Performance Simulation of Priority-based CSMA/CA and Pseudo-Access Point Routing Protocol

Moyu Yang

A Thesis in

The Department of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of Master of Applied Science at

Concordia University

Montréal, Québec, Canada

April 2004

Abstract

Performance Simulation of Priority-based CSMA/CA

and Pseudo-Access Point Routing Protocol

Moyu Yang

The ad hoc wireless Local Area Network (WLAN) has gained a lot of interest in the research community due to its special properties, such as deployment flexibility. With current demands, ad hoc WLANs are being developed to provide better quality to users.

Firstly, a simulation model is built in C++ programming language with object-oriented method. The software of the WLAN simulator is designed to provide highly detailed and accurate statistical information, and can serve as a general platform for further simulation studies. Secondly, a newly proposed double window algorithm introduces a priority-based window alternative to the standard CSMA/CA. Simulation results demonstrate that networks achieve a better Quality of Service (QoS) by using the new priority-based window. Thirdly, the Pseudo Access Points (PAP) routing protocol is presented to enhance the traditional proactive routing protocols. The simulations confirm that the PAP routing protocol is better than the generic θ routing protocol under different mobility scenarios. It provides higher throughput, less failed routing, less variance of the hop count, etc, and thus offers a better Quality of Service. The performance evaluation of the PAP routing protocol is also emphasized. The investigation shows that the performance of the PAP routing protocol can be improved through the manipulation of various input parameters.

Acknowledgements

I would like to thank Dr. A.K. Elhakeem for his continuous guidance and suggestions on my research. He always helped me to approach the problems from different perspectives and to define the scope of what I needed to accomplish. He always pointed out my mistakes and suggested solutions to the problems I encountered. His suggestions and observations were extremely helpful throughout this thesis.

I would like to thank Minjie Qiu from Cisco Ottawa Inc. for her comments about the implementation of the wireless LAN simulator.

I would like to express my appreciation to my wife, my parents, and my brother for their unlimited encouragement and support.

Finally, I want to acknowledge the financial support of the Concordia University for the Teaching Assistantship and the Research Assistantship.

Table of Contents

List of Abbreviations

ACK	Acknowledgement
AIFS	Arbitration Inter Frame Space (802.11e)
AP	Access Point
CA	Collision Avoidance
CDF	Complementary Cumulative Distribution Function
CFP	Contention Free Period
CF-Poll	Contention Free – Poll
CF-End	Contention Free – End
CP	Contention Period
CSMA	Carrier Sense Multiple Access
CW	Contention Window
CWmax	Contention Window Maximum
CWmin	Contention Window Minimum
DCF	Distributed Coordination Function
EDCF	Enhanced DCF (802.11e)
HC	Hybrid Coordinator (802.11e)
HCF	Hybrid Coordination Function (802.11e)
IEEE	Institute of Electrical and Electronics Engineers
ISM	Industrial, Science, Medical
LRE	Limited Relative Error
MAC	Medium Access Control

MSDU	MAC Service Data Unit
NAV	Network Allocation Vector
PAP	Pseudo Access Point
PC	Point Coordinator
PCF	Point Coordination Function
PDU	Protocol Data Unit
PF	Persistence Factor (802.11e)
PHY mode	Physical Layer mode, coding and modulation scheme
PIFS	PCF Inter Frame Space
(Q)BSS	(QoS-supporting) Basic Service Set (802.11e)
QoS	Quality of Service
RTS/CTS	Request to Send/Clear to Send
SDU	Service Data Unit
SIFS	Short Inter Frame Space
TBTT	Target Beacon Transmission Time
TC	Traffic Category (802.11e)
TXOP	Transmission Opportunity (802.11e)
WLAN	Wireless Local Area Network

List of Figures

List of Tables

Chapter 1

Introduction

In this thesis we concentrate on indoor or close range networks, often called Wireless Local Area Networks (WLANs). Recently, hardware prices have dropped drastically for infrastructure equipment, and as a result of this, WLANs are deployed almost everywhere [1]. The most common standard for these networks today is the IEEE 802.11 standard [2]. Furthermore, WLANs are very popular due to their flexible nature, and the inherent possibility for wireless nodes to be mobile. There exist other standards such as HiperLan/2 [3] and HomeRF [4], but they are not widely used. This chapter gives the related background knowledge.

1.1. Topologies of Wireless LANs

Wireless LANs (WLAN) [14] allow for more flexible communication, since the nodes are not limited to a fixed physical location. There are two categories of mobile wireless LANs defined in the IEEE 802.11 standard: infrastructure WLAN and Ad Hoc (Peer-to-peer) WLAN.

1.1.1. Infrastructure wireless LANs

The infrastructure WLAN consists of stationary base stations and mobile endpoints. Base stations, also called Access Point (AP), are fixed and connected to the wired backbone, acting as gateways between the mobile endpoints and the wired backbone. A mobile endpoint, or mobile station (MS), in the area of the direct wireless transmission

range covered by at least one of Access Points, communicates directly and only with the base station to exchange information with other fixed and mobile end hosts. Thus, wireless communication in such networks is a single-hop communication. Figure 1.1 illustrates an example of the infrastructure WLAN.

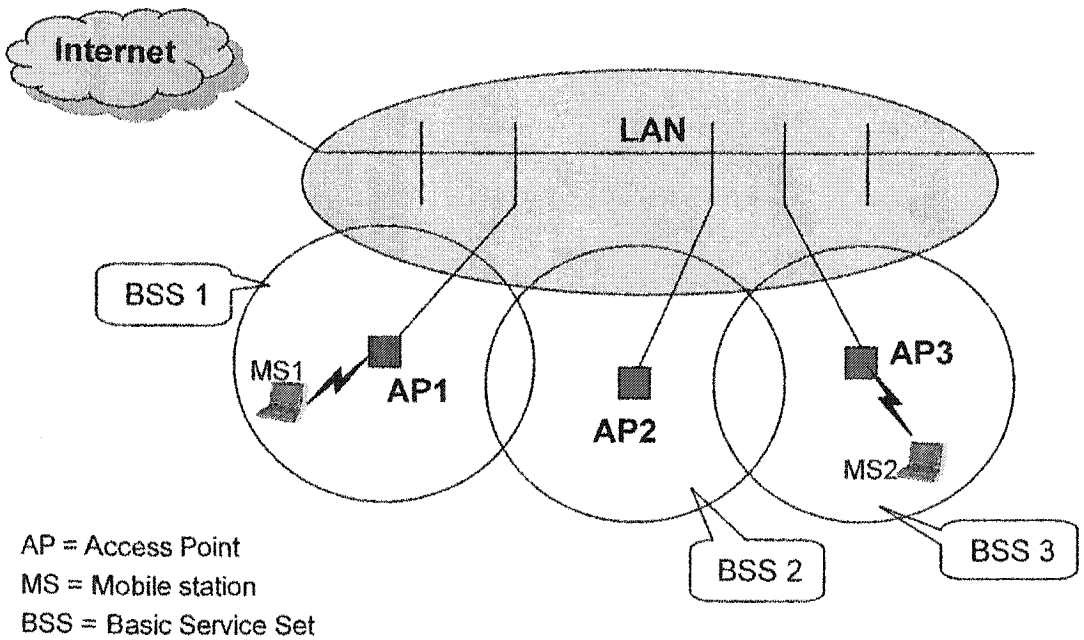


Figure 1.1 An example of the infrastructure WLAN

1.1.2. Ad hoc (Peer-to-peer) wireless LANs

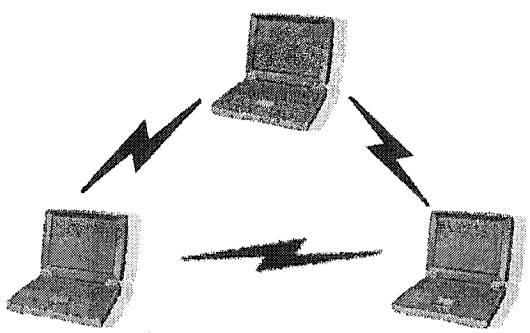


Figure 1.2 An example of Ad Hoc (Peer-to-peer) WLAN

Ad Hoc WLAN [15], a peer-to-peer network, consists of only mobile nodes, without

Access Point (AP) and wired backbone in charge of information exchange and network administration. Each mobile node not only operates as a host but also as a router, responsible for forwarding packets to other mobile nodes in the network that may not be within the direct wireless transmission range of each other. Wireless communication in such a network is a multi-hop communication. Figure 1.2 gives an example of Ad Hoc WLAN.

1.1.3. Hybrid wireless LANs

Currently most wireless networks are infrastructure WLAN. To accommodate mobility, hand-over can be performed between two base stations as the wireless station moves from the coverage area of one base station to another, enabling the communication to seamlessly continue. Ad hoc WLAN has also gained a lot of interest in the research community. Because of the special properties of ad hoc networks such as quick topology changes due to the mobility of the nodes, ordinary routing protocols fail to give good performance. There exist several ad hoc routing protocols that enable ad hoc networks.

We can set up a hybrid between the two worlds of the infrastructure WLAN and ad hoc WLAN. In such a network, ad hoc routing is used to extend the range of infrastructure wireless networks that allow a larger area to be covered. Such networks thus have different requirements than traditional networks. For such a network to be possible to create, special routing protocols are needed. Figure 1.3 gives an example of the hybrid WLAN.

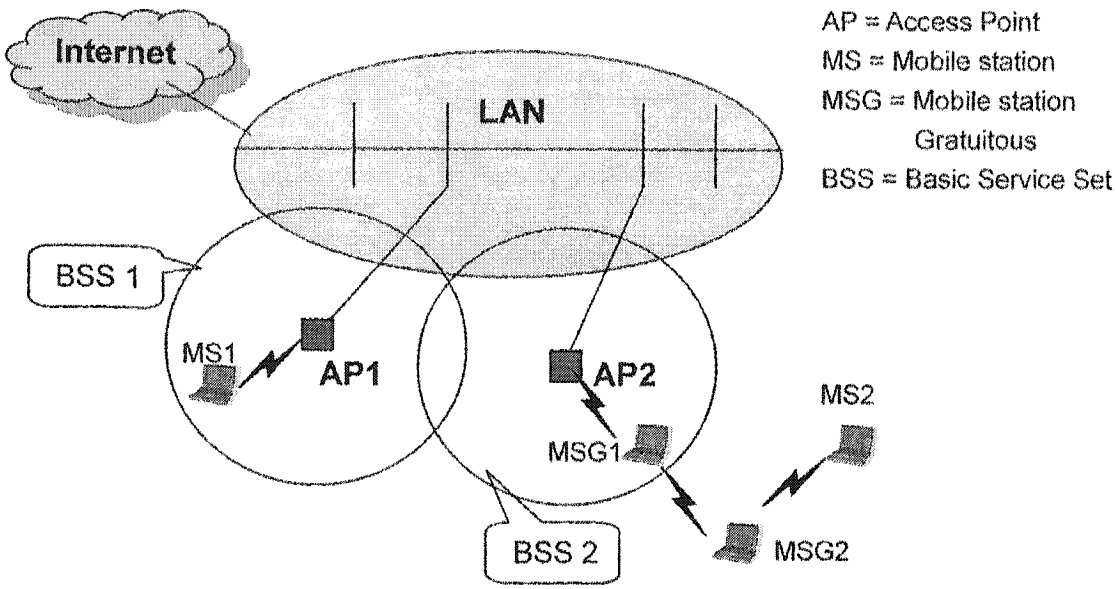


Figure 1.3 An example of the hybrid WLAN

1.2. Physical Layer

The IEEE-802.11 PHY [5] is responsible for mapping the IEEE-802.11 MAC frame unit into a format suitable for sending and receiving messages via a wireless medium, between two or more stations using one of the following implementations: Frequency Hopping Spread Spectrum (FHSS), Direct Sequence Spread Spectrum (DSSS), or Infrared (IR). The FHSS utilizes the 2.4 GHz Industrial, Scientific, and Medical (ISM) band (2.4000-2.4835 GHz). The band is divided into frequency channels with 1 MHz bandwidth each. A frequency-hopping sequence consists of a permutation of all frequency channels. Three different hopping sequence sets are defined in the specification, with 26 hopping sequences per set. With the FHSS implementation, the carrier frequency is hopped with a pre-defined hop rate according to a hopping sequence. Different hopping sequences enable multiple BSSs to coexist in the same geographical area, which may

become important to alleviate congestion and maximize the total throughput in a single BSS. Two access rates, 1 Mbit/s and 2 Mbit/s, are specified using 2-level Gaussian Frequency Shift Key (GFSK) and 4-level GFSK modulation respectively.

The DSSS implementation also uses the 2.4 GHz ISM frequency band. The band is similarly divided into frequency channels, but with 11 MHz bandwidth each. The spreading is done by chipping each data symbol at 11 MHz in one channel with a pre-defined 11-bit chip sequence. The DSSS also provides both 1 Mbit/s and 2 Mbit/s access rates with Differential Binary Phase Shift Keying (DBPSK) and Differential Quadrature Phase Shift Keying (DQPSK) modulation schemes respectively.

The IR implementation uses wavelengths from 850 nm to 950 nm for signaling. It is designed for indoor use only and operates with non-directed transmissions. Two access rates are also specified for IR: 1 Mbit/s and 2 Mbit/s using 16-Pulse Position Modulation (PPM) and 4-PPM respectively. It requires line of sight or reflected transmission.

1.3. Medium Access Control (MAC) Layer

The MAC sub-layer [17] is responsible for channel access procedures, protocol data unit (PDU) addressing, framing, error checking, fragmentation, and reassembly of MAC Service Data Unit (MSDU). The IEEE 802.11 MAC protocol is specified in terms of coordination functions that determine when a station in a BSS is allowed to transmit and when it may be able to receive PDUs over the wireless medium. The distributed coordination function (DCF) provides support for asynchronous data transfer of MAC SDUs on a best-effort basis. Under the DCF, the transmission medium operates in the

contention mode exclusively, requiring all stations to contend for the channel for each packet transmitted. The IEEE 802.11 also defines an optional point coordination function (PCF), which may be implemented by an AP, to support connection-oriented time-bounded transfer of MAC SDUs. Under the PCF the medium can alternate between the contention period (CP) and the contention free period (CFP). The medium usage is controlled by the AP, thereby eliminating the need for stations to contend for channel access.

The 802.11 MAC architecture is depicted in Figure 1.4, which shows that the DCF sits directly on top of the physical layer and supports contention services. The PCF is required to coexist with the DCF and logically sits on top of the DCF.

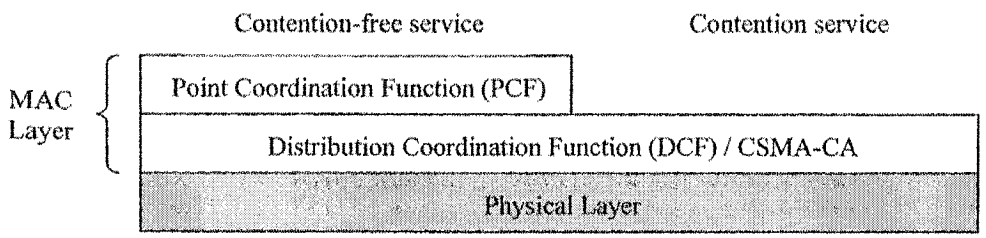


Figure 1.4: IEEE 802.11 MAC architecture

1.3.1. CSMA/CA MAC Protocol

Most wired LANs use Carrier Sense Medium Access with Collision Detection (CSMA/CD) [16] as the MAC protocol. Carrier sense means that the station will listen before it transmits. If someone is already transmitting, the sender waits and tries again later. When two stations send at the same time, transmissions collide and information will be lost. Collision detection handles this situation by listening to the signal it is

transmitting to ensure everything is going right. Whenever a collision occurs, nodes stop and try again later, which is determined by the backoff algorithm shown in Figure 1.5.

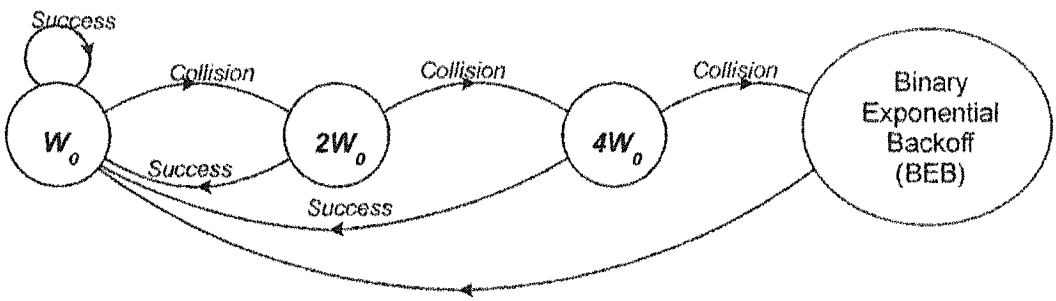


Figure 1.5: Binary Exponential Backoff (BEB)

In the wireless medium, the Carrier Sense Medium Access with Collision Avoidance (CSMA/CA) is used instead of the CSMA/CD. The first reason is that it is difficult to detect collisions in a radio environment, so it is not possible to abort transmissions that collide. The second reason is that the radio environment is not as well controlled as a wired broadcast medium, and transmissions from users in other LANs can interfere with the operation of the CSMA-CD. The third reason is that radio LANs are subject to the hidden-node problem which will be described in the next section. The CSMA/CA protocol uses only one frequency, and the single-spreading code is utilized by all stations for transmission from one node to the other. Thus, only one station can successfully transmit in the network at any time.

1.3.2. The Hidden Node Problem

The hidden node problem is one of the most common problems [11] in wireless networks (besides the high bit-error rates). This problem comes from the fact that every wireless node has limited radio transmitting range and that every wireless node can thus

not be expected to be able to communicate with every other node in the network. The phenomenon can be easily explained with the following example (depicted in Figure 1.6).

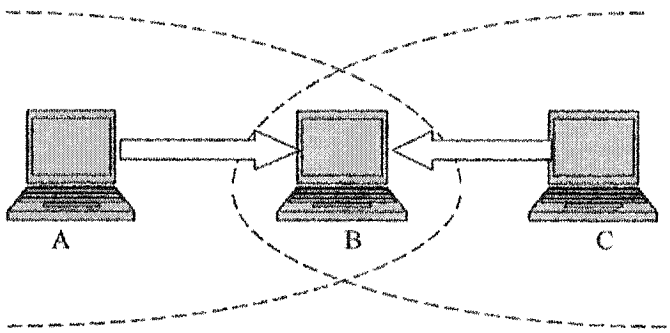


Figure 1.6: The hidden node problem

Let's suppose that a WLAN consist of three nodes, among which node A and C only are able to communicate with node B and not with each other. If node A has an ongoing transmission with B, this could easily be interrupted by a transmission from C to B, since C cannot hear the transmission from A. One could say that node C is hidden from node A.

The IEEE 802.11 provides a solution to this by using a "Request To Send - Clear To Send" (RTS/CTS) scheme [17]. In the RTS-CTS negotiation algorithm the node that wants to send data sends a RTS frame to the destination node which responds with a CTS frame if the medium is idle. Upon reception of the CTS frame, the first station transmits its data frame, and if the CTS is not received, it tries again at a later time. The RTS and CTS frames contain the duration of the coming data transmission, which allows stations overhearing the RTS/CTS exchange to refrain from transmitting during this time. To further explain this, consider the scenario in Figure 1.7. Here, any transmission between A and B could be interrupted by a sudden transmission from C to B. Using the RTS-CTS

scheme here would eliminate such interruptions, since C will be able to hear the CTS response from B and thus to know how long B will be busy in a transmission with A.

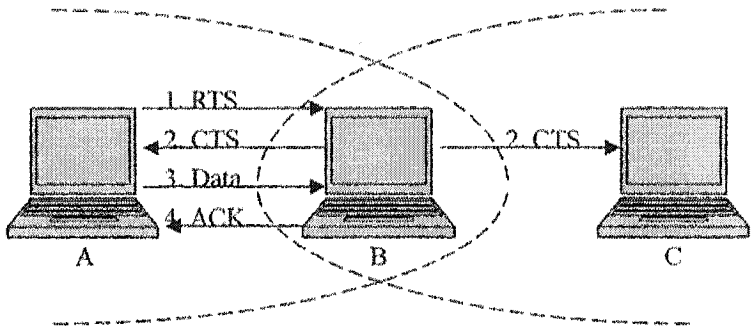


Figure 1.7: The RTS-CTS scheme

This scheme reduces the probability for collisions but introduces a lot of packet overhead. Therefore, it is recommended to be used only for data frames larger than some threshold. In addition, it costs even more if a large frame is corrupted due to a collision.

1.3.3. Distributed Coordinator Function

The Distributed Coordinator Function [17] is a basic access mechanism to support the asynchronous data transfer on a best-effort basis. All stations are required to support the DCF. The access control in ad hoc networks uses only the DCF. Infrastructure networks can operate using just the DCF or a coexistence of the DCF and the PCF. The DCF uses the CSMA/CA algorithm to mediate the access to the shared medium. When the MAC layer gets some data that should be transmitted, it senses the medium for a DCF inter-frame space 1 (DIFS) period of time as shown in Figure 1.8, and then the frame is transmitted. Otherwise, a backoff time B (measured in time slots) is chosen randomly in the interval (0, CW), where CW is the so called Contention Window.

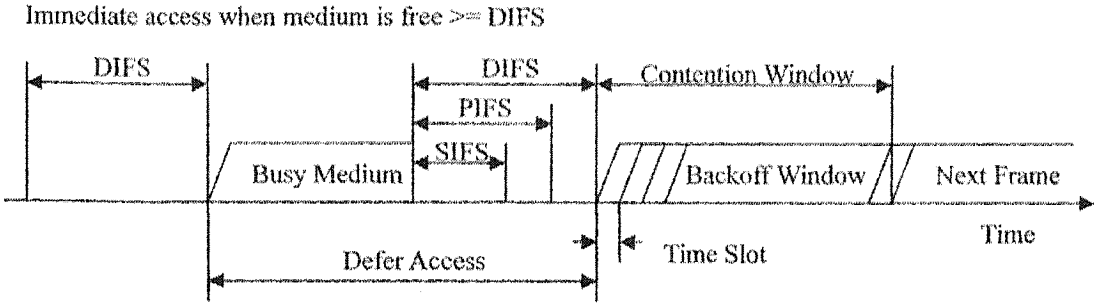


Figure 1.8: Inter-frame space relationships

After the medium has been detected idle for at least a DIFS, the backoff timer is decremented by one for each time slot and the medium remains idle. When the backoff timer reaches zero, the frame is transmitted. If the frame is successfully received at the destination, an acknowledgement frame (ACK) is sent to the sender. Upon detection of a collision (which is detected by the absence of an acknowledgement frame), the contention window is doubled according to Equation 1.1.

$$CW_i = 2^{k+i-1} \times CW\,\text{min} - 1 \qquad (1.1)$$

Here i is the number of attempts (including the current one) to transmit the frame that has been done, and k is a constant defining the minimum contention window, CWmin. A new backoff time is then chosen and the backoff procedure starts over. Since the contention window is doubled for every collision, the probability that the two colliding nodes will choose the same backoff time decreases. The backoff mechanism is also used after a successful transmission before sending the next frame and thus it can also reduce the probability for collisions. After a successful transmission, the contention window is reset to CWmin.

1.3.4. Point Coordinator Function

The PCF [17] is an optional capability that can be used to provide connection-oriented, contention-free services by enabling polled stations to transmit without contending for the channel. The centralized, polling-based access mechanism requires the presence of a base station that acts as Point Coordinator (PC) in the AP. If the PCF is supported, both the PCF and the DCF coexist. In this case, time is divided into super-frames as shown in Figure 1.9. Each super-frame consists of a contention period where the DCF is used and a contention free period (CFP) where the PCF is used. The CFP is started by a special frame (a beacon) sent by the base station. Since the beacon is sent using ordinary DCF access method, the base station has to contend for the medium, and therefore the CFP may be shortened.

The PC keeps a list of mobile stations that have requested to be polled to send data. During the CFP, it sends poll frames to the stations when they are clear to access the medium. To ensure that no DCF stations are able to interrupt this mode of operation, the IFS between PCF data frames is shorter than the usual DIFS. This space is called a PCF inter-frame space (PIFS). To prevent the starvation of stations that are not allowed to send during the CFP, there must always be room for at least one maximum length frame to be sent during the contention period.

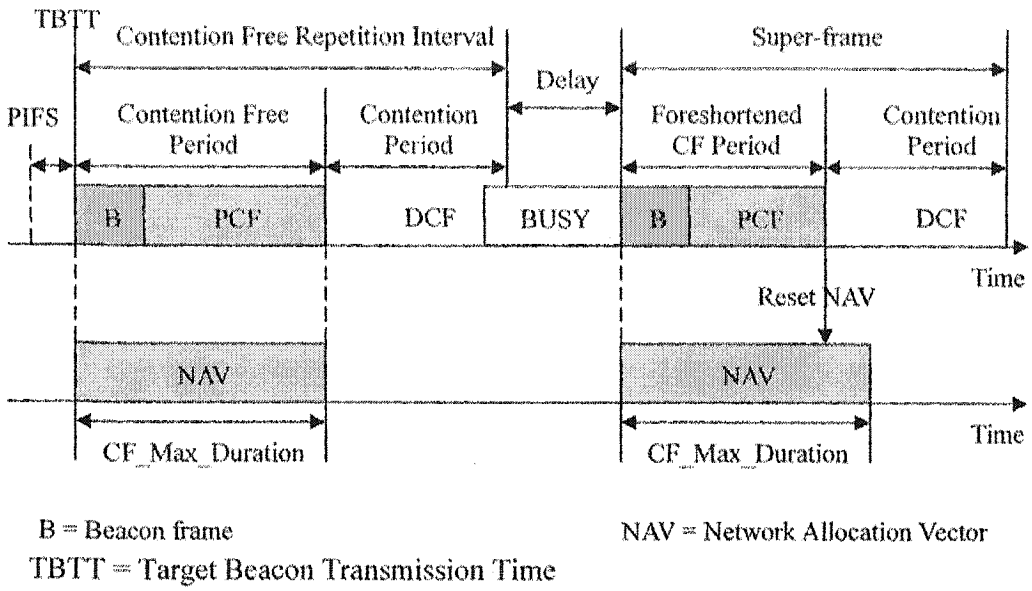


Figure 1.9: The super-frame of IEEE 802.11

At the beginning of each CFP repetition interval, all stations in the BSS update their NAV to the maximum length of the CFP, i.e. CFP_Max_Duration. During the CFP, stations may transmit only to respond to a poll from the PC or to transmit an acknowledgement one SIFS interval after receipt of an MPDU. At the start of the CFP, the PC senses the medium. If the medium remains idle for a PIFS interval, the PC transmits a beacon frame to initiate the CFP. In case the CFP is lightly loaded, the PC can foreshorten the CFP and provide the remaining bandwidth to contention-based traffic by issuing a CF-End or CF-End+ACK control frame. This action causes all stations that receive the frame in the BSS to reset their NAV value. An example is shown in Figure 1.10.

RTS/CTS frames are not used by the point coordinator. After the PC issues a poll, the intended CF-aware station may transmit one frame to any station as well as piggyback any ACK of a frame received from the PC. When a frame is transmitted to a

non-CF-aware station, the station sends its ACK using DCF rules. The PC keeps control of the medium by only waiting the PIFS duration before proceeding with its contention-free transmissions.

