is expected to run TDMA to achieve higher throughput, because the CSMA/CA service rate is less than the TDMA service rate at this traffic level (due to data rate and number of contenders). CSMA/CA is expected in a medium traffic region. TDMA is then favored in a low density region.

The real-time throughput result is shown in Figure 6.23. The scenario with switching enable achieves larger throughput than both scenarios in which the TDMA or CSMA/CA protocol was run individually in the entire network. The reason is that

with activated protocol switching, TDMA is able to be used in various traffic density regions when needed. So is CSMA/CA. If only one protocol is used in the entire network, the network-wide throughput is affected either by the throughput in high traffic density region or in low traffic density region. With protocol switching enabled, it is possible to provide the best achievable throughput at both high and low traffic density regions. When the traffic profile is flipped over completely (from profile (1) to profile (2) in Figure 6.22), the switching logic is able to adjust appropriate protocols to accommodate the changes.

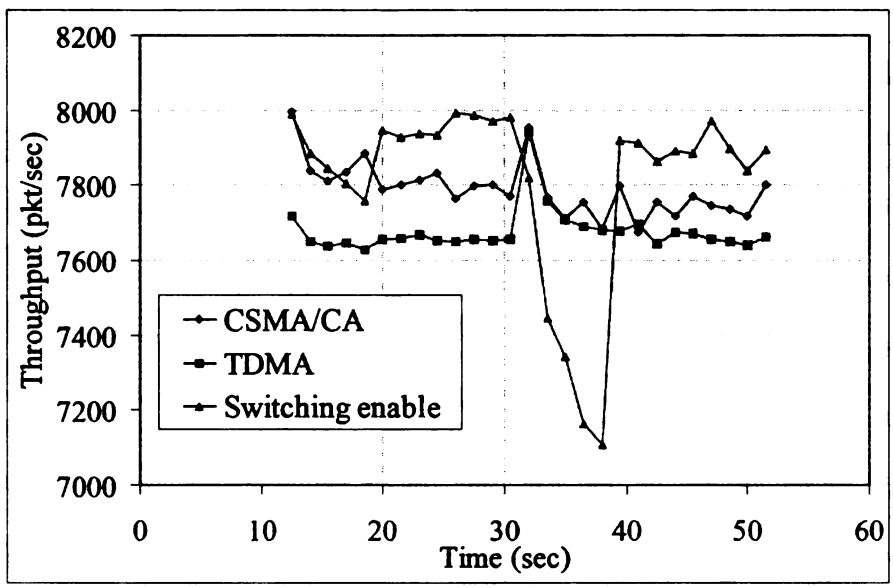


Figure 6.23: Real-time throughput with varying spatial traffic profile

Results in Figure 6.23 show the adaptation process due to switching. Note that during the actual switching, there is a temporary throughput drop. While the protocols of nodes are switching, certain packets experience collisions, and thus MAC layer drops and queue drops give rise to reduced service rate. After the transient state is over, the throughput benefit of switching resumes.

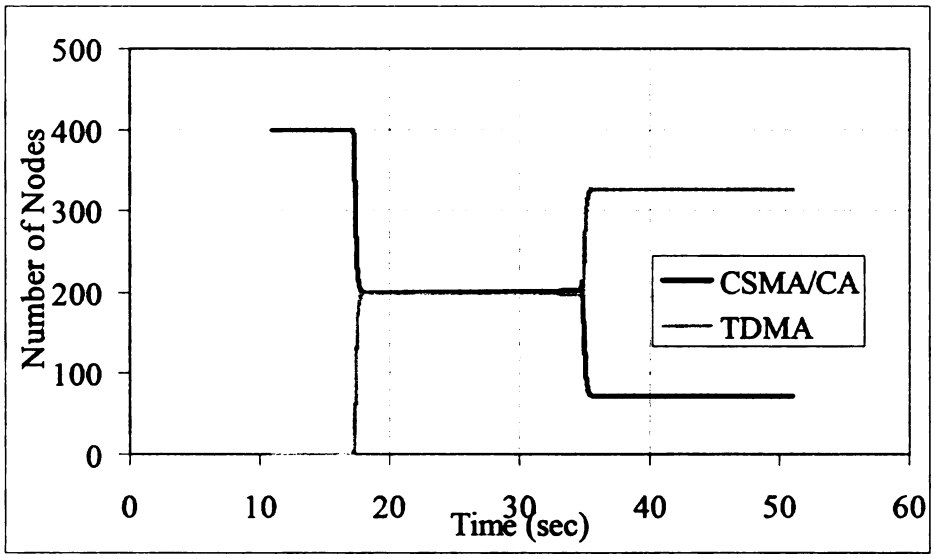


Figure 6.24: MAC protocol dynamics with varying spatial traffic profile

Two observations can be made from the reported in Figure 6.24. First, nodes change their protocols in a period of few seconds after the traffic profile actually changes. Second, the protocol distribution is different in these two traffic density profiles across time. During the time that the network is working in the first traffic density profile, the number of nodes running TDMA is equal to the number of nodes running CSMA/CA. However, during the time running the second traffic density profile, there are 328 nodes running TDMA and 72 nodes running CSMA/CA. The variance of network MAC protocol distribution in two traffic density profiles is coming from the irregularity of the network connectivity, although the traffic pattern is similar to each other in both traffic scenarios. Each node changes its protocol based on the switching decision threshold shown in Figure 6.9 and the logic in Figure 6.10. Due to the topology irregularity, even nodes in the same traffic density region experience different number of contender nodes (n) and queue utilization (ρ), which are used in the switching decision threshold and the switching logic. The

different views of n and ρ cause nodes to run different protocols as expected. For example, some nodes in the medium traffic density region run TDMA instead of expected CSMA/CA.

