***GANGs*: An Energy Efficient Medium Access Control Protocol**

by

Yawen Dai Barowski

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
May 14, 2010

Keywords: MAC, protocol, energy, efficient, GANGs

Approved by

Abstract

Recent advances in wireless communications and electronics have enabled the development of low-cost sensor networks. Low energy storage capacity is one of the critical features of nodes in these networks. Communication protocols at different layers have been proposed in order to reduce energy consumption. In this dissertation, existing work on energy efficiency is discussed. A new energy efficient MAC layer protocol, named *GANGs* is proposed. *GANGs* is a self-organized cluster-based hybrid MAC protocol. It incorporates both contention-free (*TDMA*) and contention-based (*IEEE 802.11*) medium access schemes. The contention-free scheme is deployed by cluster head nodes to construct a contention-free network backbone. The contention-based scheme is deployed by ordinary nodes to communicate with cluster head nodes. This dissertation begins by discussing the existing work on throughput analysis through modeling IEEE 802.11 single hop networks under saturated traffic conditions, develops a mathematical model for the performance analysis of single-hop IEEE802.11 networks under unsaturated traffic, and then extends the model to 3 dimensions to analyze the performance of multi-hop IEEE 802.11 networks and networks using *GANGs* protocol. The performance metrics in the analysis cover not only throughput, but also message delay, queue length, packet drop rate, and energy consumption. With the new hybrid *GANGs* MAC protocol, data collisions for traffic forwarded from cluster to cluster will be eliminated, and thus a significant amount of energy is expected to be saved. The protocol will then be implemented and evaluated through *NS2* simulation.

Acknowledgements

I am heartily thankful to my advisor, Dr. Biaz, for his encouragement, supervision and support. Also to my husband, Larry Barowski, for his unconditionally standing by me during the course of my Ph.D. program. Lastly, I offer my appreciation and respect to Dr. Chang, Dr. Cross, Dr. Chapman, Dr. Hamilton, and Dr. Agrawal.

Table of Contents

List of Figures

Chapter 1

Objectives and Related Work

1.1 Objective of the Dissertation

In wireless ad-hoc networks such as sensor networks, the transmission medium (usually air) is shared by all nodes within in each others' transmission ranges. Due to the fixed amount of energy available and scarcity of the medium spectrum, original data sent by source nodes may not reach the destination nodes (sinks) in one hop. In this case, data will have to be carried over the wireless network through relay nodes. Even if the nodes have enough transmission power to directly communicate with sinks, it is not practical to have thousands of nodes attempting to use the spectrum at the same time. As the density of nodes increases, the spectrum becomes a bottleneck. There are two types of medium access control techniques: contention and contention free. Contention protocols such as Aloha or CSMA (Carrier Sense Multiple Access) let nodes compete freely for the medium. Collisions are caused by multiple nodes transmitting at the same time. If packets collide, they cannot be received properly and must be retransmitted. Retransmissions consume extra energy in the nodes. Contention free MAC protocols schedule the use of the medium to eliminate the occurrence of collisions. Gupta and Kumar [22] have shown that the per-node throughput of nodes capable of transmitting W bits per second over a wireless network with n nodes is O(W/sqrt(n log n)) if a non-contention MAC algorithm is used. As the number of nodes increases, the nodes will eventually be unable to communicate. Even if nodes are optimally placed and traffic is optimally routed, the throughput becomes, at best, O(W/sqrt(n)) per node. This bound is quite restrictive and makes purely non-contention-based MAC protocols inadequate for sensor networks. These bounds have been shown to improve with mobility [21], with relay infrastructures, and with contention-based MAC protocols [19].

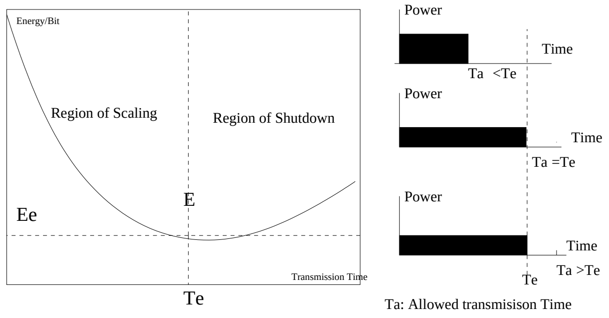
The MAC protocol in a wireless multi-hop self-organized sensor network must achieve two goals. The first is the creation of the network infrastructure from unconnected nodes. The MAC scheme must establish communication links between the nodes. This forms the basic infrastructure needed for hop-by-hop wireless communication and provides the sensor network self-organizing capability. The second goal is to fairly and efficiently share among nodes the communication resources, such as the medium and the energy stored in the nodes [1]. We propose a *hybrid* MAC protocol that combines non-contention (e.g., using time division multiplexing access (TDMA)) and contention (e.g., using CSMA/CA [7]) techniques to achieve both connectivity and energy efficiency. Our proposed protocol uses *non-contention* MAC protocols for relay nodes and *contention* MAC protocols for source nodes. The relay nodes and the contention-free protocol deployed in them construct a contention-free network backbone to establish the communication infrastructure. The backbone provides connectivity as well as energy efficiency, since no collisions will occur and no retransmissions will be needed for the forwarding traffic on the backbone. Section 1.4 describes the details of our proposed protocol.

This dissertation is outlined as follows. In Section 1.2 and 1.3, we discuss the current techniques for saving energy in sensor networks, and related work. In Section 1.4, we give the details of our proposed MAC protocol, *GANGs*. In order to evaluate our protocol, we propose a mathematical model for obtaining performance analysis in Section 2.

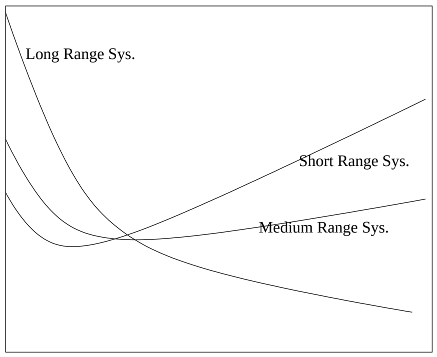
1.2 Techniques for Energy Efficiency

A sensor network is composed of a large number of sensor nodes that are densely deployed either inside the phenomenon being observed or very close to it. The positions of sensor nodes are not necessarily engineered or predetermined. Because of this, general-purpose sensor network protocols and algorithms must be self-organizing. The sensor network results from the cooperative effort of sensor nodes, which can be categorized as: *data source*, a node that generates data; *data sink*, a node that collects data; and *ordinary node*, a node that participates in data forwarding from a data source to a data sink. Although many protocols and algorithms have been proposed for traditional wireless ad-hoc networks, they are not well suited to the unique features and requirements of sensor networks. The number of sensor nodes in a sensor network is commonly several orders of magnitude higher than in an ad-hoc network. Sensor nodes are densely deployed, and are limited in power, computational capacity, and memory. New protocols and algorithms must be designed with consideration for these differences, especially the limited and non renewable power storage of the sensor nodes.

Energy in sensor nodes is consumed by the functions of multiple components such as processing units, radios, sensors, actuators and power supplies. Typically, actuators consume the most energy, followed by radios. Processor and sensor power consumption are usually less important [17]. There are two approaches for achieving energy efficiency. One is radio management. This approach is usually implemented at the *physical layer* of the nodes. Different methods are used to reduce the energy directly consumed by sending or receiving a packet. Another approach for achieving energy efficiency is to exploit application-specific information such as location, timing, specific features of the application, or neighbor information to enable *application layer*, *network layer*, *MAC layer* or *cross-layer* optimizations. The idea is to reduce redundant data transmissions which are caused by collisions, routing requirements, etc.



(a) Energy Vs. Transmission Time



(b) Performance Vs. Range

Figure 1.1: Energy Efficient Techniques: Scaling and Shutdown

For the first approach, *shutdown* and *scaling* are the main techniques for minimizing energy consumption. In the shutdown technique, a node operates at a fixed transmission rate and power level and shuts down the radio after transmission, thereby reducing superfluous energy consumption. Scaling is based on the relation between performance and energy requirements. The node changes properties such as modulation and error coding, trading energy consumption for transmission time. Figure 1.1 part (a) shows the relationship between transmission time on the x-axis and required energy per bit reliably transmitted on the y-axis. For a typical system, there is a minimal energy per bit (E_e), at point E. If the transmission time is longer or shorter than T_e, the energy consumed for each bit is more than

E_e. If the node is allowed to send data for a shorter time than $T_e(T_a < T_e)$, no energy can be saved by scaling. If the node is allowed to send data for a longer period than $T_e(T_a > T_e)$, then the node can use scaling to keep the transmission time near T_e, and shutdown for the rest of the allowed time. Scaling can be done by altering modulation or coding. A modulated radio signal consists of different symbols, which can be of different shape, different frequency, etc. Each symbol represents several bits of data. In modulation scaling, the number of bits represented by each symbol is varied. By decreasing the number of bits per symbol, the transmission time for a fixed amount of data can be increased, and the transmission power can be decreased. In code scaling, the amount of coding overhead, such as error detection or correction, is varied. By increasing the coding overhead, the transmission time for a fixed amount of data can be increased and the transmission power decreased. Both scaling methods attempt to adjust the transmission time to meet the optimal transmission time requirement. [36].

The choice of radio management method is based on the energy-transmission time curve of the system. The shape of the curve depends on the relative importance of RF and electronics, and is a function of the transmission range, as shown in Figure 1.1 part (b). For long-range systems, the energy consumed per bit decreases as the transmission time increases, so these systems have an operational region where they benefit from scaling. For short-range systems, the minimal required energy-per-bit point occurs at a short transmission time point, so these systems have an operational region where scaling is not beneficial and the best strategy is to transmit as quickly as possible then shut down.

For the second approach, using application-specific information for optimization, energy efficiency can be achieved by using the following techniques:

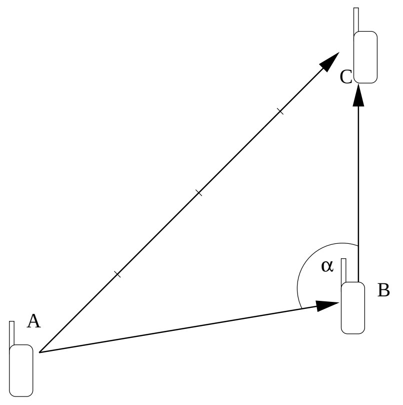


Figure 1.2: Energy Efficient Techniques: Indirect Path

1. Pre-process raw data before transmission, trading communication energy for computation energy. This is called data aggregation.

2. Shut down some unnecessary neighbor nodes.

3. Only forward data to a specific neighbor or set of neighbors according to location information or some other metric.

4. Use indirect routes instead of direct routes. This is based on the fact that under some conditions, using an indirect path consumes less energy than using a direct path. Suppose there are three nodes A, B and C, as shown in Figure 1.2. A sends data to C. In order for C to receive the data, the energy of the signal must be at least e. Since the power of the signal decreases as the magnitude of the square of the distance, A must send out a signal of power $e * AC^2$. If A sends the data to B and B sends the data to C, then the power needed would be,

$$e \cdot AB^2 + e \cdot BC^2 - 2 \cdot AB \cdot BC \cdot \cos\alpha$$

which is less than $e \cdot AC^2$ when α is greater than 90 degrees. Thus the indirect path is more efficient.

5. Increase the packet size. This is based on the fact that turning on a transmitter consumes energy. If the energy to turn on a transmitter is significant compared to the transmission energy, then sending a longer packet will save energy.

One technique in the second approach is MAC layer optimization, which is at the heart of this dissertation. The MAC protocol controls access to the shared transmission medium. Avoiding contention on the communication medium is the goal of the MAC layer optimization. Less contention implies less retransmission. It not only directly reduces the energy consumed by retransmitting the packet, but also affects the operation of the upper layer protocol. For example, if *TCP* protocol is deployed at the upper layer, contention at the medium not only increases the retransmission times at the MAC layer, the packet loss or transmission timeout caused by the contention will also invoke the TCP recovery scheme, thus causing more retransmissions and affecting TCP performance. Our proposed MAC protocol is aimed at reducing contention on the medium.

1.3 Related Work

Several protocols have been proposed to optimize the energy consumption of sensor networks. They cover the areas of network layer protocols, MAC layer protocols, and cross-layer protocols.

1.3.1 Network Layer Optimization

Network layer protocols are designed to handle multi-hop data transmissions. They address the problem of how to route data from sources to destinations that are not within transmission range. Routing schemes that network layer protocols use to solve the problem directly affect energy efficiency. For example, *flooding* and *TCP/IP* are two traditional network protocols. With a flooding protocol, data is transmitted to every node. Lots of energy is consumed through redundant data transmissions. With TCP/IP, the data flow from sources to destinations is direction-oriented, so energy is saved relative to a flooding protocol.

Diffusion [26], Rumor [10], and GPSR [28] are candidates for network layer optimization. **Diffusion** protocol is a task-based network layer protocol. Information in a diffusion based sensor network consists of *interest packets* and *data packets*. *Interest packets* are packets informing sensor nodes about the data that the sink wants to collect. The data sink broadcasts interest packets. Data packets are sent back to the sink by the source when interest packets reach a source that can provide the desired information. Each node that participates in the data forwarding remembers the paths by which data packets arrive, and records the shortest path. The shortest path between the source and the sink is found by reversing the shortest path that is recorded by the intermediate nodes. After the shortest path is found, data packets are primarily transmitted on that path. Diffusion avoids flooding of data packets, and thus saves energy.

In a sensor network in which **Rumor** protocol is deployed, information consists of *queries* and *events*. *Events* are the data that needs to be collected. *Queries* are messages sent out to retrieve events. Rumor is a logical compromise between flooding queries and flooding event notifications. Event flooding creates a network-wide *gradient field*. When a query is generated it can be sent on a random walk until it finds the event path, instead of being flooded through the network.

GPSR heavily uses geography to achieve scalability. It assumes that all wireless routers know their positions and node sources can determine the locations of node destinations. The main idea is to forward packets to the neighbor closest to the destination.

1.3.2 MAC Layer Optimization

Medium access control (MAC) protocol, which lies between data link layer and network layer protocols in the OSI seven-layer model, is designed to arrange access to the medium for all nodes. While the network layer optimizations attempt to optimize the network topology to minimize data flooding and thus decrease energy consumption, MAC layer optimizations focus on decreasing the contention and certain other types of unnecessary energy consumption.

MAC protocols address four main sources of energy consumption: collisions, overhearing, control overhead, and idle listening. *Collisions* occur when neighbor nodes transmit at the same time and the transmitted data is corrupted. *Overhearing* is caused by nodes wasting energy listening to data not meant for them. *Control overhead* consists of the hand-shaking (RTS/CTS) signal sent out to make sure the transmission medium is available before sending data. Idle listening means that nodes have their radios on when there is no data transmission occurring.

There are two categories of existing MAC protocol. The first category includes contention-based MAC protocols such as IEEE802.11 [25]. The main problem with these protocols is that they consume energy by idle listening. PAMAS [37] is based on IEEE 802.11, and uses two different radio channels for signaling and transmitting data. Since it performs signaling on a different channel than the data transmission channel, nodes know whether or not the data is intended for them. This avoids overhearing among neighbor nodes, but does not address the idle listening problem. In the second category are contention free AC protocols such as TDMA. TDMA reduces the energy consumption because it eliminates contention.

Two problems with the TDMA protocol are that it does not scale well and that it requires centralized control of all nodes. The S-MAC [40] protocol is designed specifically for sensor networks. It uses RTS/CTS to avoid collisions. Overhearing is handled by turning off a node's radio when a transmission is not meant for it. Control overhead is handled by message passing. Only one pair of RTS/CTS along with some ACKs are sent during a data transmission burst. Periodic sleeping and listening are used to reduce idle listening. LEACH [23] is a cluster-based MAC protocol. Nodes elect themselves periodically and randomly as cluster-heads, and the nodes in each cluster adopt a TDMA scheme. LEACH does not address inter-cluster communication, so it is not very practical for sensor networks. **ASCENT** [13] is a sub-layer protocol which is designed to work between the network layer protocol and the underlying MAC protocol. Nodes in the sensor network in which ASCENT is deployed select one set of neighbors to be active. Passive neighbors only listen, and do not transmit. If nodes experience degraded performance, they send help messages to some passive neighbors to wake them up. The awakened nodes begin to participate in data forwarding.

As we mentioned in Section 1.1, the goal of this dissertation is to create an energy-efficient protocol at the MAC layer.

1.3.3 Optimization with Node Clustering

The main purpose of node clustering mechanisms is to provide a way for ad-hoc networks to arrange their nodes to work in groups, and keep inter cluster information exchange among cluster heads in order to reduce redundant network traffic while preserving network connectivity. Node clustering mechanisms can provide improved network performance through different network layers.