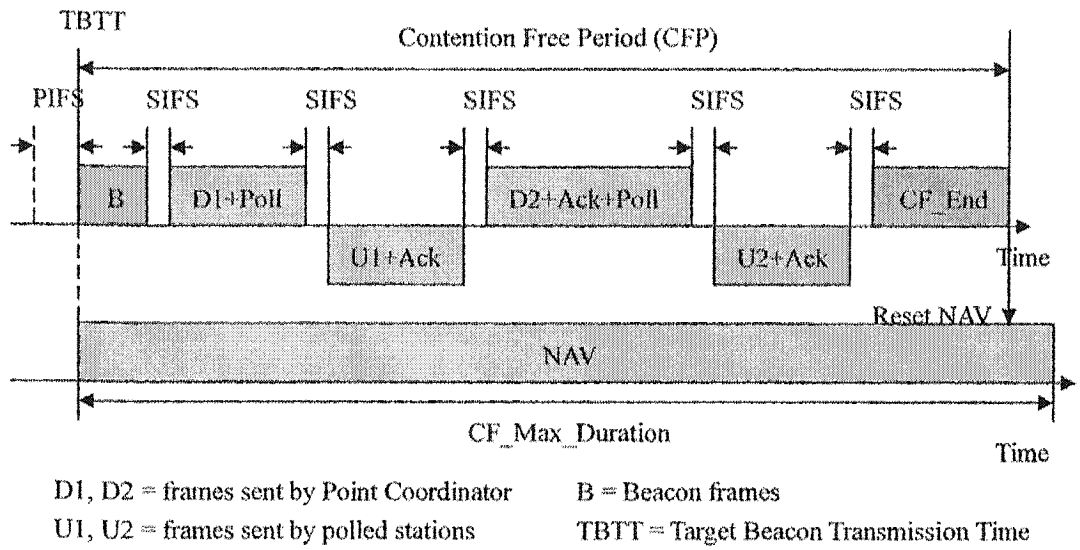


Figure 1.10: Point coordination frame transfer

1.4. Routing Protocols in Ad Hoc WLANs

Conventional routing protocols use either distance vector or link-state algorithms to determine the most efficient path to a destination. Distance vector algorithms require each router to maintain a route table containing all possible destinations along with an associated metric that is collected on a periodic basis. The routing overhead remains constant regardless of the amount of the host movement. This type of method is closely associated with the distributed Bellman-Ford routing algorithm. A version of Bellman-Ford [15] is still being used today with the Router Internet Protocol (RIP) [17]. In RIP, for each entry the next hop to the destination is stored along with a metric to reach the destination. The metric can be based on distance, total delay, or the cost of sending the

message. Each node shares its internal information periodically through update broadcasts to neighboring nodes. The routers utilize the updates to constantly revise their routing tables for shortest-path calculations. Link-state algorithms operate in a similar manner but are event driven by changes in the link status of nodes. Path-finding algorithms provide a hybrid approach utilizing both distance vector and link-state algorithms. Although distance vector and link-state algorithms are very effective for achieving routing optimization, the overhead associated with these techniques is considerable and exhibits slow convergence due to topological changes. In RIP, a conventional protocol, routing updates are produced on a periodic basis. RIP does not scale well to large networks, because each network node requires N iterations to detect a node that is disconnected, where N represents the number of nodes. This is known as the count to infinity problem. On-demand protocols have clear proportional increase in overhead due to node mobility.

Wireless links have significantly lower capacity than their wired counterparts. After the effects of multiple access, fading, noise, interference, etc., the capacity of a wireless link may be variable and the link direction may be unidirectional. In such an environment, congestion is prone to happen. Besides that, node mobility also challenges the multi-hop communication in a Mobile Ad Hoc Network (MANET) [19].

Typically, nodes in a MANET rely on battery with limited power during moving, and the network topology may change frequently, rapidly and unpredictably. All these features cause the routes between the communication pairs to fail easily, resulting in frequent route updates. Ad hoc routing protocols can be generally summarized in three categories as shown in Figure 1.11: proactive (also called table-driven), reactive (also

called on-demand), and hybrid (proactive and reactive).

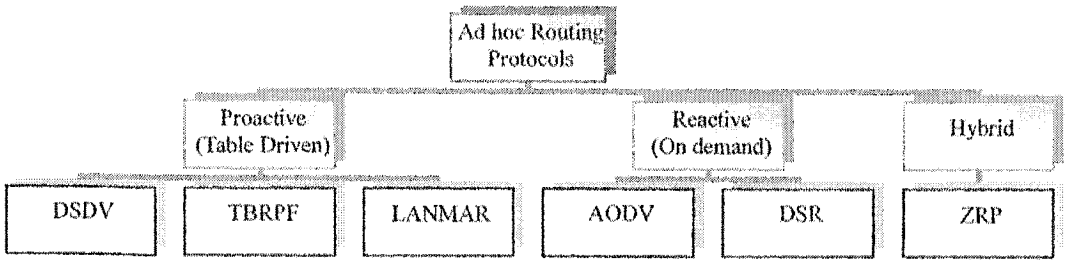


Figure 1.11: Categories of Ad hoc Routing Protocols

1.4.1. Proactive Routing Protocols

The proactive routing protocols attempt to keep up-to-date routing information between any pair of mobile nodes. Routing-update messages are propagated throughout the whole network to get a consistent view of the network topology.

DSDV (Destination-Sequenced Distance-Vector Routing) is a distance vector routing protocol based on the classical Bellman- Ford routing algorithm [17], which requires each node in the network to broadcast routing-update messages periodically to update the routing table in which routes to all the possible destinations are recorded. The key design of DSDV is that, in addition to the routing table, each node also has a monotonically increasing even sequence number, which increments whenever a new routing-update message is sent out, thus letting other nodes know which routing information is fresher and avoiding routing loops. So in a routing table, in addition to the information about the destination node address, the hop count to the destination, and the next hop to that destination, the currently known largest sequence number of the destination is also contained.

1.4.2. Reactive Routing Protocols

Reactive routing protocols create routes only when desired by the source node. The route discovery follows a Request-Reply cycle and starts only on demand, that is, when a node requires a route to the destination and finds no existing route. In such a situation, the node initiates a route discovery process by broadcasting a Route Request. This process is complete once one or more routes to the destination are found as Route Replies propagate back to the source. After the route is created, it is maintained and updated till the destination is no longer accessible by any possible route or the source no longer needs that route.

Ad hoc On-demand Distance Vector (AODV) [20] routing protocol is briefly described as follows. The protocol is, as the name suggests, an on-demand ad hoc routing protocol. When a node S needs a route to some destination D, it broadcasts a route request (RREQ) message to its neighbors, including the last known sequence number for that destination. The RREQ is flooded through the network until it reaches a node that has a route to the destination. Each node that forwards the RREQ sets up a reverse route for itself back to node S. When the RREQ reaches a node with a route to D, that node generates a route reply that contains the number of hops necessary to reach D and the sequence number for D most recently seen by the node generating the reply, and unicasts that reply to S. All nodes that forward this reply back to the source S create a forward route to D. The sequence number associated with the routes is used to prevent routing loops from occurring, and a route with a higher sequence number is always preferred over a route with the lower sequence number. In order to maintain routes, AODV periodically

transmits a HELLO message, with a default rate of once per second. Failure to receive three consecutive HELLO messages from a neighbor is taken as an indication that the link to the neighbor in question is down. If available, information from the link layer can be used instead of HELLO messages to detect link breakages, reducing the overhead. When a link goes down, any upstream node that has recently forwarded packets to a destination using that link is notified via a route reply with infinite metric for the destination. The node S must then acquire a new route to the destination using the procedure above if it still wishes to communicate with that node.

1.4.3. Hybrid Routing Protocols

As explained above, both a purely proactive and purely reactive approaches to implement a routing protocol for a MANET have their advantages and disadvantages. A hybrid routing protocol combines the advantages of both into one scheme, taking advantage of proactive discovery within a node's local neighborhood, and using a reactive protocol for communication between these neighborhoods.

The Zone Routing Protocol (ZRP) [23] is such an example in MANET. The ZRP, as a hybrid routing protocol, is not so much a distinct protocol as it provides a framework for other protocols. The separation of a node's local neighborhood from the global topology of the entire network allows for applying different approaches – and thus taking advantage of each technique's features for a given situation. These local neighborhoods are called zones; each node may be within multiple overlapping zones, and each zone may be of a different size. The "size" of a zone is not determined by geographical measurement but is given by the number of hops to the perimeter of the zone. By dividing

the network into overlapping and variable-sized zones, the ZRP avoids a hierarchical map of the network and the overhead involved in maintaining this map. Instead, the network may be regarded as flat, and route optimization is possible if overlapping zones are detected. While the idea of zones often seems to imply similarities with cellular phone services, it is important to point out that each node has its own zone, and does not rely on fixed nodes. Figure 1.12 shows an example of the routing zone of node A.

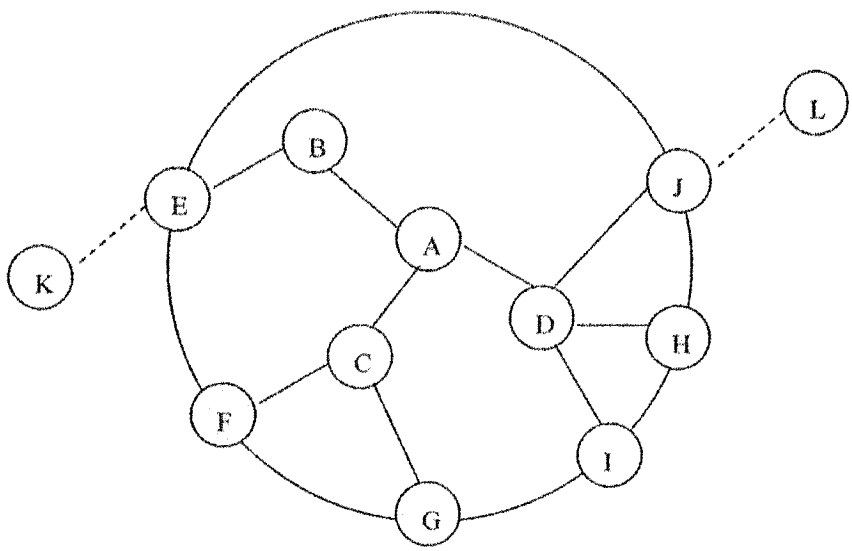


Figure 1.12 Routing Zone of node A

1.5. QoS Mechanisms

Quality-of-service (QoS) [8] is the qualitatively or quantitatively defined performance agreement between the service provider and user applications based on the connection requirements. The QoS requirements of a connection are a set of constraints such as bandwidth constraint, delay constraint, jitter constraint, loss ratio constraint, and so on. Because of the rising popularity of multimedia applications and real-time services, which require strict delay constraints, together with the potential commercial usage of Ad-Hoc networks, QoS support [10] and effective routing protocols in the WLAN have

become interesting topics.

1.5.1. QoS Support Mechanisms at MAC Layer

IEEE 802.11 Task Group E currently defines QoS enhancements to the above-described 802.11 MAC, called 802.11e [6]. The QoS support is realized with the introduction of Traffic Categories (TCs) as shown in Figure 1.14. MSDUs are now delivered through multiple backoff instances within one station, each backoff instance parameterized with TC-specific parameters. In the CP, each TC within the stations contends for a Transmission Opportunity (TXOP) and independently starts a backoff after detecting the channel being idle for an Arbitration Inter-frame Space (AIFS). A single station may implement up to 8 transmission queues realized as virtual stations inside a station with QoS parameters [7] that determine their priorities. If the counters of two or more parallel TCs in a single station reach zero at the same time, a scheduler inside the station avoids the "virtual collision".

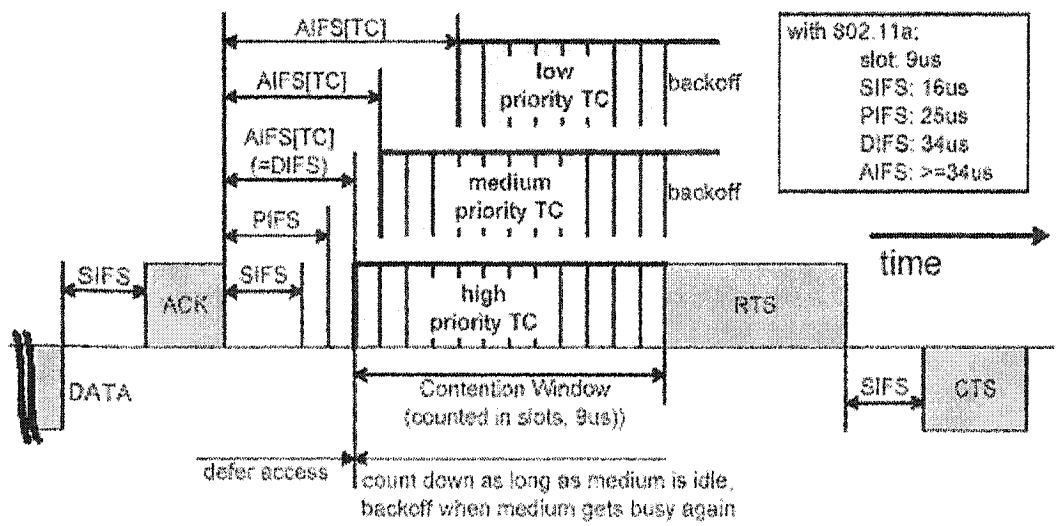


Figure 1.13: Multiple backoff of MSDU streams with different priorities.

1.5.2. QoS Routing in Ad-Hoc Networks

The QoS routing in Ad-Hoc network is difficult. First, to support the QoS, the link state information such as delay, bandwidth, jitter, cost, loss ratio and error ratio in the network must be available and manageable. However, getting and managing the link state information in the WLAN is by all means not trivial because the quality of a wireless link changes with the surrounding circumstances. The larger the size of the network, the more difficult it is to gather the up-to-date information. Second, the resource limitations and the mobility of hosts make things more complicated. The challenge that the QoS routing faces is to implement the QoS functionality with limited resources in a dynamic environment.

Chapter 2

Simulation Model

In this chapter, my thesis approach - the Pseudo Access Point (PAP) routing protocol and the double window algorithm – is presented. A general generic WLAN simulation software is designed using object-oriented programming technology in c++. Furthermore, the PAP routing protocol and the double window algorithm are also integrated, and a detailed design description and simulation model of the PAP are given in this chapter. Unified Model Language (UML) is employed.

2.1. Thesis Approach

2.1.1. Double Window Algorithm: QoS MAC

As the MAC protocol of WLAN, the standard CSMA/CA is a single window algorithm. In addition, although 802.11e provides a kind of QoS enhancement at the MAC layer, the "priority" defined in 802.11e is based on traffic categories of application. In other words, the CSMA/CA algorithm still has only one window for medium contention. The window is an exponential backoff window. The MAC coordination calculates the random backoff time using the following formula:

Backoff Counter = Random(0, CW)

Backoff Time = BackoffCounter × aSlot Time

Random() is a pseudo-random integer drawn from a uniform distribution over the interval [0,CW], in which CW (collision window) is an integer within the range of aCWmin and aCWmax. The random number drawn from this interval should be statistically independent among stations. aSlotTime is a constant.

Under low utilization, stations are not forced to wait very long before transmitting their frames. On the first or second attempt, a station will make a successful transmission within a short period of time. If the utilization of the network is high, the protocol holds stations back for longer periods of time to avoid the probability of multiple stations transmitting at the same time. This mechanism does a good job of avoiding collisions; however, stations on networks with high utilization will experience substantial delays while waiting to transmit frames.

The problem of the single window algorithm is that new generated packets have greater opportunity to use the channel than collided packets, because collided packets have a larger collision window than new generated packets. Whenever a collision occurs between new generated packets and previous collided packets, they will backoff together. Therefore, the earlier a packet is generated, the less opportunity it has to be transmitted, especially under high utilization. The double window algorithm in this thesis, a minor enhancement for the single window algorithm, has two windows: one is the backoff window which is the same as the standard CSMA/CA; the other is the priority window which is initialized by zero. When a collision occurs, the node with higher priority window size will be the winner to use channel and others will backoff. When a node

backoffs, its backoff window will binary exponentially backoff as defined in the standard CSMA/CA, while the priority window will increase by one.

2.1.2. Pseudo Access Point (PAP) Routing Prorocol

The PAP routing protocol is a proactive protocol. The key concept used in the protocol is Pseudo Access Points (PAPs), which are selected nodes that forward messages. All PAPs build a virtual backbone [22] of the network, usually called the infrastructured ad hoc network. By doing so, Global Positioning System (GPS) [24][25] is used to get each node's location. Based on this information, PAPs can be selected. The PAP routing operates as a table-driven protocol. For route calculation, each node calculates its routing table using a "Shortest Hops Path" algorithm based on the network topology.

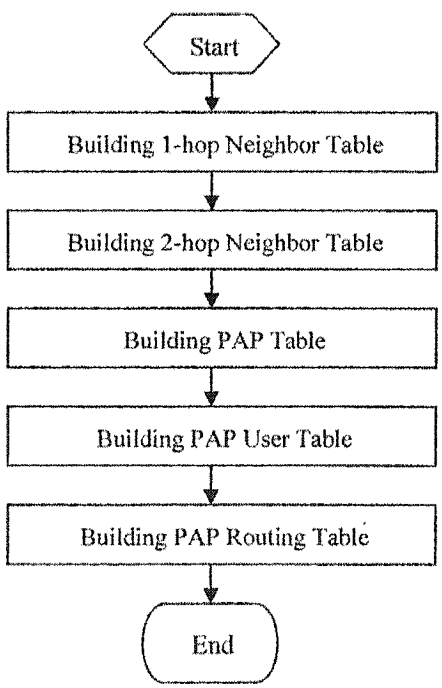


Figure 2.1: Flowchart of building the PAP routing table

The algorithm finds the minimum hop paths from the source node to all the destinations. PAP selection is the key point in the PAP routing protocol. The PAP set is

selected so that it covers all nodes that are two hops away. This means that the union of the neighbor sets of the PAPs contains the entire 2-hop neighbor set of a node. Each node selects its PAPs independently as illustrated in Figure 2.1.

The proposed heuristic is as follows:

1. Start with an empty PAP set

2. For each node y in the 1-hop neighbor set N, calculate D(y) – the degree (the number of neighbors) of y

3. Select as PAPs those nodes in N which provide the "only path" to some nodes in the 2-hop neighbor set N2

4. While nodes in N2 which are not covered exist, select a 1-hop neighbor as a PAP, which reaches the maximum number of uncovered nodes in N2.

5. As an optimization, process each node y in PAP. If PAP\{y} still covers all nodes in N2, y should be removed from the PAP set.

At step 4 of the above proposed heuristic of the PAP routing, if there is a tie, which node should be selected as a PAP? We suggest 3 ways to break the tie: Node with More Neighbors First, Node with Lower Moving Speed First, and Node with More Available Buffer First. Only one method can be selected at a time.

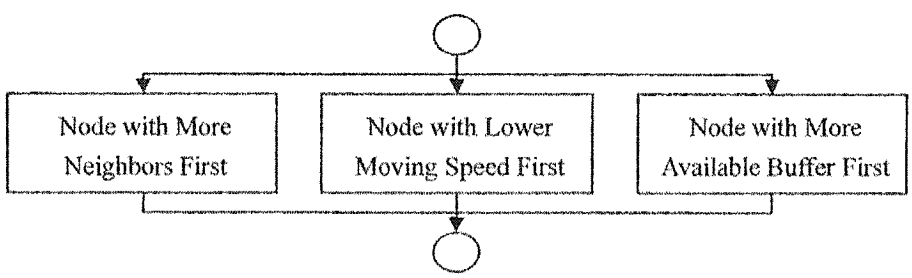


Figure 2.2: Three ways to break PAP tie

For the criteria of Node with More Neighbors First, if there is a tie, the one with higher number of neighbors is chosen. An example of how this algorithm works is given below in Figure 2.3 based on a network.

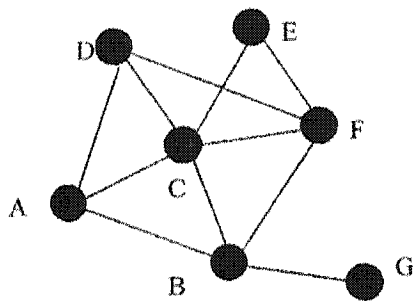


Figure 2.3: Network Example for PAP Selection

Nodes	1 hop Neighbors	2 hop Neighbors	PAP(s)
B	A, C, F, G	D, E	C

Table 1: PAP with higher number of neighbors is selected

From the perspective of node B, both C and F cover all of node B's 2-hop neighbors. However, C is selected as B's PAP as it has 5 neighbors while F only has 4 (the number of C's neighbors is higher than that of F).

For the criteria of Node with Lower Moving Speed First, if there is a tie, the one with lower moving speed is chosen. For the criteria of Node with More Available Buffer First, if there is a tie, the one with more available output buffer is chosen. In the chapter of simulations, we compare these algorithms to determine each performance.

2.2. Assumptions

A good ad hoc test bed is the basis for doing any wireless LAN simulations [27]. Several assumptions have been made to reduce the complexity of the simulation software. A short description of each of the assumptions is provided below:

- The "hidden terminal" problem is not addressed in the simulation software;
- No station operates in the "power-saving" mode (PS-Mode) [26]. By requiring all stations to be "awake" at all times, transmitted packets can be received immediately by the destination station;
- The network of nodes represents a random graph model, in which nodes are placed randomly in a given region;
- All nodes are identical, but they function independently on each other;
- If node is mobile, each node independently decides its movement: its speed and the direction;
- The nodes access the transmission channel using CSMA/CA technique;
- All the node links are bi-directional;
- One packet contains one message;
- Data and control messages have the same priority. Also, control messages have the same transmission time as data messages;
- A node can't receive and transmit messages at the same iteration. It can only receive OR transmit message during one iteration;
- Acknowledgement (ACK/NAK) and retransmission are not required.
- Transmission rate: all nodes can transmit messages in the rate of 1M bps.

- Transmission range: each node can transmit messages in the range of 250 m. Only the nodes within the distance of 250m can hear (receive) the transmitted messages.

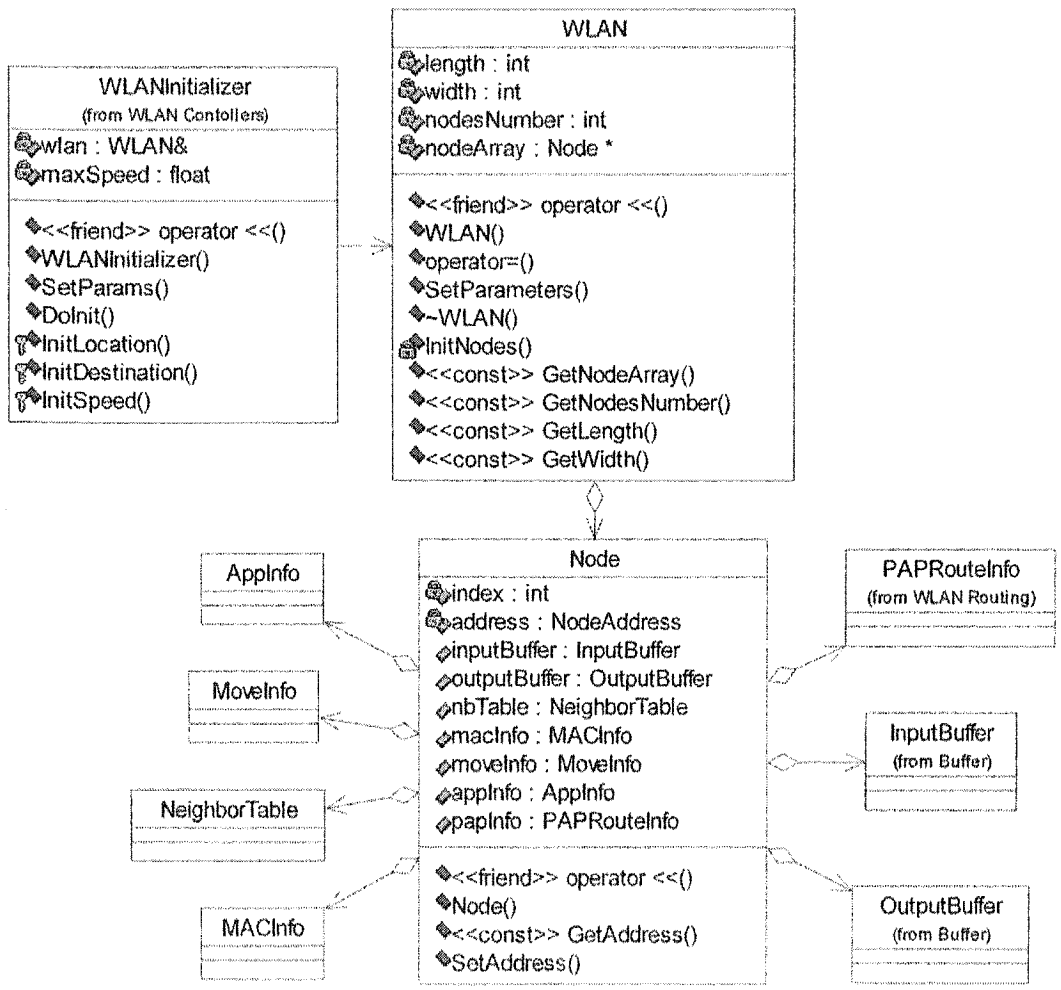
2.3. Design of the Network Node

Each WLAN includes many nodes (population). Each node has many attribute information, such as traffic information (AppInfo), routing information (PAPRouteInfor, NeighborTable), movement information (MoveInfo), MACInfor, Input/Output buffer, etc. The class diagram is shown in Figure 2.4.

Each WLAN is a randomly generated network for the simulation. It can generate any number of nodes within an arbitrarily determined area with random graph. For example, we can generate a random graph network with 200 nodes within 2000*2000 m^2 area. Attributes of WLAN and nodes are defined as follows:

- Node number: Node number is the node index beginning from zero.
- Address: This is the node address. To simplify the simulations, the format of the node address is not the MAC address or IP address but just the node index.
- MoveInfo includes attributes related with the node movement information, such as position (node position in the given area), speed (node moving speed), and direction (node moving direction).
- Input/Output buffer: The output buffer stores the messages to be sent, while the input buffer stores the received messages.

- Neighbor table: It contains all the neighbor node addresses. Owing to the node mobility, the neighbor table will change from time to time.
- PAPRouteInfo contains all attributes related only to the PAP routing protocol: 2-hop neighbor table contains all the 2–hop neighbor node addresses; PAP table contains all the addresses of the PAP that have been selected by this node; PAP user table contains all the addresses of the PAP users that select this node as their PAP; topology table contains the topology information of the network; and routing table contains the routing information.



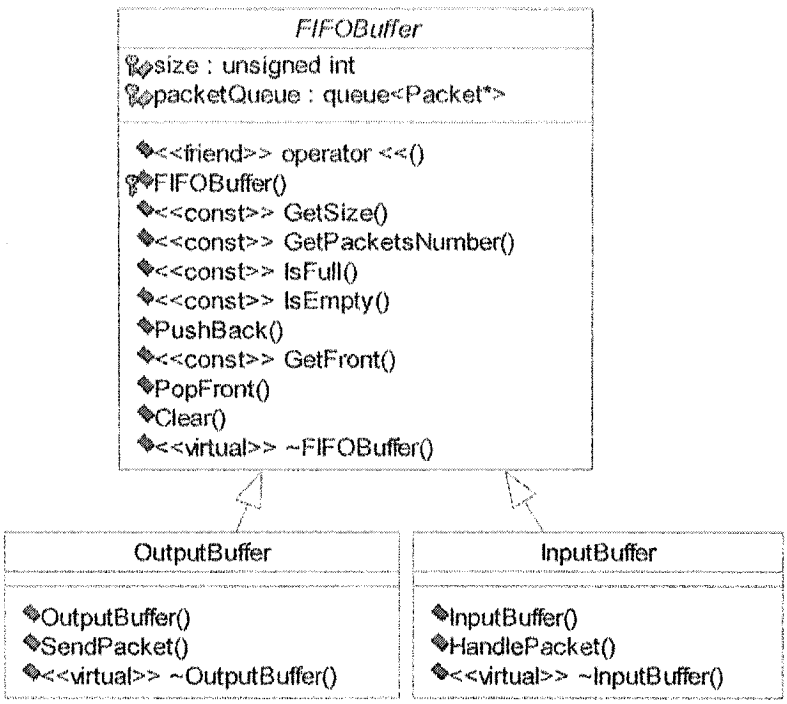


Figure 2.4: Class diagram of nodes

2.4. Design of Node Mobility

2.4.1. Uniform Random Number Generator

The uniform random number generator is fundamental to the whole simulation. The type of this generator is Mersenne twister (MT) [12] which is in rather good performance. Experts consider this an excellent random number generator.

The generator, a Singleton [13] class, will be used by many modules. The function Random() gives a 32-bit floating point random number in the interval 0~1; IntRandom() gives a 32-bit integer random number in the interval (min, max).; and BitsRandom() gives 32 random bits. From then on, we will symbol the generator as U(0,1).