6.5 Conclusions and Ongoing Work

We present a distributed Dynamic MAC Protocol Switching paradigm, which achieves the goal of maximizing network throughput in the presence of traffic and topology heterogeneity. The key idea behinds Dynamic MAC Protocol Switching is to use local neighborhood information and traffic density to decide the most favorable MAC protocol under current traffic conditions for each active node. Instead of only the sink node constantly monitoring incoming traffic as in the protocol Funneling-MAC [42], each node in the proposed scheme constantly monitors the local traffic, computes the most favorable MAC protocol, and makes decision on switching within its neighborhoods. Simulation experiments demonstrate that with the novel dynamic protocol switching logic, the MAC layer is able to achieve maximum throughput in response to the network traffic heterogeneity, including data rate, number of contenders and traffic spatial patterns.

Ongoing work includes refining transmission rules for CSMA/CA to achieve higher throughput in Multi-MAC Operating Mode. For examples, the transmission rules can include a smarter deferring mechanism when overlapping with TDMA slots, and giving TDMA priority with less than one probability. We also intend to study the processes and effects of protocol switching notification. Finally new protocol switching criteria could be introduced to have better performance.

Chapter 7

Summary and Future Work

7.1 Summary

In this thesis we present a MAC self organization framework for both intra-MAC and inter-MAC for wireless ad hoc and sensor networks. These frameworks are developed to address the specific design issues raised by unique constraints of the wireless ad hoc and sensor networks. Such constraints include ad hoc deployment, energy limitation, topology changes due to mobility, infrastructure failure, communication errors and traffic heterogeneity.

The objective of the proposed *ISOMAC* is to address the intra-MAC self organization problems with energy limitations, network dynamics, and ad hoc deployment. In *ISOMAC*, a fixed length bitmap vector is used in each packet header for exchanging relative slot timing information across immediate and up to 2-hop neighbors. It is shown that by avoiding explicit timing information exchange, *ISOMAC* can work without network-wide time synchronization which can be prohibitive for severely cost-constrained sensor nodes in very large networks. Through simulation experiments it was shown that *ISOMAC's* performance without time synchronization is just marginally worse than that with synchronization. Results demonstrate that with in-band bitmap vectors of moderate length, *ISOMAC* converges reasonably quickly - approximately within 4 to 8 TDMA frame duration. Also, if the bitmap is restricted within 10% of packet duration, the energy penalty of the in-band information is quite negligible.

In Chapter 4, a Minimized Slot Misordered Routing (MSMR) protocol has been proposed for end-to-end delay mitigation in sensor networks with TDMA MAC. Different from the existing D-MAC [34], which resolves the problem by delay-optimized slot allocation based on pre-set routes, MSMR seeks an opposite point of view by finding the delay-optimized routing based on pre-set TDMA slot allocation. MSMR has been targeted towards delay sensitive sensor network applications such as tactical surveillance, intrusion detection, and industrial process monitoring. Delay reduction in MSMR is accomplished by computing least cost routes with a link cost formulation based on the degree of misordering of the TDMA slots of nodes across a link. It has been shown that with realistic TDMA models and sensor event generation rates, in a 5000-node sensor network, MSMR can reduce the reporting delay of emergency events by up to 3.5 seconds, which is a significant amount of time for possible remedial actions. We have also shown that while controlling end-to-end delay due to slot misordering, at higher event rates MSMR can give rise to undesirable queuing delay as a result of congestions. We propose a centralized *Balanced* MSMR routing protocol that can strike a balance between queuing delay and slot misordering delay by adjusting link costs based on instantaneous node level congestions.

We adapt the concept of *ISOMAC* logic in the context of vehicular networks and propose the Vehicular Self-Organizing MAC (*VeSOMAC*) protocol in Chapter 5. *VeSOMAC* is designed to be vehicle location and movement aware so that the MAC TDMA slots in a vehicle platoon can be time ordered based on the vehicles' relative

locations. Simulation results of both highway and urban intersection scenarios demonstrate that unlike the 802.11 style contention based protocols, *VeSOMAC* can offer better vehicle safety and data transfer throughput through smaller and bounded packet latency. It has been also shown that the protocol convergence during topology changes is fast including platoon mergers and vehicle passing.