From the network layer point of view, in a clustered network routing information can be collected and maintained by each cluster head. A normal node can get the information through one-step traffic or simply use its cluster head to forward all its traffic. From the MAC layer point of view, node clustering mechanisms could provide a way to balance contention-based networks and contention-free networks. While contention-based networks have the benefit of less overhead energy used in establishing topology and resource allocation, contention-free networks have the benefit less energy consumption caused by communication collisions. By arranging the network nodes in clusters, and adopting different MAC mechanisms within individual clusters and among different clusters, communication traffic can be controlled and constrained to smaller scopes. With less distributed traffic, network contention will be reduced, and thus energy efficiency will be improved. However, energy consumption overhead during cluster establishment, cluster maintenance procedures, and possible cluster re-establishment procedures need to be considered when evaluating total system energy consumption.

The cluster establishment procedure is primarily a cluster head election procedure. Many approaches have been proposed, focusing on different aspects such as network connectivity, network mobility, and network energy consumption [2, 4, 3, 35, 6, 5, 14, 20]. They all involve heuristic searches that are based on characteristics of the network nodes. Some of them make use of static node information such as node ID; some of them make use of the dynamic information such as degree of network connectivity, inter node distance, and node energy status; some of them take into account multiple characteristics with associated weight factors to determine the selection of cluster head nodes.

In the identifier-based clustering algorithm proposed by Baker and Ephremides [4, 3], each node is assigned a unique id, and this id is used as the metric for cluster head election. A node with the minimum id value among neighbor nodes is elected as the cluster head. Supporting nodes that lie in the overlapped area of two clusters will serve as *gateway* nodes to ensure network connectivity. Node ids do not encode actual node characteristics. Nodes with lower id values will frequently serve as cluster heads, and thus consume more energy than others. On this basis, the algorithm is considered to be "biased". In the connectivity degree-based algorithm proposed by Gerla and Parekh [20], the degree of a node, indicating how many neighbor nodes it can reach, is used as the metric for cluster head election. A node with maximum connectivity degree is elected as a cluster head. As in the identifier-based algorithm, no cluster heads are directly connected, and nodes that lie in the overlapped areas will serve as *gateways*. It is observed that due to the frequent cluster head updates caused by node mobility and connectivity changes, as the node number in a cluster increases the system will experience low throughput and degraded performance. In the node-weight-based algorithm proposed by Basagni, *distributed clustering algorithm* (DCA) and *distributed mobility-adaptive clustering algorithm* (DMAC) [6], a node is assigned a "weight" based on certain criteria including node mobility, and the node that has the highest weight among neighbor nodes is elected as the cluster head. Node weights are evaluated and modified periodically. This algorithm focuses more on handling the connectivity and mobility than throughput and energy optimization. A heuristic algorithm that takes multiple node characteristics into account, *weighted clustering algorithm* WCA, is proposed in [14]. The algorithm uses four main criteria to determine whether a node should be elected as a cluster head. These criteria are degree of connectivity, average distance to all neighbors, average node speed (mobility), and cumulative duration of a node having served as a cluster head (energy awareness). Different weight factors are applied to the four criteria to evaluate the node. The weight factors can be adjusted according to the system parameters to achieve less frequent cluster head re-elections. The node that has the least weight among neighbors is elected as the cluster head. It is observed that the use of WCA results in fewer cluster re-elections than the connectivity-based and identifier-based algorithms described above.

1.4 *GANGs* Protocol

We propose to create a cluster-based MAC protocol that takes advantage of the mechanisms of both contention-based and TDMA protocols. Our assumption is that for most nodes, forwarded traffic is much heavier than originated traffic. That is, most of the bandwidth is used to forward other nodes' data. The purpose of our protocol is to avoid contention for forwarded traffic. The network of sensor nodes is divided into clusters. Each cluster has a cluster head. Cluster heads form the backbone of the sensor network. This backbone carries forwarding data from one cluster to another. Nodes in one cluster only talk to their cluster heads. The backbone of cluster heads can be exploited by a network layer protocol to optimize routing. The TDMA scheme is adopted for communication between cluster heads. A contention-based scheme is used among nodes within each cluster. Time is divided into frames. Each frame is divided into several slots. There are two kinds of slots, contention-free TDMA slots and contention-based slots. A contention-free TDMA slot is dedicated to cluster heads. A contention slot is a piece of time that is shared among all the nodes in the cluster for exchanging data with their cluster heads. Each frame has several TDMA slots and one contention slot. The number of TDMA slots depends on the number of connections with neighboring cluster heads. The radio for each node (other than a cluster head) is turned off during all TDMA slots and turned on during the contention slot. A cluster head's radio is always on. Each cluster head communicates with its neighbor cluster heads during TDMA slots. It sends out its data to its neighbor cluster heads during its dedicated TDMA slot and listens to the data from neighbor cluster heads during their TDMA slots. Thus, the bandwidth for communication between cluster heads is reserved, and contention caused by the large traffic among cluster heads will be reduced.

We call our protocol *GANGs*. Each cluster acts like a gang. The most powerful node (with highest remaining energy) will be elected as cluster/gang head. Each cluster head controls its own cluster, and negotiates with other clusters/gangs through their cluster heads. The gangs construct a network in which a transmission can reach every cluster. Because cluster heads are doing more work and consuming more energy than other nodes, a cluster head will eventually become less powerful than another node in its cluster. When this happens, a more powerful node will take over the position and cause a reconfiguration of clusters/gangs.

1.4.1 A *GANGs* Scenario

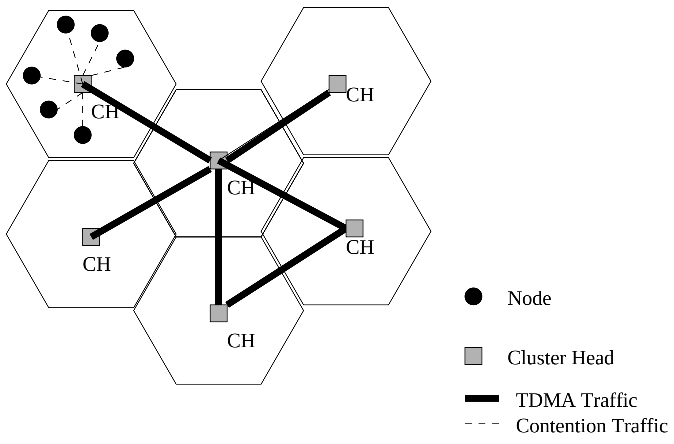


Figure 1.3: *GANGs* Protocol Scenario

In all the figures in this dissertation, cluster heads are represented by shadowed squares and ordinary nodes are represented by filled circles. A dashed and external circle represents the radio range of a node. Figure 1.3 illustrates the *GANGs* protocol. Cluster heads construct the contention-free network backbone. Within each cluster, nodes communicate with the cluster head using a contention-based protocol.

1.4.2 Time Frame of *GANGs*

1. For Cluster Head

   The time frame for cluster heads is shown in the upper part of Figure 1.4. Nodes A,B,C,D,E, and F are cluster heads. Each of them uses a specified time slot to communicate with others. In the contention time slot, they communicate with the ordinary nodes within their own clusters.

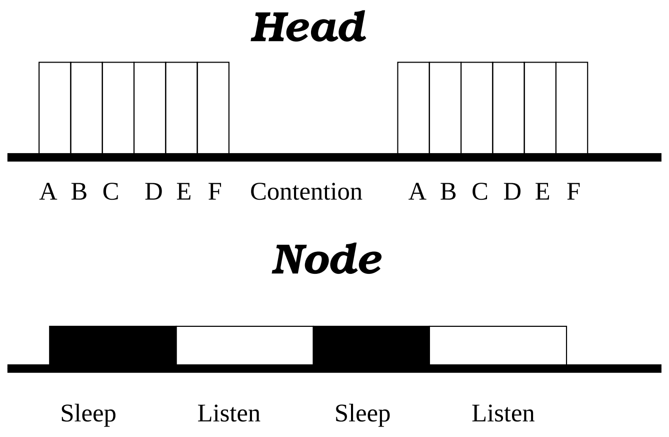


Figure 1.4: *GANGs* Protocol: Time frame for cluster head & node

2. For Ordinary Nodes

   The time frame for ordinary nodes is shown in the lower part of Figure 1.4. When cluster heads are talking to each other, ordinary nodes turn off the power and enter *sleep* mode. They wake up during the contention time slot and begin to communicate with their cluster heads.

1.4.3 Establishing Clusters

1. *Local maximum stage*

   When a node enters the network, (see Figure 1.5) it communicates with its neighbors and provides its energy information. At the beginning of setup, the node that has the maximum energy among all its neighbors, the "local maximum", claims that it is a cluster head and sends this claim to its neighbors. Its neighbors decide whether or not they will accept this cluster head. In the following description, a cluster head will simply be referred as a "head". Note that no head is within any other's range at this stage

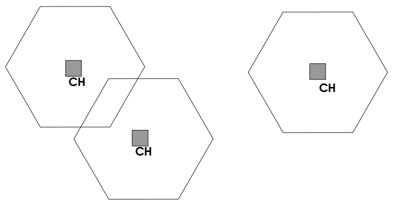


Figure 1.5: *GANGs* Protocol: Local Maximum Stage

2. *Inter − Cluster stage* Add more cluster heads to construct the back bone

   After the first stage, a node that is not a head will be in one of the following three situations: situation 1, it is in the range of one head and accepts the head; situation 2, it is in the range of multiple heads and needs to choose one head among them, as shown in Figure 1.6 case 1; or situation 3, it is not in the range of any head, as node A shown in Figure 1.6 case 2. In Figure 1.6 case 1, nodes A and B are in the intersection of two clusters. Both nodes receive claims from heads C1 and C2. These nodes are aware of each other's energy information. The node that has more power, node A in this case, will claim to be a new head.

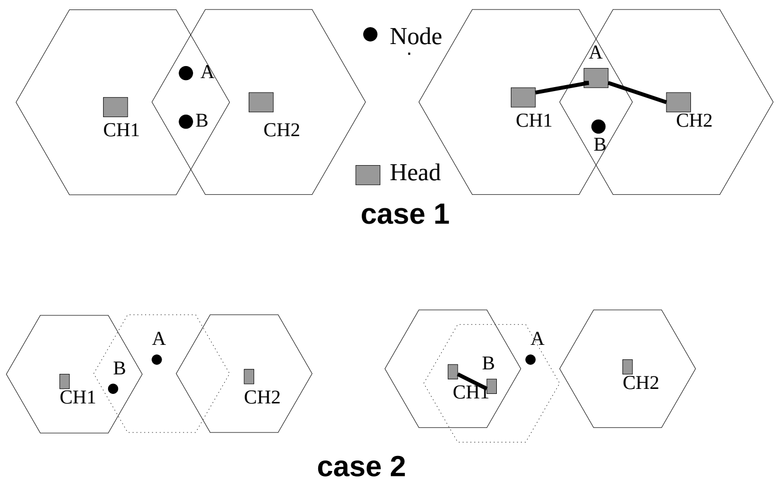


Figure 1.6: *GANGs* Protocol: Inter-Cluster Stage

In Figure 1.6 case 2, node A is not in the range of any head. Within its own range, which is represented by the dashed hexagon, there is a local maximum node B. B accepted a head during the local maximum stage. In this situation, node A sends a message to B to demand head service. B then proclaims itself a head and the topology changes. Each node that has not had a head yet will fall into situation 2 or 3 specified above. In this way, the backbone will gradually be constructed. According to the paper, ”optimum transmission radii for packet radio networks or why six is a magic number” [31], if the network degree is approximately six, the probability that the network is fully connected is about 0.95. So we can assume that if each head has about six neighbor heads, the cluster heads on the backbone will be fully connected, and thus the whole sensor network will be fully connected.

3. *Reconfiguration stage*

The energy consumption of heads is usually higher than for ordinary nodes. After a while, the head may not be the most powerful node in its cluster. When a node is more powerful than its cluster head, and conditions based on energy information and other metrics are satisfied, reconfiguration will be invoked. The current local maximum will elect itself and start the reconfiguration.

1.4.4 Arrange TDMA Schedule

Because of the required synchronization between nodes, TDMA networks are not scalable. In a sensor network, synchronization is not required for the entire network. Only synchronization between neighboring cluster heads is needed. After the clusters are established, we only need to consider the TDMA schedule among the cluster heads that comprise the backbone.

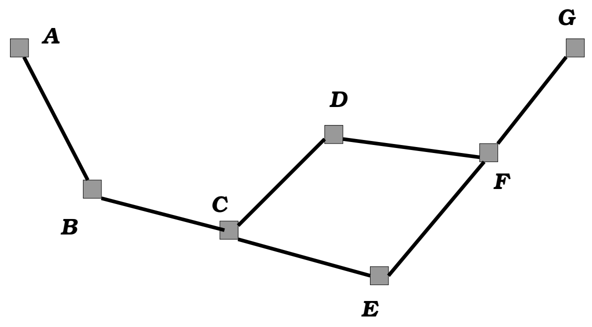


Figure 1.7: *GANGs* Protocol: TDMA Schedule between Cluster Heads

For the situation in Figure 1.7, a sample schedule could be arranged as follows:

A: AB**

B: ABC*

C: EBCD

D: *FCD

E: EFC*

F: EFGD

From each cluster head's point of view, four TDMA time slots are assigned for peer cluster heads. For example, cluster head A acknowledges that slot 1 is for itself, slot 2 is for neighbor B, and the following two slots are assigned to some other cluster heads that are not in its own transmission range, but may be in its neighbors' transmission ranges.

According to the schedule, we could arrange a time frame of specific length, call it T:

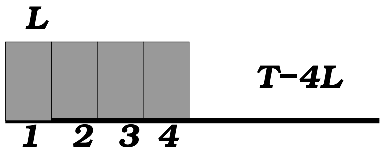


Figure 1.8: *GANGs* Protocol: TDMA Time Frame

Shown in Figure 1.8. Each T-4*L period of time is used for contention-based traffic. Heads send TDMA schedules to their nodes, and the nodes will shutdown during TDMA slots and wake up during contention slots. Every cluster has the same frame length. It is not necessary that every cluster have the same contention slot length. The actual contention slot length should be based on the connection information of the current head, such as how many neighbor heads it has.

Two items should be considered in order to set up the TDMA schedule:

1. Time slot arrangement

   Each head knows its neighbors' information and the total number of neighbors. For example, in Figure 1.7 Node A has one neighbor and Node C has three neighbors. Each head randomly chooses a number from one to the number of its neighbors plus one. Node A randomly chooses a number from one to two and Node C randomly chooses a number from one to four. Node A and C send out the numbers to their neighbors. If their chosen numbers are the same, the head that has either fewer or more neighbors will change its selection. A good algorithm is required to achieve fast scheduling.

2. Synchronization method

   Cluster heads adjust according to the information of their neighbors. Other nodes follow the head to which they belong. We are still considering whether or not it is necessary to reserve a time slot for signaling.

In Chapters 2 and 3, mathematical models will be reviewed and established to analyze the performance of $IEEE802.11$ and $GANGs$ protocols under both saturated and unsaturated traffic, and in both single-hop and multi-hop networks. In Chapter 5, $GANGs$ protocol will be implemented in *NS2* simulation networks and evaluated.

Chapter 2

Mathematical Approach to Performance Analysis

2.1 Overview

The performance of IEEE 802.11 in multi-hop wireless networks depends on the characteristics of the protocol itself, and on characteristics of the upper layer routing protocol. Extensive work has been done to analyze and evaluate the performance of single hop networks under saturated traffic conditions, through both simulation and mathematical modeling. Little work has been done on the analysis of the performance of IEEE 802.11 protocol under *unsaturated* traffic conditions that arise in multi-hop networks. Our intention is to establish a model to describe the protocol behavior under such a situation. In this chapter, existing work on performance analysis is discussed, and an analytical model and scenarios are established to analyze the IEEE 802.11 protocol under unsaturated traffic conditions for single-hop networks. In chapter 3, the model is extended to analyze the behavior of IEEE802.11 and the proposed *GANGs* protocol in multi-hop networks. *NS2* simulations with different network configurations validate the proposed models for performance metrics such as throughput, message delay, average queue length, and energy consumption. Simulation results show that the proposed models work well.

2.2 Introduction

IEEE 802.11 [25] medium access control (MAC) protocol is currently the most popular random access MAC layer protocol used in wireless ad-hoc networks. It uses a distributed coordination function (DCF) as the primary mechanism for accessing the medium. DCF has two modes: the basic broadcast mode, and the MACAW [7, 27] based RTS/CTS mode (Request To Send/Clear To Send). The efficiency of the IEEE 802.11 protocol directly affects utilization of channel capacity and system performance. Performance analysis of IEEE 802.11 has been done experimentally and analytically: *saturated* throughput of IEEE 802.11 has been extensively investigated [8, 11, 15, 33, 41]. Bianchi [8] proposed a two-dimensional Markov chain model to analyze the performance of the IEEE 802.11 exponential backoff scheme, and to evaluate the saturated throughput. There are inaccuracies in Bianchi's model, mentioned in [41] which also proposes modifications.

Other performance metrics such as message delay, data loss, power consumption, and scalability, were investigated in [9, 12, 16, 24, 29, 32, 39].

In all previous work, one or more performance aspects were reported for *single hop* IEEE 802.11 networks under *saturated* traffic conditions. Inspired by Bianchi's saturated throughput model, we propose a model to describe the behavior of IEEE 802.11 under different offered traffic loads. This model shows the effect of the offered load on transmission probability. We also propose a three dimensional model to attempt to describe the behavior of multi-hop 802.11 networks. The 3D model allows the modeling of not only data sources (as in Bianchi's model) but also relay stations that forward traffic.