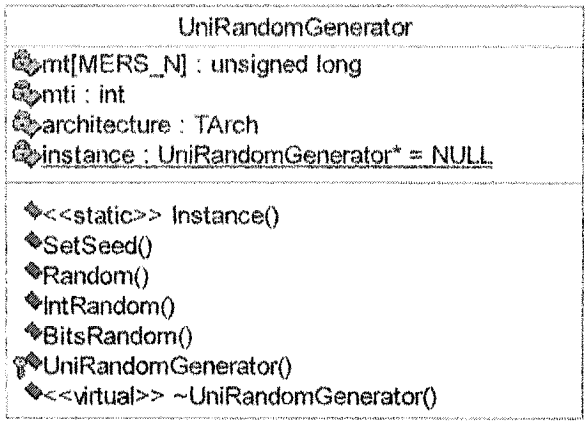


Figure 2.5: Class diagram of uniform random generator

2.4.2. Location Initialization and Updating

(I) Initial Location (executed only once at the beginning of the simulation):

The default area is a 1000m × 1000m square. Then for each user, call U (0, 1) two times to get two uniform random variables - Var1 and Var2. Thus the initial location is (Xi, Yi) = (Var1 × 1000, Var2 × 1000).

(II) Moving Speed Initialization (executed only once at the beginning of the simulation):

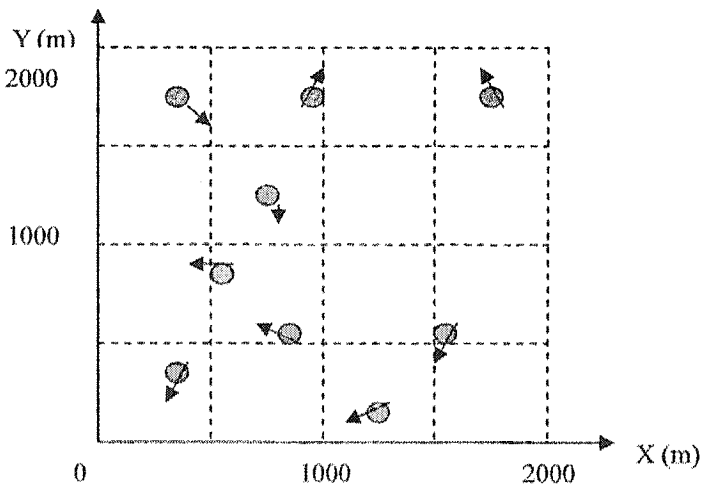


Figure 2.6: Example of mobile nodes

The default maximum speed is 20m/s, e.g. $20\times60\times60$ = 72Km/H. Then for each user, call U (0, 1) to get a uniform random variable Var. Thus the user's moving speed is Vi = Var×20m/s. This speed is fixed for the rest of the simulation.

(III) Mobility:

After a certain number of iterations (updatePeroid), each node's mobility will be checked. Each user calls U (0, 1) to get a uniform random variable Var. If the Var is greater than the movementThreshold (0 ~ 1), the node will be in the mobile status.

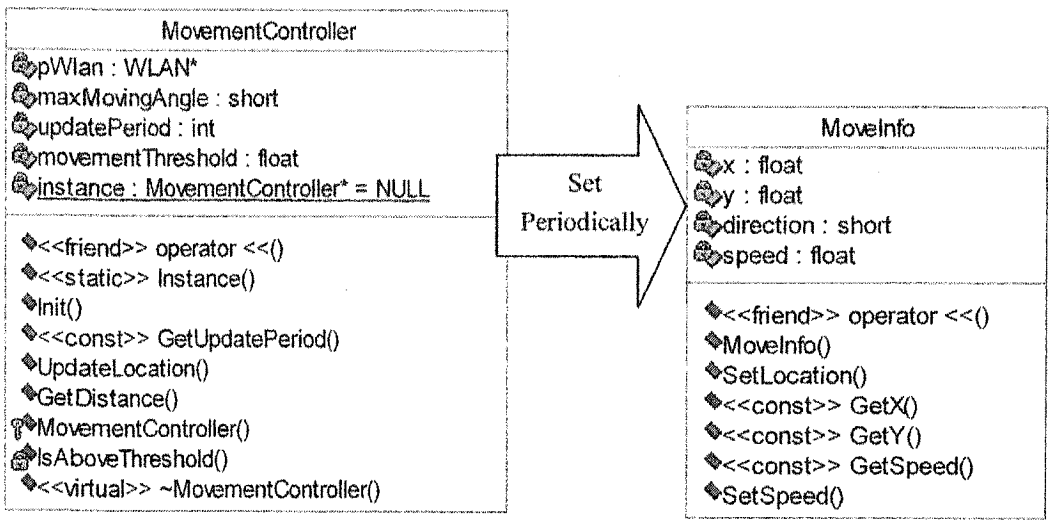


Figure 2.7: Class diagram of mobile controller

(IV) Current Location Computation:

When a node is in the mobile status, it calls U (0, 1) to get a uniform random variable Var. Thus the moving orientation is $\theta = \text{Var}\times 360°$. Then the current location is given by

$$Xi = Xi\_old + \cos\theta \times Vi \times \Delta t$$

$$Yi = Yi\_old + \sin\theta \times Vi \times \Delta t$$

2.5. Design of the Physical Layer: Channel

All mobile nodes have one or more network interfaces that are connected to a channel (see the following Figure 2.8). A channel represents a particular radio frequency with a particular modulation and coding scheme. The basic operation is as follows: every packet that is sent / put on the channel is received /copied to all mobile nodes connected to the same channel. When a mobile node receives a packet, it first determines if it is possible to receive the packet. This is based on the transmitter range, the distance that the packet has traveled, and the channel quality.

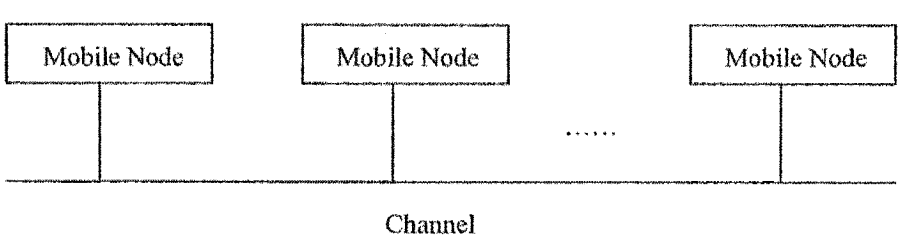


Figure 2.8: Shared media model

A two-state continuous-time Markov chain [18] is used to represent the burst error model characterizing fading in the communication channel. The state "GOOD" indicates that the channel is operating with a very low bit error rate The state BAD indicates the channel is operating in a fading condition with a higher error rate. The transition rate from state GOOD to state BAD is denoted by α, while the transition rate from state BAD to state GOOD is denoted by β. A frame is considered to be corrupt if it contains one or more bit errors.

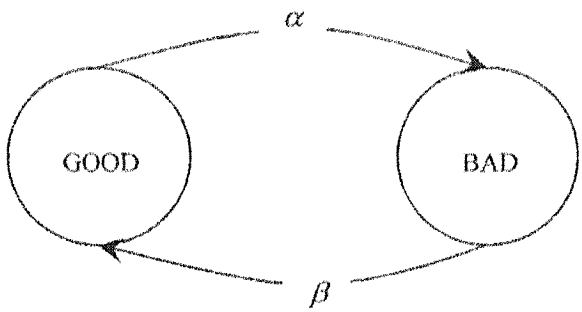


Figure 2.9: A two-state continuous-time Markov chain

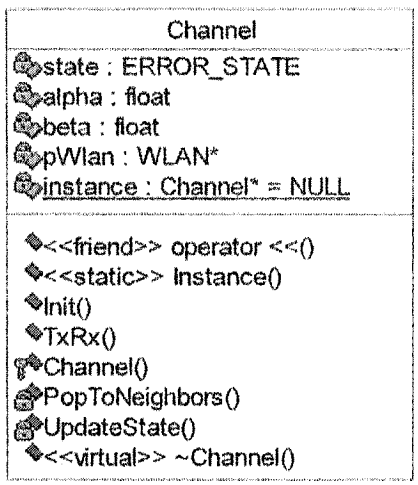


Figure 2.10: Class diagram of the Channel module

The Channel singleton is responsible for transferring a packet from a sender to its one-hop neighbors depending on this model. The simulation model uses the above error model to determine whether each transmitted packet has been transmitted successfully or not. A frame can be sent to its neighbors successfully when the channel is in state GOOD, and will be dropped during state BAD.

We use channel transmission rate 1Mbps and packet size 20000 bits. So the time to transmit one packet is 0.02 s. We call this one iteration and the program runs 10000 iterations in order to get a stable network performance.

2.6. Design of the MAC Layer

2.6.1. Flowchart of the CSMA/CA

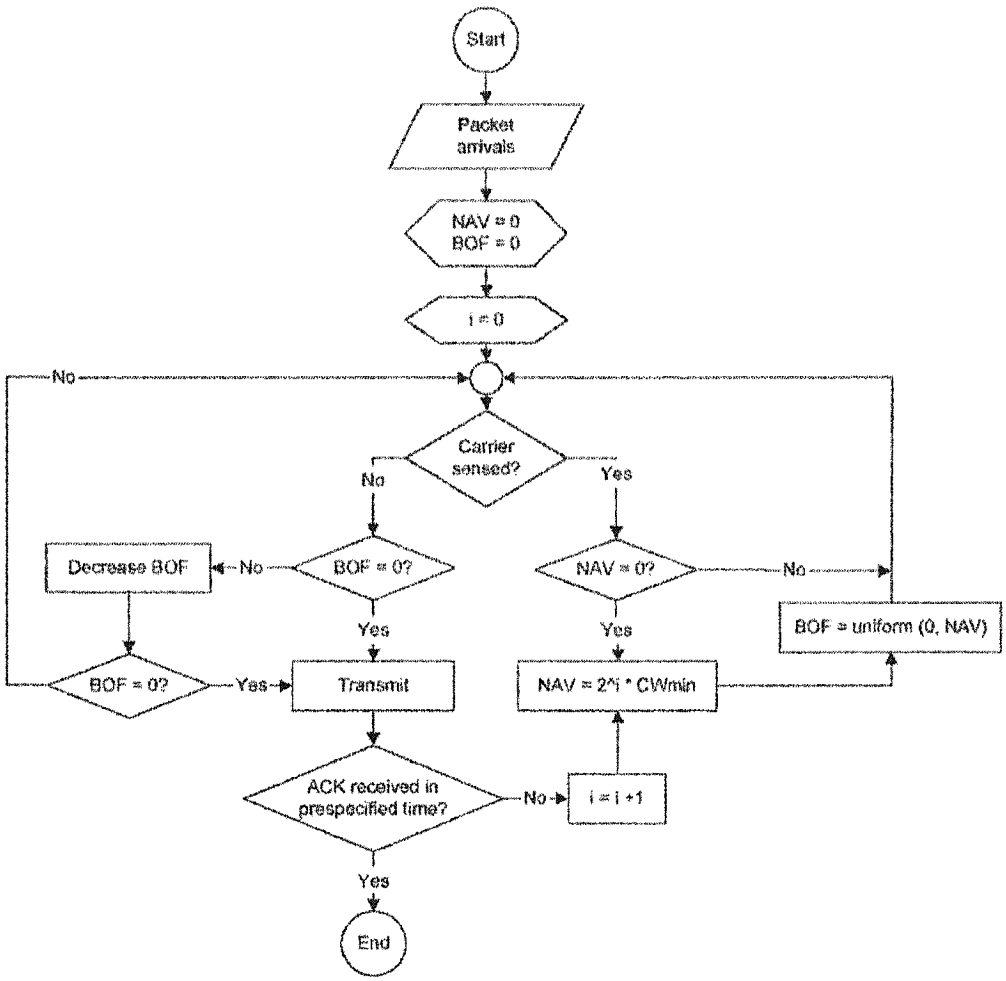


Figure 2.11: Flow chart of CSMA-CA

The CSMA/CA algorithm is the key to the MAC layer and can be readily represented in the form of a flowchart as shown in Figure 2.11. In this figure, the NAV is the Network Allocation Value and denotes the width of the interval from which a uniformly distributed backoff period (BOF) is selected. In case of a collision, the NAV is increased using binary exponential backoff (NAV $= 2^{i} \times CW\min$), where CWmin is the minimum

backoff window, which is the size of the NAV when i is equal to zero. In practice this increase is stopped when "i" reaches an upper value.

2.6.2. State Transition Diagram for the CSMA/CA

The operation of the CSMA-CA can be described in terms of the state transition diagram shown in Figure 2.12.

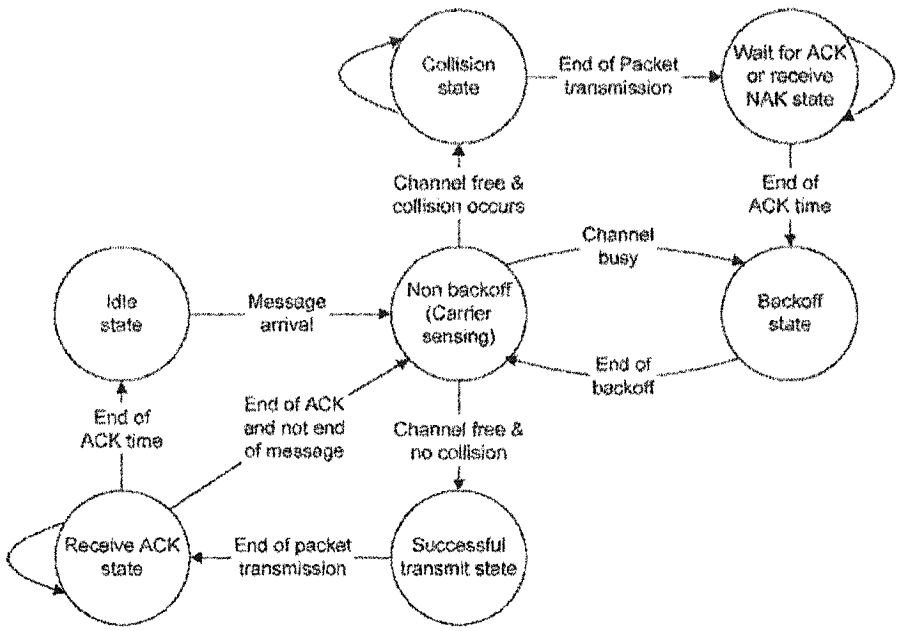


Figure 2.12: State transition diagram for CSMA-CA.

Initially, each station is in the idle state. When a new message arrives, it is stored in a transmit buffer and the station moves to a non-backoff carrier sensing state. Depending on whether the channel is busy or not busy, the station moves either to a backoff state or to a transmit state. When in the backoff state, a random time –uniformly chosen from a progressively increasing interval – is used for backoff. So long as the station is in the backoff state, it continues to sense the channel and decrement the backoff time only when the channel is free. When the backoff time decreases to zero, the station moves from the

backoff state to the non-backoff state. On the other hand, when a station is in the transmit state, two possibilities exist: either the transmission is done successfully (as indicated by the reception of an ACK signal), or the transmission is not successful due to the collision with transmissions from other station(s) (as indicated by the absence of the ACK or the reception of the NAK). In the first case, the station moves to the ACK state, while in the second case the station moves to the collision state. When in the collision state, the backoff interval is increased, a new backoff time is selected, and the station moves to the backoff state. On the other hand, a station in the ACK state may either return to the non-back-off carrier sensing state (if more units of the original message are to be transmitted), or else return to the idle state (in case the message transmission is completed).

2.6.3. Class Diagram of the MAC Layer

The MAC Layer includes two parts: the entity part and the controller part. The entity part contains the static information of the MAC Layer, while the controller part is responsible for the behaviors of the MAC layer. Figure 2.13 is the class diagram of the MAC Layer. The QoSCollisionWindow adds a priority window to the collision window, and the QoSMAController module implements the Double Window Algorithm.

2.7. Design of the Routing Unit

2.7.1. Design of the Default Routing Protocol: θ Routing Protocol

The generic θ Routing Protocol is default for the simulator. This protocol will be

used to be compared with the PAP routing protocol. It operation is given in Figure 2.14.

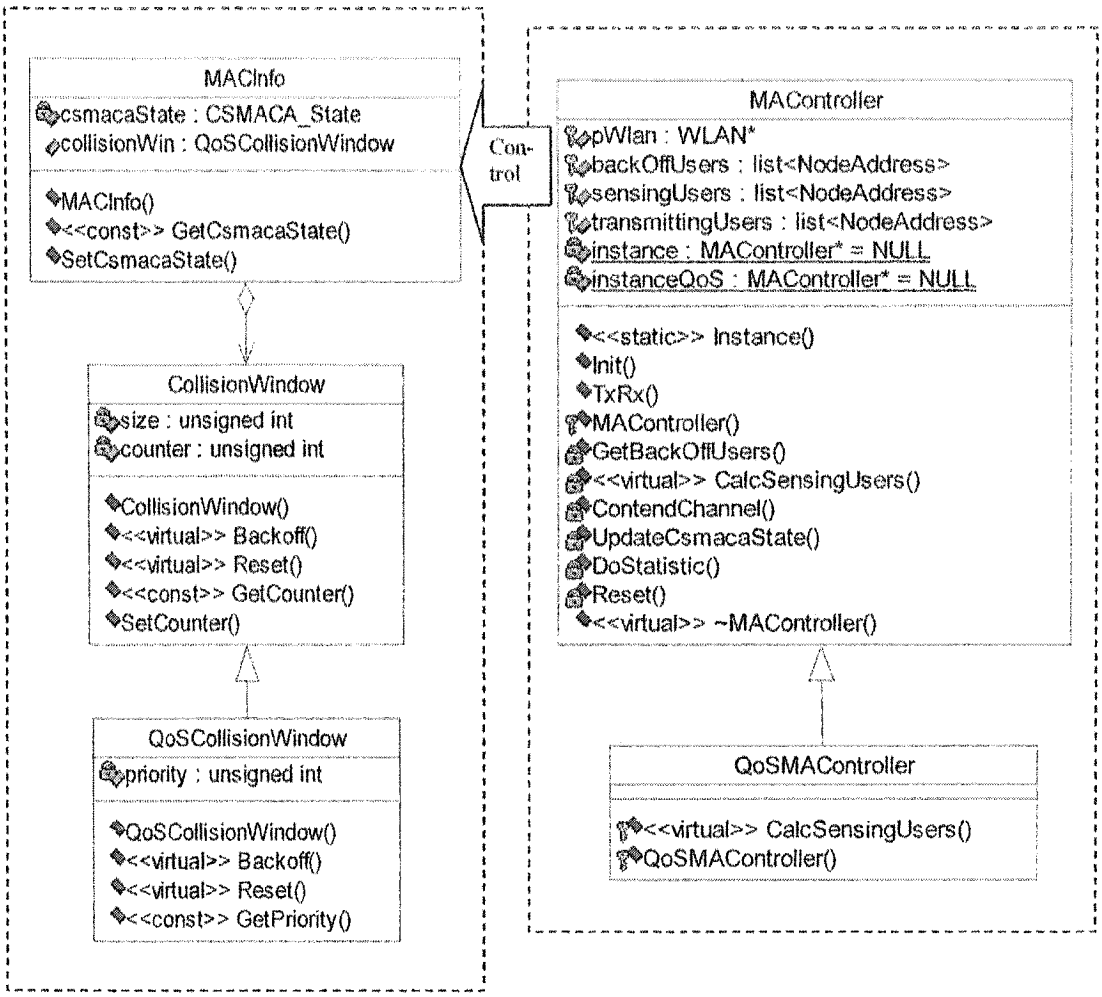


Figure 2.13: Class diagram of the MAC Layer

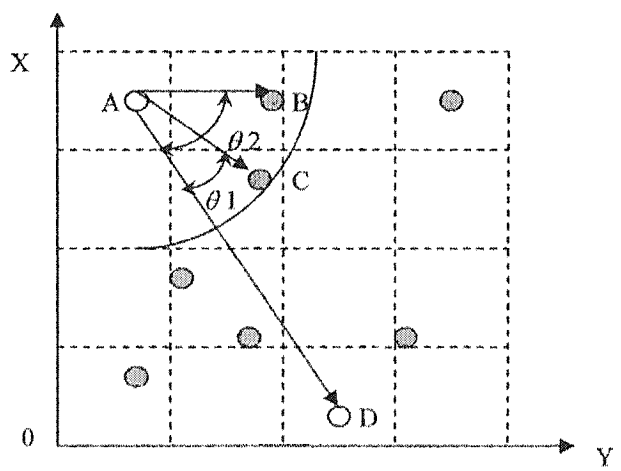


Figure 2.14: Operation of θ Routing Protocol

A is the source and D is the destination. A can not send packets to D via any other nodes except either B or C, because A, B, and C are in the same cluster while others are not, e.g. AB<=250m, AC<=250m, while AD>=250m. In the θ routing, A will send packets to D via C but not via B because $\theta 1 < \theta 2$. When C gets these packets from A, he will repeat the same operations to find the next node to D.

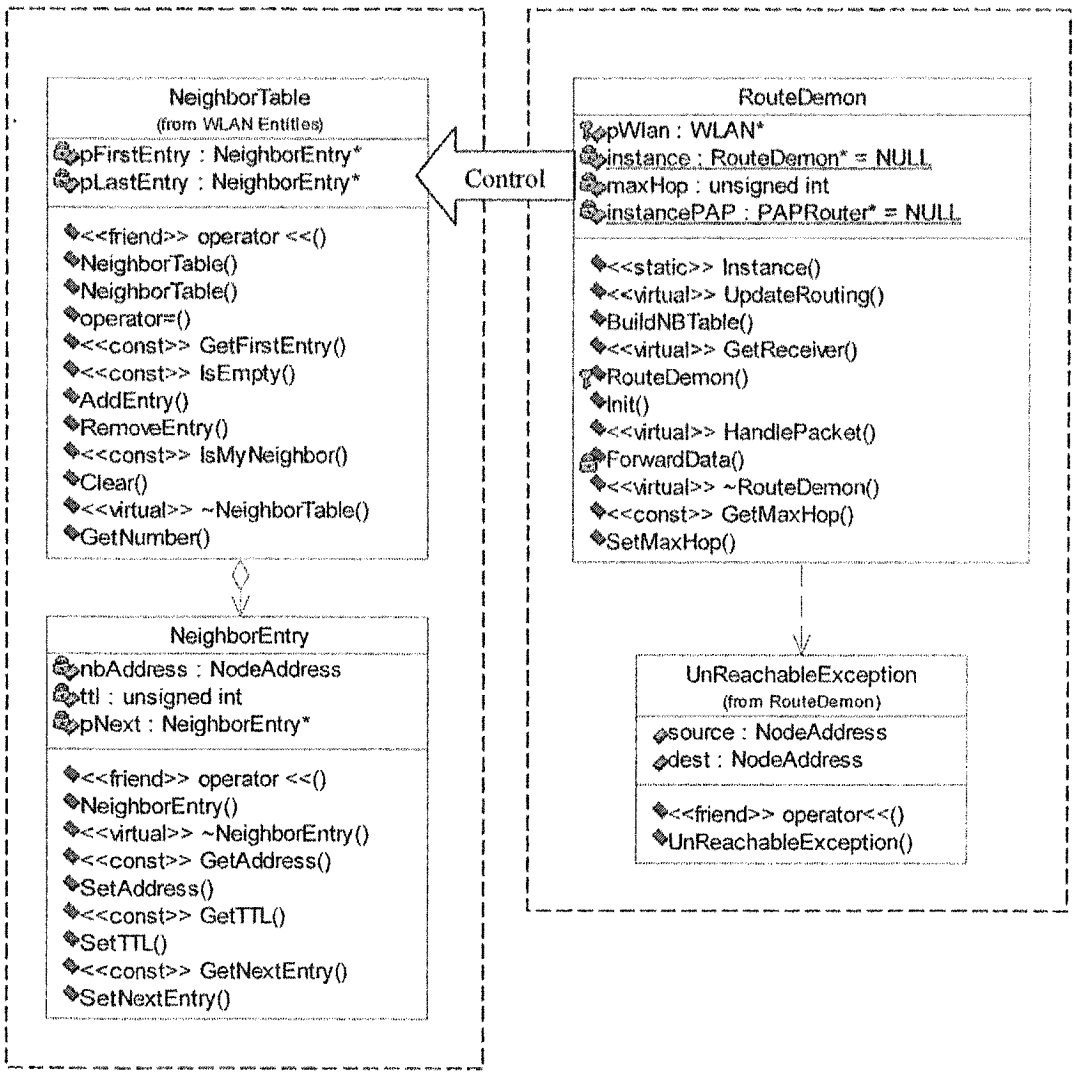


Figure 2.15: Class diagram of the RouteDemon module

Figure 2.15 is the class diagram. The base class RouteDemon is based on the θ routing protocol. Users can create their own route demon to evaluate new routing

protocols by inheriting the RouteDemon.

In the neighbor table, each node records the information about its one-hop neighbors. The information is recorded in the Neighbor table as a neighbor entry. The neighbor table may have the following format to record these entries:

	nbAddress	TTL	pNext
1	11	3	18
2	25	5	9
...			

Each entry in the table consists of nbAddress, TTL, and pNext. It specifies that the node with address nbAddress is a one-hop neighbor to this local node. The pNext is a pointer pointing to the next entry. Each neighbor entry has an associated holding time TTL, upon expiry of which it is no longer valid and hence removed.

The neighbor information can be easily retrieved through the GPS facility [25], which can tell each node's position. Therefore, the neighbor table doesn't contain a sequence number value to ensure its freshness. Of course, the neighbor information can also be retrieved by exchanging HELLO packets, but this is not our focus.

2.7.2. Design of the PAP Routing Protocol

To evaluate the PAP routing protocol, class PAPRouter is designed and inherited from the RouteDemon as follows. The PAP Routing protocol defines rules of how to select the PAP, how to calculate one PAP's users, how to break the tie of the PAP confliction, how to optimize the PAP list, etc. After the PAP user table is built, the routing table will be built on it.

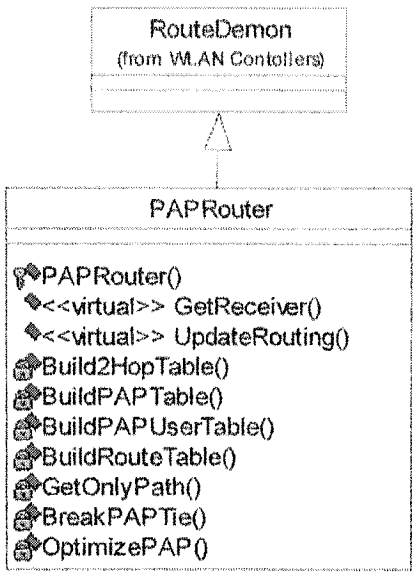


Figure 2.16: Class diagram of the PAPRouter module

Each node maintains a routing table which allows it to route messages to the other destinations in the network. The routing table is re-calculated locally each time the neighbor table is changed. The update of this routing table does not generate or trigger any messages to be transmitted, neither in the network nor in the one-hop neighborhood. The following procedure is executed to calculate (or re-calculate) the routing table:

1. All entries in the routing table are removed.
2. The new entries are recorded in the table starting with the one-hop neighbors (h = 1) as the destination nodes. For each neighbor entry in the neighbor table, a new route entry is recorded in the routing table where both dest_address and next_address are set to the address of the neighbor and the distance is set to 1.
3. Then the new route entries for the destination nodes which are (h + 1) hops away are recorded in the routing table. The following procedure is executed for each

value of h, starting with h = 1 and incrementing it by one each time. The execution will stop if no new entry is recorded in iteration.

a. For each topology entry in the topology table, if its dest_address does not correspond to the dest_address of any route entry in the routing table AND its last_PAP corresponds to the dest_address of a route entry whose distance is equal to h, then a new route entry is recorded in the routing table where :
b. the dest_address in the routing table is set to the dest_address in the topology table;
c. the next_address is set to the next_address of the route entry whose dest_address is equal to the last_PAP;
d. the dist_address is set to h + 1.

2.7.3. Design of the PAP Routing Entities

The PAP routing protocol includes six main entities: two-hop neighbor table, PAP table, PAP user table, topology table, and routing table. The class diagram of these entities follows.

2.7.3.1. Two-hop neighbor table: TwoHopTable

In the 2-hop neighbor table, each node records the information about its 2-hop neighbors. The information is recorded in the 2-hop neighbor table as a 2-hop neighbor entry. The TwoHopTable is a list of Two-hop neighbors.