In Chapter 6, a distributed Dynamic MAC Protocol Switching logic is presented. The key idea behind our Dynamic Protocol Switching is to use local neighborhood information to decide the most favorable MAC protocol under current traffic conditions for each active node. Each node constantly monitors the local traffic, computes the most favorable MAC protocol, and makes decision on protocol switching within its neighborhood. The experiments demonstrate that with the novel dynamic MAC protocol switching logic, the MAC layer is able to enhance throughput in response to the network traffic heterogeneity, including data rate, and the number of contenders.

7.2 Future Work

Here, we highlight some of the key items tat we plan to pursue in the future:

- Develop a theoretical approach to prove the convergence of the asynchronous *ISOMAC* protocol using some methods from the multi-agent system[94-105] in embedded control system theories [106-108].
- For our proposed centralized Balanced *MSMR* routing protocol, future work includes developing a distributed Balanced MSMR protocol. The goal is to avoid network wide information flooding and to let the sensor nodes compute

the delay optimal routing locally.

- The extensions of the *VeSOMAC* framework include border ITS scenarios involving vehicle-to-roadside communication, bi-directional and multiple highway lanes, and the presence of multiple simultaneous emergency events. We also intend to evaluate it in the presence of more detailed physical layer models including fading and multi-paths. Finally, variable traffic rates will be introduced in *VeSOMAC* by allocating multiple slots to a vehicle in each *TDMA* frame.
- Ongoing work of Dynamic Protocol Switching includes refining transmission rules for CSMA/CA to achieve higher throughput in Multi-MAC Operating Mode. For examples, the transmission rules can include a smarter deferring mechanism when overlapping with TDMA slots, and giving TDMA priority with less than one probability. We also intend to study the processes and effects of protocol switching notification. Finally new protocol switching criteria could be introduced to have better performance.

Appendix A

A Model for *ISOMAC* Convergence in Linear Networks

We construct an analytical model for *ISOMAC's* convergence in a linear network as described in Section 3.3.2.5. The purpose of this model is to validate the experimental convergence results presented in Figure 3.6:b. Consider the linear topology in Figure A.1:a, in which there are two disconnected physical clusters $\{a_1, a_2, \ldots\ldots a_n\}$ and . Within each cluster, nodes are in allocation steady state, and as shown in Figure A.1:b, the inter-slot timing separation between connecting nodes is kept fixed at 3 slots, which is what was used for the experiments described in Section 3.6.2. Also, it is assumed that the separation of slots between the two clusters is larger than the bitmap size.

We study the allocation dynamics after node a_0 joins right in between the clusters so that they are forced to merge to a single connected graph. When a_0 joins, according to *ISOMAC*-A described in Section 3.3.2.6, a_0 will choose a slot right at the middle of those of a'_1 and a_1. Since in this model the topology and slot allocation are symmetric across the clusters, and the slot of a_0 never moves, we analyze the slot movement by observing it only in one cluster. In Figure A.1:c, slot movements of the right hand cluster is demonstrated, keeping the slot location of a_0 fixed.

After a_0 joins the network in step-0, the separation between slots of a_0 and a_0, a_1, a_2, ... a_n are defined as $a_0(0)$, $a_1(0)$, $a_2(0)$, ... $a_n(0)$. In step-1, in order to satisfy the bitmap constraint, a_0, a_1, a_2, ... a_n.

sequentially pick up new slots. After step-1, the separation durations are defined as $a_0(1)$, $a_1(1)$, $a_2(1)$, ... $a_n(1)$. To state generally, the separation between node a_0 and node a_i in step j is $a_i(j)$. With the stated initial inter-node separation of 3 slots, we can write $a_0(0)=0$, $a_i(0)=T+3\times i$, for i= 1, 2,... n, where T is the half of the initial timing separation between the two disconnected clusters, i.e. $T=a_1(0)-a_0(0)$.