Section 2.3 of this paper briefly describes the IEEE 802.11 DCF scheme, stressing key elements related to this paper. In Section 2.5, work related to this paper [8] and our model for analyzing the protocol under *unsaturated* traffic loads is discussed. This model is extended in Chapter 3 to a three dimensional model that could be used to model IEEE 802.11 in multi-hop networks.

2.3 *IEEE* 802.11 Mechanism

Contention based MAC $IEEE$802.11 has two working modes, PCF and DCF. PCF is point coordination function mode, which is designed for infrastructure network configurations. This access method uses a point coordinator that operates at the base station to determinate which wireless station has the right to transmit. DCF is distributed coordination function, which is known as "carrier sense multiple access with collision avoidance", or $CSMA/CA$ in $IEEE$802.11. DCF is the mode of interest in this dissertation.

DCF allows for automatic medium sharing between compatible PHYs through the use of the $CSMA/CA$ and a random backoff time following a busy medium condition. Carrier sense is performed both through a physical mechanism at the PHY layer and a virtual mechanism at the MAC layer. The virtual carrier sense mechanism is achieved by distributing reservation information announcing the impending use of the medium. The exchange of the RTS and CTS frames prior to the actual data frame is one means of distributing this medium reservation information. A wireless station desiring to send data should invoke the carrier sense mechanism to determine the state of the medium. If the medium is idle for a specific length of time, the wireless station should generate a random backoff period for an additional deferral before transmission. Figure 2.1 illustrates the access procedure of DCF.

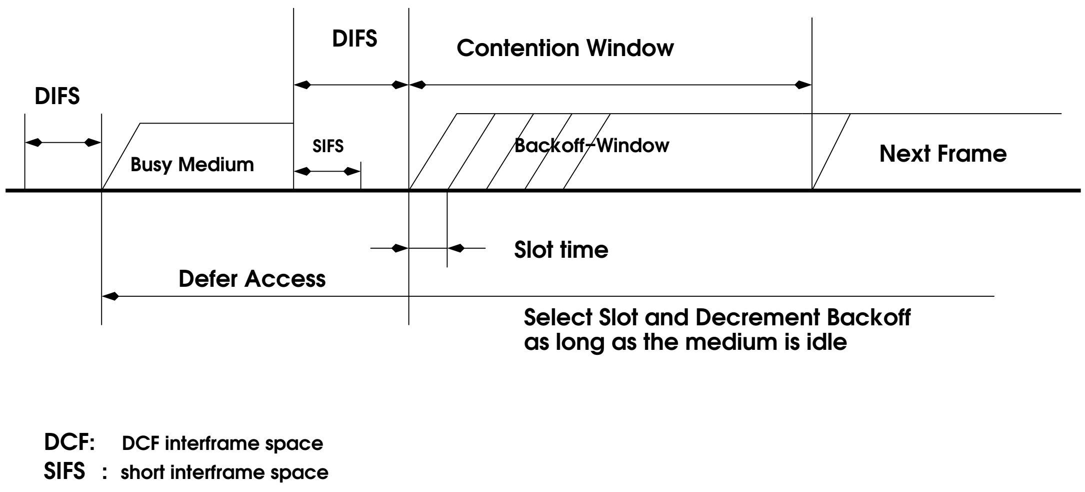

DIFS
DIFS
Contention Window
SIFS
Busy Medium
Backoff-Window
Next Frame
Slot time
Defer Access
Select Slot and Decrement Backoff as long as the medium is idle
DCF: DCF interframe space
SIFS : short interframe space

Figure 2.1: Basic Access Method of DCF

The backoff procedure is invoked when the medium is found busy. The MAC sets its $Backoff\ Timer$ to a random backoff time using the formula,

$$Backoff \quad Time = Random() * aSlotTime$$

where $Random()$ in an integer in the range of minimum contention window size to maximum contention window size. $aSlotTime$ is determined by the PHY layer. A MAC performing the backoff procedure uses the carrier sense mechanism to determine whether there is activity during each backoff slot. If no medium activity is indicated, the backoff procedure will decrement the backoff time by $aSlotTime$. Otherwise the backoff procedure is suspended and the backoff timer will not decrement. Figure 2.2 illustrates a successful transmission between source and destination.

If a transmission succeeds, the wireless station will follow the same procedure for the next transmission. If a transmission fails, the DCF will repeat the procedure with an exponential backoff mechanism. That is, at the first transmission the range of $Random(0)$ is from zero

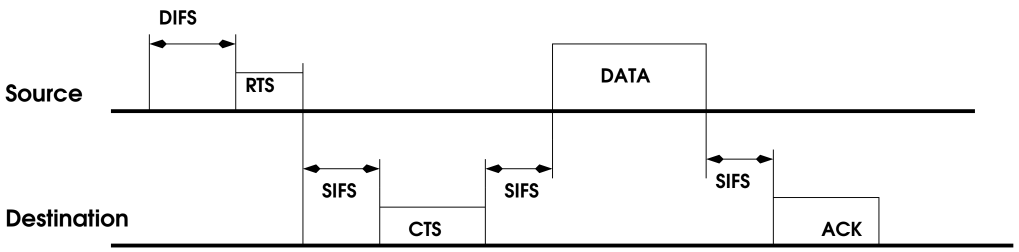


Figure 2.2: Data transmission of DCF

to CW_0, contention window at stage 0. After each failure, the range of $Random()$ is set to zero to,

$$Range \quad at \quad nth \quad transmission = 2^{n-1} * CW_0$$

We use the term $BackoffStage$ to describe the retransmission times. The first transmission attempt is called stage 0, the first retransmission is called stage 1, and so on. So at different stages, the range of contention window sizes are different. After each successful transmission the stage is reset to 0.

2.4 Related Work on Performance Analysis

IEEE 802.11 [25] medium access control (MAC) protocol is currently the most popular random access MAC layer protocol used in wireless ad-hoc networks. It uses a distributed coordination function (DCF) as the primary mechanism for accessing the medium. DCF has two modes: the basic broadcast mode, and the MACAW [27, 7] based RTS/CTS mode. The efficiency of the IEEE 802.11 protocol directly affects utilization of channel capacity and system performance. Performance analysis of IEEE 802.11 has been done experimentally and analytically.

Throughput of IEEE 802.11 has been investigated in [11], [8], [33], [15], and [41]. Cali, Conti, and Gregori [11] established a theoretical upper bound for IEEE 802.11 protocol capacity, showed that the standard may perform poorly, and proposed an appropriate tuning of the backoff algorithm to allow throughput to approach the theoretical upper bound. Bianchi [8] proposed a two-dimensional Markov chain model to analyze the performance of the IEEE 802.11 exponential backoff scheme and to evaluate saturated throughput. There are inaccuracies in Bianchi's model, mentioned in [41], which also proposes modifications. In [33], Robinson and Randhawa extended Bianchi's DCF model to analyze the IEEE 802.11 enhanced distributed coordinated function (EDCF). They modified Bianchi's original model to include post-collision period treatment, which is not valid in the DCF definition. In [15], Chhaya and Gupta proposed a method that uses location information to estimate the lower bound of system throughput. All of this work is based on single hop networks.

Other performance metrics such as message delay, data loss, power consumption, and scalability, were investigated in [12], [32], [39], [24], [29], [16], [9]. Marcelo [12] derived the average delay based on Bianchi's model and results. Marcelo used a probabilistic approach to analyze the average service time for each packet, then derived the packet access delay. [32] summarized Bianchi's work, provided an estimation of the average backoff window size during each transmission, and thus derived the average delay. [39] considered another approach: the station was considered as a server and a G/G/1 queue model was used to analyze the average queue length and delay under unsaturated traffic. In [29], a full state Markov bit error model was established to model the behavior of 802.11b MAC-to-MAC channels for both bit errors and packet loss. Power consumption of wireless networks was investigated in [18] and [9]. [18] provided a linear model to estimate the power consumption of wireless interfaces based on the location of stations and the size of packets, but this model focuses on single packet transmissions. [9] introduced a power saving mechanism, Distributed Contention Control (PS-DCC).

In all work discussed above, one or more performance aspects were reported for single hop IEEE 802.11 networks under saturated traffic conditions. In [38], throughput for CSMA systems under different traffic loads was mathematically analyzed, and closed-form throughput formulas for finite and infinite numbers of stations were presented. Lee and Kim [30] made use of the results from Takagi and Kleinrock's work [38] and added consideration of the capture effect in the throughput and delay analysis of the CSMA/CA system. [42] took into account the delay results from [41], and presented numerical results to show the impact of a finite number of stations on performance. [30] and [42] are inspired by [38]. For the analysis in [38], the system idle periods, in which no station has a packet, are exponentially distributed, each empty station receives a packet with probability g, which determines the offered traffic load, and each station is a P-persistent station that sends packets with probability P in any time slot in which it has packets. These analyses assume that P is independent from g, the offered traffic load. However, for IEEE 802.11, the probability that a station sends a packet in any time slot when it has a packet does depend on the offered traffic load. When the offered traffic load is high, more collisions will occur and the backoff procedure will be more likely to enter a higher stage. The transmission probability will thus decrease. We will take this issue into consideration in our proposed model.

2.5 Bianchi's Model

Bianchi [8] proposed a two-dimensional Markov chain model to analyze the performance of the IEEE 802.11 protocol. His work is often referred to in other related work. Our proposed model is also inspired by his work. The model is shown in Figure 2.3. Two parameters, *backoff stage* and *backoff counter value* are used to describe the state of an IEEE 802.11

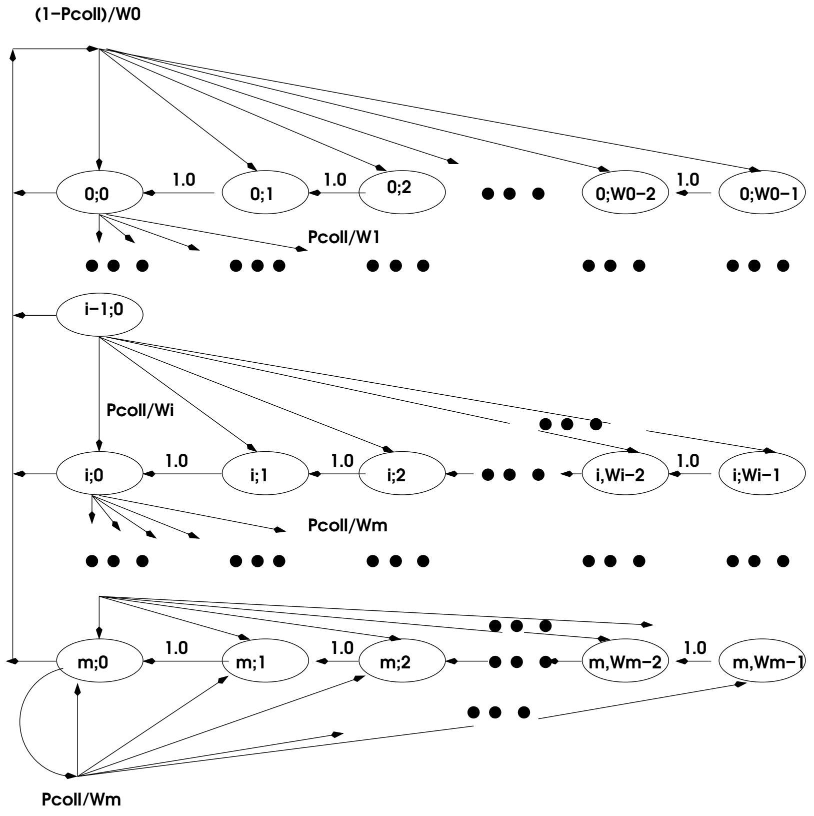


Figure 2.3: Bianchi's Model for IEEE 802.11

station. The pair *(backoff stage, backoff counter value)*, referred as (b, c), describes a given state of a station where c can take any value between 0 and W_b. Backoff stage b varies from 0 to a maximum backoff stage MBS. Bianchi assumes that a station will transit from state (b, c) to state $(b, c-1)$ with probability 1.0. This assumption implies that the backoff counter value is decremented at each time slot under all conditions. This violates the IEEE standard 802.11[25]. IEEE 802.11 specifies that the back off counter value is decremented only if the medium is sensed idle. Bianchi accommodates this violation by considering an average slot time, rather than the constant slot time σ set by the physical layer.

If a station reaches state $(b, 0)$ (i.e., the backoff counter value becomes zero), the station will send out a packet. If the packet collides (with probability P_{coll}) then the station will enter with probability $\frac{P_{coll}}{W_{b+1}}$ some state $(b+1, c)$ with a higher backoff stage. If the packet does not collide, the station will return to some state $(0, c)$ with backoff stage 0 (recall that in such a state, c can be any value between 0 and W_0) with probability $\frac{1.0 - P_{coll}}{W_0}$.

From the model it can be derived that the probability that a station transmits a packet in any time slot is

$$\tau = \textstyle\sum_{b=0}^{MBS} \pi(b; 0)$$

The probability that a sent packet collides is

$$P_{coll} = 1 - (1 - \tau)^{n-1}$$

If n is the total number of stations, the expression above reflects the fact that a packet collides when at least one other station is also sending. We denote $\pi(b, 0)$ as the probability that the station stays in state (b,0).

Instead of the system time slot σ, Bianchi [8] uses the average time slot T_{slot}, which he defines as the average time duration for a station to transit from one state to another. T_{slot} can be derived as follows. If there is no medium activity on the channel, the time slot is the system time slot σ. If the medium is busy, the slot time is either the time to complete a successful transmission or the time to perform a failed transmission. The average time slot is

$$T_{slot} = (1 - P_{tr}) * \sigma + P_{tr} P_s T_s + P_{tr}(1 - P_s) T_f$$

where

$$P_{tr} = 1 - (1 - \tau)^n \quad P_s = \tfrac{n\tau(1-\tau)^{n-1}}{1-(1-\tau)^n}$$

$$T_s = RTS + DIFS + CTS + SIFS + T[P] + SIFS + ACK + DIFS$$

$$T_f = RTS + DIFS$$

P_{tr} is the probability that there is at least one transmission in the considered time slot, and P_s is the probability that a transmission occurring in the considered slot is successful given that there is a transmission in that slot. T_s is the time needed to complete a successful transmission. T_s includes the time to send RTS, CTS, ACK and data packets plus the defer and interframe time. T_c is the time for a failed transmission. σ is the system time slot duration and $T[P]$ is the time needed to transmit the payload. The propagation delay is ignored.

The saturated throughput is derived by Bianchi as,

$$S = \frac{P_s P_{tr} E[P]}{T_{slot}}$$

where $E[P]$ is the average payload length.

2.6 Mathematical Model for Unsaturated Traffic

Bianchi's model structure is excellent and provides a good estimate of the saturated throughput. However, we pointed out that Bianchi's model does not accurately model IEEE 802.11. In this section, we will first explain the difference between our model and Bianchi's, and then we will derive the expressions for the probability of transmission τ, the probability of collision P_{coll}, the average access delay D_{access}, and the average energy consumption E. Other characteristics of the station are derived too. Finally, we use *NS2* simulations to check and validate our model and analysis.

As in Bianchi's paper [8], we assume that all stations always have at least one packet to send. Let us denote the state space as $\mathcal{R}$,

$$\mathcal{R} = \{(\mathrm{b,c}) \colon MBS \geq b \geq 0, c \geq 0\}$$

where b is the current *backoff stage* and c is the current *backoff counter value.* c takes any value from 0 to W_b where $W_b = W_0 * 2^b$. MBS is the maximum backoff stage of the IEEE 802.11 protocol. The behavior of a station can be described as a chain of states on the time axis. When the station is in some state, different actions such as receiving a packet, sending a packet, or decrementing the counter will lead the station to different states with some specific probability. The next state depends only on the current state, not on any previous state (the state chain is a Markovian chain).