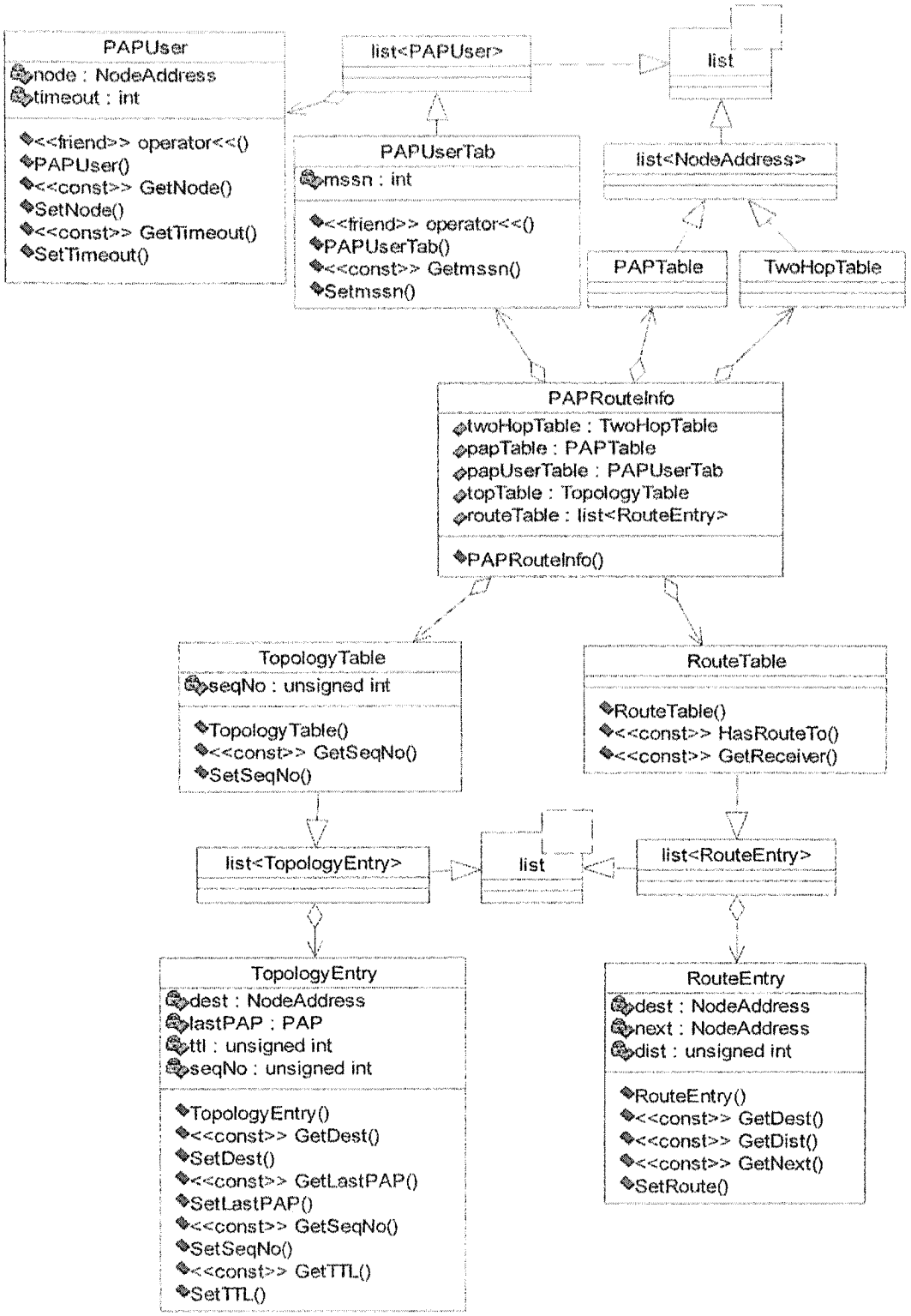


Figure 2.17: Class diagram of PAP routing protocol entities

2.7.3.2. PAP table: PAPTable

Each node selects one of its one-hop neighbors as its PAP and put the PAP information in the PAP table. The information is recorded as a PAP entry in the PAP table, which is a list of PAP.

2.7.3.3. PAP user table: PAPUserTab

Each node may have several PAPs and each PAP may have several users. After selecting PAPs, each node also builds its PAP user table. This table gives us an idea of how many one-hop neighbors treat this node as their PAP. The PAP user table may have the following format to record the entries:

Index	Node address	TTL
1	12	1
2	25	1
3	...	...

A sequence number MSSN is assigned to this table. It specifies that the PAP users are most recently modified with the sequence number MSSN. The node modifies its PAP user table according to its neighbor table which can be built by exchanging the HELLO messages or through the GPS, and it increments this sequence number on each modification.

2.7.3.4. Topology table: TopologyTable

The Topology Table is an intermediate table, which is based on the PAP user table

and used for building the routing table. Each node in the network maintains a topology table, in which it records the information about the topology of the network obtained from the PAP user table. Based on this information, the routing table is calculated. A node records information about the PAPs of other nodes in the network in its topology table as a topology entry, which may have the following format:

sequence_num

	dest_address	last_PAP	sequence_num	TTL
1	8	24	24489	5
2	17	9	24460	4
...				

Each entry in the table consists of dest_address, last_PAP, sequence_num, and TTL. It specifies that the node dest_address has selected the node last_PAP as a PAP. Therefore, the node dest_address can be reached in the last hop through the node last_PAP. Furthermore, each topology entry has an associated holding time TTL, upon expiry of which it is no longer valid and hence removed. Thirdly, the topology table also contains a sequence number value sequence_num. Every time when a node updates its topology table, this sequence_num is incremented to a higher value.

2.7.3.5. Routing table: RouteTable

The routing table is based on the information contained in the neighbor table and the topology table. Therefore, if any of these tables is changed, the routing table is re-calculated to update the route information about each destination in the network. The route entries are recorded in the routing table in the following format:

	dest_address	next_address	distance
1	34	15	4
2	13	27	3
...			

Each entry in the table consists of dest_address, next_address, and distance. The distance specifies that the node identified by the dest_address is estimated to be "distance" hops away from the local node. The one hop's neighbor node with address next_address is the next hop node in the route to the dest_address. Then new entries are recorded in the table for each destination in the network for which the route is known. All the destinations for which the route is broken or partially known are not entered in the table.

2.8. Design of the Application Layer

2.8.1. Node State: Active/Idle

The probability of a node generating traffic is λ (calls) per iteration < 1. For each large iteration, each idle user executes the following flowchart:

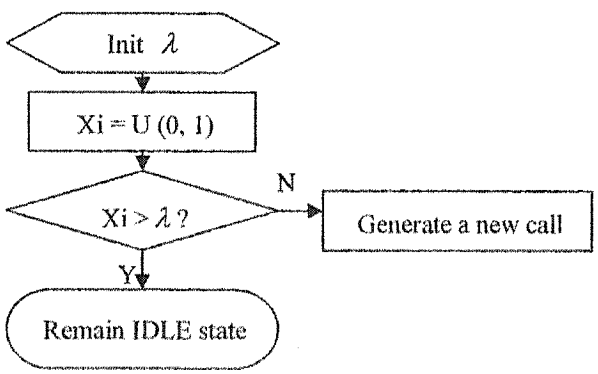


Figure 2.18: Flowchart of node state (Active/Idle)

A node calls function U (0, 1) to get a uniform random variable. If the variable is less than the load factor λ, then a new call is generated; otherwise, the node remains the IDLE state. We assume a node which has generated a call remains active to the end of the simulation.

2.8.2. Traffic State: On/Off

When a node is in state "Active", it may create a packet if its traffic state is "ON". At the state "ON", packets are generated, while at the state "OFF", no packets are generated.

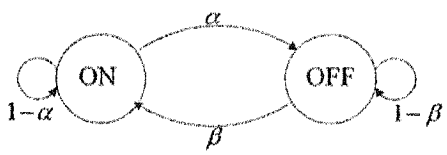


Figure 2.19: State transmission diagram of traffic state (ON/OFF)

It is easy to get the probability of a node's traffic state:

$$P(ON) = \beta / (\alpha + \beta) = Pu, \text{ and}$$

$$P(OFF) = \alpha / (\alpha + \beta) = 1 - Pu.$$

During every packet time, if a node is OFF, it calls U (0, 1) to get Xi; and then if $Xi \leq \beta$, it becomes ON and generates one packet during every smallest iteration. If a node is ON, it generates a new packet, and calls U (0, 1) to get Yi; and then if $Yi \leq \alpha$, it becomes OFF. Each node maintains a buffer where all packets generated by itself will be sent on the 802.11 MAC Layer.

All the generated messages will be put into the output buffer waiting for transmission. And the input buffer will detect and receive messages from the channel. The messages in the input buffer will be processed one by one according to their message type. Messages

in both buffers will be processed or transmitted according to the principle of "First In First Out (FIFO)".

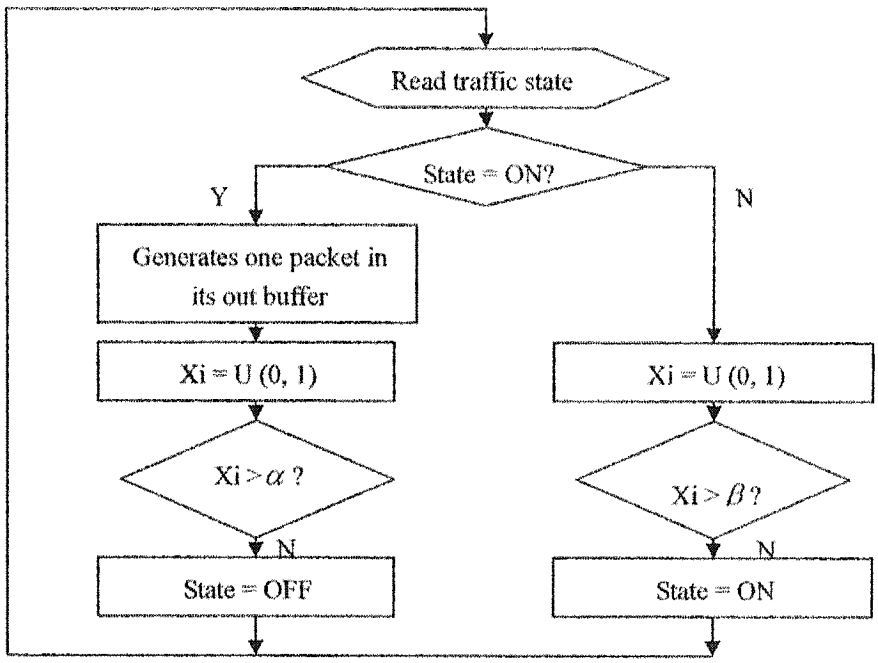


Figure 2.20: Flowchat of traffic state (ON/OFF)

2.8.3. Definition of Packet

Each packet includes source, destination, sender, and receiver. The source Address is the address of the node that originally generates the message; the sender address is the address of the node that is sending the message during certain iteration; the destination address is the destination of the message; and the receiver address is the next hop where the message will go. The sender and the receiver are always neighbors. An example of their relationship is given in Figure 2.21.

- Source: A
- Pair of sender ->receiver: A->B, B->C, and C->D
- Destination: D

Figure 2.21: Example of different addresses

Packets will be generated by the DataProducer module and be transmitted to the DataConsumer module. The class packet is a base class and can have many sub-classes, such as DataPacket, HelloPacket, and PAPPacket. Figure 2.22 is the class diagram.

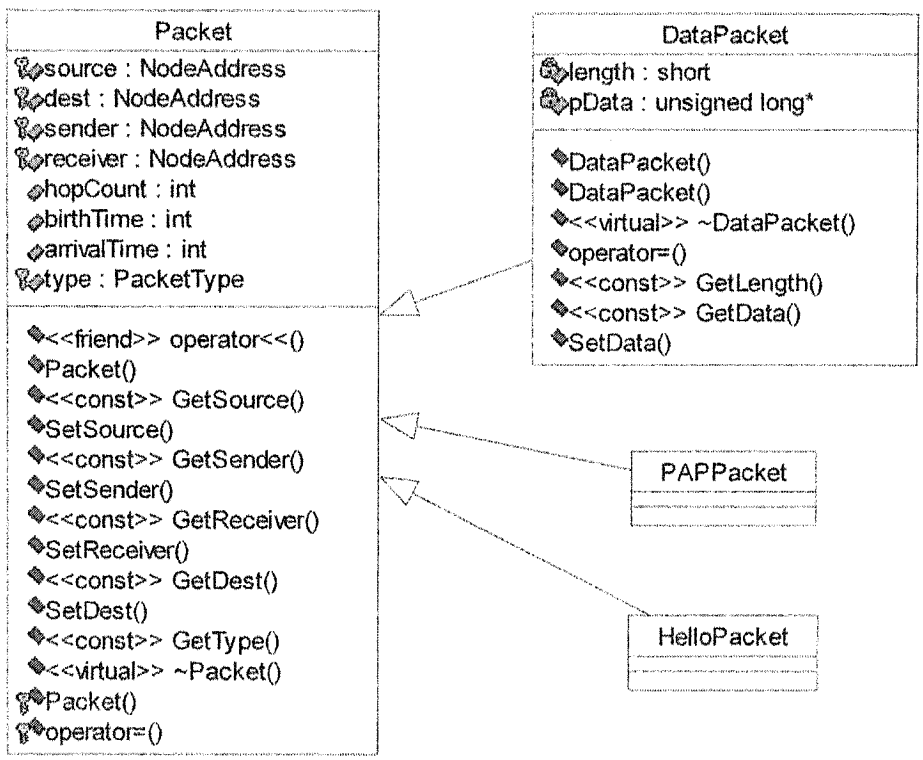


Figure 2.22a: Class Diagram of Packet

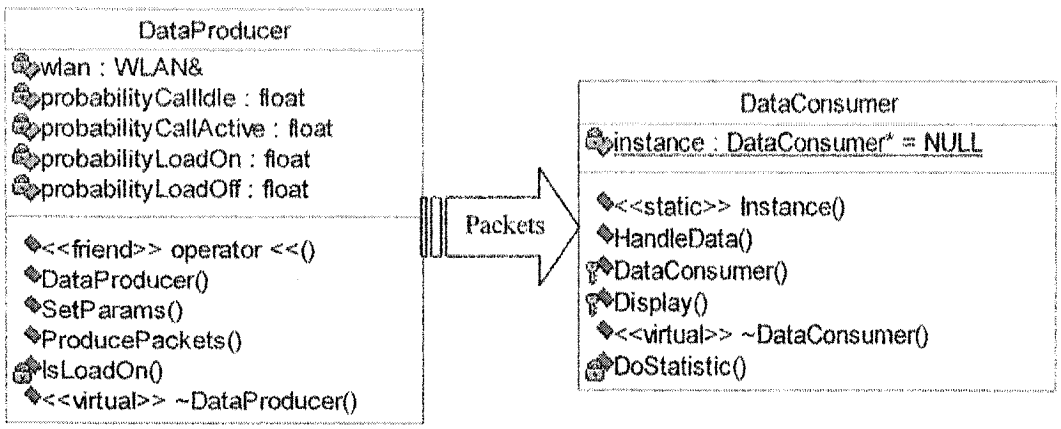


Figure 2.22b: Software module of the Application Layer

The field PacketType is reserved for users to define types of packets in the simulation; the hop counter counts the number of hops through which the packet travels; the birth time records the iteration during which a packet is generated; the arrival time records the iteration during which the packet reaches its receiver or destination. The difference between the final arrival time and the birth time is the end-to-end delay of one packet. In the DataPacket, data is generated randomly by the UniRandomGenerator.

We assume that all packets generated by each node have only one destination node to simplify our simulation. At the start of the simulation, create a pair of nodes (source and destination) only once. For instance, if there are 200 nodes in a WLAN, for each new call of the user j, call U(0,1) to create a uniform random variable Xj; then, its corresponding destination is Dj = 200*Xj.

2.9. Definition of Input/Output Parameters

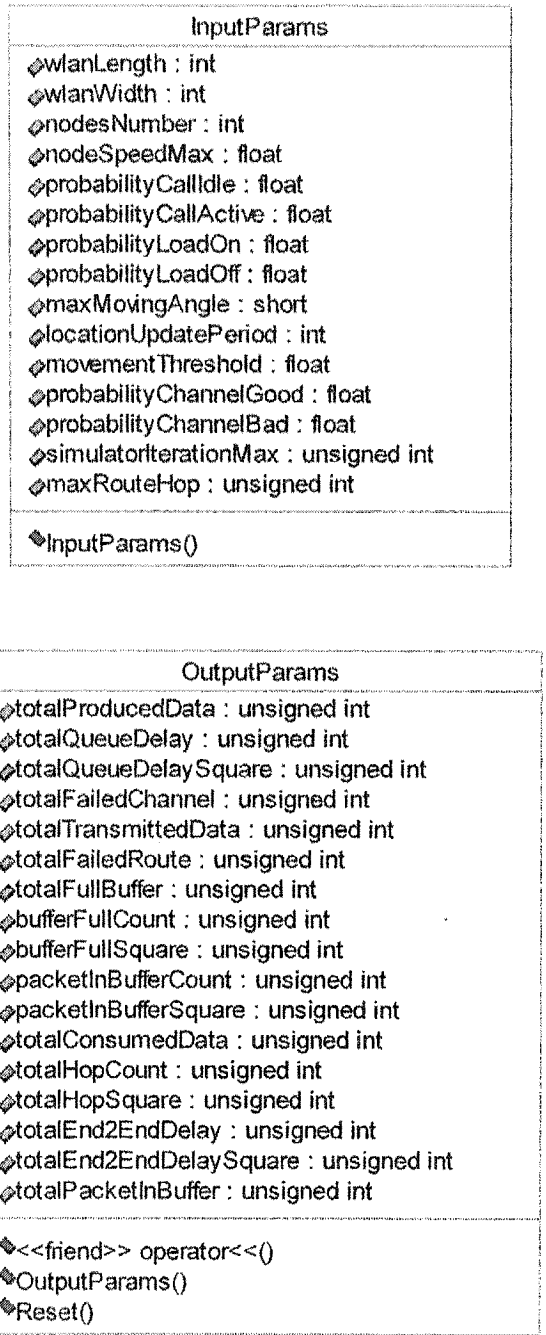


Figure 2.23: Class diagram of input/output parameters

There are some input variables in our simulation, by proper setting of which we can achieve better performance in our simulation. The following is the input parameters by category:

- Node density: Three parameters are compulsory for the WLAN simulation: wlanLength, wlanWidth, and nodesNumber. All nodes are generated initially within a certain area, such as 2000*2000 m^2. It is easy to get the node density from the Number of the nodes and the area.

  Node density = nodesNumber / (wlanLength * wlanWIdth)

- Node mobility: After every period (locationUpdatePeriod), the WLAN simulator will check each node's mobility. By the parameter of the movementThreshold which ranges from 0 to 1, and the UniRandomGenerator, we can know whether a node is in movement or not. Each mobile node moves at a speed from 0 to the maximum value (nodeSpeedMax). Each mobile node may move at a direction randomly from 0 to the maxMovingAngle (less than 360 degree). The UniRandomGenerator will give the specific maximal speed and maximal moving angle. These parameters will be used by the MovementController module.
- Traffic load: We define three main traffic load parameters in our simulation: probabilityCallActive, probabilityLoadOn (beta), and probabilityLoadOff (alpha). These parameters are used for configuring the DataProducer module.
- Iteration number: The program runs in iterations. In each iteration one node can send or receive only one packet. In order to have a stable network performance,

the total number of iterations should be large enough. Here we set it 10000 after several tests.

- Max hop count: Each packet has Max hop count to determine how long the packet can travel. If a packet has traveled this Max hops, it is discarded from the network.

The following table is the given value range of input parameters:

	Parameter Name	Unit	Value Range
1	wlanLength	m	500 ~ 3500
2	wlanWidth	m	500 ~ 3500 (=wlanLength)
3	nodesNumber		1 ~ 600
4	nodeSpeedMax	m/s	0 ~ 40
5	probabilityCallIdle		0.0 ~ 1.0 (not effective)
6	probabilityCallActive		0.0 ~ 1.0
7	probabilityLoadOn (beta)		0.0 ~ 1.0
8	probabilityLoadOff (alpha)		0.0 ~ 1.0
9	maxMovingAngle		0.0 ~ 1.0
10	locationUpdatePeriod	iteration	40
11	movementThreshold		0.0 ~ 1.0
12	probabilityChannelGood		0.0 ~ 1.0
13	probabilityChannelBad		0.0 ~ 1.0
14	simulatorIterationMax		10000
15	maxRouteHop		(10 ~ 100%) * nodesNumber

Table 2: Given value range of input parameters

The output parameters will be set by the WLANsimulator module after each simulation and will be used by the WLANAnalyzer module to do some statistics.

2.10. Definition of Statistic Parameters: StatisticParams

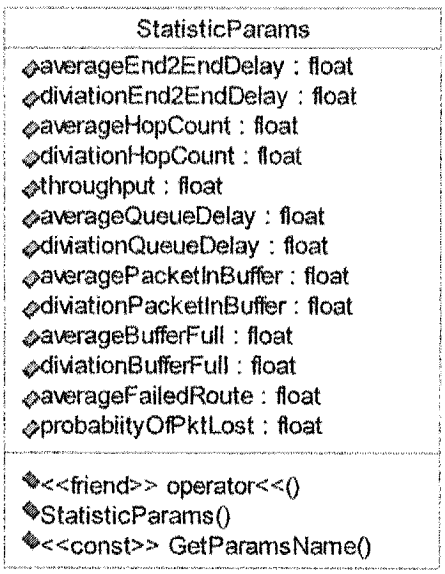


Figure 2.24: Class diagram of statistic parameters

Choosing the correct metrics [22] to use in the evaluation of the WLAN is vital to the result and the validity of the evaluation. The metrics we have used are end-to-end delay, queuing delay, throughput, etc. Based on a group of the value of output parameters, the WLAN analyzer can do some statistics. The statistic result will be saved in a corresponding output file. The class diagram of statistic parameters is shown in Figure 2.24.

2.10.1. Average and Variance of End-to-End Delay

The end-to-end delay is defined as the total time that one packet takes from its generation to its destination. It includes the queuing delay (the time a packet waits in the buffer) and the transmission delay. The mean of the end-to-end delay is calculated by dividing the sum of all end-to-end delays of all packets by the total number of successfully transmitted packets. The sample variance of the end-to-end delay is calculated by dividing the square sum of the difference between the average end-to-end delay and the individual end-to-end delay by the total number of successfully transmitted packets. The formulas for the average of the end-to-end delay ($\overline{D_{end2end}}$) and the variance of the end-to-end delay ($v_{end2end}$) are as follows:

$$\overline{D_{end2end}} = \frac{\sum_{i=1}^{Ns} D_i}{Ns}$$

$$v_{end2end} = \frac{\sum_{i=1}^{Ns} (\overline{D_{end2end}} - D_i)^2}{Ns}$$

Ns is the total number of successfully transmitted packets in the whole program iterations of all nodes (Sum of total successful packets). D_i repre sents the end-to-end delay of each successfully transmitted packets. It is equal to the difference between the end iteration and the beginning iteration contained in the packet. The beginning iteration and the end iteration represent the birth time and the final arrival time of each packet respectively.

2.10.2. Average and Variance of Queuing Delay

Queuing delay is defined as the time duration during which one packet waits for transmission. The mean of the queuing delay equals the sum of the queuing delays of all packets in all program iterations of all nodes divided by the total number of packets. The variance of the queuing delay equals the sum of the squares of the difference between the average queuing delay and each individual queuing delay in all program iterations and over all nodes divided by the total number of messages. The formulas for Average Queuing Delay ($\overline{D_{queue}}$) and Variance of Queuing Delay (v_{queue}) are listed as follows:

$$\overline{D_{queue}} = \frac{\sum_{i=1}^{N} D_i}{N}$$

$$v_{queue} = \frac{\sum_{i=1}^{N} (\overline{D_{queue}} - D_i)^2}{N}$$

Here N is the total number of packets; D_i represents the queuing delay of the i^{th} packets.

2.10.3. Average and Variance of Hop Counter

After data packets are generated, the node looks up its routing table to find a route to the destination. If the destination is not a neighbor, data packets first go to the next hop in the route and continue going on to the next hop until they reach the destination node. In other words, packets need to travel several hops to reach the destination. For simulation purposes, we set the hop counter to record the number of hops through which packets need to go until they finally reach the destination. The mean of the hop counter is defined

as the ratio of the sum of all hop-counters of all successfully transmitted packets divided by the total number of successfully transmitted packets. Correspondingly, the variance of the hop counter is defined as the sum of the square difference between the average hop counter and the individual hop counter divided by the total number of successfully transmitted packets by all nodes in the whole simulation time. The formulas for Average Hop Counter (AHC) and Variance of Hop Counter (VHC) are:

$$AHC = \frac{\sum_{i=1}^{Ns} HC_i}{Ns}$$

$$VHC = \frac{\sum_{i=1}^{Ns} (AHC - HC_i)^2}{Ns}$$

Here HC_i is the hop counter of each successfully transmitted packet; Ns is the total number of successfully transmitted packets.

2.10.4. Average and Variance of Buffer Overflow Probability

Buffer overflow probability is 1 when the buffer is full, otherwise is 0. The mean of the buffer overflow probability equals the sum of the individual buffer overflow probability over all program iterations and over all buffers divided by the total number of buffers and the total number of iterations. The variance of the buffer overflow probability equals the sum of the square difference between the average buffer overflow probability and the individual buffer overflow probability over all program iterations and over all nodes divided by the total number of iterations and by the number of buffers. The

formulas for Average buffer overflow probability (ABF) and Variance of buffer overflow probability (VBF) are listed as follows:

$$ABF = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{Nb} BF_{ijk}}{n * m * Nb}$$

$$VBF = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{Nb} (ABF - BF_{ijk})^2}{n * m * Nb}$$

m is the maximum program iterations and n is the total number of nodes. BF_{ijk} represents the buffer overflow probability of the k^{th} buffer in the j^{th} node and the i^{th} iteration. Nb is the number of buffers in each node. It is 2 in our simulation.

2.10.5. Average and Variance of Packet Number in Buffer

The mean of packets in buffers equals the sum of packets in buffers over all program iterations and over all nodes divided by the total number of buffers and by the number of iterations. The variance of the number of packets in the buffer equals the sum of the square of the difference between the average packets in buffers and the individual number of packets in a certain buffer in all program iterations and among all buffers divided by the total number of iterations and the number of buffers. The formulas for Average of packets in buffers (APB) and Variance of packets in buffers (VPB) are listed as follows:

$$APB = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{Nb} PB_{ijk}}{n * m * Nb}$$

$$VPB = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{Nb} (APB - PB_{ijk})^2}{n * m * Nb}$$

m,n,Nb are the same as before. PB_{ijk} represents the packets in the k^{th} buffers of the j^{th} node in the i^{th} iteration.

2.10.6. Throughput

Wireless bandwidth is a scarce resource, so efficiency of such bandwidth is vital. Throughput is defined as the total number of successfully delivered data packets divided by the total number of data packets generated over all nodes and all iterations. The formula for Throughput is listed as follows:

$$Throughput = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} SD_{ij}}{\sum_{i=1}^{m}\sum_{j=1}^{n} DA_{ij}}$$

m is the maximum program iterations and n is the total number of nodes. SD_{ij} represents successfully transmitted data packets of the j^{th} node in the i^{th} iteration. DA_{ij} represents all data packets transmitted by the j^{th} node in the i^{th} iteration.

2.10.7. Average of Failed Routing

The average of failed routing is an important index to evaluate the performance of routing protocols. The mean of failed routing is calculated by dividing the sum of all failed routings of all nodes by the total number of generated packets. The formula for the Average of failed routing (AFR) is as follows:

$$AFR = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} N_{fr}}{Np}$$

m is the maximum program iterations and n is the total number of nodes. N_{fr} is the number of the failed routing per node in one iteration. Np is the total number of produced packets by all nodes in one simulation.