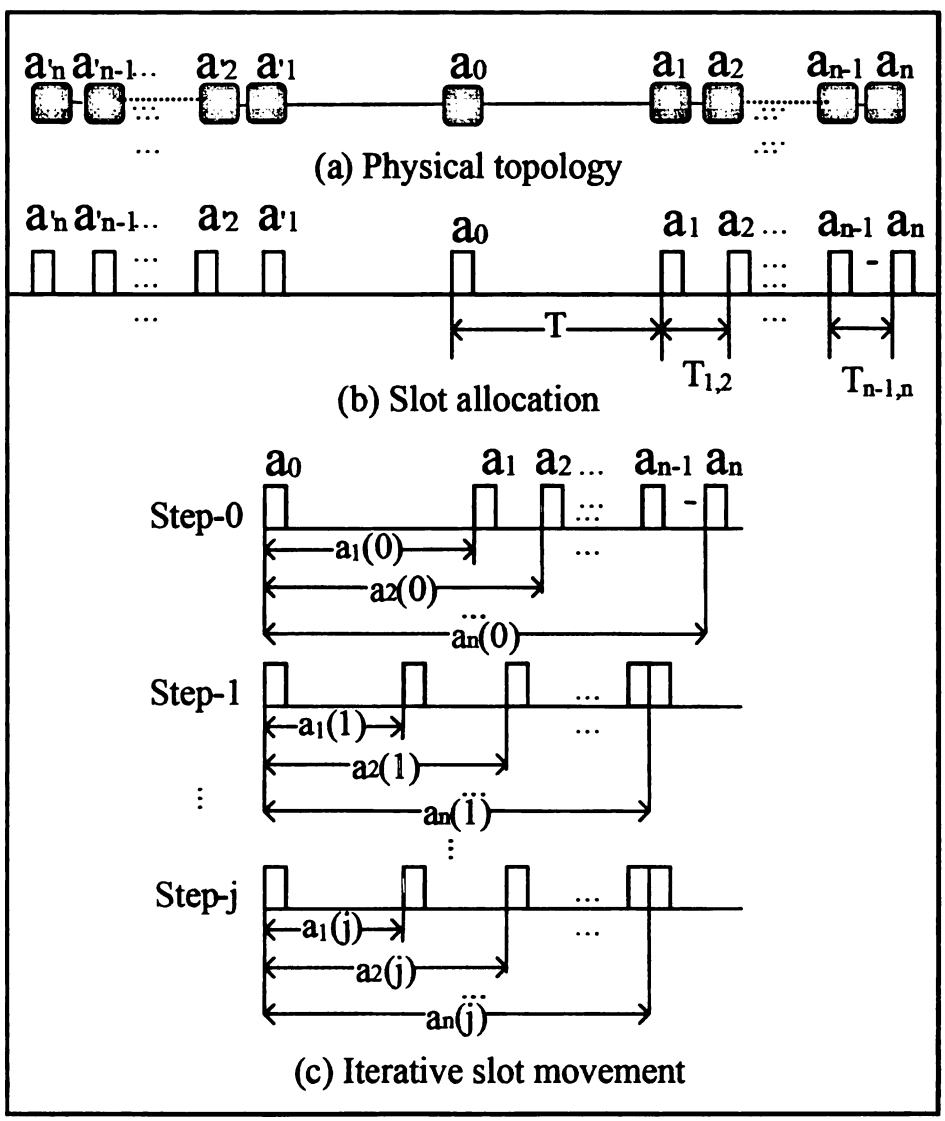


Figure A.1: Convergence model in linear networks

According to the *ISOMAC*-A protocol, as described in Section 3.3.2.6, the

quantity $a_i(j)$ will reduce, although non-monotonically, with increasing iteration step *j*. We make a simplifying assumption that the edge node a_n will always follow node a_{n-1} 's slot at each allocation iteration step. This implies $a_n(j) = a_{n-1}(j)$ for all *j*. With this assumption, the quantity $a_i(j)$ becomes monotonically decreasing and the protocol becomes analytically tractable. Now, $a_i(j)$ can be written as: $a_0(j) = 0, a_i(j) = \frac{1}{2}(a_{i-1}(j) + a_{i+1}(j-1))$ .... (A.1), for i = 1, 2,.... n-1. With this iteration model, the protocol is said to have converged when the condition $a_1(j) < B$ is true. This is because at any protocol step-*j*, $a_1(j)$ represents the largest inter slot separation between any two connecting nodes in the cluster. Therefore, if this separation satisfies the bitmap constraint then all $a_i(j)$ values will satisfy the bitmap constraint, thus the protocol will have converged.

We use the iterative expression in Equation A.1 to compute the convergence time from the model, and compare that with experimental results for different cluster sizes in Figure 3.6:b. Observe that the experimental results are always slightly larger that the values from the model.

The following assumptions in the model account for this difference. The first assumption that $a_n(j) = a_{n-1}(j)$ in the model results in a faster convergence than what happens in the experiments. This is because in the experiments the edge node does not always move. When the distance between edge node and its neighbors is smaller than the bitmap range, the bitmap constraint is satisfied for the edge node and therefore, it does not move. This slows down the rate of reduction of the inter

cluster separation, which is the speed of convergence in this scenario.

Another assumption here is that nodes' slot movements at two sides of the center node a_0 are symmetric, which means slots used by the two clusters are non-overlapping. In the experiments, however, this may not always be true. For example, when n=1, two nodes at each side may end up picking an overlapped slot, which would require four more frames for resolving such a collision. The collision probability will decrease with larger bitmaps. That is why with increasing bitmaps, the experimental results have better agreement with this model (Figure 3.6:b).

Bibliography

[1] P. Gupta and P. R. Kumar, "Capacity of wireless networks," IEEE Transactions on Information Theory, vol. 64, pp. 388 - 404, March 2000.

[2] J. Li, C. Blake, D. S. J. D. Couto, H. I. Lee, and R. Morris, "Capacity of Ad Hoc Wireless Networks," in the 7th ACM International Conference on Mobile Computing and Networking, Rome, Italy, July 2001.

[3] "Mote Documentation and Development Information," http://www.cs.berkeley.edu/~awoo/smartdust.