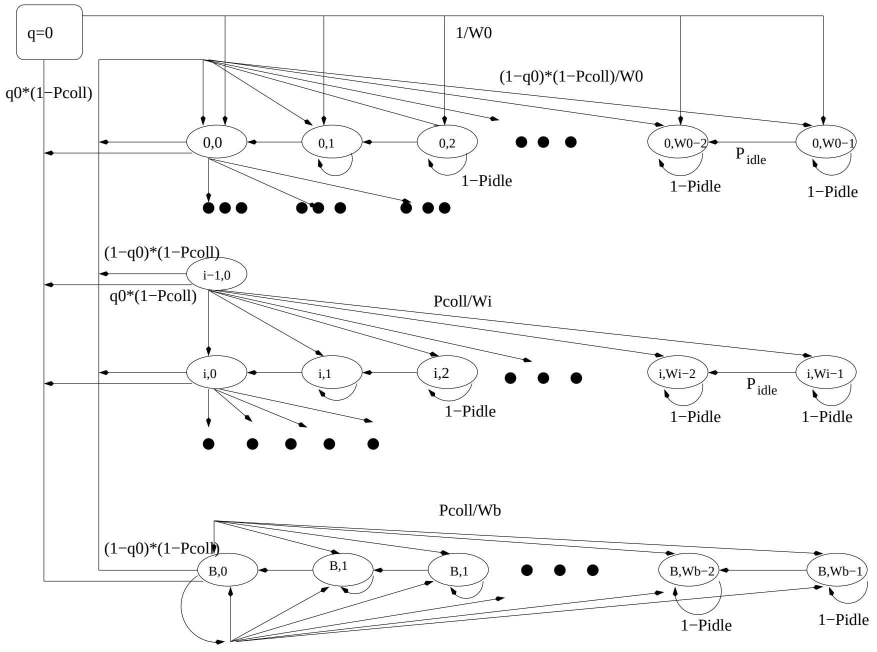


Figure 2.4: Overview of the Station Model

One difference between Bianchi's model and ours is the transition probability from state (b, c) to state $(b, c - 1)$, which is also addressed in [41]. Bianchi's model assumes that the probability of the transition from state (b, c) to state (b,c-1) is 1.0. A second difference is that Bianchi's Markov chain assumes a constant time T_{slot} duration from one state to another where T_{slot} is defined as the average slot time. This does not reflect accurately the behavior of an IEEE 802.11 station. Therefore, we consider for our model that the time duration for the station to transit from one state to another is different for different transitions. This time duration is randomly distributed and makes the chain a *semi-Markov process*. With the stationary distribution of this semi-Markov process, we can derive the frequency probability and time proportion of each state for further analysis.

Our model also takes into account the transition time, transition probability, and transition energy in order to derive expressions for the probability of collision P_{coll}, the average throughput, the average access delay D_{access}, and energy consumption E. Figure 2.4 outlines our model.

2.6.1 Notations and Terminology

b	backoff stage of the state
c	backoff count of the state
MBS	maximum backoff stage of the model
CW_0	maximum contention window size at stage 0
MaxState	total number of possible states that nodes go through
CW_i	maximum contention window size at stage i
τ	probability that a node sends a packet in any time slot
P_{coll}	probability that a sent packet collides
P_{succ}	probability that there is a successful packet in the medium in any time slot
P_{fail}	probability that there is a corrupted packet in the medium in any time slot
P_{idle}	probability that the medium is idle in any time slot
T_s	time to transmit a successful packet

T_f time to transmit a collided packet

Et_s energy used to transmit a successful packet

Et_f energy used to transmit a corrupted packet

Er_s energy used to receive a successful packet

$Eovr_s$ energy used to overhear a successful packet

$Eovr_f$ energy used to overhear a corrupted packet

E_{idle} energy consumption when a node is idle for aSlotTime

π_i frequency probability of a node being in state i

μ_i mean time of duration a node staying in state i

P_i time proportion of a node staying state i

P_{ij} probability of a node jumping from state i to state j

t_{ij} transition duration of a node jumping from state i to state j

ϵ_{ij} transition energy consumption for a node jumping from state i to state j

$\pi(q;b;c)$ frequency probability of a node being in state $(q;b;c)$

$P(q;b;c)$ time proportion of a node staying state $(q;b;c)$

2.6.2 The System Model

In order to analyze the protocol under *unsaturated* traffic, we make the same assumption as is done in [38]. We assume that stations are statistically identical, each station has idle periods that are exponentially distributed, and packet length is constant. For a station under unsaturated traffic, the transition from state $(b, 0)$ to state $(0, c)$, which means the station reaches the next transmission cycle after successfully sending out a packet, is not guaranteed, as it is in Bianchi's model. This is only true when the station has at least one buffered packet. We add an additional state ($q = 0$) in our model to handle the situation in which the station has no buffered packet. Let λ be the offered load of each station, and q_0 the probability that a station has no buffered packet. In Figure 2.4, all states and state transitions except state ($q = 0$) are based on the condition that there is at least one packet to be sent. When a station gets to state $(b, 0)$ and sends a packet, it will reach state $(b+1, c)$ if the packet collides and needs to be retransmitted. If the packet is successfully sent, it will reach state ($q = 0$) or state $(0, c)$ depending on whether or not there is any buffered packet. When a station is in state ($q = 0$), which means it currently has no packet to send, it will stay there until a packet arrives, then it will reach one of the $(0, c)$ states and start a transmission cycle. The average packet arrival interval is $\frac{1}{\lambda}$, so the transition from state ($q = 0$) to state $(0, c)$ has transition probability $\frac{1}{W_0}$ and transition duration $\frac{1}{\lambda}$. Please refer to the appendix for a detailed description of states and state transitions. The state transition diagram shown in Figure 2.4 is governed by the following transition probabilities and durations.

1. The backoff count is decremented and the station makes a transition from state (b, c) to state $(b, c-1)$ when the medium is idle

   $P\{(b, c-1)|(b, c)\} = P_{idle}$

   $t\{(b, c-1)|(b, c)\} = \sigma$

   $\epsilon\{(b, c-1)|(b, c)\} = E_{idle}$

2. The backoff counter suspends and the station stays in state (b, c) when the medium is busy

   $P\{(b, c)|(b, c)\} = 1 - P_{idle}$

$t\{(b,c)|(b,c)\} = \frac{P_{succ}*T_s+P_{fail}*T_f}{1-P_{idle}}$

$\epsilon\{(b,c)|(b,c)\} = \frac{P_{succ}*Eovr_s+P_{fail}*Eovr_f}{1-P_{idle}}$

3. The station sends a packet and the packet collides, the station reaches state $(b+1,c)$

   $P\{(b+1,c)|(b,0)\} = \frac{P_{coll}}{CW_{b+1}} \quad 0 \leq b \leq (B-1)$

   $P\{(B,c)|(B,0)\} = \frac{P_{coll}}{CW_B}$

   $t\{(b+1,c)|(b,0)\} = t\{(B,c)|(B,0)\} = T_f$

   $\epsilon\{(b+1,c)|(b,0)\} = \epsilon\{(B,c)|(B,0)\} = Et_f$

4. The station sends a packet successfully and reaches state $(0,c)$ since it has more packets to send.

   $P\{(0,c)|(b,0)\} = \frac{(1-q_0)*(1-P_{coll})}{CW_0} \quad 0 \leq c \leq CW_0$

   $t\{(0,c)|(b,0)\} = T_s \quad 0 \leq b \leq B$

   $\epsilon\{(0,c)|(b,0)\} = Et_s \quad 0 \leq b \leq B$

5. The station sends a packet successfully and reaches state $(q=0)$ since it has no more packets to send.

   $P\{(q=0)|(b,0)\} = q_0 * (1-P_{coll}) \quad 0 \leq b \leq B$

   $t\{(q=0)|(b,0)\} = T_s$

   $\epsilon\{(q=0)|(b,0)\} = Et_s$

6. The station has an arrival packet and leaves state $(q=0)$

   $P\{(0,c)|(q=0)\} = \frac{1}{CW_0} \quad 0 \leq c \leq CW_0$

$t\{(0,c)|(q=0)\} = \frac{1}{\lambda}$

$\epsilon\{(0,c)|(q=0)\} = E_{idle}$

Let us denote $P_{(b,c)}$ as the probability that the station reaches state (b,c). From the model we can compute that under the condition that there is at least one packet to send, the station has probability τ to send a packet in any time slot.

$$\tau = \sum_{b=0}^{B} P_{(b,0)}$$

Then the probability that a station will send a packet in any time slot is,

$$p = (1-q_0) * \tau$$

In a system that consists of n stations, the probability that a sent packet collides is,

$$P_{coll} = 1 - (1-p)^{(n-1)}$$

For the whole system, the probabilities of a successful packet, failed packet, and no packet in any time slot are P_{succ}, P_{fail}, and P_{idle}, respectively,

$$P_{succ} = n * p * (1 - P_{coll})$$

$$P_{idle} = (1-p)^n$$

$$P_{fail} = 1.0 - P_{succ} - P_{idle}$$

For probability q_0, let us denote the average access delay, the time between when a packet reaches the MAC layer and when it is successfully sent, as D_{access}. D_{access} is also the packet service time if we treat each station as an $M/M/1/N$ queue system, in which N is the maximum queue length of the queue. For an $M/M/1/N$ queue, the probability that there is no packet in the queue is:

$$\rho = \lambda * D_{access}$$

$$q_0 = \frac{1-\rho}{1-\rho^{N+1}}$$

2.6.3 Energy Estimation

In [18], Feeney and Nilsson gave a linear model for Lucent $IEEE802.11$ $2Mbps$ PC cards. According to this model, the energy consumption in $IEEE802.11$ networks can be associated with the size of sent packets.

$$E = a * Size + b$$

a is the energy consumption per byte, and b is the overhead for sending a packet. a and b are different for sending, receiving, and overhearing conditions. They also depend on whether or not the station is within the range of the data source and data destination. The idle state energy consumption does not depend on the packet size. The paper gives an estimation of idle state consumption rate e. We borrow this model for calculating per-packet energy consumption. In addition to the transition duration for each state, our model gives the transition energy consumption for each state. For example, in the source station model, the transition from state (b, c) to state $(b, c-1)$ will consume $e * \sigma(w.sec)$. The transition from state $(b, 0)$ to state $(0, c)$ will consume $a * L + b(w.sec)$, and the transition from state $(b, 0)$ to state $(b+1, c)$ will consume $a^{'} * l + b^{'}(w.sec)$, where l is the number of bytes sent during a failed transmission. The average consumption in each state E_i can be calculated in the same way as the average state duration, μ_i.

$$E_i = \sum_j P_{ij}\epsilon_{ij}$$

The energy consumption of a station during queue empty state is not very straightforward. For source stations, when they are in the empty queue state there are two cases affecting energy consumption. In the first case, other stations have packets and there is transmission activity on the medium; the station's energy consumption includes the overhearing of packets on the medium. In the second case, all stations are empty and there is no transmission activity on the medium; the station's energy consumption only includes idle energy consumption. Let e_i denote the average energy consumption rate of any state i, and π_i denote the time proportion of each state i, then the average energy consumption rate of the station, e, is,

$$e_i = \frac{E_i}{\mu_i} \qquad e = \sum_i P_i * e_i$$

2.6.4 Solutions and Results

With the diagram and conditions mentioned above we can obtain the stationary probability distribution of the model. In the description of the models and the calculation of the parameters, we use unknowns such as π_is and τ to recursively represent themselves. In order to solve those unknowns and get the stationary distribution π, system equations are established for all nodes.

$$\pi_i = \sum_{j=1}^{MaxStates} \pi_j P_{ji} \tag{2.1}$$

$$\sum_i \pi_i = 1.0 \tag{2.2}$$

$$\tau = \sum_{b=1}^{MBS} \pi(b;0) \tag{2.3}$$

for all $i \in [0, MaxStates]$, in which $MaxStates$ is the total number of states.

The characteristics of the system matrix include: 0 or less than 1 values on the diagonal, and the sum of absolute values on the same row is less than 1. According to the Gerschgorin circle theorem, all the eigenvalues of the system are less than 1. Thus if Jacobi iteration is used to solve the system, the calculation should converge.

There is still an unknown, D_{access}. Suppose that a packet is successfully sent on the first try and the time it takes is TS. Otherwise, if it fails on the first try, which takes time TF, then it will have to wait for the station to reach the next sending state $(b, 0)$ before it is sent again. In order to derive D_{access} we need to express the average time between two sending states. Let us denote D as the average time between two sending states. In practice, D is also the time that a station takes to complete a backoff procedure after a failed transmission. Consider that each transmission starts with a backoff procedure. We have,

$$TF = T_f + D$$

$$TS = T_s + D$$

Let $\mathcal{R}_\tau$ be the set of sending states, i.e.,

$$\mathcal{R}_\tau = \{(b; c) : c = 0\}$$

The probability that a station is in $\mathcal{R}_\tau$ is τ. Suppose $\tau_i \in \mathcal{R}$ and $\tau_j \in \mathcal{R}$ are two consecutive states (in $\mathcal{R}_\tau$) that the station goes through. Between two consecutive visits to $\mathcal{R}_\tau$ (τ_i and τ_j), the expected number of visits to any state $k \notin \mathcal{R}_\tau$ is $\frac{P_k}{\tau}$. Assume that the average time a station will stay in state k is μ_k. We can derive the time D between two consecutive sending states as

$$D = \textstyle\sum_{k;k\in\mathcal{R},k\notin\mathcal{R}_\tau} \frac{P_k \mu_k}{\tau}$$

D is also the time between two consecutive transmissions of any packet. The probability that a packet would be successfully sent on the first try is $(1 - P_{coll})$, on the second try is

$P_{coll} * (1 - P_{coll})$, and so on. The probability that a packet would be sent successfully on the i^{th} try is $P_{coll}^{i-1} * (1 - P_{coll})$. If a packet is sent successfully on the first try, it takes TS. If on the second try, it takes $(TF + D + T_s)$, which is $(TF + TS)$, and so on. If a packet is sent successfully on the i^{th} try, it takes $((i-1) * TF + TS)$. The average access delay can be derived as

$$\sum_{n=1}^{N} P_{coll}^{n-1}(1 - P_{coll})(TS + (n-1) * TF).$$

where N is the number of retransmission times minus one. When N goes to infinity, the access delay will be

$$D_{access} = TS + \frac{P_{coll} * TF}{1 - P_{coll}}.$$

D_{access} can be expressed through P_{coll}, and the stationary distribution can thus be obtained.

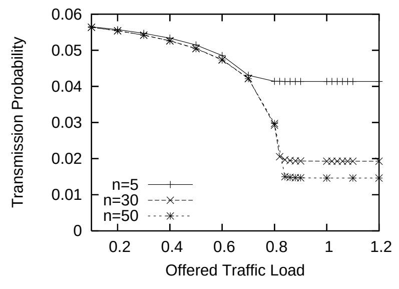
The average system throughput should be the sum of the throughputs of all stations. For a single station, the throughput should be the throughput it has during the time it has packets to send averaged over the total time.

$$T_{slot} = \sum_i P_i \mu_i$$

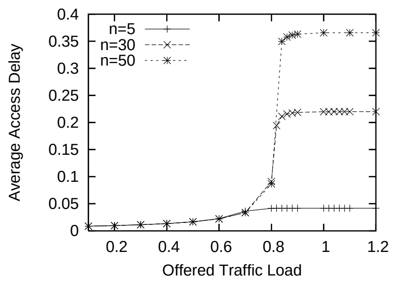
$$Thr = 8 * L * \{\frac{(1 - P_{(q=0)}) * \tau (1 - P_{coll}) * n}{T_{slot}}\}$$

where L is the payload length.

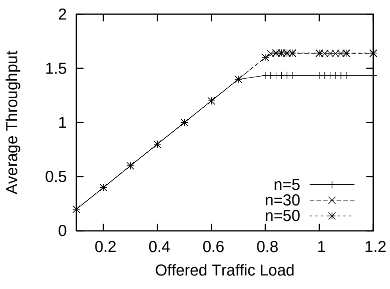
From a single station's point of view, there are four kinds of time slots. A slot in which there is a successful packet, a slot in which there is a collided packet, a slot in which the backoff count is decremented, and a slot in which there is no packet to send. T_{slot} can be seen as the average slot time. Figures 2.5(a)-(c) show some analytic results from the model. The x-axis in each figure is the normalized offered traffic load. Figure 2.5a) show that the



(a)Transmission Prob.

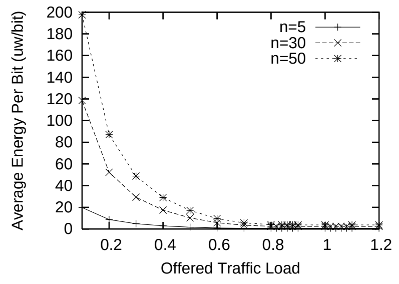


(b)Average Access Delay

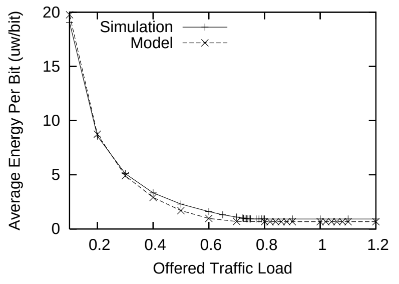


(c)Average Throughput

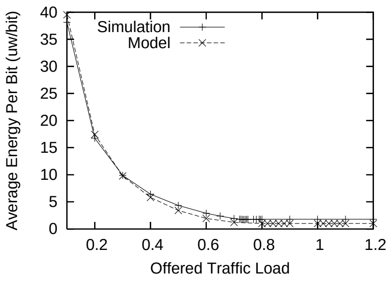
Figure 2.5: Station Model: Analytic Results



(a)epb.