2.10.8. Probability of Lost Packets

The probability of lost packets is calculated by dividing the sum of all lost packets by the total number of generated packets. Lost packets can result from failed routes, bad channels, and buffer overflows. The formula for the Probability of lost packets (P_{lost}) is as follows:

$$P_{lost} = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}(N_{fr} + N_{bc} + N_{bo})}{Np}$$

m is the maximum program iterations and n is the total number of nodes. N_{fr} is the number of lost packets due to the failed routing per node in one iteration. N_{bc} is the number of lost packets due to bad channels per node in one iteration. N_{bo} is the number of lost packets owing to buffer overflows per node in one iteration. Np is the total number of produced packets by all nodes in one simulation.

2.11. Design of the WLAN Analyzer

Figure 2.25 shows the block diagram of all units of the WLAN simulation software. These units can be classified into three packages: WLAN Routing package, WLAN entity package, and WLAN controller package. For example, the WLAN controller package has modules (ended with "er" except RouteDemon) in Figure 2.25.

Based on these three packages, the WLANSimulator module is designed. The module

gets a group of input parameters from the WLANAnalyzer module and sends results through output parameters to the WLANAnalyzer module. After the WLANSimulator's running for a certain times, the WLANAnalyzer outputs statistics outcomes to an output file. Figure 2.26 is the software architecture of the WLAN Analyzer.

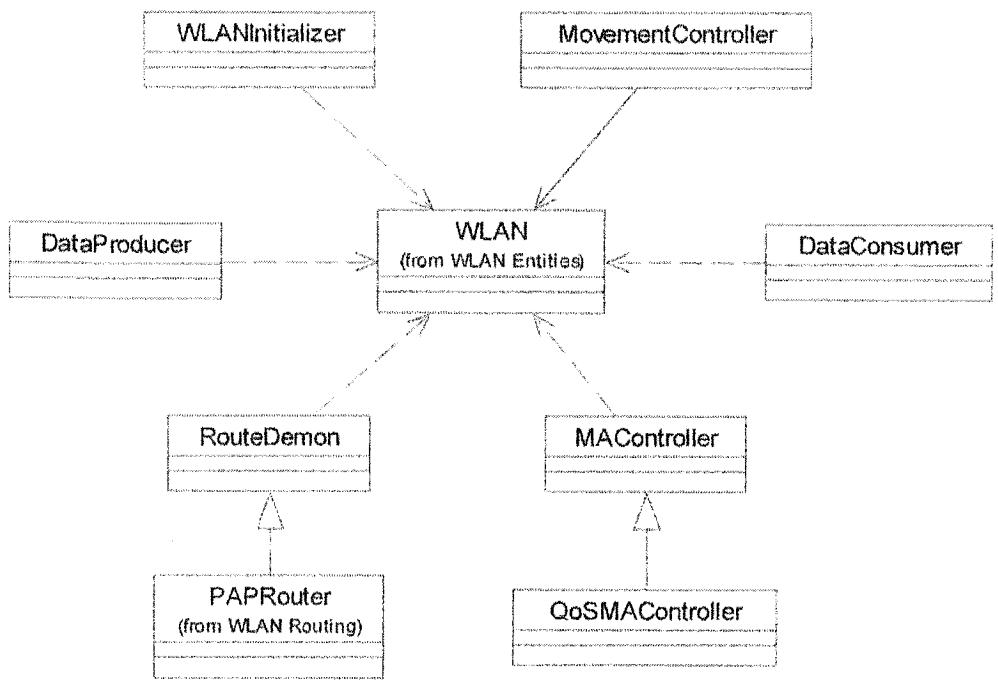


Figure 2.25: Modules in WLAN Controller package

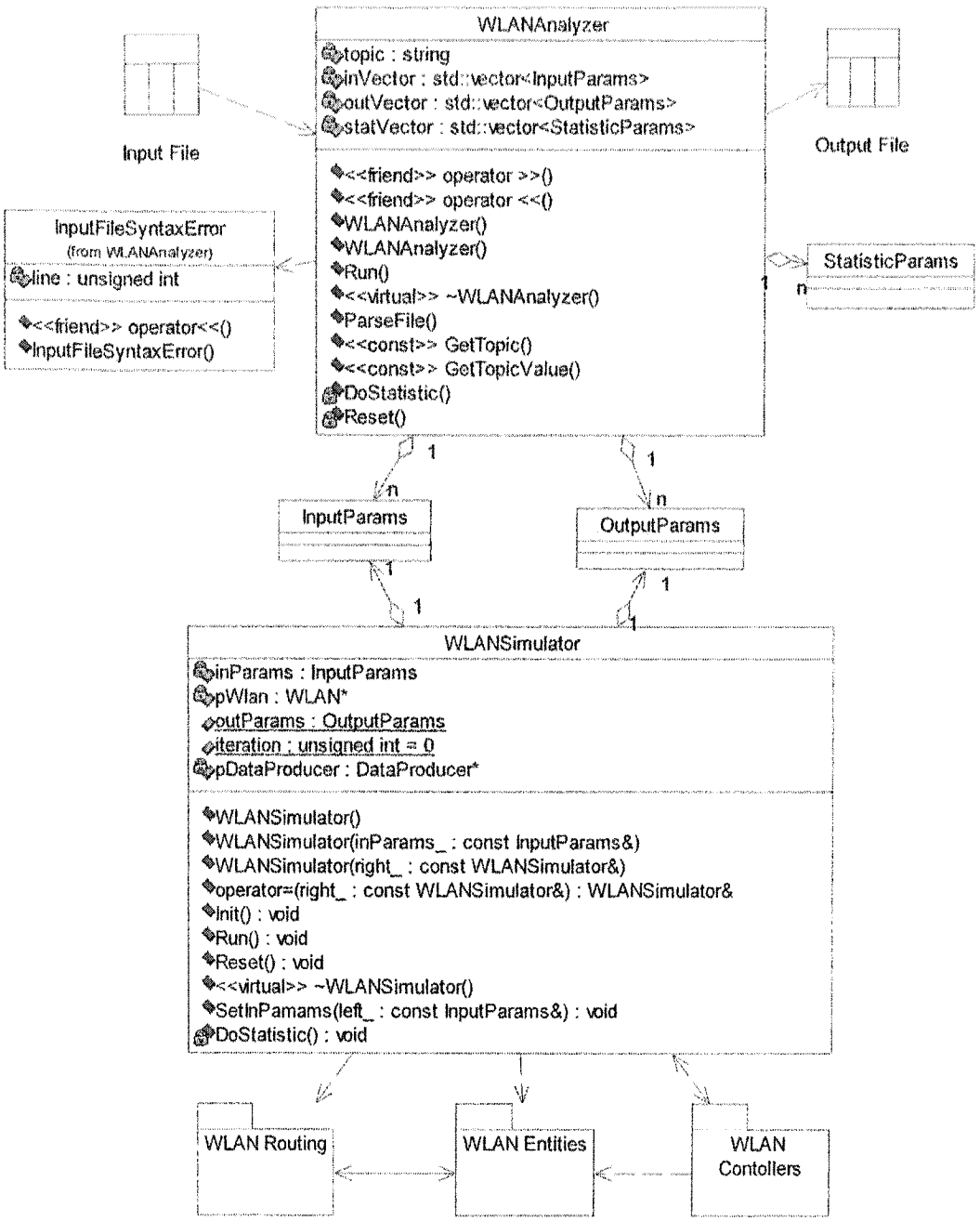


Figure 2.26: Software architecture of WLAN Analyzer

Chapter 3

Simulation Results

The simulations in this chapter are conducted to get a comparison of how much better/worse the double window algorithm, a modified CAMS/CA algorithm in this thesis, is than the standard CSMA/CA algorithm. Also we propose a comparison of how much better/worse the PAP routing protocol is than the θ routing protocol, and to get a performance evaluation of the PAP routing protocol. We have executed 4 different crucial types of simulations:

1. Offered load simulations: We vary the load probability that we offer to the network to simulate the performance difference between the modified CSMA/CA algorithm and the standard CSMA/CA algorithm when, for instance, when the load is high. The load probability P(ON) = $\beta / (\alpha + \beta)$ as defined in Figure 2.19.
2. Mobility simulations: We vary the mobility (move velocity & probability) to see how it affects the PAP routing protocol and the θ routing protocol that we are evaluating.
3. Network size simulations: We vary the number of nodes in the network with constant area and vary the network area with constant number of nodes to see how the PAP routing protocol behaves.
4. Channel quality simulations: We vary the channel quality from bad to good to see how it affects the behavior of the PAP routing protocol.

The simulations have been conducted on an AMD Athlon™ processor at 900 MHz, 320

Mbytes of RAM running Microsoft Windows XP professional 2002 with Service Pack 1.

3.1. Evaluation of the Double window Algorithm for CSMA/CA

The double window algorithm suggested in this thesis is a QoS enhancement to the standard CSMA/CA algorithm. This simulation is to verify the creative idea of the enhancement. Performance comparison is carried out under different load probabilities. The input parameters that have been used for the simulation are shown in the following table.

Table 3: Parameters used during offered load simulations

Parameters	Value
Environment size	1000m X 1000m
Number of nodes	50
Traffic type	Variable Bit Rate (VBR)
Call active probability	0.1
Mobile threshold	20%
Maximum speed	20 m/s
Maximum move angle	360
Location update period	40 (iterations)
Channel good quality	0.999
Maximum hop limit	20
Simulation time	10000 (iterations)

This simulation of the effects of the offered load is very crucial. We have compared the throughput, the end-to-end delay, the queuing delay, the number of packets in buffer, and the buffer overflow percentage between the standard CSMA/CA MAC protocol and the QoS MAC protocol (double window algorithm).

Figure 3.1 shows that both throughputs decrease when the offered load is reduced. The reason is clear: the buffer overflow will occur more frequently when the offered load becomes higher, so more packets will be dropped. It can be seen from Figure 3.4 that the

average number of packets in buffers increases along with the augment action of the offered load, because the channel is a shared medium, and all packets can not be sent at the same time but have to execute the CSMA/CA algorithm (backing off exponentially) to contend for the channel. More packets in the buffer will bring about two results: one is the longer queuing delay; the other is the higher buffer overflow percentage. As a result, it is reasonable in Figure 3.3 that the average queuing delay extends as a consequence of the increase of the offered load. It is also logical in Figure 3.5 that the average buffer overflow percentage increases as the offered load expands. Furthermore, the longer queuing delay will cause a longer end-to-end delay. Figure 3.2 shows that both the average end-to-end delays are extended when the offered load increases, as more packets will be queued in the buffer and more collisions and backing-offs in the channel will occur. Until now, we can not get any performance difference between the two policies, namely the double window algorithm and the single window algorithm, from Figure 3.1 to Figure 3.5. In other words, the double window algorithm can not improve the performance on the throughput, the average end-to-end delay, the average queuing delay, the average number of packets in the buffer, and the average buffer overflow percentage.

However, Figure 3.6 to Figure 3.9 provide us with extremely valuable information. These four figures represent the standard deviation of the packets in buffers, the standard deviation of the queuing delay, the standard deviation of the buffer overflow, and the standard deviation of the end-to-end delay respectively. All of these four figures display an interesting result: the standard deviation of the new QoS MAC is less than that of the standard MAC, although the difference is not large. The less the variance of the

end-to-end delay and the queuing delay is, the better the quality of service (QoS) for real-time traffic. In addition, the difference of the variance of delays is rather trivial. The reason is that the double window algorithm is only a little modification to the standard CSMA/CA: the new added window will be effective only if the collision window of at least two hosts in one cluster timeouts at the same time; in other words, the new added window will be effective after the collision window rather than before it, and the exponential backoff algorithm itself remains unaltered.

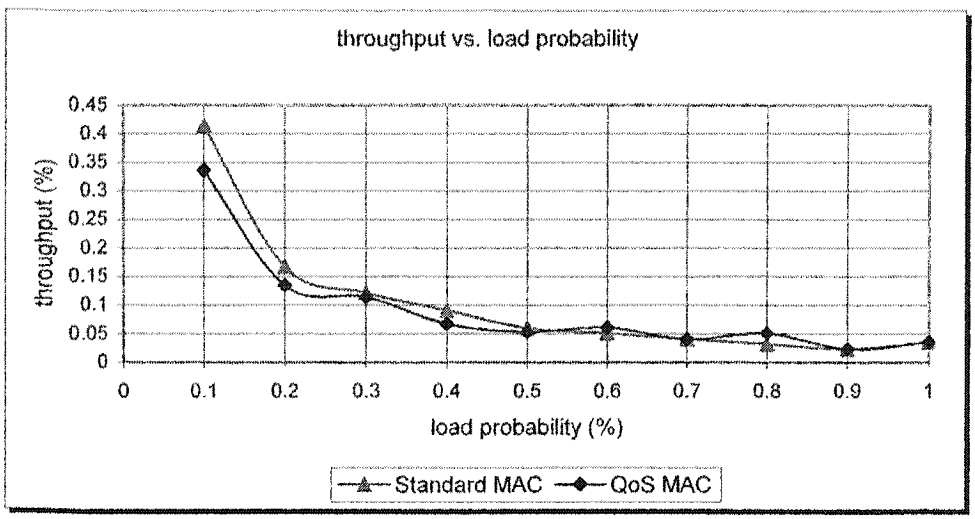


Figure 3.1: Throughput vs. load probability

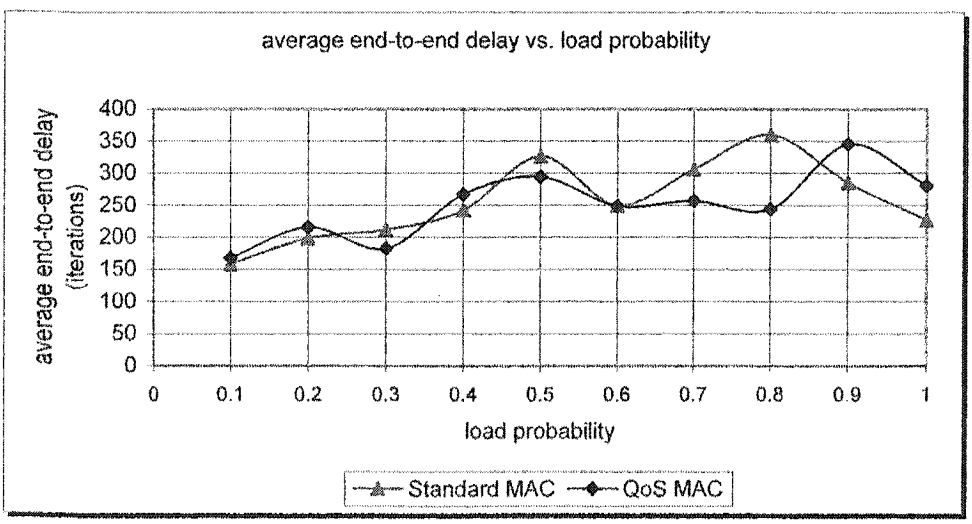


Figure 3.2: Average end-to-end delay (iterations) vs. load probability

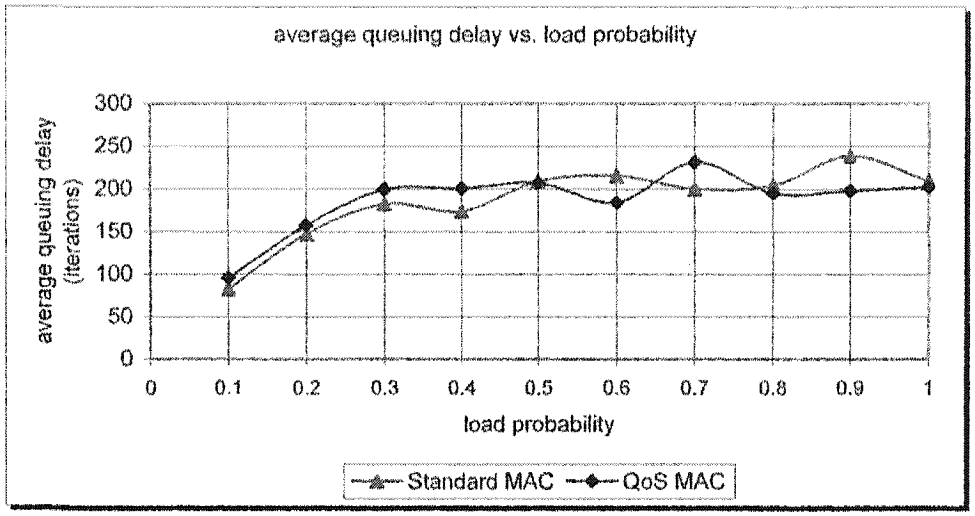


Figure 3.3: Average queuing delay (iterations) vs. load probability

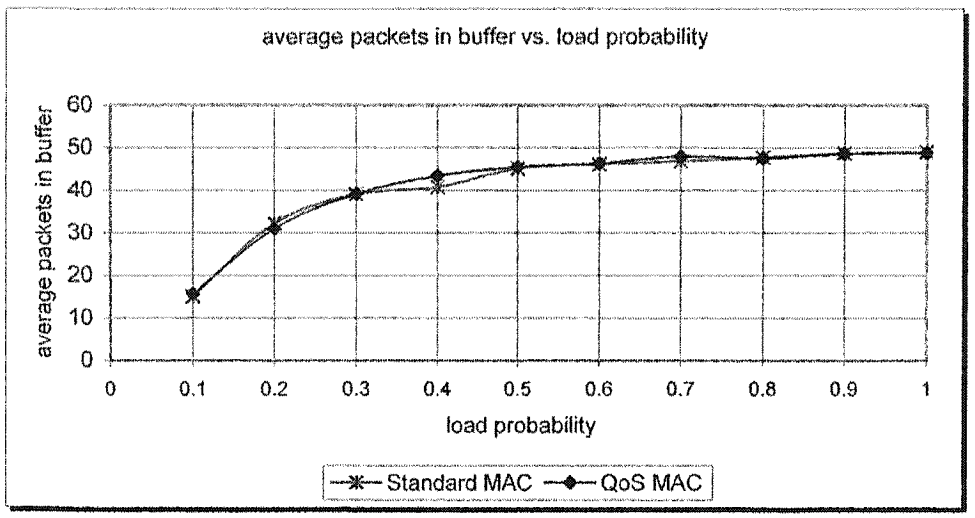


Figure 3.4: Average packets in buffer vs. load probability

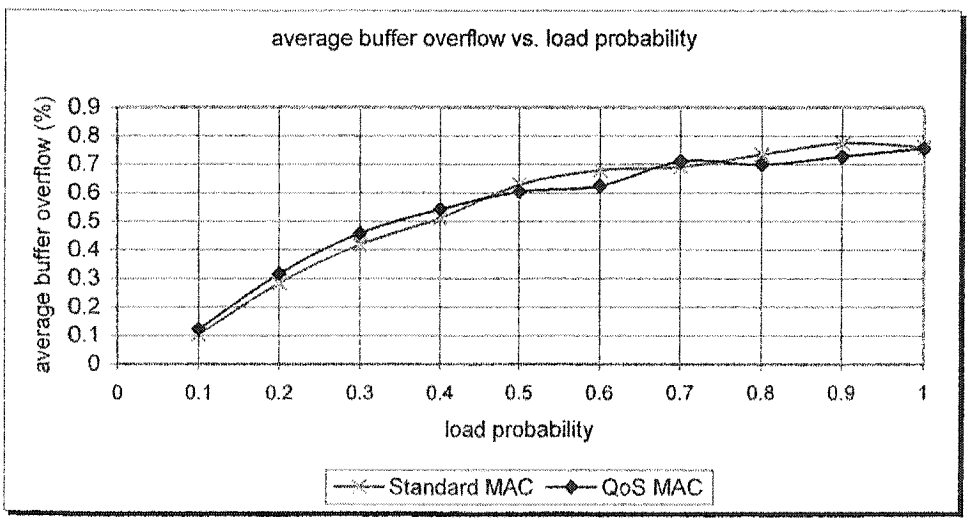


Figure 3.5: Average buffer overflow (%) vs. load probability

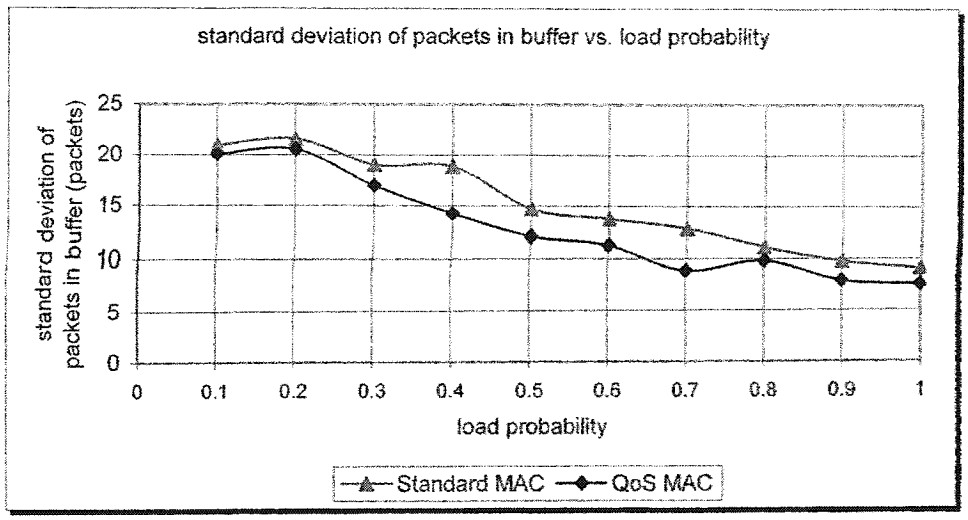


Figure 3.6: Standard deviation of packets in buffers vs. load probability

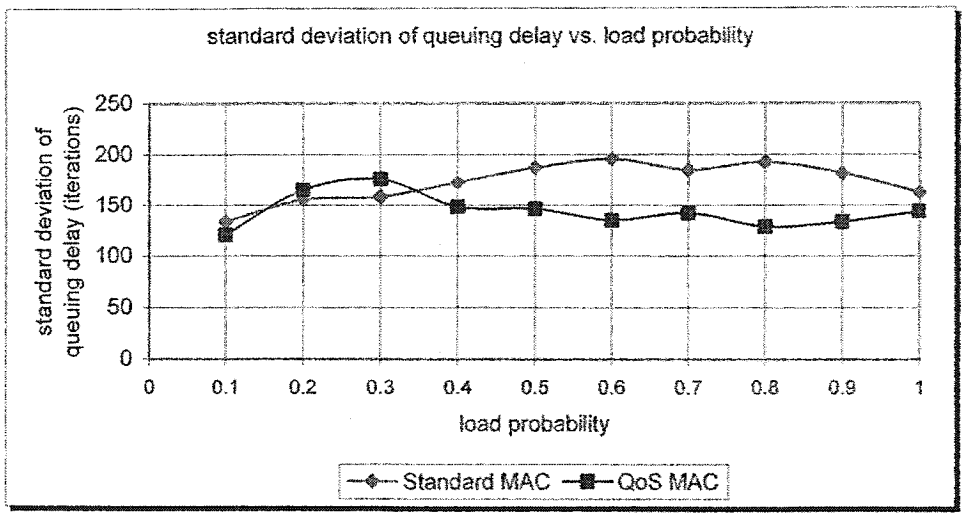


Figure 3.7: Standard deviation of queuing delay (iterations) vs. load probability

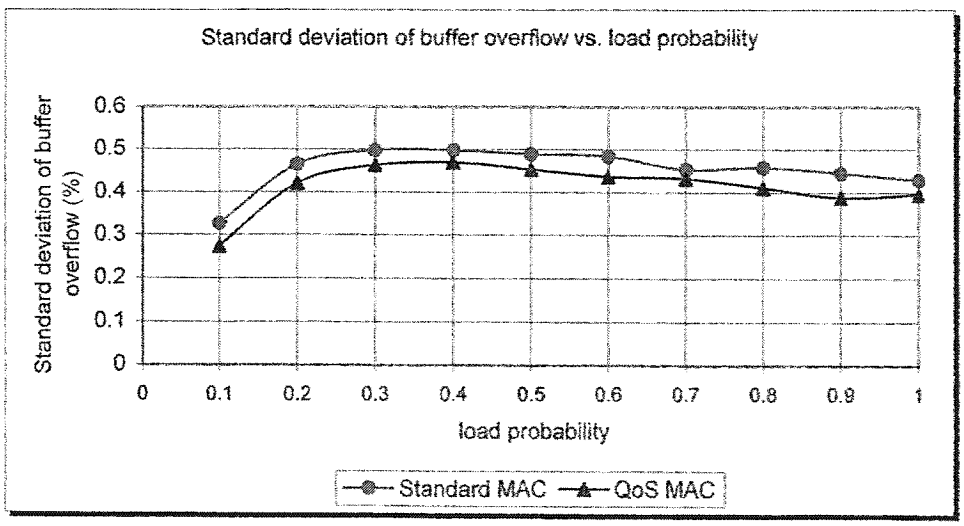


Figure 3.8: Standard deviation of buffer overflow (%) vs. load probability

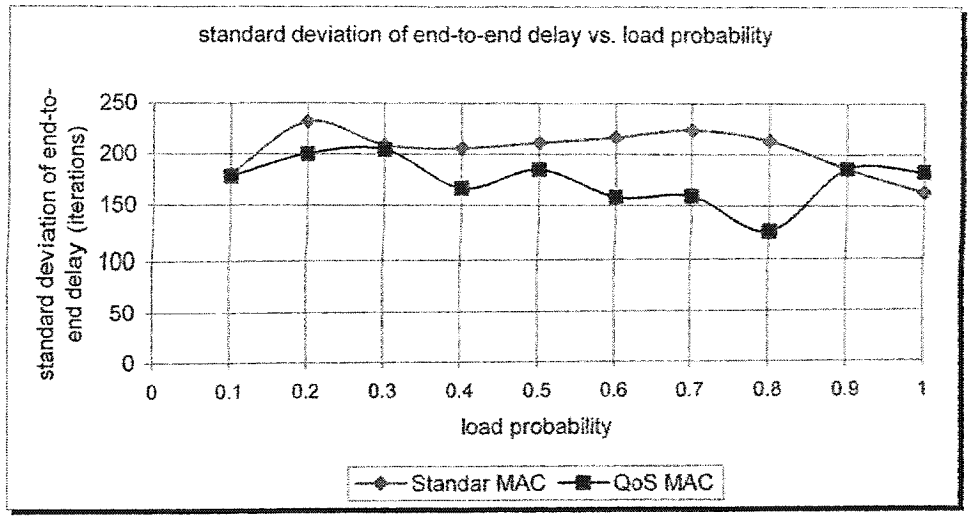


Figure 3.9: Standard deviation of end-to-end delay (iterations) vs. load probability

3.2. Comparison of PAP Routing Protocol with θ (angle) routing protocol

Mobility is probably one of the most important characteristics when evaluating ad-hoc network routing protocols. This will affect the dynamic topology; links will go up and down. Performance is compared under different node move probabilities and under different move velocities.

3.2.1. Performance Evaluation under Different Node Move Probabilities

One of the purposes of this simulation is to evaluate the PAP routing protocol by comparing the θ (angle) routing protocol under different node move probabilities. The larger the move probability is, the higher the mobility. The input parameters that have been used for the simulation are shown in Table 4.

Table 4: Parameters used during move probability simulations

Parameters	Value
Environment size	1000m X 1000m
Number of nodes	50
Traffic type	Variable Bit Rate (VBR)
Call active probability	5%

Load	5%
Maximum speed	20 m/s
Maximum move angle	360
Location update period	40 (iterations)
Channel good quality	0.9999
Maximum hop limit	20
Simulation time	10000 (iterations)

Figure 3.10 demonstrates the relationship of the throughput vs. the move probability. Obviously, the throughput of the PAP routing protocol is much higher than that of the θ (angle) routing protocol. It is due to the fact that the PAP routing protocol has knowledge of the whole network topology and almost always can find a new route if a link breakage is detected, while the θ routing protocol is only dependent on the smallest angle. Thus, when a link is broken, the θ routing protocol may not be able to find an efficient route even if the route exists. Secondly, the throughput of the θ routi ng protocol decreases as the mobility increases, whereas the throughput curve of the PAP improves: a higher value can be reached when the move probability goes from 0.3 to 0.7. Some packets are dropped just because an effective route can not be found. A certain higher mobility in the PAP routing protocol can help a node get new neighbors and have relatively stable number of neighbors as indicated in Figure 3.11; nevertheless, the throughput will decrease if the mobility is too much evident, e.g. above 0.7 in this simulation.