[4] www.xbow.com.

[5] I. F. Akyildiz, W. Su, and Y. Sankarasubramaniam, "A Survey on Sensor Networks," in IEEE Communications Magazine. vol. 40, 2002, pp. 102--114.

[6] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," Computer Networks: The International Journal of Computer and Telecommunications Networking, vol. 38, pp. 393-422, March 2002.

[7] H. Karl and A. Willig, "A short survey of wireless sensor networks," August 2003.

[8] N. Xu, "A Survey of Sensor Network Applications," University of Southern California 2003.

[9] W. Ye, J. Heidemann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," in INFOCOM, June 2002, pp. 1567-1576.

[10] S. Singh and C. S. Raghavendra, "PAMAS: Power aware multi-access protocol with signaling for ad hoc networks," ACM Computer Communication Review, vol. 28, pp. 5-26, July 1998.

[11] J. Rao and S. Biswas, "Biologically Inspired Mobility Models for Energy Conservation in Sensor Networks," in The Second IEEE Workshop on Embedded Networked Sensors,, Australia, 2005.

[12] J. Rao and S. Biswas, "Controlled Node Mobility: A Mechansim for Energy Load Distribution in Sensor Networks," in IEEE Globecom 2006, 2006.

[13] N. Bulusu, J. Heidemann, and a. D. Estrin, "GPS-less Low Cost Out Door Localization for Very Small Devices," Technical report 00729, Computer Science Department, USC April 2000.

[14] S. Gage, "Computational Ecology and Visualization Laboratory," http://envirosonic.cevl.msu.edu/Home.asp.

[15] S. Kumar, V. S. Raghavan, and J. Deng, " Medium Access Control Protocols for Ad-Hoc Wireless Networks: A Survey," Elsevier Ad-Hoc Networks Journal, vol. 4, pp. 326-358, May 2006.

[16] W. Ye and J. Heidemann, "Medium Access Control in Wireless Sensor Networks," Information Sciences Institute, University of Southern California October, 2003.

[17] A. Tanenbaum, Computer Networks, 4 ed.: Prentice Hall PTR, 2002.

[18] I. Demirkol, C. Ersoy, and Fatih Alagöz, "MAC Protocols for Wireless Sensor Networks: A Survey," in IEEE Communication Magazine, April, 2006.

[19] P. Naik and K. M. Sivalingam, "A survey of MAC protocols for sensor networks," in Wireless sensor networks, 2004, pp. 93 - 107.

[20] C. Gershenson, "Design and Control of Self-organizing Systems." vol. PhD: Vrije Universiteit Brussel, 2007.

[21] C. Gershenson, "A general methodology for designing self-organizing systems," ECCO, Vrije Universiteit Brussel May 2005.

[22] T. Dam and K. Langendoen, "An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks," in 1st ACM International Conference on Embedded Networked Sensor Systems, Nov. 2003, pp. 171-180.

[23] V. Rajendran, K. Obraczka, and J. J. Garcia-Luna-Aceves, "Energy-Efficient, Collision-Free Medium Access Control for Wireless Sensor Networks," in International Conference on Embedded Networked Sensor Systems, Nov. 2003, pp. 181-192.

[24] Z. Chen and A. Khokhar, "Self organization and Energy Efficient TDMA MAC Protocol by Wake up for Wireless Sensor Networks," in 1st Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks 2004 (IEEE SECON' 04), Oct. 2004, pp. 335-341.

[25] D. Bertsekas and R. Gallager, Data Networks, 2nd ed.: Prentice Hall, 1992.

[26] H. Karvonen and C. Pomalaza-Raez, "A cross layer design of coding and awake/sleep periods in WSNS," in the 17th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC '06), Helsinki, Finland, September 2006, pp. 1-5.

[27] C. Buratti, A. Giorgetti, and R. Verdone, "Cross-layer design of an energy-efficient cluster formation algorithm with carrier-sensing multiple access for wireless sensor networks," EURASIP Journal on Wireless Communications and Networking vol. 5, pp. 672 - 685, October 2005.

[28] V. Srivastava and M. Motani, "Cross-layer design: a survey and the road ahead," IEEE Communications Magazine, vol. 43, pp. 112- 119, December 2005.

[29] S. Cui, R. Madan, A. J. Goldsmith, and S. Lall., "Cross-layer Energy and Delay Optimization in Small-scale Sensor Networks," IEEE Transactions on Wireless Communications, vol. 6, pp. 3688--3699, 2007.

[30] C.-F. Chou and K.-T. Chuang, "CoLaNet: a cross-layer design of energy-efficient wireless sensor networks," in IEEE Systems Communications (ICW '05), Montreal, Canada, August 2..5, pp. 364- 369.

[31] H. Kwon, T. H. Kim, S. Choi, and B. G. Lee, "A cross-layer strategy for energy-efficient reliable delivery in wireless sensor networks," IEEE Transactions on Wireless Communications, vol. 5, pp. 3689–3699, 2006.