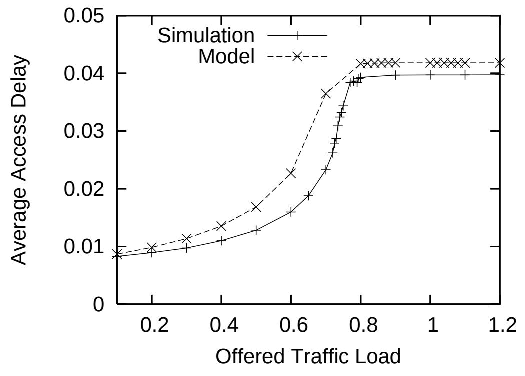
(b)epb. $n = 5$



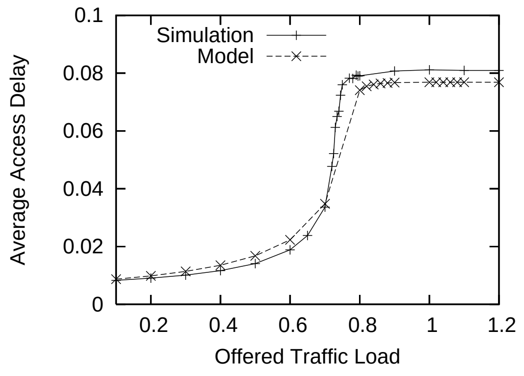
(c)epb. $n = 10$

Figure 2.6: Station Model: Energy Consumption Analysis

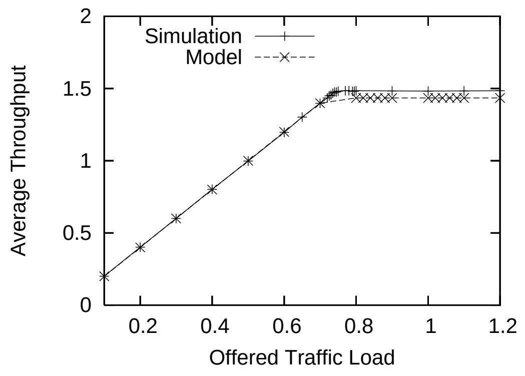
probability that a station sends a packet in any time slot when it has packets to send, τ, is not independent of the offered traffic load. As the offered load increases, τ decreases. For network size less than 50, the breaking point is around 0.65. After the overall traffic load exceeds 0.65, the average access delay increases steeply, and the throughput becomes saturated. After the load exceeds 0.8, the average access delay and τ do not change much. Figures 2.7(a)-(d) plot both *NS2* simulation results and analytical results for network size 5 and 10. The simulation results fit quite well with the analytic results. When the network size increases, the *NS2* simulation results fit well with the analytic results when the offered load



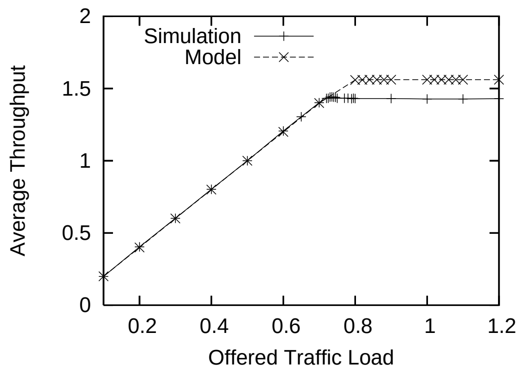
(a) Average Access Delay $n = 5$



(b) Average Access Delay $n = 10$



(c)Average Throughput $n = 5$



(d)Average Throughput $n = 10$

Figure 2.7: Station Model: Simulation Results

is less than 0.65 or greater than 0.80. However, they show a relatively large deviation from the analytic results when the load is between 0.65 and 0.8. This is because when the offered load is greater than 0.65, the system is close to the saturation condition. The average access delay of each station is close to or greater than the packet arrival rate, and the estimation of ρ and q_0 may not be accurate any more.

Figure 2.6.a) shows the analytic results for energy consumption of all source station networks under different traffic loads. Figures 2.6.b) and 2.6.c) show the simulation results when n is 5 or 10. *epb* in the figures means energy per bit. *epb* increases as the network size increases and decreases as the traffic load increases.

Chapter 3

Extended Mathematical Model for Mutli-hop Networks

3.1 Overview

In the IEEE 802.11 protocol model for single-hop wireless networks, every station is assumed equivalent. In most work described above, every station is assumed to be a data source that sends out saturated traffic. In multi-hop wireless network, each station in the network is not necessarily a data source. It may act like a data source for a period of time when it has original data to send, while at other times it may act like a relay station that simply helps other stations to forward data. In this case, it is inappropriate to model every station as a saturated data source at all times.

We propose a general scenario for the modeling of mutli-hop wireless networks. We make the following assumptions. From a single station's point of view, statistically, at any time in the network, among all stations that are within its transmission range, a certain number of stations act like data sources that inject saturated data traffic. These are called source stations. Also, a certain number of stations act like data relays that forward a certain amount of traffic within the network. These are called relay stations. Source stations will not forward data and relay stations will not generate data.

Key to the modeling of multi-hop networks is a treatment of the upper layer routing protocols that can affect network performance through the way they forward packets and the impact that has on the traffic load. The model proposed takes into account the impact of the upper layer routing protocol by introducing a packet acceptance factor with which each relay station accepts packets from the wireless medium before forwarding them.

3.2 Multi-hop Wireless Network Scenario

For the source stations, the two dimensional model described in Chapter 2 is sufficient to describe the behavior of the IEEE 802.11 MAC protocol. For relay stations, the previous model fails to correctly describe the behavior of the protocols after one packet is successfully sent to the network. In the previous model, under a saturated traffic situation, the protocol enters another cycle of packet transmission once the previous packet is sent. Under an unsaturated traffic situation, each station has a defined traffic load. The protocol enters another cycle of packet transmission as long as there is a packet in the queue. A relay station listens to the medium, get packets from it, and forward the packets it receives. Whether or not the MAC protocol enters another transmission cycle depends on whether or not there are packets in the queue at the link layer. The number of packets a relay station receives and *accepts to forward* depends on the upper layer routing protocol. For example, in the flooding protocol, a station broadcasts all its own packets and forwards packets from/to all its neighbors. A station will *accept* 100% of the traffic within its range. In the diffusion routing protocol [26], as well as most routing protocols, a station will forward most of its traffic to the neighbor station on its estimated shortest path to the destination, and very little traffic to other neighbor stations. So the station on the shortest path will accept the packet with a probability that is much higher than those of the other stations. Thus, for a multi-hop data transaction, the upper layer protocol determines the traffic load on the relay stations that are involved. Due to the different traffic load that falls on the each station, the status of each station's link layer queue is different. The assumption that at any time there is at least one packet in the queue is not appropriate.

Since our work focuses on the analysis of the MAC protocol, we need to find a way to isolate or take into account the upper layer protocol. That is what we propose in our three dimensional model for the relay stations in the multi-hop wireless network. We introduce a probability P_{in}. P_{in} is the probability that a relay station will accept to forward (receive and later forward) a packet under the condition that there is a successful packet from other stations in the medium. P_{in} could be different for different relay stations. The different

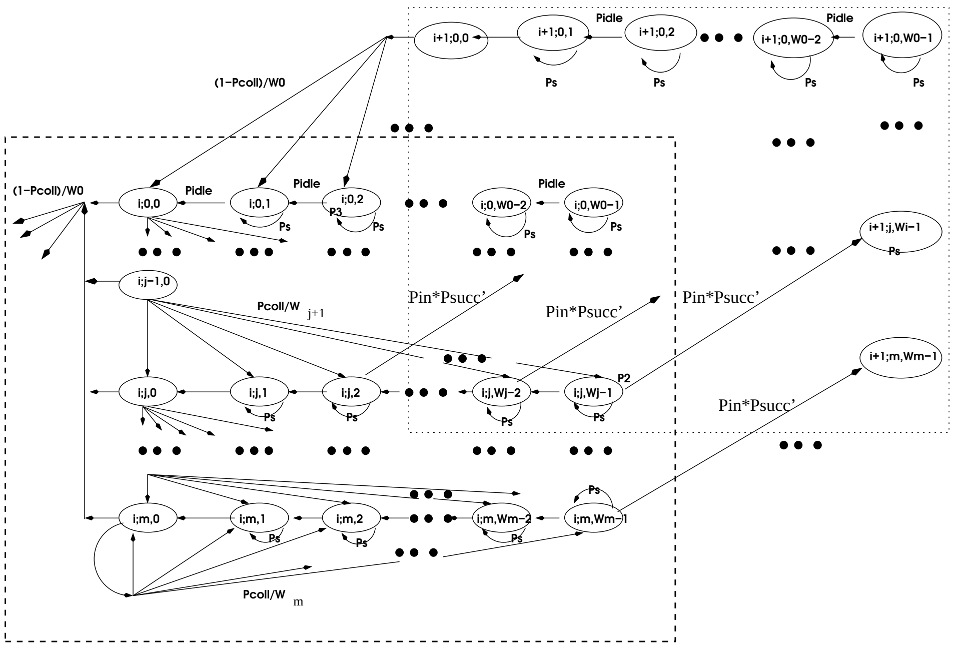


Figure 3.1: Overview of the Relay Station Model

routing protocols distribute the traffic load among the stations in different ways. P_{in} reflects that distribution.

For the work that has been done on the performance analysis of the IEEE 802.11 protocol, saturated throughput is of the most interest. For a network in which all stations behave as saturated data sources, queue length of stations is not a concern due to the nature of the saturated traffic. For multi-hop wireless networks that include both source stations and relay stations, the average queue length, average delay, and packet drop rate at relay stations are of great interest. In order to mathematically analyze those performance features, we add a dimension to our original model in consideration of queue length.

3.3 Model for the Relay Station

Figure 3.1 outlines the model for the relay station. Let us denote state space $\mathcal{R}$,

$$\mathcal{R} = \{(q, b, c) : Q \geq q \geq 0, B \geq b \geq 0, c \geq 0\}$$

where q is the current queue length, Q is the maximum queue length, b is the current backoff stage, and c is the current backoff counter value.

The foreground plane in Figure 3.1 represents the two dimensional Markov model with a queue length $q = i$. The model is extended in depth toward the background with increasing queue length q. The background plane is the two dimensional Markov model with queue length $q = i + 1$. Within the (b, c) plane with a fixed value q, the model is similar to the two dimensional model. A station in a state on the (b, c) plane with queue value i (i.e., in state (i, b, c)) will transit to a state on the (b, c) plane with queue length $i+1$ if it accepts a packet. A station in a state on the (b, c) plane with queue value $i + 1$ (i.e., in state $(i + 1, b, c)$) will transit to some state $(i, 0, c)$ if it completes a successful transmission. If a station stays in some state $(0, b, c)$ that has a queue length of zero, then the station has no packet to send. Therefore, we assume that states $(0, b, c)$ must have backoff stage $b = 0$ and that a station in one of these states will not transit to any other state unless the station gets a successful packet from the medium. This feature of the relay stations is different from that of source stations since the source stations have new packets from the application layer.

As an example of how these three parameters describe the behavior, let us suppose that at any moment the state of the station is (q, b, c). As time progresses, the backoff count may go to zero, after which one packet will be sent. If the packet is sent successfully, the station transitions to state $(q - 1, 0, c)$, otherwise the station transitions to state $(q, b + 1, c)$. If a station accepts a successful packet, it transitions to state $(q + 1, b, c)$.

One thing worth mentioning here is the queue length consideration in the model. By queue length, we mean the total number of packets that exist in the link layer queue and MAC layer. This declaration makes the above model more reasonable. After the MAC layer fetches a packet from the link layer queue, the packet will be buffered in the MAC layer until it is either sent or dropped. It is possible that there is still one packet in the MAC layer when the link layer queue is empty. From the above model, we notice that the protocol will not do anything once it enters a state in which the queue length is zero until it gets one packet from the medium. This is only true when we include the packet in the MAC layer in the calculation of queue length.

3.3.1 Notations and Terminology

The descriptions in this chapter follow the same notations and terminology defined in Chapter 2, as well as the additional definitions for the extended model.

q queue length of the state

MQL maximum queue length of the MAC layer

nd total number of relay nodes

n total number of source nodes

P_{in} probability that a node accepts a successful packet

P'_{succ} probability that there is a successful packet in the medium at some time slot, and the packet is not from the current node

P^r_{coll} probability that a packet sent by a relay node collides

q^r_0 probability that a relay node is empty

$\overline{Q}$ average queue size of a relay node

$\pi(q;b;c)$ frequency probability of a node being in state $(q;b;c)$

$P(q;b;c)$ time proportion of a node staying state $(q;b;c)$

3.3.2 Parameters in the Models

When both models are used together to analyze the network described in section 3.2, parameters such as transition probability and transition duration should be justified with consideration of the effects that both relay stations and source stations have on each other. The changes are made through the system parameters P_{succ},P_{fail} and P_{idle}, as well as model parameters such as P_{coll}. Let us suppose that in the whole system, there are nd relay stations in addition to the n source stations, and that each relay station has accept probability $P_{in}(i)$. The state transition diagram shown in figure 3.1 is governed by the following transition probabilities and durations that are different from those previously mentioned in the source station model.

1. The station gets one successful packet from the medium and puts it in the queue

   $P\{(q+1,b,c)|(q,b,c)\} = P^{'}_{succ} * P_{in}(i)$

   $t\{(q+1,b,c)|(q,b,c)\} = T_s$

2. the station sends out a packet and the packet collides

   $P\{(q,b+1,c)|(q,b,0)\} = \frac{P_{coll}(i)}{CW_{b+1}}$

   $P\{(q,B,c)|(q,B,0)\} = \frac{P_{coll}(i)}{CW_B}$

   $t\{(q,b+1,c)|(q,b,0)\} = T_f$

   $t\{(q,B,c)|(q,B,0)\} = T_f$

3. the station sends out a packet successfully

   $P\{(q-1,0,c)|(q,b,0)\} = \frac{(1-P_{coll}(i))}{CW_0}$

   $t\{(q,0,c)|(q,b,0)\} = T_s$

4. the station has no packet, and waits for a packet to come in

   $P\{(1,0,c)|(0,0,c)\} = P^{'}_{succ} * P_{in}(i)$

   $t\{(q=0)|(b,0)\} = T_{empty}(i)$

A relay station will accept the successful packet from *other* stations with probability $P_{in}(i)$. $P^{'}_{succ}$ is the probability that there is a successful packet in the medium and that the packet is not from the current station. T_{empty} is the time that a relay station has to wait to get a packet from the medium. For a relay station i, the probability of sending a packet on any slot is,

$$\tau(i) = \sum_{b=0}^{B} \sum_{q=1}^{Q} P_{(q,b,0)}$$

The probability that a sent packet collides is,

$$P^r_{coll}(i) = 1.0 - (1-p)^{(n)} \prod_{j=0, j\neq i}^{nd} (1-\tau(i))$$

And,

$$P'_{succ} = n * p * (1.0 - P_{coll}) + \sum_{j=0, j\neq i}^{nd} \tau(i) * (1.0 - P^r_{coll}(i))$$

For the whole system,

$$P_{succ} = n * p * (1.0 - P_{coll}) + \sum_{j=0}^{nd} \tau(i) * (1.0 - P^r_{coll}(i))$$

$$P_{idle} = (1-p)^{(n)} \prod_{j=0}^{nd} (1-\tau(i))$$

For the source station, P_{coll} should be changed to,

$$P_{coll} = 1.0 - (1-p)^{(n-1)} \prod_{j=0}^{nd} (1-\tau(i))$$

T_{empty} is critical in the analysis under unsaturated traffic conditions. As the source station has traffic source rate λ, the traffic for relay stations depends on the traffic in the medium. If there is at least one station in the system with a packet to send, statistically, the time an empty station needs to wait to take in a packet is the same as the time it waits to take in a packet at any non-empty state. If all stations are empty, the whole system will have to wait until at least one source station has an arrival packet before there will be a new packet on the medium. The waiting time for this case should be the average source station arrival interval. Let's denote $\pi_{(q,b,c)}$ as the time proportion that a relay station stays in state (q, b, c), and $q^r_0(i)$ as the probability that a relay station is empty.

$$\pi_{(0,0,c)} = \frac{\sum_{i=0}^{CW_0} P_{(0,0,c)} * \mu_{(0,0,c)}}{\sum_{i\in\mathcal{R}} P_i * \mu_i}$$

$$q^r_0(i) = \sum_{c=0}^{CW_0} \pi_{(0,0,c)}$$

Then, the probability that all stations are empty when a relay station is in an empty state, $\pi_0(i)$ is,

$$\pi_0(i) = q_0^n * \prod_{j=0;j\neq i}^{nd} q_0^r(i)$$

and the waiting time in this case is,

$$t_0 = \frac{1}{n*\lambda}$$

When not all stations are empty, the waiting time can be calculated as,

$$t = P_{succ} * T_s + P_{fail} * T_f + P_{idle} * \sigma$$

T_{empty} can then be calculated as,

$$T_{empty}(i) = \pi_0(i) * t_0 + (1 - \pi_0(i)) * t$$

T_{empty} is important in calculating relay throughput and energy consumption under unsaturated traffic conditions. The throughput of a relay station is,

$$Thr = 8 * L * \frac{(1-\sum_{i=0}^{CW_0} P_{(0,0,c)})*\tau(1-P_{coll})}{T_{slot}}$$

The average access delay can be calculated as in the source station model, and the average queue length is,

$$\overline{Q} = \sum_{q,b,c} \pi_{(q,b,c)} * (q-1)$$

$(q-1)$ is used here for consistency with the note about the queue length mentioned above. The queue length in our model includes the extra space in the MAC layer. The $\overline{Q}$ here only refers to the link layer queue length.