Figure 3.11 gives one an idea about the average failed routing curves. It can be found that the average of the failed routes of the θ routing protocol is much higher than that of the PAP and increases dramatically as the mobility increases. Also, for the PAP routing protocol, we can see that mobility has less effect on average failed routes, which stems from the fact that a relatively constant number of neighbors helps in finding new routes.

Figure 3.12 explains the average end-to-end delay (iterations) vs. the move probability. From this figure we see that the average end-to-end delay of the PAP is higher than that of the θ routing protocol, since packets can travel more hops by the PAP than by the θ routing protocol, as shown in Figure 3.14. Likewise, Figure 3.13 demonstrates that the standard deviation of the end-to-end delay via the PAP is also higher than that of the θ routing protocol.

Figure 3.14 illustrates the relationship between the average hop count per packet and the mobility. It can be seen that the average hop count via the PAP is approximately higher than that of the θ routing protocol and basically keeps consistent. It can be explained as follows: the average failed route of the θ routing protocol is much higher than that of the PAP, while the throughput of the θ routing protocol is lower than that of the PAP. Compared with the θ routing protocol, the PAP routing protocol not only can help packets reach their neighbor destination but also can help packets travel more hops and arrive their distant destination. Figure 3.15 shows that the standard deviation of the hop count of the PAP is lower than that of the θ routing protocol.

In one word, the PAP routing protocol can supply higher throughput and fewer failed route than the θ routing protocol under the move probability. Thus the quality of service (QoS) of the PAP routing protocol is superior to that of the θ routing protocol.

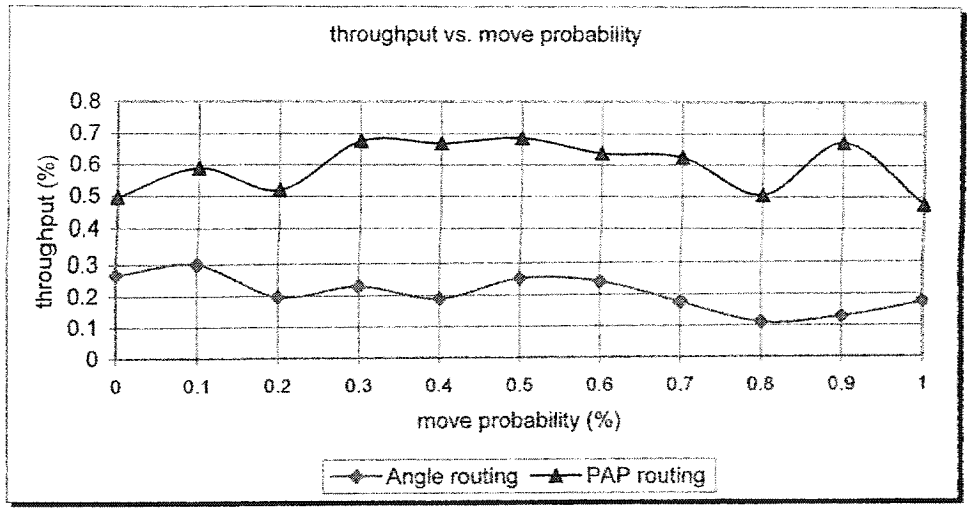


Figure 3.10: Throughput vs. move probability

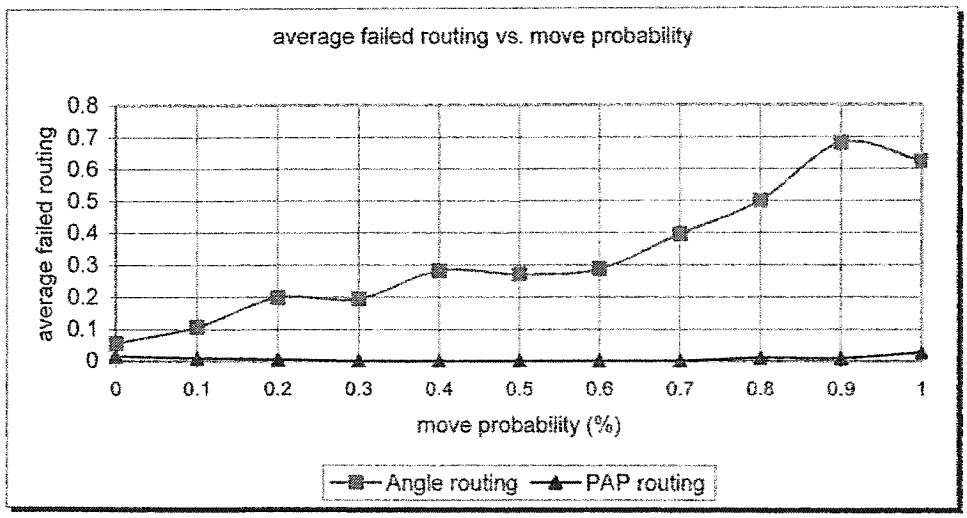


Figure 3.11: Average failed routing vs. move probability

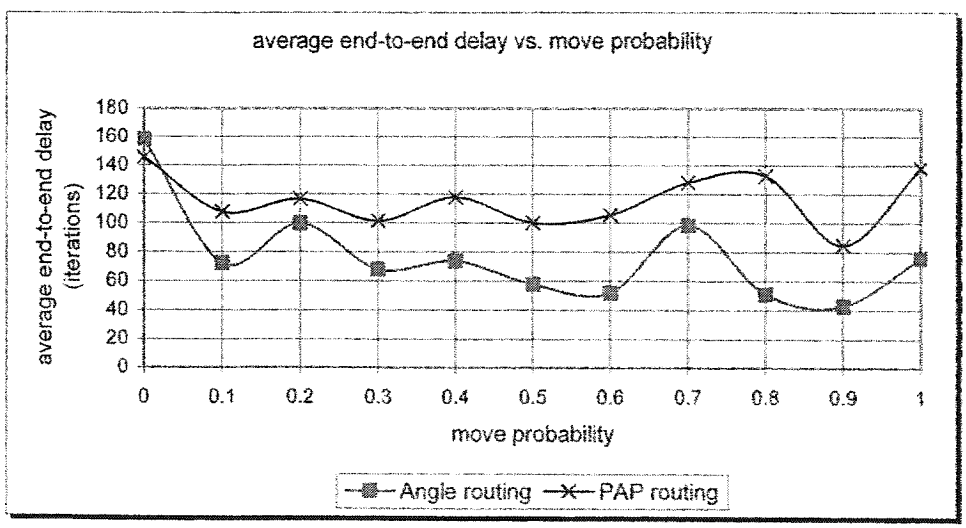


Figure 3.12: Average end-to-end delay (iterations) vs. move probability

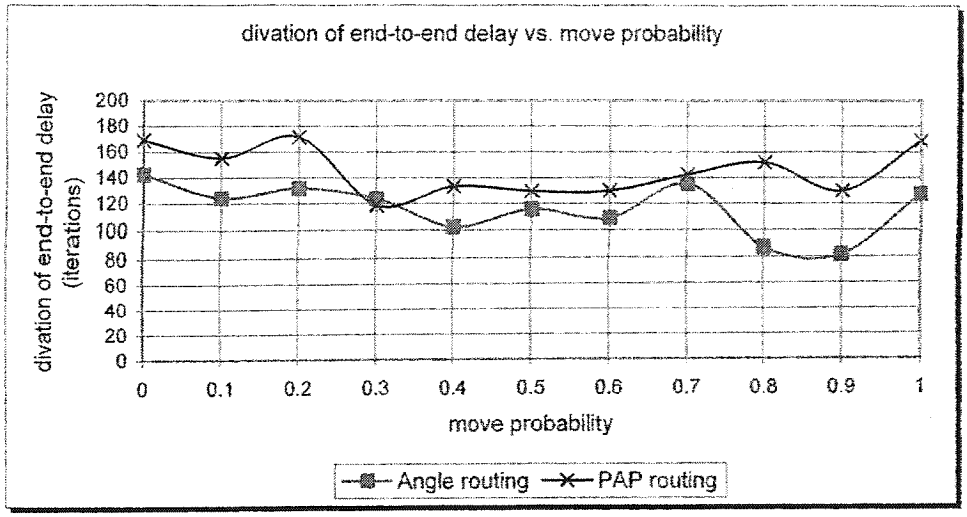


Figure 3.13: Standard deviation of end-to-end delay (iterations) vs. move probability

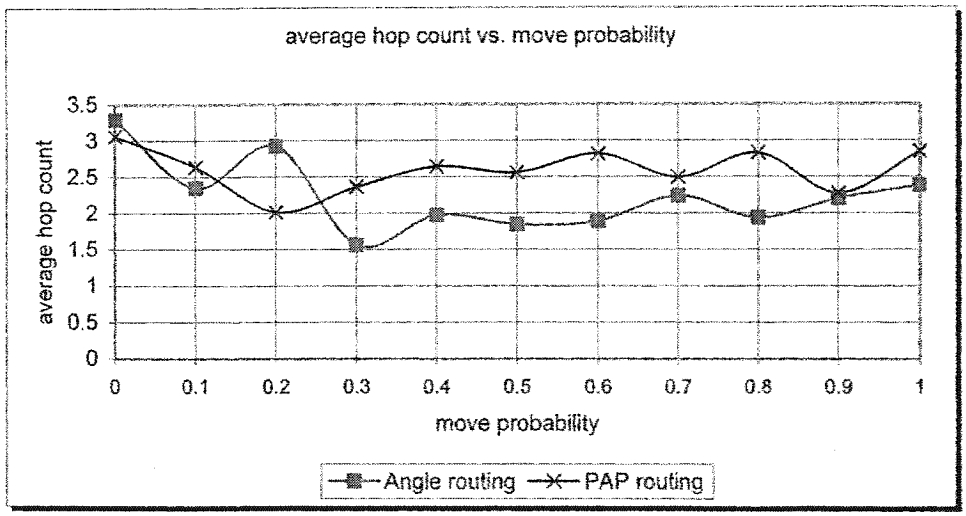


Figure 3.14: Average hops vs. move probability

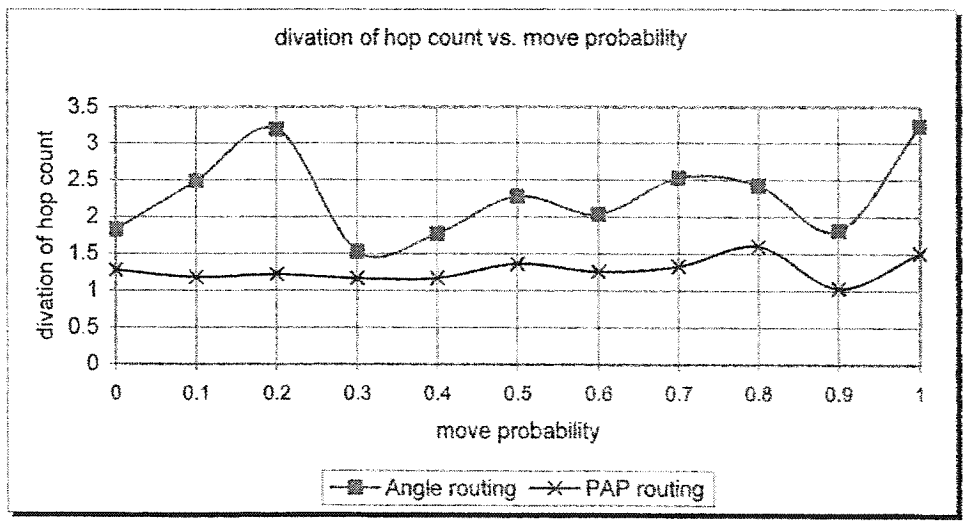


Figure 3.15: Standard deviation of hops vs. move probability

3.2.2. Performance Evaluation under Different Node Move Speeds

Performance contrast between the PAP routing protocol and the θ routing protocol have just been described under different move probabilities. Now, we will distinguish these two routing protocols under different node maximum move speeds. The input parameters that have been used for the simulation are shown in the following table.

Table 5: Parameters used during maximum move velocity simulations

Parameters	Value
Environment size	1000m X 1000m
Number of nodes	50
Traffic type	Variable Bit Rate (VBR)
Call active probability	10%
Load	10%
Probability of movement	20%
Maximum move angle	360
Location update period	40 (iterations)
Channel good quality	0.9999
Maximum hop limit	20
Simulation time	10000 (iterations)

Figure 3.17 and 3.18 show the throughput and the average failed routes tendency correspondingly as the maximum speed improves. These curves have roughly the same tendency as the simulations under different move probabilities. It seems understandable that the larger the move velocity is, the higher the mobility.

Figure 3.19 and Figure 3.20 show that the average queuing delay and the standard deviation of the end-to-end delay of the PAP routing protocol is higher than those of the θ routing protocol. It can be explained as following: the longer queuing delay will result in the larger variance of the end-to-end delay. In addition, the average queuing delay and the standard deviation of the end-to-end delay of both protocols, are virtually straight

lines. Hence, it can be concluded that these parameters are nearly unrelated with the move speed, the reason of which is related with the relative speed and the absolute speed. The speed in our simulation is the absolute speed, which is not sufficient to give a picture of the average speed of the distance change between the nodes. For instance, two nodes keep standing still when they move parallel with the same speed and the same move orientation, because their relative speed is zero. Figure 3.16 illustrates the relative speed and the absolute speed.

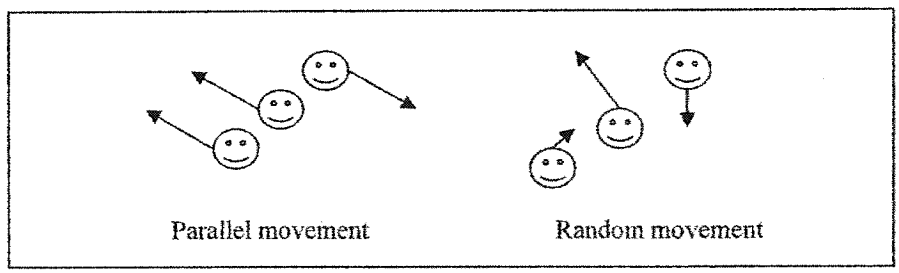


Figure 3.16: Relative speed vs. absolute speed

Additionally, the PAP routing protocol and the θ routing protocol in our simulations are based on the Global Positioning System (GPS) [25] to get the mobility information. This gives rise to some new issues: the GPS can augment each node's economic cost and can enlarge its size, especially for such devices as mobile phones. Secondly, other telecommunication protocols are also required for communication between the nodes and the GPS network. The principal advantage of the GPS-based routing protocols [24] is that the network overhead is too trivial to be considered, because the routing overhead is not demanded for finding the route. However, if the PAP is not a routing protocol based on the GPS, the routing control packets, such as HELLO packets, are compulsory to get the neighbor's information and topology change, and then when the network mobility increases, the network overhead will increase accordingly.

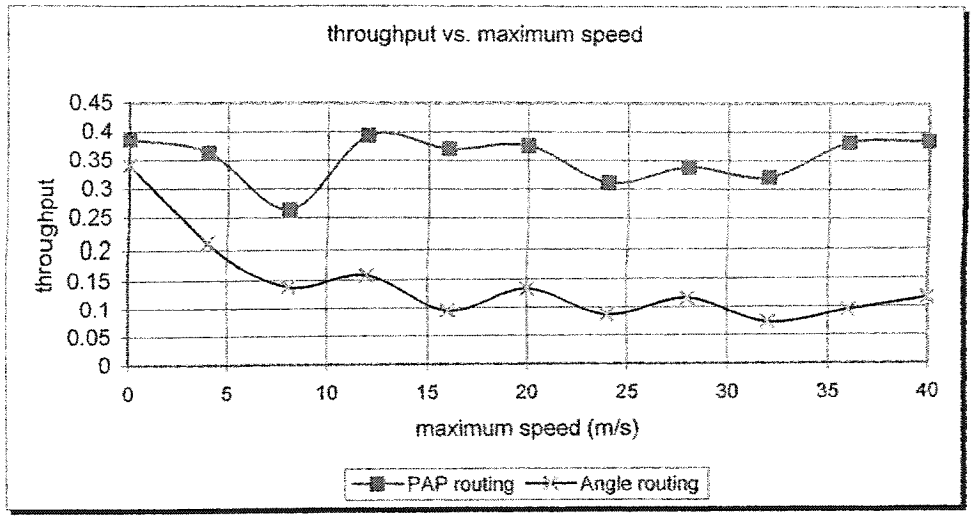


Figure 3.17: Throughput vs. maximum speed (m/s)

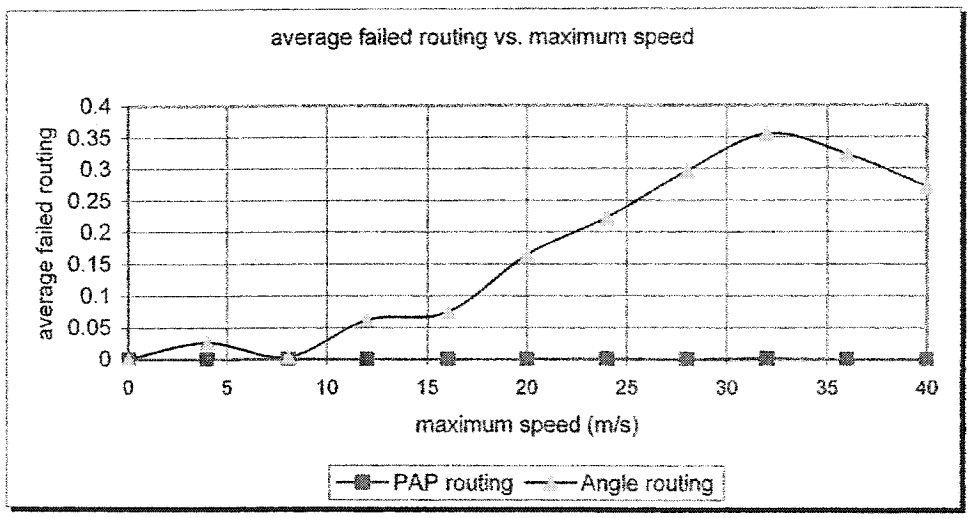


Figure 3.18: Average failed routing vs. maximum speed (m/s)

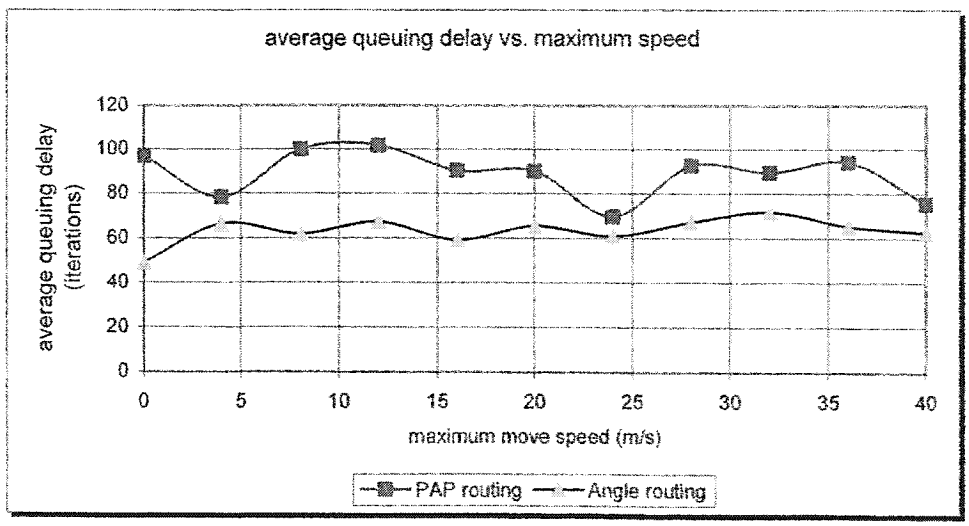


Figure 3.19: Average queuing delay (iterations) vs. maximum speed (m/s)

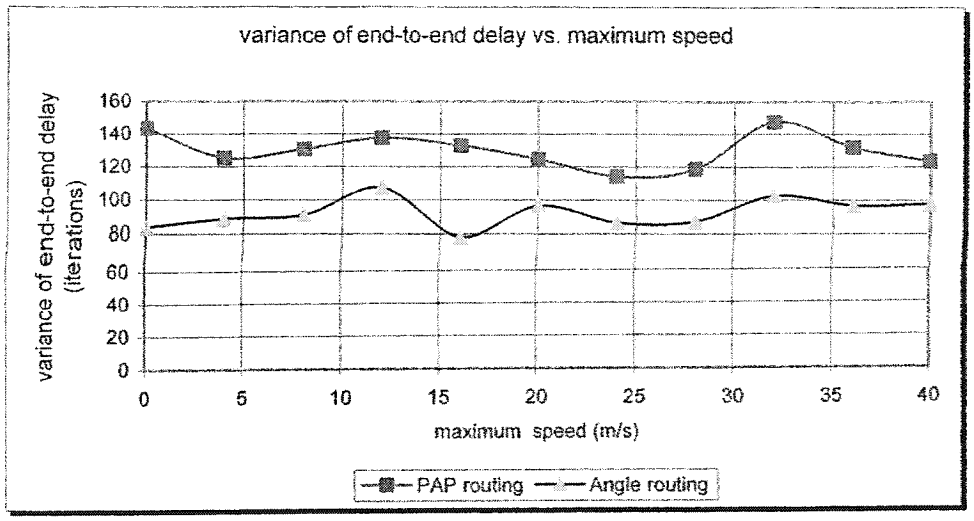


Figure 3.20: Standard deviation of end-to-end delay (iterations) vs. maximum speed (m/s)

3.3. Evaluation of PAP Routing Protocol

We have already made some evaluation about the PAP routing protocol through contrasting it with the θ routing protocol. In this part, more simulations will be conducted to continue evaluating the PAP protocol. The double window algorithm is employed in all simulations.

To compare the performance of the three ways to break the PAP tie, that is, Node with More Neighbors First, Node with Lower Moving Speed First, and Node with More Available Buffer First, we have carried out quite a few simulations. However, we can not get any useful information because there is no performance difference among these three ways. At last, we suggest a novel way to break the PAP tie, namely, choosing any one randomly as the PAP when there is a tie at step 4 of the proposed heuristic of the PAP routing protocol in Chapter 2.1.2. But we still can not find differences on the fourth way, as the probability of the occurring ties is not quite significant. Even when a tie turns out, it is difficult to select which node is better than others. The node with more one-hop

neighbors may not have enough buffers that can easily bring about the buffer overflow, whereas the node with more buffers available may not have sufficient one-hop neighbors or its mobility is large. Furthermore, the node with a lower speed doesn't mean its relative speed is also lower or its mobility related to its neighbors is smaller. Therefore, their influence on the performance is too minor to be discerned. After all, various parameters affect each other and thus have a significant and complicated effect on the network performance. More complex and comprehensive explanations still need further investigations. All simulations in this part employ the fourth way to break the ties: select any one randomly as the PAP.

3.3.1. Network Performance under Different Network Areas

In this simulation, we modify three parameters at the same time, i.e. enlarge the environment size from 500m by 500m to 4000m by 4000m with step 500m, raise the number of nodes with constant density $48/Km^2$, and extend the maximum hop limit by 25% of the number of nodes. Of course, the environment size is our focus in this simulation. The input parameters that have been used for the simulation are shown in the following table.

Table 6: Parameters used during network area simulations

Parameters	Value
Density of nodes	48 / km*km
Traffic type	Variable Bit Rate (VBR)
Call active probability	10%
Load	10%
Probability of movement	20%
Maximum move angle	360
Maximum speed	20 m/s
Location update period	40 (iterations)

Channel good quality	0.9999
Maximum hop limit	25% * Number of nodes
Simulation time	10000 (iterations)

Figure 3.21 shows that the average end-to-end delay and its standard deviation increase rapidly when the network size expands within around $2\,Km^2$, and then increase gradually when the network size expands beyond $2\,Km^2$. Likewise, the average of the queuing delay and its standard deviation have similar tendency as shown in Figure 3.22. Unquestionably, the longer the queuing delay is, the longer the end-to-end delay, but it must be noted that the end-to-end delay also depends on the transition delay: the more hops are traveled, the longer the end-to-end delay is. Figure 3.23 gives us the answer that the increase tendency of the average hop count and its standard deviation is similar to that of the end-to-end delay curves when the environment size expands. Of course, it is reasonable that a larger network area has more hops traveled.

Figure 3.24 depicts the dependency of the number of packets in buffers per node on the network area. It is obvious that the average number of packets in buffers and its standard deviation first increase promptly, but then slowly at the point of about $2\,Km^2$ when the network size expands. More packets in buffers can most probably result in larger buffer overflow probability. Figure 3.25 confirms our reasoning. Moreover, more packets will be dropped when the buffer overflow percentage increases, and it will finally cause the decrease of the throughput. Figure 3.26 validates this logic. It must be noted that the throughput in Figure 3.26 is rather low, which is not because lots of packets can not find their effective routes, but because large numbers of them are still on the way. As shown in Figure 3.24, on the average when the network area is larger than $2\,Km^2$, 20 plus packets are still in buffers and need forwarding or transmitting. Figure 3.26 also confirms

that the probability of the lost packets increases along with the enlargement of the network area. That is the direct reason of the decrease of the throughput.