[32] X. Lu, Y. Sun, J. Yu, and W. Yan, "An energy efficient cross-layer routing algorithm for wireless sensor networks," in the 1st IEEE Conference on Industrial Electronics and Applications (ICIEA '06), Singapore, May 2006, pp. 1-5.

[33] I.-Y. Kong and W.-J. Hwang, "Lifetime maximization by cross-layer interaction in wireless sensor networks," in the 8th International Conference Advanced Communication Technology (ICACT '06), , Phoenix Park, Korea, February 2006, pp. 2055-2060.

[34] G. Lu, B. Krishnamachari, and C. S. Raghavendra, "An Adaptive Energy-efficient and Low-Latency MAC for Data Gathering in Wireless Sensor Networks," in 18th International Parallel and Distributed Processing Symposium, Apr. 2004, p. 224.

[35] S. Cui, R. Madan, A. J. Goldsmith, and S. Lall, "Energy-delay Tradeoff for Data Collection in Sensor Networks," in Internation Communication Conference (ICC2005), South Korea, 2005.

[36] N. H. Vaidya, "Mobile Ad-hoc Networks: Routing, MAC and Transport Issues," http://www.crhc.uiuc.edu/~nhv.

[37] Q. Xu, T. Mak, J. Ko, and R. Sengupta, "Vehicle-to-Vehicle Safety Messaging in DSRC," in the 1st ACM International Workshop on Vehicular ad hoc networks, October 01-01, 2004.

[38] R. M. Yadumurthy, A. Chimalakonda, m. Sandashivaiah, and R. Makanaboyina, "Reliable MAC Broadcast in Directional and Omni-directional Transmissions for Vehicular Ad Hoc Networks," in the 2nd ACM International workshop on Vehicular ad hoc networks, 2005.

[39] T. Liu, J. A. Sylvester, and A. Polydoros, "Performance Evaluation of

R-ALOHA in Distributed Packet Radio Networks with Hard Real-time Communication," in IEEE Vehicle Technology Conference 1995.

[40] R. Verdone, "Multihop R-ALOHA for Intervehicle Communications at Millimeter Waves," IEEE Transaction on Vehicular Technology, vol. 4, 1997.

[41] S. Katragadda, G. M. C.N.S., R. R. M.S., K. S. M, and R. Sachin, "A Decentralized Location-based Channel Access Protocol for Inter-vehicle Communication," in Vehicular Technology Conference (VTC 2003-Spring), 2003.

[42] G.-S. Ahn, E. Miluzzo, A. T. Campbell, S. G. Hong, and F. Cuomo, "Funneling-MAC: A Localized, Sink-Oriented MAC For Boosting Fidelity in Sensor Networks," in Fourth ACM Conference on Embedded Networked Sensor Systems (SenSys 2006) Boulder, Colorado, USA, November 2006.

[43] K. Sohrabi, J. Gao, V. Ailawadhi, and G. J. Pottie, "Protocols for Self-Organization of a Wireless Sensor Network," IEEE Personal Communications, vol. 7, pp. 16-27 Oct. 2000.

[44] T. Wu and S. Biswas, "A self-reorganizing Slot Allocation Protocol for Multi-cluster Sensor Networks," in Fourth International Symposium on Information Processing in Sensor Networks(IPSN 2005) Apr. 2005, pp. 309-316.

[45] J. Reason and J. M. Rabaey, "A Study of Energy Consumption and Reliability in a Multi-Hop Sensor Network," in ACM SIGMOBILE Mobile Computing and Communications, Jan. 2004, pp. 84-97.

[46] A. Woo and D. E. Cullar, "A Transmission Control Scheme for Media Access in Sensor Networks," in the 7th annual international conference on Mobile computing and networking, 2001, pp. 221-235.

[47] N. Abramson, "The Aloha system - Another Alternative for Computer Communication," in Fall Joint Computer Conference, AFIPS Conference 1970, pp. 281-285.

[48] L. Kleinrock and F. Tobagi, "Carrier Sense Multiple Access for Packet Switched Radio Channels," in International Conference on Communications, Minneapolis, Minnesota, Jun. 1974, pp. 21B-1 -21B-7.

[49] "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification," LAN MAN Standards Committee of the IEEE Computer Society, 1997.

[50] ETSI, "Broadband Radio Access Networks (BRAN); HIPERLAN type 2 Technical Specification; Physical (PHY) layer." vol. DTS/BRAN-0023003, Aug. 1999.

[51] F. Sivrikaya and B. Yener, "Time Synchronization in Sensor Networks: A Survey," IEEE Networks, pp. 45-50, July/August 2004.

[52] B. Sundararaman, U. Buy, and A. D. Kshemkalyani, "Clock Synchronization in Wireless Sensor Networks: A Survey," Ad-Hoc Networks,, vol. 3, pp. 281-323, May 2005.

[53] Y. Zhang, M. Roughan, C. Lund, and D. L. Donoho, "Estimating Point-to-Point and Point-to-Multipoint Traffic Matrices: An Information-theoretic Approach," IEEE/ACM Transactions on Networking, vol. 13, pp. 947-960, Oct. 2005.