3.4 Solutions and Results

System equations for relay nodes in the extended model are,

$$\pi_i = \sum_{j=1}^{MaxStates} \pi_j P_{ji} \tag{3.1}$$

$$\sum_i \pi_i = 1.0 \tag{3.2}$$

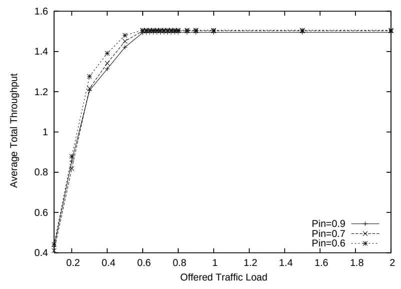
$$\tau = \sum_{q=1}^{MQL} \sum_{b=1}^{MBS} \pi(q; b; 0) \tag{3.3}$$

for all $i \in [0, MaxStates]$, in which $MaxStates$ is the number of states of the relay node.

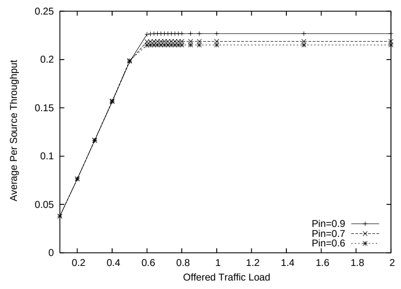
According to the assumption in our models, for all our simulations and analyses we use packet size 1500 bytes, bandwidth $2MBps$, initial window size 32, maximum backoff stage 3, and maximum queue limit 30.

For multi-hop network analysis, we fix the source stations at 5 and the number of relay station nodes at 2 or 5. We use varied values of P_{in} among relay stations and traffic load for source stations. One relay station among all will have a higher P_{in} value and the rest will have the same, smaller P_{in}. In the following figures, a result labeled "$P_{in} = a$" is from the test in which the highest P_{in} value is a. Results labeled "$P_{in} = aH$" and "$P_{in} = aL$" are for relay stations with high and low P_{in} values respectively, for tests where a is the highest P_{in} value. Let a denote this P_{in} for the rest of our description.

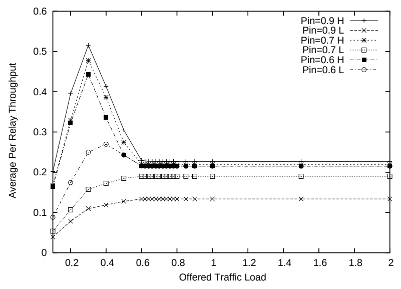
Figures 3.2 to 3.4 show the analytic results for a system that consists of 5 source stations and 2 relay stations. Figures 3.5 to 3.7 show the analytic results for a system that consists of 5 source stations and 5 relay stations. The high P_{in} values vary from 0.6 to 0.95. The low P_{in} values are from 0.05 to 0.25 for the first case and from 0.01 to 0.08 for the second.



(a)throughput

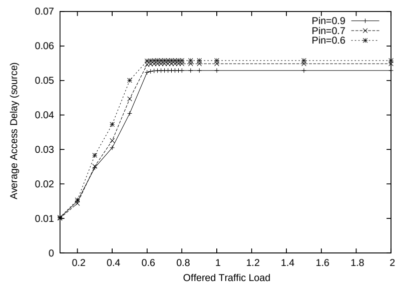


(b)source throughput

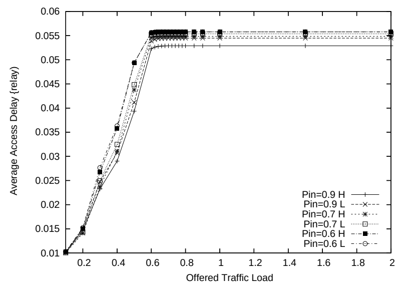


(c)relay throughput

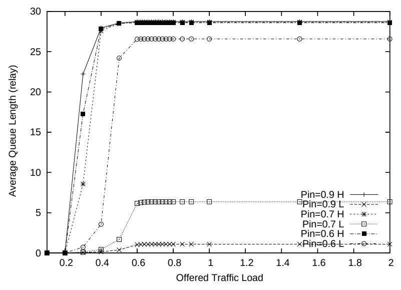
Figure 3.2: Relay Station Model: Throughput Analysis $n = 5, nd = 2$



(a)source access delay



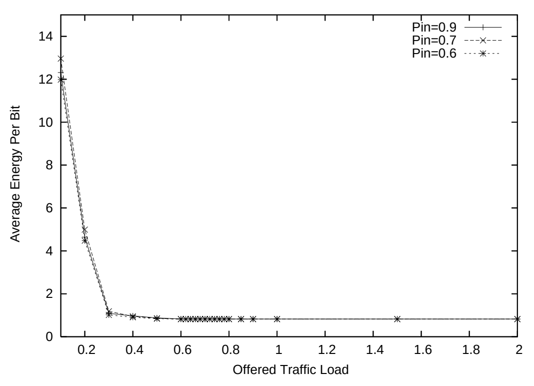
(b)relay access delay



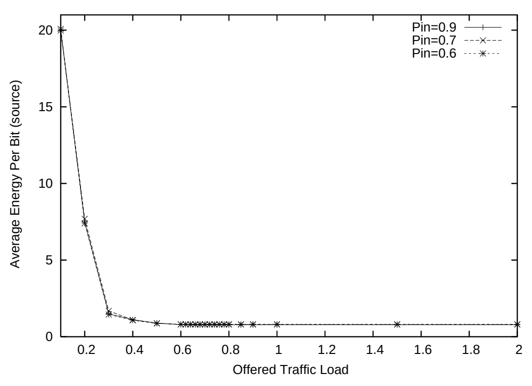
(c)relay average queue

Figure 3.3: Relay Station Model: Delay & Queue Analysis, $n = 5, nd = 2$

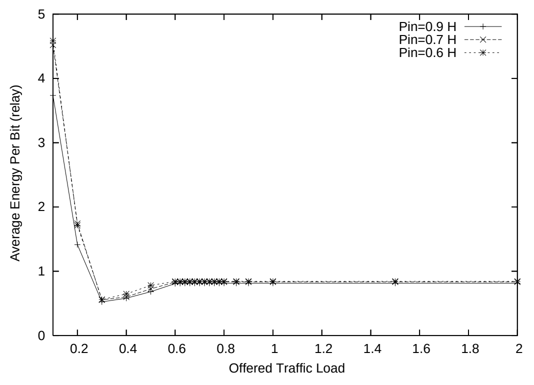
From the results, we notice that the overall throughput is slightly higher when a is lower, which means the traffic load is more evenly distributed among all the relay stations. The throughput from all source stations is actually lower when a is lower. If the purpose of the communication is to send source stations' data to some destinations, lower throughput from source stations means lower efficiency. The bandwidth is consumed by redundant data transmission. The access delay of both source stations and relay stations are lower when a is higher. This is because more traffic is going through some main path, less stations are sending packets at the same time, and there are fewer collisions. The average queue length



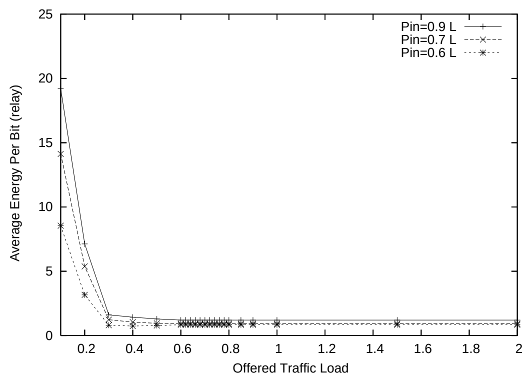
(a) total epb



(b) source epb



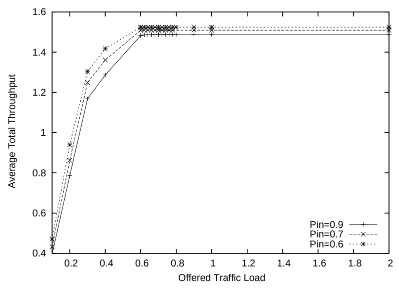
(c)relay epb, high P_{in}



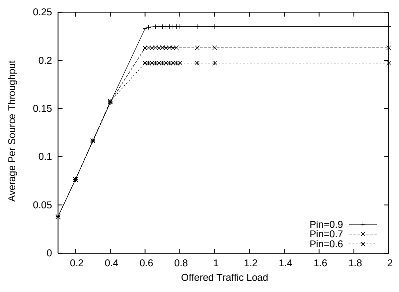
(d)relay epb, low P_{in}

Figure 3.4: Relay Station Model: Energy Consumption Analysis, $n = 5, nd = 2$

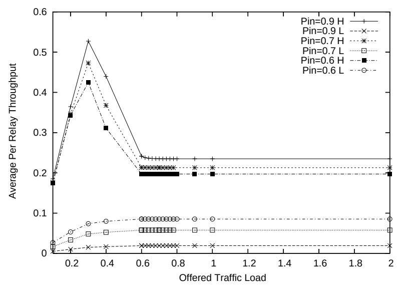
increases rapidly when a increases, which presents a problem for the upper layer protocol. The average waiting time before transmitting a packet is the product of average queue length and average access delay. A change in a causes an opposite change in queue length and access delay. An optimal choice of a can help to achieve the lowest message delay. As for energy consumption, the relay station that has a lower P_{in} value consumes more energy to transmit one bit of information than those that have a higher P_{in}. This result is consistent with the original energy consumption model we adopted. The data from the original model does show that a station consumes more energy in an idle state than in a sending or receiving state when the packet size is above some value (which is less than 1500 bytes). So even if the relay stations with lower P_{in} value spend less time transmitting packets, they do consume a certain amount of energy that could be significant.



(a)throughput

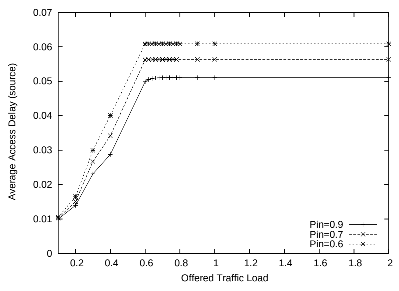


(b)source throughput

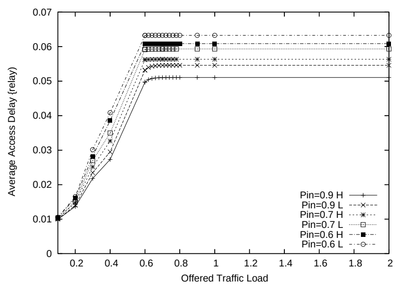


(c)relay throughput

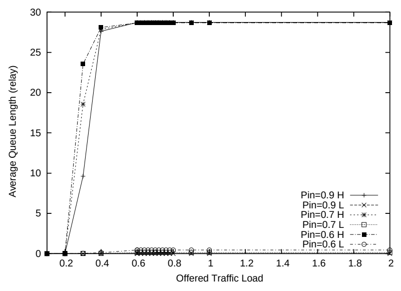
Figure 3.5: Relay Station Model: Throughput Analysis, $n = 5, nd = 5$



(a)source access delay



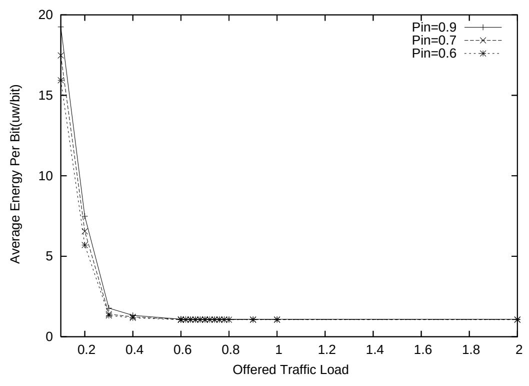
(b)relay access delay



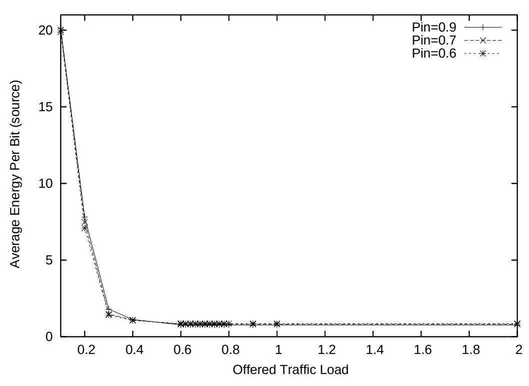
(c)relay average queue

Figure 3.6: Relay Station Model: Delay & Queue Analysis, $n = 5, nd = 5$

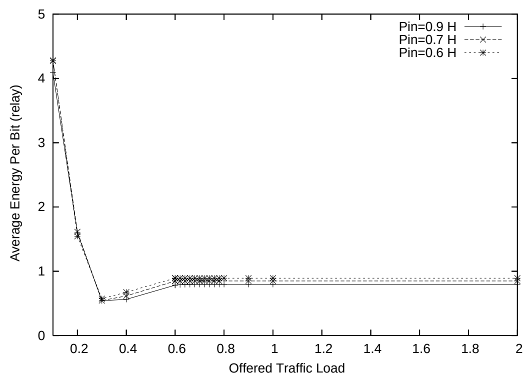
Figure 3.5 to 3.7 show the same trend as in figure 3.2 to 3.4. Comparing those two sets of results, we can see that when the traffic load is distributed on more relay stations, the overall throughput when the traffic load is unsaturated is higher, and average access delay increases faster when a decreases. The system consumes more energy to send one



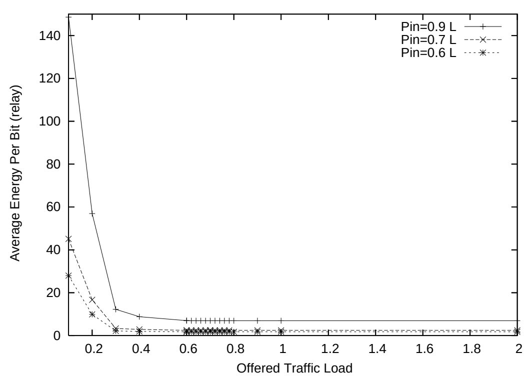
(a) total epb



(b) source epb



(c)relay epb, high P_{in}

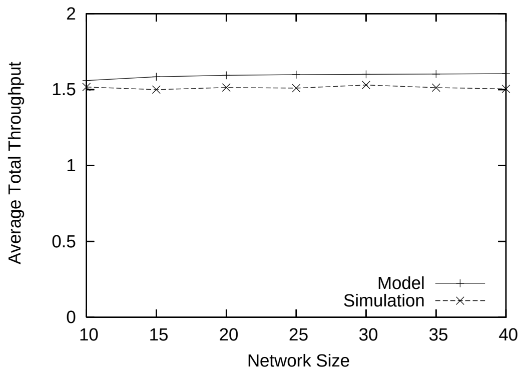


(d)relay epb, low P_{in}

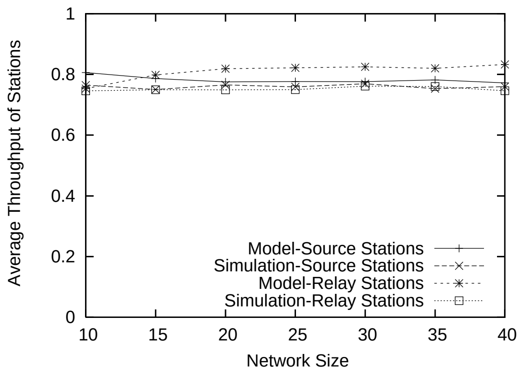
Figure 3.7: Relay Station Model: Energy Consumption Analysis, $n = 5, nd = 5$

bit of information. The reason is that if the same traffic is distributed on more stations, each station will have less traffic and more idle time. For the reason we mentioned in the discussion of the energy consumption model, the station consumes more energy in the idle state.