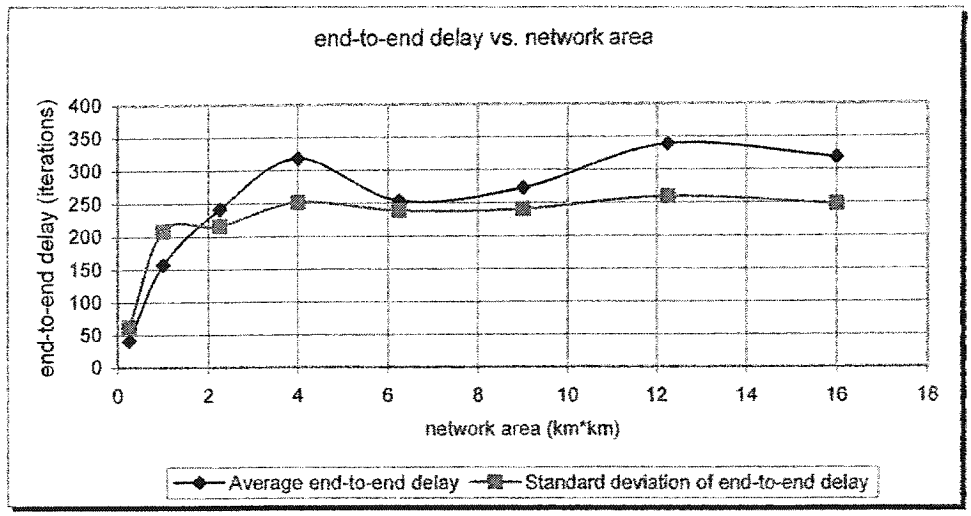


Figure 3.21: End-to-end delay (iterations) vs. network area

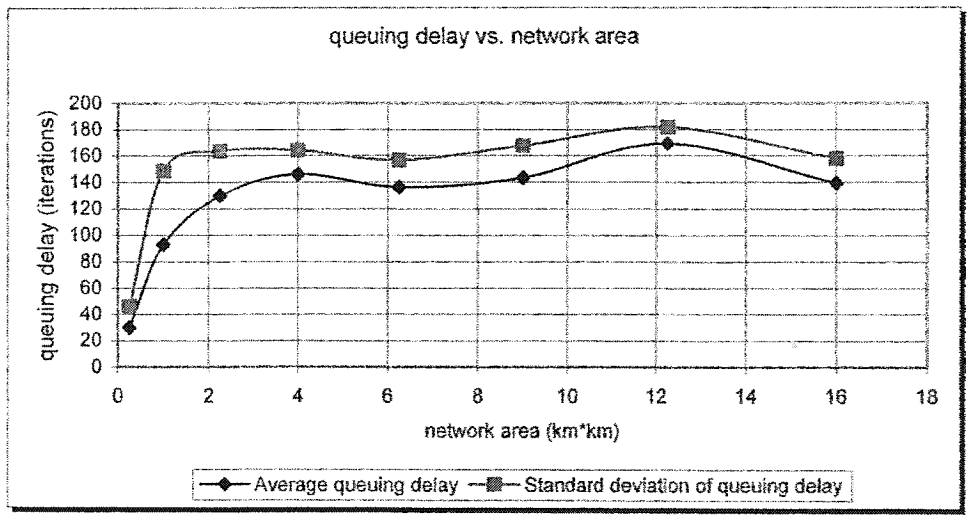


Figure 3.22: Queuing delay (iterations) vs. network area

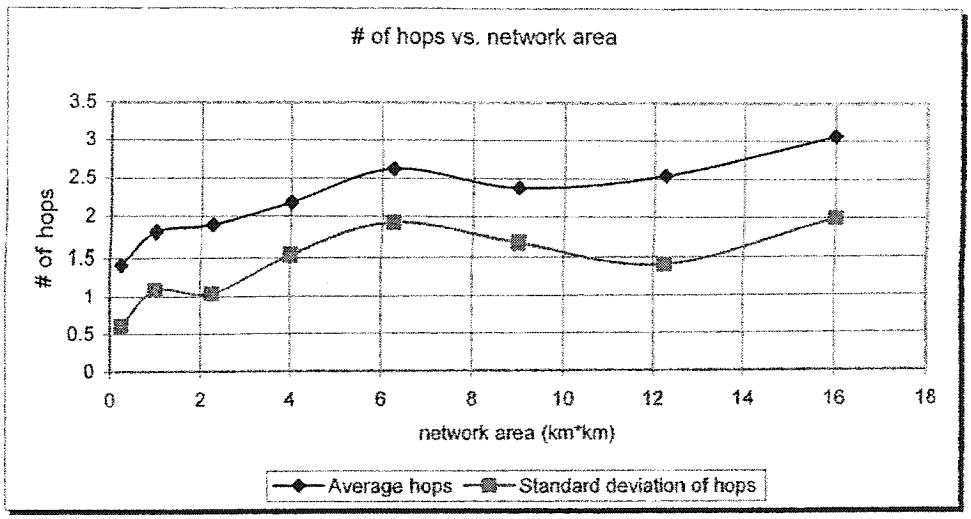


Figure 3.23: Hop count vs. network area

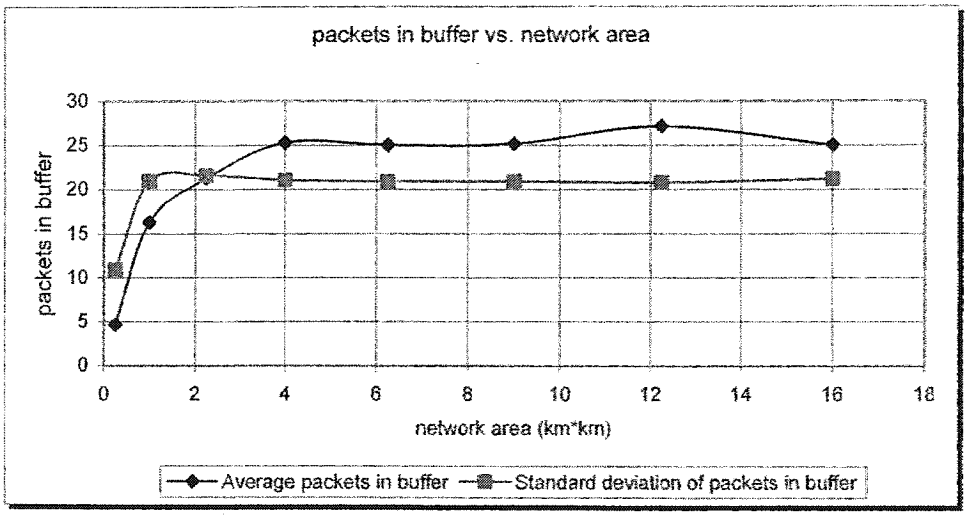


Figure 3.24: Packets in buffer vs. network area

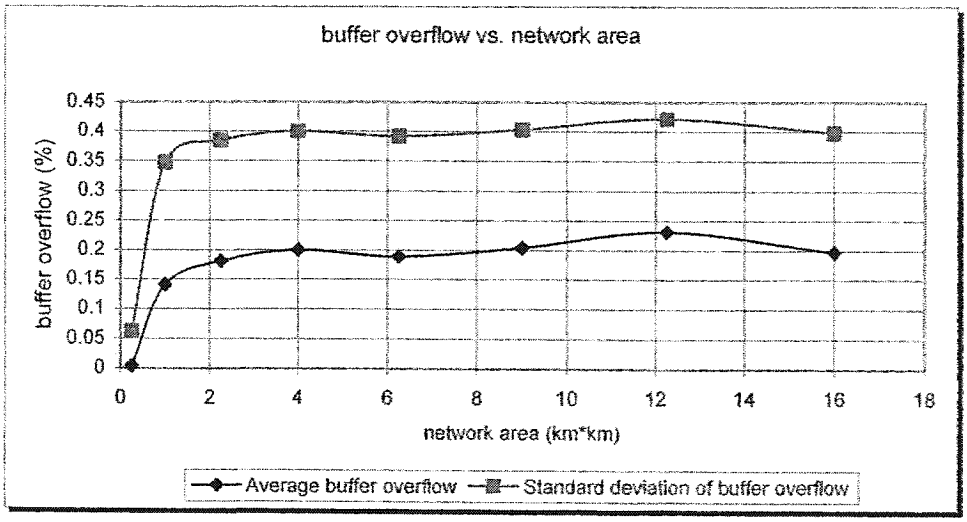


Figure 3.25: Buffer overflow (%) vs. network area

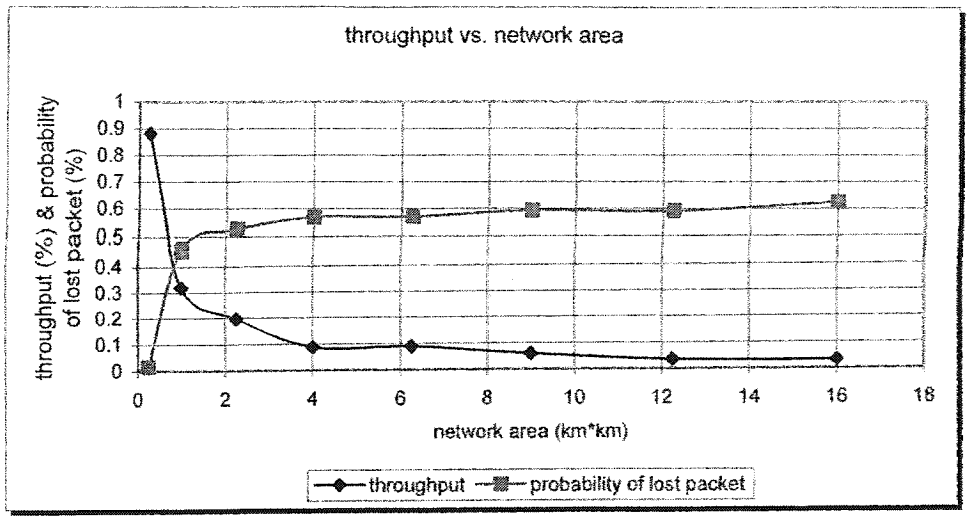


Figure 3.26: Throughput vs. network area

3.3.2. Network Performance under Different Node Density

In the previous simulation, we vary the network area and the number of nodes with constant node density. The simulation in this part, however, aims to evaluate the network performance under different node density. The input parameters that have been used for the simulation are shown in the following table.

Table 7: Parameters used during node density simulations

Parameters	Value
Environment size	1000m X 1000m
Traffic type	Variable Bit Rate (VBR)
Call active probability	10%
Load	15%
Probability of movement	20%
Maximum move angle	360
Maximum speed	20 m/s
Location update period	40 (iterations)
Channel good quality	0.9999
Maximum hop limit	40% * Number of nodes
Simulation time	10000 (iterations)

Figure 3.27 proves that the average end-to-end delay and its standard deviation increase accordingly when the node density increases. With the comparable increase slope,

Figure 3.28 illustrates that the average queuing delay and its standard deviation also increase as a result of the increase of the node density. Figure 3.29 shows that the hop count and the node density are not heavily dependent on each other. Consequently, the enlargement of the end-to-end delay is for the most part due to the enlargement of the queuing delay as the node density increases. In addition, when a node's density increases, each node will have more neighbors, and the size of such tables as the one-hop neighbor table, the two-hop neighbor table, the PAP table, and the routing table of each node will enlarge rapidly. The searching time for these tables will hence also become longer.

Generally, the queuing delay largely relies on the number of packets in buffers, i.e. when there are more numbers of packets in buffers, the queuing delay will tend to be longer. As Figure 3.30 shows, the average number of packets in buffers increases when the node density augments. This result is compatible with Figure 3.28 about the queuing delay. On the other hand, the buffer overflow probability will be larger when there are more numbers of packets in buffers. Figure 3.31 proves that the augment of the node density brings about the increase of the average buffer overflow percentage and its standard deviation.

Figure 3.32 displays that the average throughput of each node diminishes though the node density increases. The most important reason is that more packets will be dropped when the buffer overflow percentage increases. Besides, quite a few packets are still in buffers. Figure 3.32 also shows that the probability of the lost packets enlarges together with the augment of the node density, which is the reason for the throughput tendency.

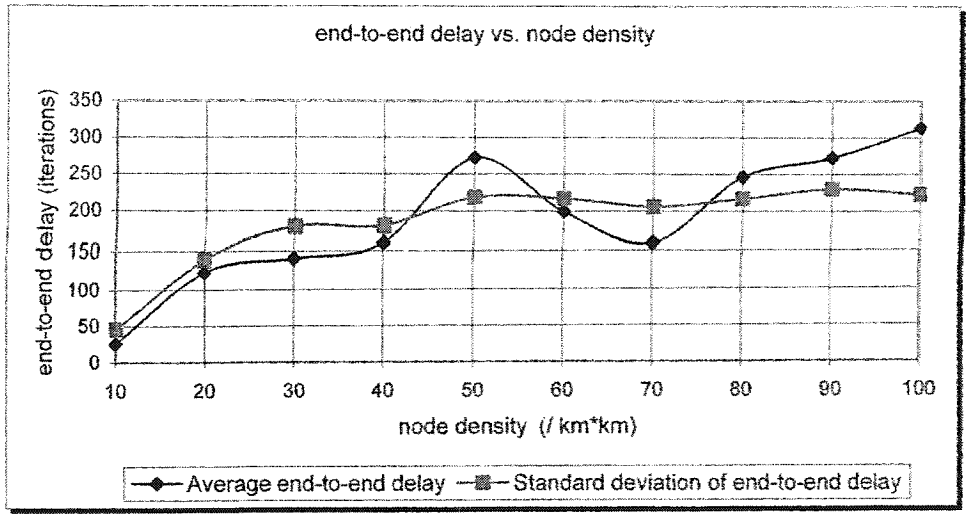


Figure 3.27: End-to-end delay (iterations) vs. node density

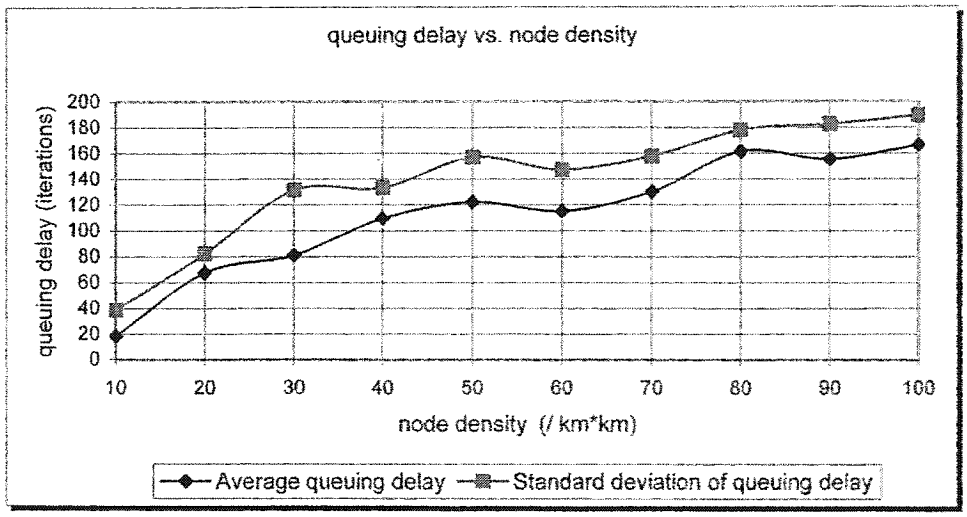


Figure 3.28: Queuing delay (iterations) vs. node density

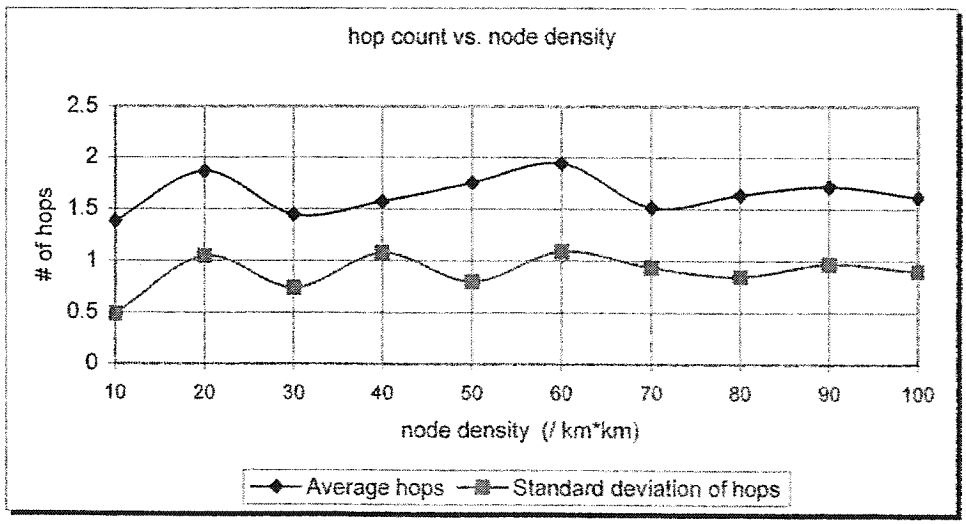


Figure 3.29: Hop count vs. node density

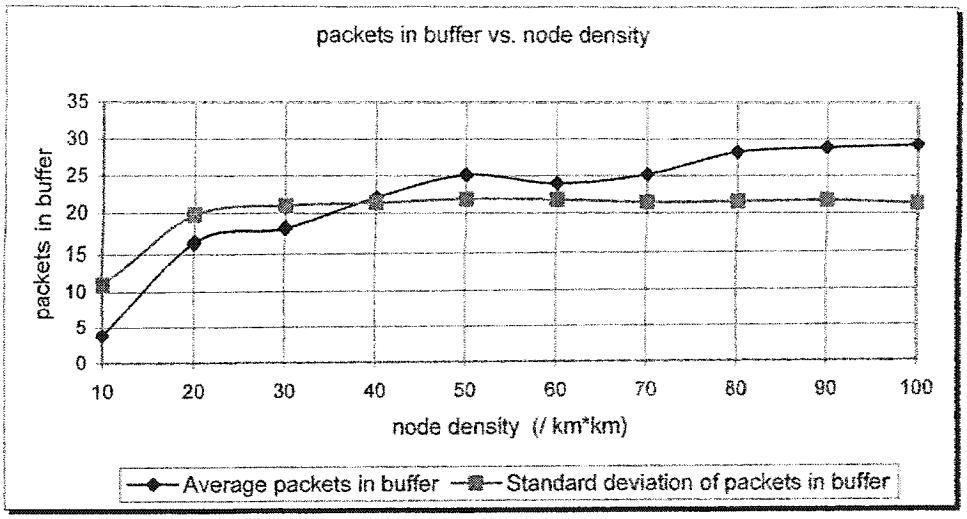


Figure 3.30: Packets in buffer vs. node density

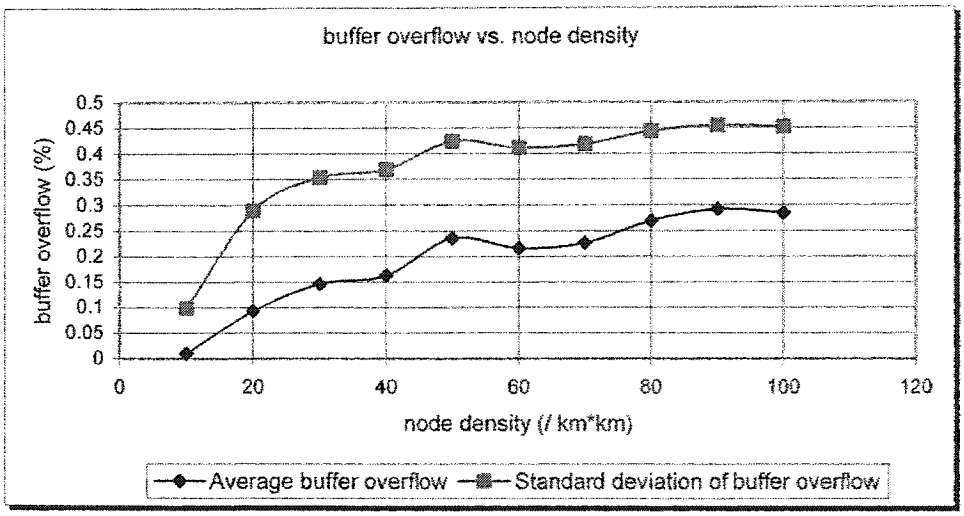


Figure 3.31: Buffer overflow (%) vs. node density

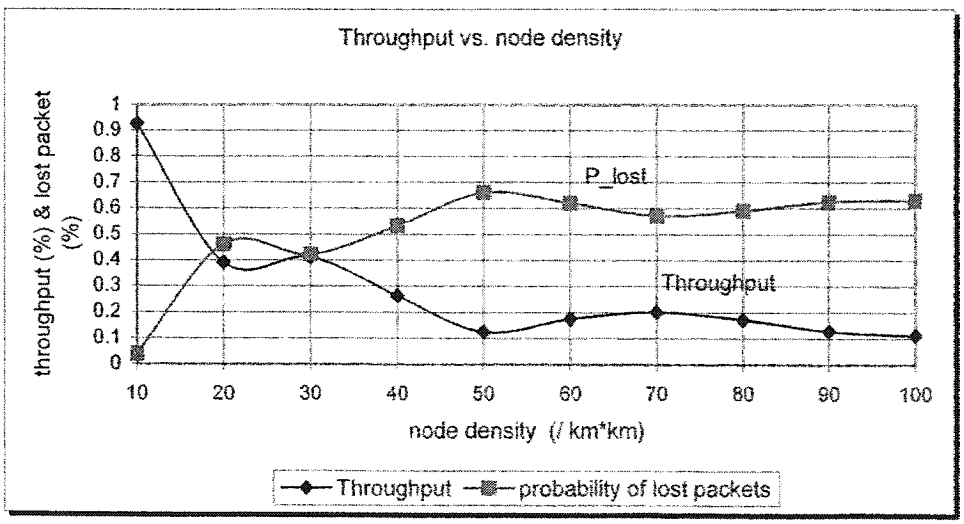


Figure 3.32: Throughput vs. node density

3.3.3. Network Performance under Different Channel Qualities

The channel quality is defined as the probability of the channel in good state. In this simulation, we alter the channel quality to evaluate the network performance. The input parameters that have been used for the simulation are shown in the following table.

Table 8: Parameters used during Channel Quality simulations

Parameters	Value
Environment size	1000m X 1000m
Number of nodes	50
Traffic type	Variable Bit Rate (VBR)
Call active probability	10%
Load	5%, 10%, 20%
Probability of movement	20%
Maximum move angle	360
Maximum speed	20 m/s (or 72 km/h)
Location update period	40 (iterations)
Hop limit	20

Figure 3.33 shows that the average end-to-end delay extends as the channel quality improves from 0.01 to 0.99. When the channel quality becomes better, packets can travel through more hops. As a result, more packets will be in the buffer and will cause longer queuing delay as demonstrated in Figure 3.34. Undoubtedly, the dependency of the end-to-end delay on the queuing delay is extraordinarily great. Figure 3.33 and Figure 3.34 also prove that the average end-to-end delay and the average queuing delay extend as the offered load augments.

Figure 3.35 illustrates that the average hop count increases as a result of the improvement of the channel quality. When a node sends a packet to its neighbors, the packet will be dropped and can not reach its neighbors if the channel is in bad state. When a node receives a packet and forwards it, the packet will also be dropped if the channel

state changes from GOOD to BAD at that time. If a packet arrives at its destination, it means that all channels between any two nodes it has traveled are in GOOD state. Figure 3.35 also proves that the number of hops declines though the offered load augments.

As a consequence of the increase of the hop count, the average number of packets in buffers will enlarge. Figure 3.36 points up that the average number of packets in buffers increases as the channel quality improves. It can bring two results: on the one hand, the average queuing delay will also extend when the average number of packets in buffers enlarges, which is shown in Figure 3.34; on the other hand, the increase of the average number in buffers can give rise to larger probability of the buffer overflow. Figure 3.37 confirms the assumption that the probability of the buffer overflow amplifies when the channel quality gets better. Figure 3.36 and Figure 3.37 also attest that the average number of packets in buffers and the probability of the buffer overflow increase consequently as the offered load augments.

Figure 3.38 gives the curve that the throughput increases as a result of the improvement of the channel quality. Evidently, the throughput is in direct proportion to the channel quality. It can be reckoned that the throughput will become zero when the channel quality is zero, since all packets can not be transmitted successfully and finally will be dropped, even though these packets contend for the shared channels successfully and get a good route via the PAP routing protocol. Figure 3.38 also shows that the throughput lessens as the offered load augments.

It is worth noting that Figure 3.37 and Figure 3.38 are not contradictive. Usually the throughput will lessen as the buffer overflow percentage increases because more and

more packets are dropped. However, when the channel quality advances, larger numbers of packets arrive at their destination than those packets which are dropped. Figure 3.39 verifies that the probability of the lost packets declines as the channel quality gets better. Figure 3.39 also shows an interesting result: when the channel quality is below 0.5, the probability of the lost packets increases along with the decrease of the offered load; whereas when the channel quality is above 0.5, the probability of the lost packets increases along with the increase of the offered load.