[54] G. Urvoy-Keller, G. Hebutene, and Y. Dallery, "Traffic Engineering in a Multipoint-to-Point network," IEEE Journal on Selected Areas in Communications, vol. 20, pp. 834-849, May 2002.

[55] R. Nelson and L. Kleinrock, "Spatial-TDMA: A Collision-free Multihop Channel Access Protocol," IEEE Transaction on Communications, vol. 33, pp. 934-944, September 1985.

[56] A. El-Hoiyfi, "Spatial TDMA and CSMA with Preamble Sampling for Low Power Ad Hoc Wireless Sensor Networks," in International Symposium On Computer and Communication, 2002.

[57] K. Khan and H. Peyravi, "Delay and Queue Size Analysis of TDMA with General Traffic," in the Sixth International Symposium Modeling, and Simulation of Computer Systems, July 1998, pp. 217-225.

[58] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," in the 6th Annual International Conference on Mobile Computer and Networks, August 2000, pp. 56-67.

[59] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An Application- Specific Protocol Architecture for Wireless Microsensor Networks," IEEE Transaction on Wireless Communication, vol. 1, pp. 660-70, October 2002.

[60] S. D. Muruganathan, D. C. F. Ma, R. I. Bhasin, and A. O. Fapojuwo, "A Centralized Energy-Efficient Routing Protocol for Wireless Sensor Networks," IEEE Radio Communication, pp. S8-S13, March 2005.

[61] "Intelligent transportation systems," United States Department of Transportation, www.its.dot.gov/index.htm.

[62] L. Armstrong, "Dedicated Short Range Communications (DSRC)," www.leearmstrong.com/DSRC/DSRCHomeset.htm.

[63] Y. Liu, F. Dion, and S. Biswas, "Dedicated Short-range Wireless

Communications for Intelligent Transportation System Applications: State of the Art," ITS and vehicle-highway automation, vol. 1910, 2005.

[64] Y. Wang and B. Bensaou, "Achieving Fairness in IEEE 802.11 DFWMAC with Variable Packet Size," in IEEE Global Telecommunication Conference, 2001.

[65] T. Pagtzis, P. Kirstein, and S. Hailers, "Operational and Fairness Issues with Connection-Less Traffic Over IEEE802.11b," in IEEE International Conference on Communication (ICC), 2001.

[66] H. Krishnan and C. Kellum, "Use of Communication in Vehicle Safety Application," Internal Report of General Motors Company 2002.

[67] M. Ergen, D. Lee, R. Sengupta, and P. Varaiya, "WTRP-Wireless Token Ring Protocol," IEEE Transaction on Vehicular Technology, vol. 53, 2004.

[68] T. Nagaosa and T. Hasegawa, "Code Assignment and the Multicode Sense Scheme in an Inter-Vehicle CDMA Communication Network," IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer, vol. E81-A, 1998.

[69] I. Rhee, A. Warrier, J. Min, and L. Xu, "DRAND: Distributed Randomized TDMA Scheduling for Wireless Ad-hoc Networks," in the 7th ACM international symposium on Mobile ad hoc networking and computing, Florence, Italy, May 22-25, 2006,.

[70] I. Rhee, A. Warrier, M. Aia, and J. Min, "Z-MAC: A Hybrid MAC for Wireless Sensor Networks," in ACM SenSys 2005, 2005.

[71] "Michigan Auto Accident News," www.michiganautolaw.com/blog/michiganautoaccidents/index.php.

[72] X. Yang, J. Liu, F. Zhao, and N. Vaidya, "A Vehicle-to-Vehicle Communication Protocol for Cooperative Collision Warning," in the First International Conference on Mobile and Ubiquitous Systems: Networking and Services, 2004.

[73] F. Yu, T. Wu, and S. Biswas, "Towards In-Band Self-Organization in Energy-Efficient MAC Protocols for Sensor Networks," IEEE Transaction of Mobile Computing, Accepted 2007.

[74] S. Biswas and F. Yu, "An in-band self-organized MAC protocol for sensor networks," in SPIE in Wireless Sensing and Processing, Orlando FL, USA, May 12, 2006.

[75] P. Varaiya, "Smart Cars on Smart Roads: Problems of Control," IEEE transaction Automatic Control, vol. 38, 1993.

[76] "The network simulator: NS-2," www.isi.edu/nsnam/ns.

[77] "2006 michigan state police testing evaulation," www.michigan.gov/documents/MSP_Eval_146823_7.pdf.

[78] T. J. Gates, D. A. Noyce, and L. Laracuente, "Analysis of Dilemma Zone Driver Behavior at Signalized Intersections," in Transportation Research Board (TRB) of the National Academies the 86th Annual Meeting January 2007.

[79] C. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," in the ACM SIGCOMM, October 1994.

[80] H. Balakrishnan, "Opportunities and challenges in high-rate wireless sensor networking," in 29th Annual IEEE International Conference on Local Computer Networks, 2004. , November 2004.