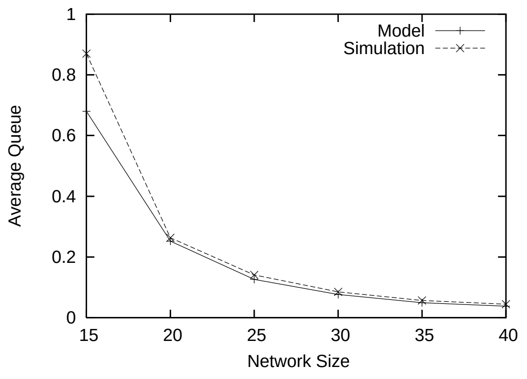
Figures 3.8 and 3.9 show some results from *NS2* simulations. In these two sets of simulations, the number of source stations are 5 and 7. The number of relay stations varies from 5 to 50. Each source station has a data connection with each relay station. So in our



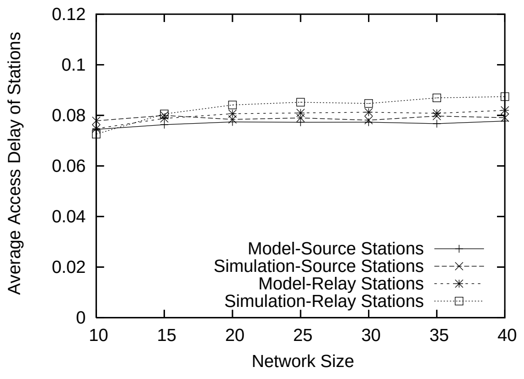
(a) throughput



(b) source/relay throughput



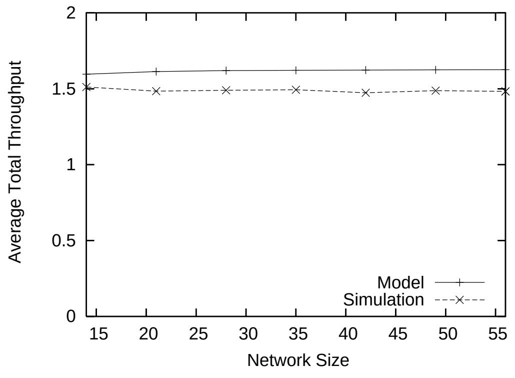
(c)relay average queue



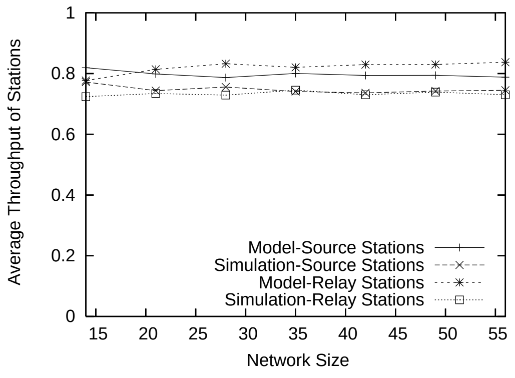
(d)source/relay access delay

Figure 3.8: Relay Station Model: Simulation Results, $n = 5$

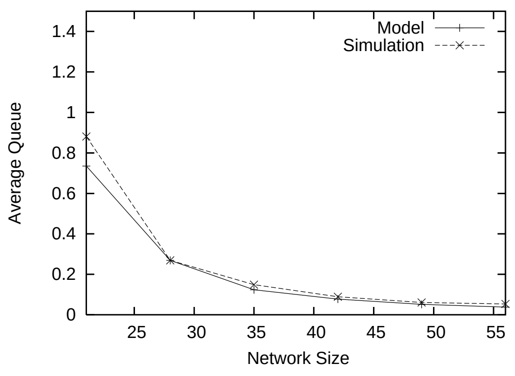
simulations, all relay stations have the same P_{in} value. Saturated traffic load is also used in the simulation. Each relay station will forward the packet it receives to a single data sink where data will stay and will not be forwarded anymore. This scenario models a simple two-hop network. The P_{in} value of each relay station is collected through the simulation and fed back to our models to validate them.



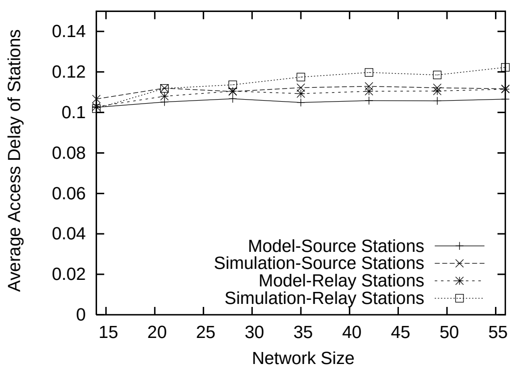
(a) throughput



(b) source/relay throughput



(c)relay average queue



(d)source/relay access delay

Figure 3.9: Relay Station Model: Simulation Results, $n = 7$

Chapter 4

Mathematical Model for ***GANGs***

4.1 Overview

For performance analysis in networks where *GANGs* protocol is deployed, we make more assumptions in addition to those described in the overview of Chapter 3. Besides source nodes and relay nodes, there is an additional kind of node called cluster head nodes. As defined before, source nodes inject data traffic into the network, and relay nodes forward data from node to node. Source nodes do not forward data and relay nodes do not generate data. Cluster head nodes behaves similarly to relay nodes. They only forward data and do not generate data.

For the purpose of modeling, the only difference between the *GANGs* protocol and *IEEE*802.11 protocol is that in *GANGs*, each node except cluster heads can be in a power on or power off state. The *ON* and *OFF* state of the nodes are synchronized. That is, the system as a whole will be in a power on or power off stage. During the power on stage, all nodes will follow *IEEE*802.11 protocol. During the power off stage, non-cluster head nodes will power off and cluster head nodes will follow contention-free *TDMA* protocol. The synchronization of the node states directly affects the results derived from the models.

For example, probabilistically, the collisions on the medium will be fewer when the ON and OFF state of the nodes are not synchronized.

The models established for source nodes and relay nodes in $GANGs$ networks can be kept the same as those in $IEEE802.11$ networks with minor changes. The model for cluster head nodes will be established in this section.

4.2 Model for $GANGs$

4.2.1 Notations and Terminology

nd total number of relay nodes

n total number of source nodes

nh total number of cluster heads

P_{off} proportion of the time that all relay nodes and source nodes are in state OFF

$aPeriod$ duration of each ON and OFF cycle

T_{off} duration of each OFF state, which is equal to $P_{off} * aPeriod$

P_{nj} probability that during any ON state, a source node will stay ON instead of jumping to the OFF state

P^r_{nj} probability that during any ON state, a relay node will stay ON instead of jumping to the OFF state

P^h_{nj} probability that during any ON state, a head node will stay ON instead of jumping to a $TDMA$ period

τ probability that a source node will send a packet in any time slot

τ^r probability that a relay node will send a packet in any time slot

τ^h probability that a head node will send a packet in any time slot

P_{coll} probability that a packet sent by a source node collides

P^r_{coll} probability that a packet sent by a relay node collides

P^h_{coll} probability that a packet sent by a head node collides

P_{in} probability that a relay node accepts a packet during contention periods

P^h_{in} probability that a head node accepts a packet during contention periods

PT^h_{in} probability that a head node accepts a packet from other heads during a $TDMA$ time slot

P_{succ} probability that there is a successful packet in the medium during any time slot

P_{idle} probability that the medium is idle during any time slot

ΔQ_t increment of queue size of a head node after each $TDMA$ time period

Thr_t throughput of each cluster head node during $TDMA$ time slots

4.2.2 Model for Relay Nodes & Source Nodes

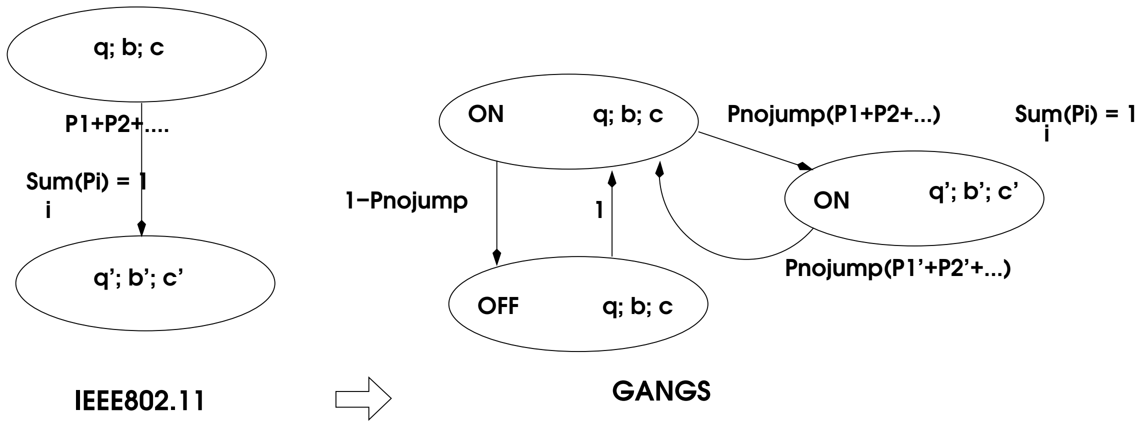


Figure 4.1: Model Modification for $GANGs$

Figure 4.1 shows the modification of the model for $GANGs$ protocol. In the $IEEE802.11$ protocol model, a node in some state $(q; b; c)$ will always transit to one of several other states

$(q'; b'; c')$s and the probabilities of transitions to those states will sum to 1.0. In the *GANGs* protocol model, a node in any *ON* state $(q; b; c;)$ will have probability $(1 - P_{nj})$ to transit to the *OFF* state, in which all q, b, and c will not be changed. We call these counterparts *ON* state $(q; b; c)$ and *OFF* state $(q; b; c)$. After a duration of $P_{off} * aPeriod$ in the *OFF* state $(q; b; c)$, the node will go back to the *ON* state $(q; b; c)$ with probability 1.0. Suppose the probability that a node in state $(q; b; c)$ will transit to state $(q'; b'; c')$ is p in the *IEEE*802.11 protocol. Then the probability that a node will transit from *ON* state $(q; b; c)$ to *ON* state $(q'; b'; c')$ is $P_{nj} * p$ in the *GANGs* protocol. Figure 4.2 shows an example of the change of state transitions of non-head nodes in *GANGs* networks.

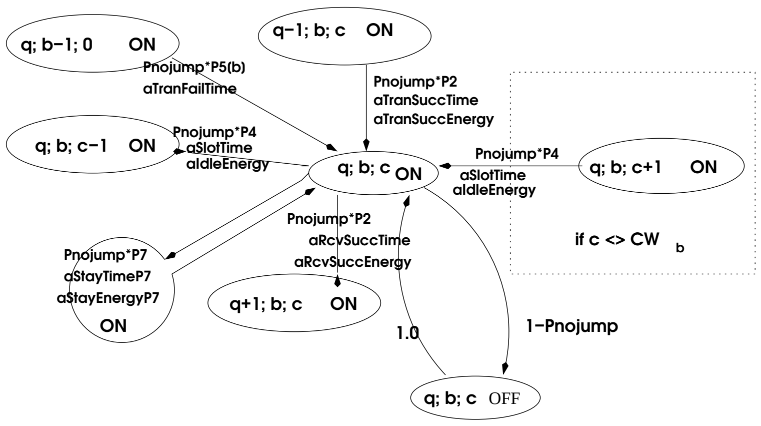


Figure 4.2: *GANGs* Protocol: Behavior of Source & Relay Node

4.2.3 Model for Cluster Heads

For cluster heads in the *GANGs* protocol, there is no *OFF* state. Head nodes are always on. When other nodes are in the *OFF* state, head nodes are in *TDMA* states. They can receive and send packets during the *TDMA* slots. When the *TDMA* slot time ends, non-head nodes will go back to the *ON* state with unchanged q, b and c, but the head nodes will have a changed queue size, q'. Figure 4.3 shows one example of the behavior of head nodes.

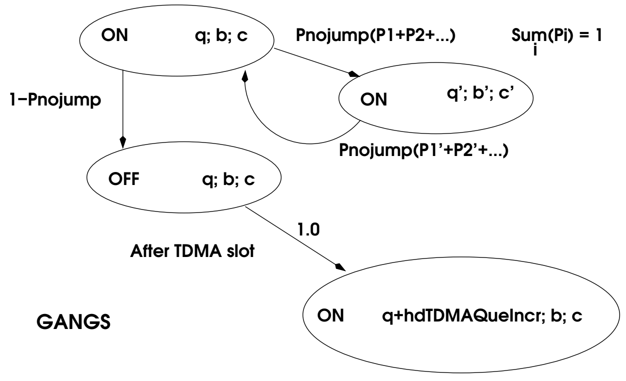


Figure 4.3: *GANGs* Protocol: Behavior of Cluster Head

In figure 4.3, our modeling of the effect of $TDMA$ on $GANGs$ protocol is shown. The statistical effect of the $TDMA$ protocol is taken into consideration. Suppose there are nh head nodes and the throughput of each node during a $TDMA$ slot is Thr_t. Let L represent the number of packets that a head node can send with that throughput. If the acceptance rate of packets to one cluster head from the neighbor cluster heads is PT^h_{in}, then during the $TDMA$ time slot each cluster head will send out L packets and receive $PT^h_{in} * L * (nh - 1)$, represented by ΔQ_t. A head node in state $(q; b; c)$ in the $TDMA$ time slot will return to the contention time slot in state $(q + \Delta Q_t; b; c)$ with probability 1.0.

4.2.4 Parameters in the Models

As was mentioned before, for any non-head node in $GANGs$ protocol, each ON state $(q; b; c)$ has a counterpart, OFF state $(q; b; c)$. Let π_i and P_i represent the frequency probability and time proportion of the ON state $(q; b; c)$. $\pi_i^{'}$ and $P_i^{'}$ represent those of the OFF state $(q; b; c)$.

From figure 4.2 and figure 4.3, we can derive,

$$\pi_i^{'} = \pi_i * P_{nj} \tag{4.1}$$

P_{off} represents the time proportion of all OFF states, which means,

$$P_{off} = \frac{\sum_i \pi_i{}' * T_{off}}{\sum_i \pi_i \mu_i + \sum_i \pi_i{}' * T_{off}} \tag{4.2}$$

Since the summation of all the probabilities is 1.0, we have,

$$\sum_i \pi_i + \sum_i \pi_i^{'} = 1 \tag{4.3}$$

We denote S as, $S = \sum_i \pi_i \mu_i$ and $P_{stateon}$ as, $P_{stateon} = \sum_i \pi_i$. From the three equations above, we can derive,

$$P_{stateon} = \frac{1.0}{1 + P_{nj}} \tag{4.4}$$

$$P_{off} = \frac{T_{off} * \sum_i \pi_i^{'}}{S + T_{off} * \sum_i \pi_i^{'}} \tag{4.5}$$

1. P_{nj}

   From the equations above, we have,

$$P_{nj} = \frac{(1 - P_{off}) * T_{off} - 2 * P_{off} * S}{(1.0 - P_{off}) * T_{off} - P_{off} * S}$$

2. $P_{coll}, \quad P^r_{coll} \quad and \quad P^h_{coll}$

   Let's think about the probability of a sent packet colliding in $GANGs$ protocol. We note again that when one node is in the ON state, all nodes are in the ON state. Let $\mathcal{R}_\tau$ represent the subspace of the state space,

$$\mathcal{R}_\tau = \{(q; b; c) : c = 0\}$$

$$\overline{\mathcal{R}_\tau} = \{(q;b;c) : (q;b;c) \notin \mathcal{R}_\tau\}$$

$$\overline{\mathcal{R}^{on}_\tau} = \{(q;b;c) : (q;b;c) \in \overline{\mathcal{R}_\tau} \quad and \quad node \quad in \quad ON \quad state\}$$

Then, the probability that a sent packet collides is,

P(packet collides | a packet is sent)

= 1 - P(packet does not collide | a packet is sent)

= 1 - P(all other nodes ∈ (q; b; c), c≠0 | a packet is sent)

= 1 - $\Pi_{j;j\neq i}$ P($Node_j \in \overline{\mathcal{R}_\tau}$ | a packet is sent)

= 1 - $\frac{\Pi_{j;j\neq i} P(Node_j \in \overline{\mathcal{R}\tau} \quad \& \quad a \quad packet \quad is \quad sent)}{P(a \quad packet \quad is \quad sent)}$

= 1 - $\frac{\Pi_{j;j\neq i} P(Node_j \in \overline{\mathcal{R}_\tau} \quad \& \quad a \quad packet \quad is \quad sent \quad \& \quad all \quad nodes \quad are \quad on)}{P(a \quad packet \quad is \quad sent)}$

= 1 - $\frac{\Pi_{j;j\neq i} P(Node_j \in \overline{\mathcal{R}_\tau} \quad \& \quad a \quad packet \quad is \quad sent \quad | \quad all \quad nodes \quad are \quad on) P(all \quad nodes \quad are \quad on)}{P(a \quad packet \quad is \quad sent)}$

= 1 - $\Pi_{j;j\neq i} P(Node_j \in \overline{\mathcal{R}_\tau} \quad | \quad all \quad nodes \quad are \quad on)$

$\cdot \frac{P(a \quad packet \quad is \quad sent \quad | \quad all \quad nodes \quad are \quad on) \cdot P(all \quad nodes \quad are \quad on)}{P(a \quad packet \quad is \quad sent)}$

= 1 - $\Pi_{j;j\neq i} P(Node_j \in \overline{\mathcal{R}_\tau} | \quad all \quad nodes \quad are \quad on)$

= 1- $\Pi_{j;j\neq i} \frac{P(Node_j \in \overline{\mathcal{R}^{on}_\tau})}{P(all \quad nodes \quad are \quad on)}$

= 1 - $\Pi_{j;j\neq i}(1.0 - \frac{\sum_i \pi(q; \quad b; \quad 0)}{P_{stateon}})$

That is,

$$P_{coll} = 1.0 - \prod_{j=0,j\neq i}^{n} (1 - \frac{\tau(j)}{P_{stateon}}) \prod_{j=0}^{nd} (1 - \frac{\tau^r(j)}{P_{stateon}}) \prod_{j=0}^{nh} (1 - \frac{\tau^h(j)}{P_{stateon}})$$

$$P^r_{coll} = 1.0 - \prod_{j=0}^{n}(1 - \frac{\tau(j)}{P_{stateon}}) \prod_{j=0,j\neq i}^{nd}(1 - \frac{\tau^r(j)}{P_{stateon}}) \prod_{j=0}^{nh}(1 - \frac{\tau^h(j)}{P_{stateon}})$$

$$P^h_{coll} = 1.0 - \prod_{j=0}^{n}(1 - \frac{\tau(j)}{P_{stateon}}) \prod_{j=0}^{nd}(1 - \frac{\tau^r(j)}{P_{stateon}}) \prod_{j=0,j\neq i}^{nh}(1 - \frac{\tau^h(j)}{P_{stateon}})$$

The state transition diagram for nodes in *GANGs* networks is governed by the following transition probabilities and durations that are different from those that have already been mentioned in the source and relay station model. Please refer to the appendix for detailed descriptions of *GANGs* network node state transitions.