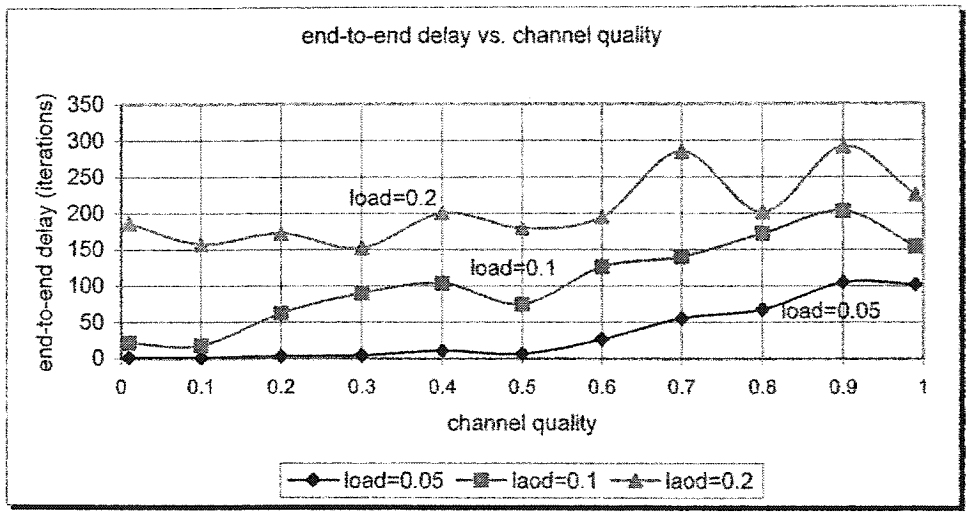


Figure 3.33: End-to-end delay (iterations) vs. channel quality

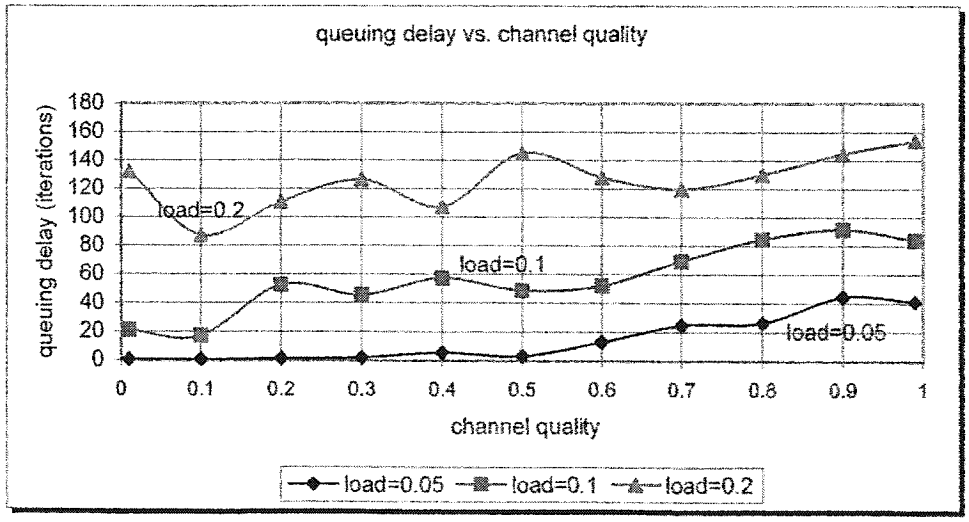


Figure 3.34: Queuing delay (iterations) vs. channel quality

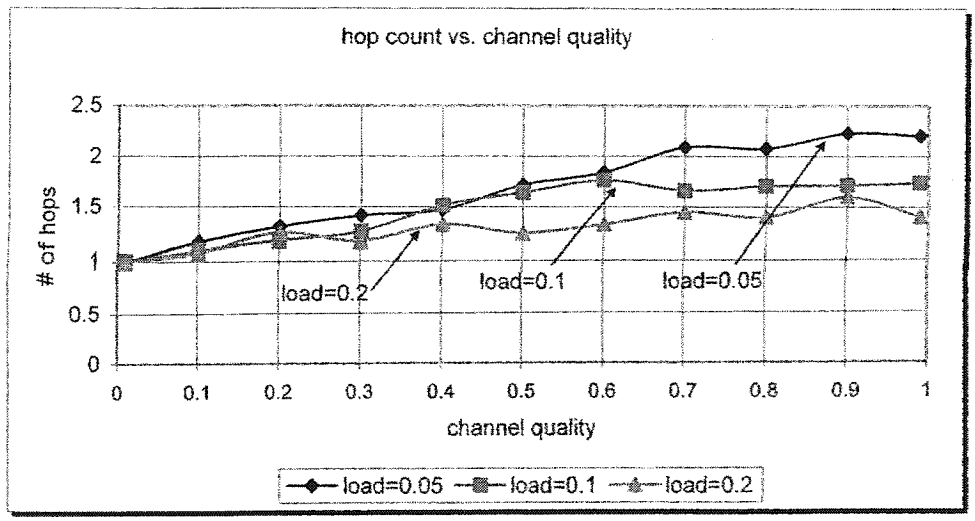


Figure 3.35: Hop count vs. channel quality

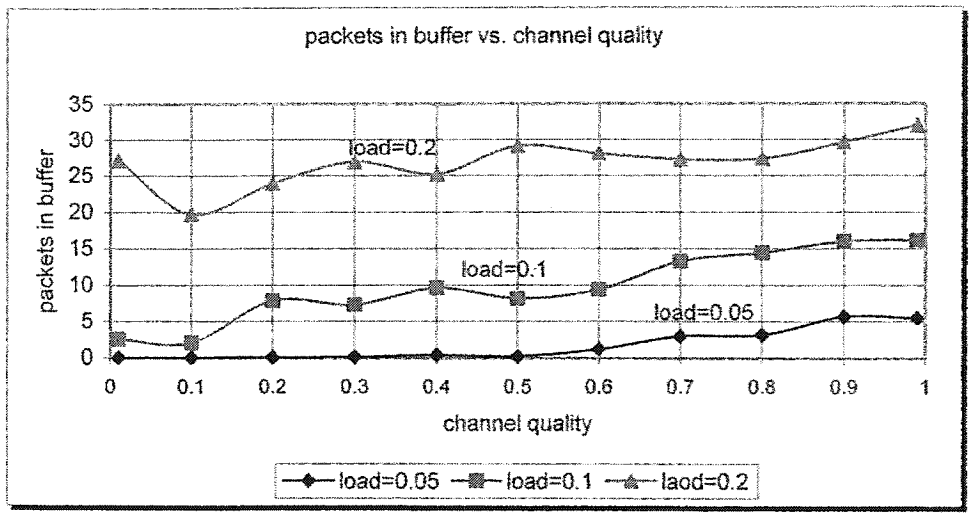


Figure 3.36: Packets in buffer vs. channel quality

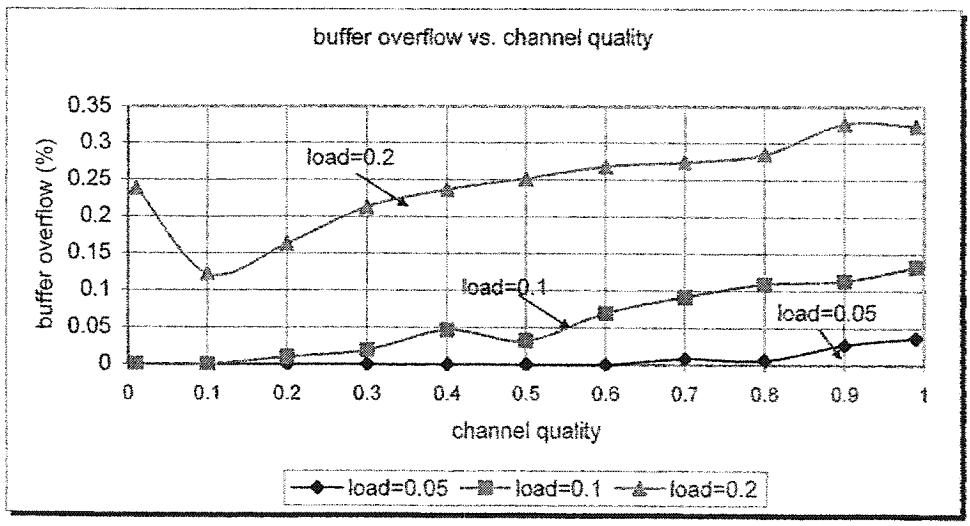


Figure 3.37: Buffer overflow (%) vs. channel quality

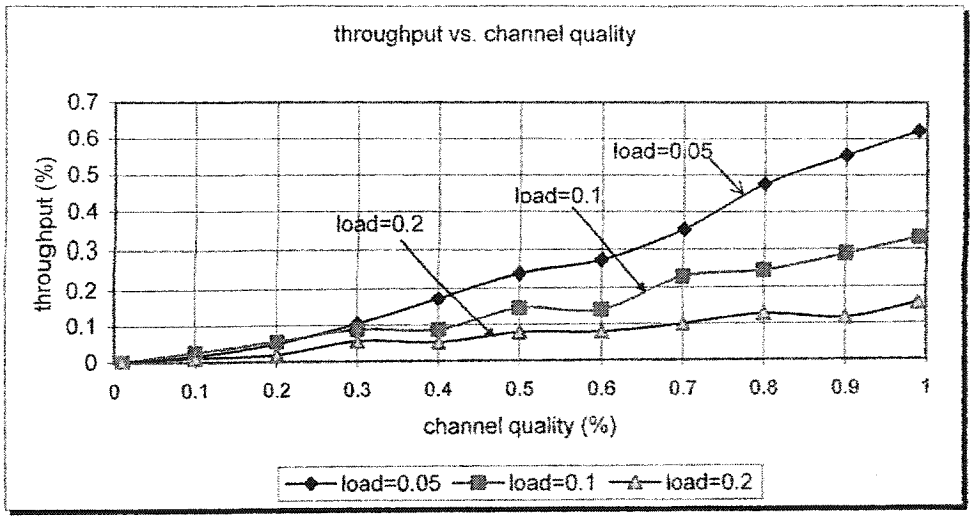


Figure 3.38: Throughput vs. channel quality

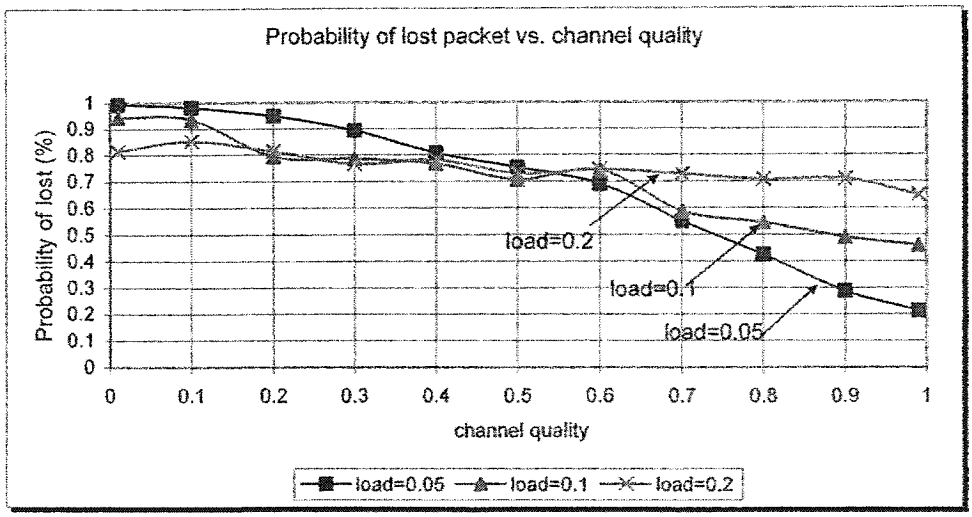


Figure 3.39: Probability of lost packet vs. channel quality

3.3.4 Network Performance under Different Hop Limits

The hop limit is defined as the maximum number of hops after which the packet will be dropped. It is used to limit looping. Packets circulating in a loop will have extraordinary high hop count and thus will be dropped due to the hop limit. All simulations in this part use the hop limit as the major input parameter and the offered load as the minor input parameter. The input parameters that have been used for the simulation are shown in the following table.

Table 9: Parameters used during hop limit simulations

Parameters	Value
Environment size	2000m X 2000m
Number of nodes	200
Traffic type	Variable Bit Rate (VBR)
Call active probability	5%
Load	1%, 3%, 5%
Probability of movement	20%
Maximum move angle	360
Maximum speed	20 m/s (or 72 km/h)
Location update period	40 (iterations)
Channel quality	0.999

Figure 3.40 reveals that the average end-to-end delay extends as the hop limit increases and keeps relatively constant when the hop limit is greater than 4. It is due to the fact that more hops need longer time to be traveled. As the offered load enlarges, the average end-to-end delay also extends, since heavier load can bring about longer queuing delay. Figure 3.41 shows the curves of the average queuing delay. The end-to-end delay is, to a large extent, dominated by the queuing delay. In consequence, the tendencies of the curves in both Figure 3.40 and Figure 3.41 are pretty similar to each other.

Figure 3.42 explains that the average hop count increases as a result of the increase of the hop limit, which is easily to be understood. This figure also says that the average hop count diminishes as the offered load amplifies. The reason is that heavier loads can create higher probability of buffer overflow, and then more packets will be dropped and will not be forwarded by intermediate nodes. Hence those packets will travel through fewer hops.

From Figure 3.41, we can see that the queuing delay extends as the hop limit expands. It is logical that longer queuing delay results from more packets in buffers. Figure 3.43 is a kind of evidence that the average packets in buffers increase as the hop limit expands

but decrease as the offered load lessens. More packets in buffers will make larger probability of the buffer overflow happen. Therefore, Figure 3.44 about the buffer overflow shows similar curves as that of Figure 3.43.

The average failed routes percentage diminishes, in Figure 3.45, as the hop limit increases. When there are fewer hop limits, the PAP routing module will have larger probability to drop a packet before it arrives at its destination because the PAP routing module thinks the packet is in circulative routes. Figure 3.45 also shows that the failed route percentage declines as the offered load amplifies. The reason is that even though only one route fails, the percentage is still large when the offered load is little. In addition, the diminishment of the average failed routes will cause the augment of the throughput. Consequently, Figure 3.46 proves that the throughput increases along with the augment of the hop limit, but decreases when the offered load enlarges. It is because more and more packets will be still in buffers of intermediate hops when the offered load enlarges. Of course, the buffer overflow probability will also expand.

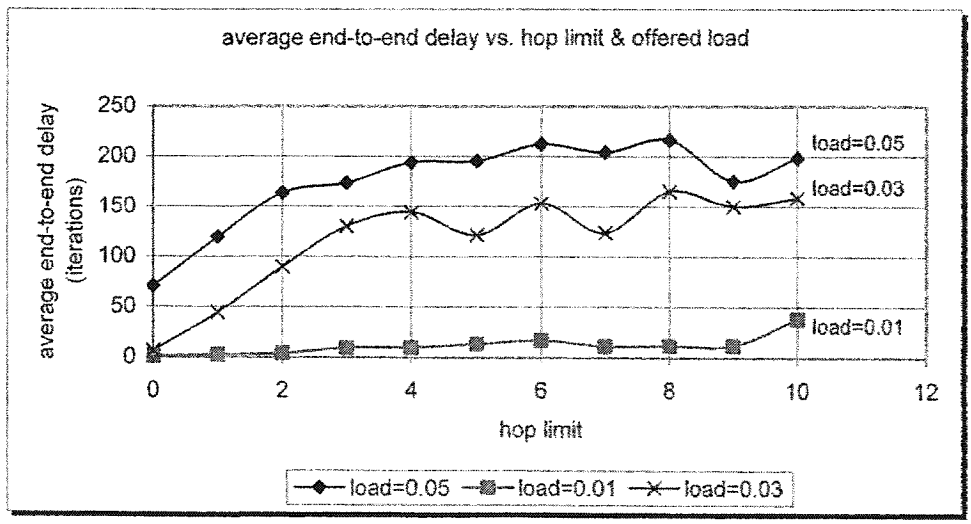


Figure 3.40: Average end-to-end delay (iterations) vs. hop limit & offered load

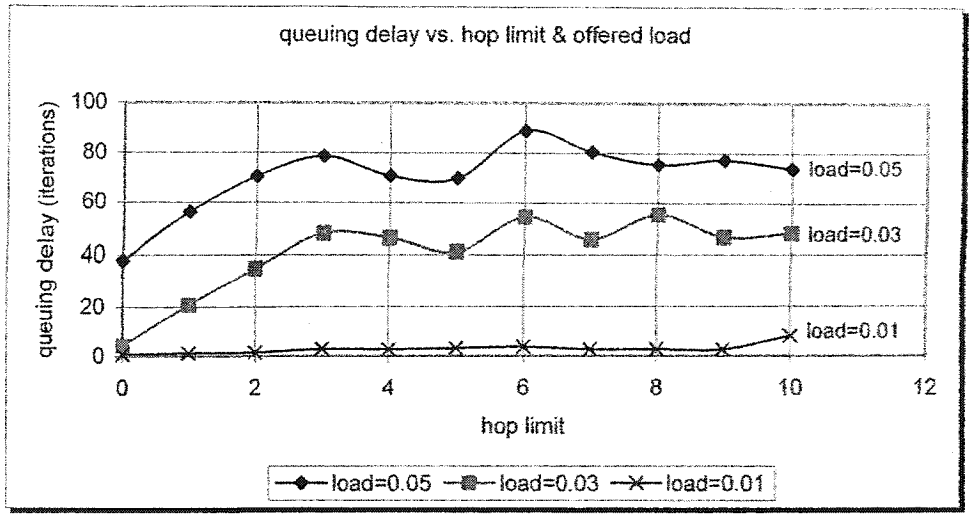


Figure 3.41: Queuing delay (iterations) vs. hop limit & offered load

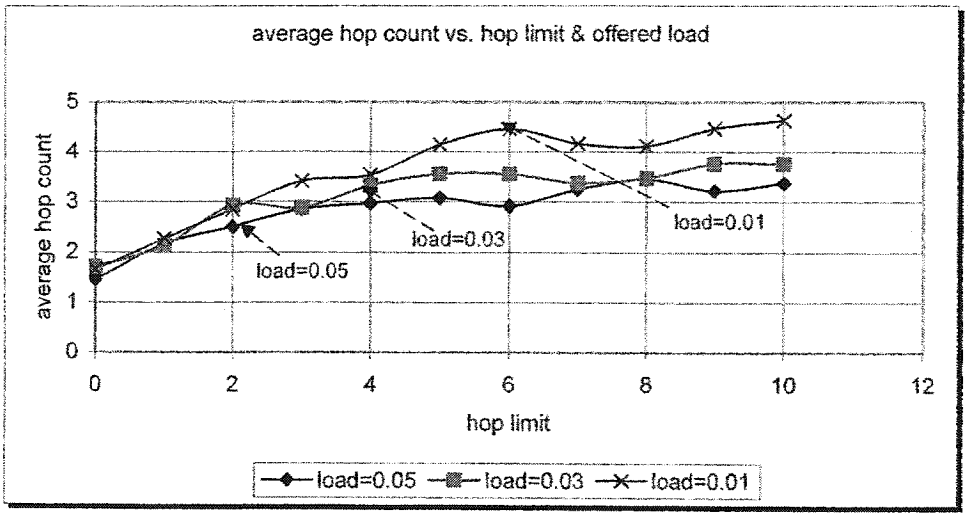


Figure 3.42: Average hop count vs. hop limit & offered load

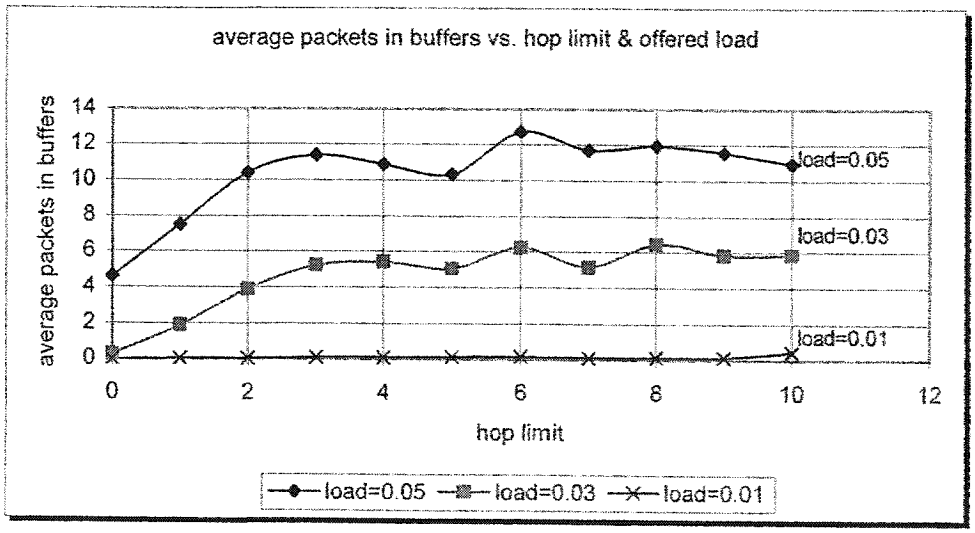


Figure 3.43: Average packets in buffers vs. hop limit & offered load

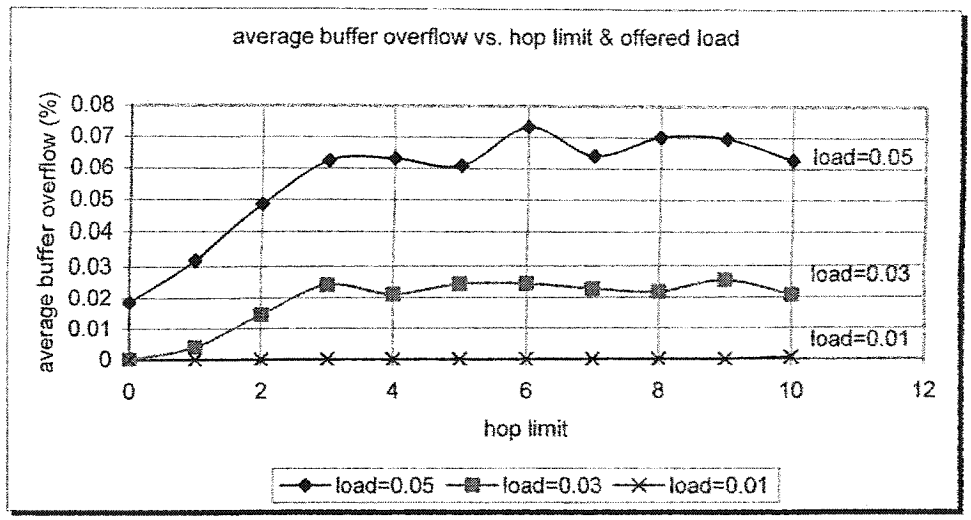


Figure 3.44: Average buffer overflow (%) vs. hop limit & offered load

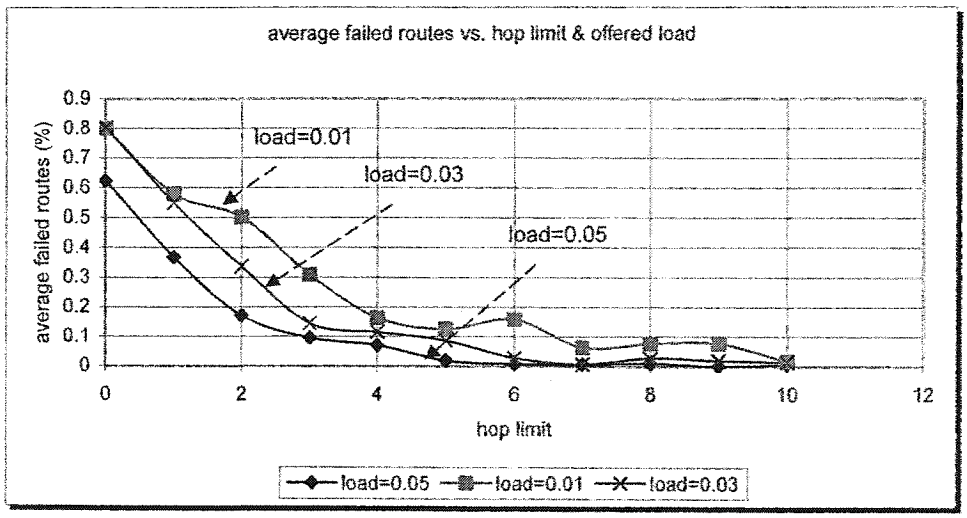


Figure 3.45: Average failed routes (%) vs. hop limit & offered load

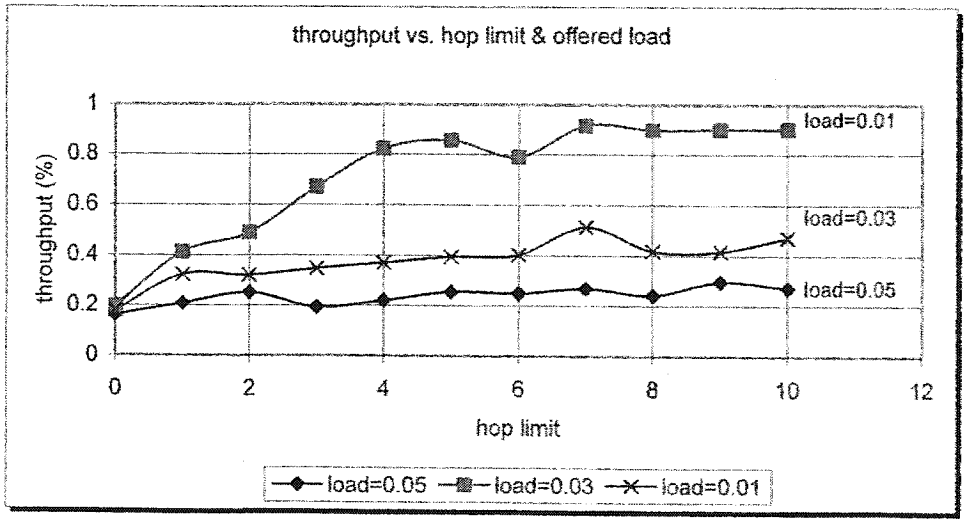


Figure 3.46: Throughput (%) vs. hop limit & offered load

Chapter 4

Conclusions & Future Work

4.1. Conclusions

A WLAN Analyzer/Simulator was designed for purposes of evaluating the performance of the new PAP routing technique, as well as the performance of the new priority based double window WLAN protocol. This simulation software can easily be extended by other researchers. Other wireless LAN researchers may easily integrate their routing protocols into it just by inheriting the RouteDemon module like PAPRouter, and may simply inherit the MAController module to do any modification on MAC layer like QoSMAController module.

Compared with the single window algorithm of the standard CSMA/CA, the newly proposed double window algorithm introduces a priority-based window. The new priority based window scheme can help the node that backs off more times than others to get the channel first. Simulations demonstrate that networks achieve a better QoS performance: less variance of the end-to-end delay, less variance of the queuing delay, less variance of packets in buffers, and less variance of the buffer overflow percentage by using the new priority window. Although this enhancement is not too much, it is still a valuable modification compared with the standard CSMA/CA.

On the routing protocols in ad hoc wireless LANs, it seems that reactive routing protocols and hybrid routing protocols are superior to proactive routing protocols on account of a large mobility. However, proactive routing protocols have less routing delay

because each node has the whole topology information, especially for some QoS-restricted applications, such as real-time audio/video. Proactive protocols have the weakness of a larger overhead because plenty of control packets like HELLO are considered necessary to exchange the topology information. In order to lessen this problem, the PAP routing protocol was recommended in this thesis.

The PAP protocol employs two techniques (namely, GPS and Pseudo Access Points) to enhance the traditional proactive routing protocols, which we have compared to the generic θ routing protocol which is also GPS based. Simulations confirm that the PAP routing protocol is better than the θ routing protocol under different mobility scenarios. It provides higher throughput, less failed routing, less variance of the hop count, etc, and thus provides better Quality of Service for applications.

Within the PAP routing protocol, the Pseudo Access Points (PAPs) are selected to imitate the infrastructural wireless LANs. These PAPs can be regarded as the backbone of the network. The network performances are evaluated under different mobility, different channel quality, different hop limit, and so on. Through a large number of qualitative simulations and analyses, we can see that the PAP routing protocol can be used as an ad hoc routing protocol under certain conditions.

4.2. Future Work

(1) Configuration of the PAP routing protocol. As the network changes constantly in such aspects as mobility, environment size, density of nodes, channel quality, and offered load, how to dynamically configure the network for best performance is still a challenge for future research. A good routing protocol is difficult to find because it should not only

perform well under certain network conditions but also under most possible network conditions. It should be able to modify the network configuration quickly and thus to work well on the changing network traffic.

(2) Modification of the PAP routing protocol. The PAP routing protocol can work without the GPS after a tiny modification: sending HELLO packets to exchange each other's location and other information. It is definite that this method may bring more overhead, however, the overhead is much less than that of the traditional proactive routing protocol, because the topology information can be sent through these PAPs but not through all nodes. More detailed theory and the related simulation lead to another topic for future work.

(3) Power-aware enhancement for the PAP routing protocol. All nodes in a network might not be created equally, especially for power. A node with high power should do most of the packet forwarding and should have higher priority to be selected as a PAP. Thus, power of nodes can be used as one criterion for selecting PAPs. More detailed issues initiate another theme for future effort.

(4) Comparisons with other routing protocols. Performance comparisons of the PAP routing protocol with other ad hoc routing protocols, especially some famous protocols like the reactive protocol of AODV and the hybrid protocol of ZRP, are an additional future work.

(5) Perfection of this WLAN simulator. The simulator designed in this thesis is based on some important assumptions. A better simulator can generate more realistic and more satisfying results. To make it ideal, a lot of issues need to be considered in future work

even for each layer, such as integrating FHSS, DSSS, and IR to the simulator at the physical layer, taking acknowledgement (ACK/NAK) and retransmission into account at the MAC layer, simulating any kind of traffic like constant/variable bit rate (CBR/VBR) at the application layer, etc.

Bibliography

[1] Bhagwat, P. and Screenan, C.J., Eds., "Future Wireless Applications, IEEE Wireless Commum., 9(1), 6-59, 2002.

[2] The Institute of Electrical and Electronics Engineers, Inc. "IEEE Std 802.11 – Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications", 1999 edition.

[3] Hiperlan/2 Global Forum (web page). URL: http://www.hiperlan2.com/web/

[4] HomeRF Wireless LAN (web page). URL: http://www.homerf.org

[5] LAN MAN Standards Committee of the IEEE Computer Society, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications", June 1997. IEEE 802.11.

[6] IEEE 802.11 WG, Draft Supplement to LAN/MAN Specific Requirements - Part 11: Wireless Medium Access Control (MAC) and physical layer (PHY) specifications: Medium Access Control (MAC) Enhancements for Quality of Service (QoS), IEEE 802.11e/D2.0, Nov. 2001.

[7] G. Anastasi, E. De Stefano, and L. Lenzini. Qos provided by IEEE 802.11 wireless LAN to advanced data application: a simulation analysis. In Workshop on Nomadic Computing, Geneva, April 5 1997.

[8] J. L. Sobrinho and A. S. Krishnakumar, "Quality-of-Service in ad hoc Carrier Sense Multiple Access Wireless Networks," IEEE JSAC, vol. 17, no. 8, Aug. 1999, pp. 1353–1414.

[9] E. Crawley et al., "A Framework for QoS-Based Routing in the Internet," RFC 2386, http://www.ietf.org/rfc/ rfc.2384.txt, Aug. 1998.

[10] C. R. Lin and J.-S. Liu, "QoS Routing in Ad Hoc Wireless Networks," IEEE JSAC, vol. 17, no. 8, Aug. 1999, pp. 1426–38.

[11] Z. J. Haas et al., "Guest Editorial," IEEE JSAC, "Special Issue on Wireless Networks", vol. 17, no. 8, Aug. 1999, pp. 1329–32.

[12] M. Matsumoto & T. Nishimura, Mersenne twister (MT), ACM Transactions on Modeling and Computer Simulation, vol. 8, no. 1, 1998, pp. 3-30.

[13] Erich etc., Design Patterns - Elements of Reusable Object-Oriented Software, Addison Wesley, July 2001.

[14] Crow, Brian P., Indra Kim Widjaja, Geun Jeong, and Prescott T. Sakai., "IEEE-802.11 Wireless local Area Networks" IEEE Communications Magazine, September 1997, vol. 35, No.9: pages 116-126.

[15] Perkins, C.E., "Ad Hoc Networks", Addison-Wesley, Reading, MA, 2001.

[16] Borko Furht and Mohammad Ilyas, "Wireless Internet Handbook – Technologies, Standards and Applications" CRC Press, 2003.

[17] Leon-Garcia and Widjaja, "Communication Networks – Fundamental Concepts and Key Architectures", McGraw-Hill, 2000.

[18] Leonard Kleinrock, "Queuing Systems – Volume 1: Theory", John Wiley & Sons, 1975.

[19] Joseph Macker and Scott Corson, IETF Mobile Ad Hoc Networks (MANET) Charter, http://www.ietf.org/html.charters/manet-charter.html.

[20] C. Perkins, E. Royer, and S. Das. Ad hoc on demand distance vector (AODV), Internet-draft, November 2002, http://search.ietf.org/internetdrafts/draft-ietf-manet-aodv-12.txt.

[21] I. Rubin, A. Behzad, R. Zhang, H. Luo, E. Caballero, TBONE, "A Mobile-Backbone Protocol for Ad Hoc Wireless Networks", IEEE AEROSPACE 2002.

[22]Scott Corson and Joseph Macker, "Mobile Ad Hoc Networking: Routing Protocol Performance Issues and Evaluation Considerations". Internet-Draft, draft-ietf-manet-issues-01.txt, March 1998.

[23] Zygmunt J.Haas and Marc R. Pearlman,"The Zone Routing Protocol for Ad Hoc Networks", Internet-Draft, draft-ietf-manet-zone-zrp-04.txt, July 2002.

[24] T. Imielinski and J.C. Navas, "GPS-based addressing and routing", Technical report LCSR-TR-262, Rutgers University, August 1996.

[25] Iowa State University GPS page, available via WWW at URL: http://www.cnde.iastate.edu/gps.html

[26] T. J. Kwon and M. Gerla, "Clustering with Power Control", IEEE MILCOM 1999.

[27] Sanghani, S., Brown, T.X., Bhandare, S., and Doshi, S., "EWANT: the emulated wireless ad hoc network test bed", IEEE Wireless Communications and Networking, Volume 3, March 2003, Pages 1844 – 1849.