[81] N. Abramson, "The Aloha system - Another Alternative for Computer Communication," in AFIPS, 1970, pp. 295-298.

[82] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in the 2nd international conference on Embedded networked sensor systems, 2004, pp. 95 - 107.

[83] F. Yu, T. Wu, and S. Biswas, "Towards In-Band Self-Organization in Energy-Efficient MAC Protocols for Sensor Networks," IEEE Transaction of Mobile Computing, vol. 7, pp. 156-170, February 2008.

[84] "Funneling-MAC Technical Report: http://www.cs.dartmouth.edu/~sensorlab/funneling-mac/TAPTR-2006-08-003.pdf.."

[85] J. V. Sudarev, L. B. White, and S. Perreau, "Performance Analysis of 802.11 CSMA/CA for Infrastructure Networks under Finite Load Conditions," in The 14th IEEE Workshop on Local and Metropolitan Area Networks, 2005. LANMAN 2005. , September 2005.

[86] M. Y. Chung, M.-H. Jung, T.-J. Lee, and Y. Lee, "Performance Analysis of HomePlug 1.0 MAC With CSMA/CA," IEEE Journal on Selected Areas in Communications, vol. 24, pp. 1411-1420, July 2006.

[87] W. Lee, C. Wang, and K. Sohraby, "On Use of Traditional M/G/1 Model for IEEE 802.11 DCF in Unsaturated Traffic Conditions," in IEEE Wireless Communications and Networking Conference (WCNC' 06), Las Vegas, Nevada, April 2006, pp. 1933-1937.

[88] Y. Barowski and S. Biaz, "The Performance Analysis of IEEE 802.11 Under Unsaturated Traffic

Conditions,ftp://ftp.eng.auburn.edu/pub/techreports/csse/04/CSSE04-08.pdf," August 2004.

[89] F. Daneshgaran, M. Laddomada, F. Mesiti, and M. Mondin, "Unsaturated Throughput Analysis of IEEE 802.11 in Presence of Non Ideal Transmission Channel and Capture Effects," IEEE Transactions on Wireless Communications, vol. 7, pp. 1276-1286, April 2008.

[90] G. Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function," IEEE Journal on Selected Areas in Communications, vol. 18, pp. 535-547, March 2000.

[91] O. Tickoo and B. Sikdar, "A queueing model for finite load IEEE 802.11 random access MAC," in IEEE International Conference of Communication (ICC), 2004, pp. 175-179.

[92] R. B. Cooper, Introduction to Queueing Theory, Third ed.: CEE Press Books, 1990.

[93] D. Bertsekas and R. Gallager, Data Networks, Second ed.: Prentice-Hall 1992.

[94] E. W. Justh and P. S. Krishnamrasad, "A simple control law for UAV formation flying," Institute for System Research Technical Report 2002.

[95] Y. U. Cao, A. S. Fukunaga, and A. kahng, "Cooperative mobile robotics: Antecedents and directions," Automation Robots, March 1997.

[96] N. Shimoyama, K. Sugawara, T. Mizuguchi, Y. Hayakawa, and M. Sano, "Collective motion in a system of motile elements," Physical Review Letters, 1996.

[97] B. T. Ludington, L.tang, and G. J. Vachtsevanos, "Target tracking in an urban warfare environment using particle filters," in Aerospace, 2005 IEEE Conference, 2005.

[98] S. Biswas, S. Gupta, F. Yu, and T. Wu, "Collaborative Multi-target Tracking Using Networked Micro-robotic Vehicles," in SPIE Defense Transformation and Net-Centric Systems 2007, Orlando FL, USA, May 2007.

[99] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," IEEE Transation on Automation Control, 2003.

[100] L. Moreau, "Staility of multiagent systems with time-dependent communication links," IEEE Transation on Automatic Control, 2005.

[101] W. Xi, X. Tan, and J. S. Baras, "Gibbs' sampler-based coordination of autonomous swarms," Automatica, 2006.

[102] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: algorithms and theory," IEEE. Transaction On Automatic Control,, March 2006.

[103] R. O. Saber and R. M. Murray, "Consensus Protocols for dynamic agent networks," in American Control Conference, 2003.

[104] A. Savvides, C.-C. Han, and M. Srivastava, "Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors," in 7th ACM MobiCom, July 2001.

[105] A. V. Savkin, "Coordinated collective motion of groups of autonomous mobile robots: analysis of Vicsek's model," IEEE transactions on Automatic Control, June 2004.

[106] T. Vicsek, A. Czirok, E. B. Jacob, I. Cohen, and O. Schochet, "Novel type of phase transitions in a system of self-driven particles," Physical Review Letters, 1995.

[107] A. Jadbabaie, N. Motee, and M. Barahona, "On the stability of the Kuramoto model of coupled nonlinear oscillators," in American Control Conference, 2004.

[108] E. W. Justh and P. S. Krishnamrasad, "Equilibria and steering laws for planar formations," in the 42nd IEEE conference On Decision and Control 2003.