1. Station i gets one successful packet from the medium and puts it in the queue

   P(a source node sends a packet | all nodes are on) $= \frac{\tau(i)}{P_{stateon}}$

   P(a replay node sends a packet | all nodes are on) $= \frac{\tau^r(i)}{P_{stateon}}$

   P(a head node sends a packet | all nodes are on) $= \frac{\tau^h(i)}{P_{stateon}}$

   Therefore, we can derive,

   $P_{succ} = \sum_{i=0}^{nd} \frac{\tau^r(i)*(1-P^r_{coll})}{P_{stateon}} + \sum_{i=0}^{n} \frac{\tau(i)*(1-P_{coll})}{P_{stateon}} + \sum_{i=0}^{nh} \frac{\tau^h(i)*(1-P^h_{coll})}{P_{stateon}}$

   $P^r\{(ON : q+1, b, c)|(ON : q, b, c)\} = P^r_{in}(i) * P_{succ} * P^r_{nj}(i)$

   $P^h\{(ON : q+1, b, c)|(ON : q, b, c)\} = P^h_{in}(i) * P_{succ} * P^h_{nj}(i)$

2. Station i sends out a packet and the packet collides

   $P\{(ON : b+1, c)|(ON : b, 0)\} = \frac{P_{nj}(i) \cdot P_{coll}(i)}{CW_{b+1}}$

$P\{(ON:B,c)|(ON:B,0)\} = \frac{P_{nj}(i)\cdot P_{coll}(i)}{CW_B}$

$P^r\{(ON:q,b+1,c)|(ON:q,b,0)\} = \frac{P^r_{nj}(i)\cdot P^r_{coll}(i)}{CW_{b+1}}$

$P^r\{(ON:q,B,c)|(ON:q,B,0)\} = \frac{P^r_{nj}(i)\cdot P^r_{coll}(i)}{CW_B}$

$P^h\{(ON:q,b+1,c)|(ON:q,b,0)\} = \frac{P^h_{nj}(i)\cdot P^h_{coll}(i)}{CW_{b+1}}$

$P^h\{(ON:q,B,c)|(ON:q,B,0)\} = \frac{P^h_{nj}(i)\cdot P^h_{coll}(i)}{CW_B}$

3. Station i sends out a packet successfully

$P\{(ON:0,c)|(ON:b,0)\} = \frac{P_{nj}(i)\cdot(1-P_{coll}(i))}{CW_0}$

$P^r\{(ON:q-1,0,c)|(ON:q,b,0)\} = \frac{P^r_{nj}(i)\cdot(1-P^r_{coll}(i))}{CW_0}$

$P^h\{(ON:q-1,0,c)|(ON:q,b,0)\} = \frac{P^h_{nj}(i)\cdot(1-P^h_{coll}(i))}{CW_0}$

4. Station i's backoff count is decremented when the medium is idle

$P_{idle} = \prod_{j=0}^{n}(1-\frac{\tau(j)}{P_{stateon}})\prod_{j=0}^{nd}(1-\frac{\tau^r(j)}{P_{stateon}})\prod_{j=0}^{nh}(1-\frac{\tau^h(j)}{P_{stateon}})$

$P\{(ON:b,c-1)|(ON:b,c)\} = P_{nj}(i) * P_{idle}$

$P^r\{(ON:q,b,c-1)|(ON:q,b,c)\} = P^r_{nj}(i) * P_{idle}$

$P^h\{(ON:q,b,c-1)|(ON:q,b,c)\} = P^h_{nj}(i) * P_{idle}$

5. Station i jumps out of ON state

$P\{(OFF:q,b,c)|(ON:q,b,c)\} = (1-P_{nj})$

$P^r\{(OFF:q,b,c)|(ON:q,b,c)\} = (1-P^r_{nj})$

$P^h\{(TDMA:q,b,c)|(ON:q,b,c)\} = (1-P^h_{nj})$

6. Station i jumps back to the ON state

   $P\{(ON: q, b, c)|(OFF: q, b, c)\} = 1$

   $P^r\{(ON: q, b, c)|(OFF: q, b, c)\} = 1$

   $P^h\{(ON: q + \Delta Q_t, b, c-1)|(TDMA: q, b, c)\} = 1.0$

7. $Other \quad Parameters$

   Once the above parameters are adjusted, the other parameters in the model will be calculated in the same way as they are in the $IEEEE802.11$ model.

4.2.5 Solution of the Models

System equations for the $GANGs$ model are same as those in $IEEE802.11$ with one additional unknown P_{nj} and therefore one additional equation. The Jacobi iteration algorithm is used to recursively solve the system equations. Solutions from the system equations are the frequency probabilities that the system stays in each state, π_i, not including $\pi_i^{'}$. Starting from this point, we can derive several system characteristics.

Time proportion of each state

As we mentioned before, for any node in $IEEE802.11$ networks the time proportion of each state can be calculated as:

$$P_i = \frac{\pi_i \mu_i}{\sum_{j=1}^{MaxStates} \pi_j \mu_j}$$

in which the μ_i's are the mean time for the node to stay in state i.

$$\mu_i = \sum_j P_{ij} * t_{ij} \quad for \quad j, \quad such \quad that \quad P_{ij} \neq 0$$

in which, P_{ij} is the probability for the node to jump from state i to state j and t_{ij} is the transition time for the node to transit from state i to state j.

For any relay node in $GANGs$ networks, we need to consider the time that the node spends in the OFF state. We know that the time proportion of all OFF states is P_{off}, thus the time proportion of all ON states is $(1 - P_{off})$. We have,

$$P_i = \frac{\pi_i \mu_i}{\sum_{j=1}^{MaxStates} \pi_j \mu_j + \sum_{j=1}^{MaxStates} \pi_j^{'} T_{off}}$$

$$P_i^{'} = \frac{\pi_i^{'} * T_{off}}{S + \sum_{j=1}^{MaxStates} \pi_j^{'} \mu_j^{'}}$$

and,

$$P_{off} = \frac{\sum_{j=1}^{MaxStates} \pi_j^{'} T_{off}}{S + \sum_{j=1}^{MaxStates} \pi_j^{'} T_{off}}$$

So we can derive,

$$P_i = \frac{\pi_i \mu_i * (1 - P_{off})}{\sum_{j=1}^{MaxStates} \pi_j \mu_j}$$

$$P_i^{'} = \frac{\pi_i * P_{off}}{(2 - P_{nj})}$$

For the convenience of discussion, we use $avgT_{state}^{r}$ to denote the average state time of a relay node,

$$avgT_{state}^{r} = \sum_i \pi_i \mu_i \quad for \quad IEEE802.11 \quad protocol$$

$$avgT_{state}^{r} = \sum_i \pi_i \mu_i + \sum_i \pi_i^{'} T_{off} \quad for \quad GANGs \quad protocol$$

Time proportions of source nodes and head nodes can be derived the same way, as can the average state time of source nodes and head nodes, denoted as $avgT_{state}$ and $avgT^{h}_{state}$ respectively.

Average Queue Length

For nodes in $IEEE802.11$ networks, the average queue length is,

$$avgQue = \sum_{b}\sum_{c}\sum_{q} P_i(q;b;c) * q$$

For relay nodes in $GANGs$, the queue length in the OFF state is the same as that of the counterpart in the ON state, so we can calculate the average queue length as,

$$avgQue^{r} = \sum_{b}\sum_{c}\sum_{q} P_i(q;b;c) * q + \sum_{b}\sum_{c}\sum_{q} P_i^{'}(q;b;c) * q$$

For cluster heads in $GANGs$ protocol, the queue size changes during the TDMA time slot. It changes from q to $(q + \Delta Q_t)$. Let's assume that once the cluster head enters a $TDMA$ time slot, it first sends out L packets then receives $L + \Delta Q_t$ packets, which leads to an additional ΔQ_t packets. Also, let's assume that the transmission of all those packets are distributed evenly in time, which means that the queue size during the $TDMA$ slot goes as: q, $q-1$, $\ldots$, $q-L$, $q-L+1$, $\ldots$, $max(q+\Delta Q_t, MQL)$. Then, the average queue length of the head nodes is,

$$avgQue^{h} = \sum_{b}\sum_{c}\sum_{q} P_i(q;b;c) * q + \sum_{b}\sum_{c}\sum_{q} P_i^{'}(q;b;c) * q^{'}$$

$$q^{'} = q - \tfrac{L}{2} + \tfrac{(nh-1)*\Delta Q_t}{2*nh} \quad for \quad q + \Delta Q_t \leq MQL$$

$$q^{'} = q - L + \tfrac{(nh-1)*MQL}{2*nh} \quad for \quad q + \Delta Q_t \geq MQL$$

in which L, as mentioned before, is the average number of packets that a head node can send during the $TDMA$ slot,

$$L = Thr_t * avgT^h_{state}$$

Average Energy Consumption Rate

Assuming that relay nodes and source nodes do not consume energy during OFF states, for all nodes in $IEEE802.11$ and all relay nodes and source nodes in $GANGs$, the average energy can be calculated as,

$$avgErg = \sum_i P_i * \varepsilon_i$$

in which the ε_i's are the mean energy consumption rates for a node staying in state i.

$$\varepsilon_i = \frac{\sum_j P_{ij} * \epsilon_{ij}}{\mu_i} \quad for \quad all \quad j, \quad such \quad that \quad P_{ij} \neq 0$$

in which ϵ_{ij} is the energy consumption for the node to transit from state i to state j.

For head nodes in $GANGs$, energy is also consumed during $TDMA$ slots. During a $TDMA$ time slot, energy consumption includes a part for sending L packets, a part for receiving the $(PT^h_{in} * 100)$ percents of packets from other head nodes, and a part for overhearing $(100 - PT^h_{in} * 100)$ percent of packets from other head nodes. Thus,

$$avgErg = \sum_i P_i * \varepsilon_i + P_{off} * \frac{\mathcal{E}}{avgT^h_{state}(i)}$$

in which, $\mathcal{E}$ is the total energy of all three parts.

Average Throughput

The system throughput should be the summation of all throughputs of relay nodes, source nodes, and head nodes in $GANGs$ networks. For $IEEE802.11$ networks,

$$avgThr = 8 * PAYLOAD * \{\sum_{i=0}^{nd} \frac{\tau^r(i)(1 - P_{coll}^r(i))}{avgT_{state}^r(i)} + \sum_{i=0}^{n} \frac{\tau(i)(1 - P_{coll}(i))}{avgT_{state}(i)}\}$$

And for $GANGs$ networks,

$$avgThr^{'} = 8*PAYLOAD*\{\sum_{i=0}^{nd} \frac{\tau^r(i)(1 - P_{coll}^r(i))}{avgT_{state}^r(i)} + \sum_{i=0}^{n} \frac{\tau(i)(1 - P_{coll}(i))}{avgT_{state}(i)} + \sum_{i=0}^{nh} \frac{\tau(i)^h(1 - P_{coll}^h(i))}{avgT_{state}^h(i)}\}$$

is only the throughput of the system in a contention slot. For the throughputs of the head nodes in a $TDMA$ slot, suppose the maximum number of packets that a cluster head can send in its own time slot is M_{thr}^h, then the throughput for a $TDMA$ time slot is,

$$M_{thr}^h = \frac{T_{off} * BandWidth}{8 * PAYLOAD * nh}$$

$$Thr_t = 8 * PAYLOAD * \sum_{i=0}^{nh} \sum_q \sum_b \sum_c \frac{\pi^{'}(q; b; c) Min(q, M_{thr}^h)}{avgT_{state}^h(i)}$$

Then, the total throughput of the system is,

$$avgThr = avgThr^{'} + Thr_t$$

Average One-Hop Delay

Average one-hop delay of a packet means the time from when a packet enters the queue and when it leaves the queue. First, we think about the time needed to transmit a packet when the packet is already at the top of the queue, called average access delay, denoted as D_{access}.

The probability for a sent packet to collide is P^r_{coll} for relay nodes. If a packet is successfully sent the first time, for which the probability is $(1 - P^r_{coll})$, the time it takes is T_s. If it collides the first time and is successfully sent the second time, the probability is $P^r_{coll}(1 - P^r_{coll})$, and so on. The probability that a packet is successfully sent the k^{th} time is $(P^r_{coll})^{k-1}(1 - P^r_{coll})$.

We know from the model that a relay node will send the packet when it is in the ON state and the backoff counter is zero. Let $\mathcal{R}^{on}_{\tau}$ represents the space where,

$$\mathcal{R}^{on}_{\tau} = \{(q; b; c) : c = 0 \quad and \quad node \quad is \quad on\}$$

The probability that the node is in $\mathcal{R}^{on}_{\tau}$ is τ. Suppose τ_i and τ_j are two consecutive states the node goes through, which are in $\mathcal{R}^{on}_{\tau}$. Then between τ_i and τ_j, the number of any state k that is not in $\mathcal{R}^{on}_{\tau}$ is $\frac{\pi_k}{\tau}$. We can derive that the time between two consecutive sending states is,

$$D = \sum_{k; k \notin \mathcal{R}^{on}_{\tau}} \frac{\pi_k \mu_k}{\tau}$$

This D is also the time between two consecutive transmissions of any packet. Then, we can derive the average access delay as,

$$D_{access} = \sum_{k=1}^{N} (P^r_{coll})^{n-1}(1 - P^r_{coll})(T_s + (k - 1) * (T_f + D))$$

in which N is the retransmission time minus one. The average one-hop delay should be the product of the average queue length and average access delay.

$$avgDelay = avgQue * D_{access}$$

When N goes to infinity, D_{access} will be,

$$D_{access} = T_s + \frac{P^r_{coll}(T_f + D)}{1 - P^r_{coll}}$$

The average delay for all source and relay nodes in $IEEE802.11$ and $GANGs$ networks can be derived as above, as can the average delay of the head nodes when all nodes are in the ON state. For head nodes in the $TDMA$ period, the average one-hop delay $avgTDelay$ is different. Each time a head node enters a $TDMA$ period it can send a maximum of M^h_{thr} packets. In order to send $avgQue^h$ packets it will need $n_1 + 1$ cycles,

$$n_1 = avgQue^h \quad mod \quad M^h_{thr}$$

$$q_1 = avgQue^h \quad \% \quad M^h_{thr}$$

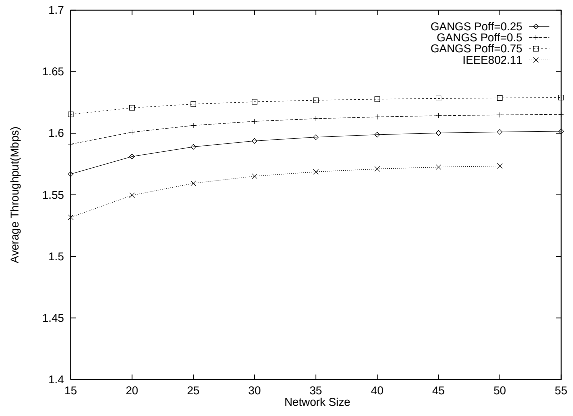
which means the node will send M^h_{thr} packets in the first n_1 cycles and q_1 packets in the last cycle. For packets sent in the first cycle, the packets will wait T_s, $2{*}T_s$, …, $M^h_{thr}{*}T_s$ to be sent. For packets sent in the second cycle, the packets will wait $aPeriod{+}T_s$, $aPeriod{+}2{*}T_s$, …, and so on. The average delay can be derived as,

$$avgTDelay = \frac{\sum_{j=0}^{n_1-1}\sum_{i=1}^{M^h_{thr}}(i * T_s + j * aPeriod) + \sum_{i=1}^{q_1}(n_1 * aPeriod + i * T_s)}{avgQue^h}$$

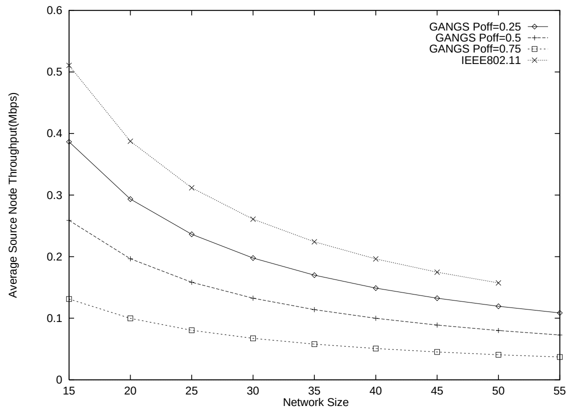
4.3 Experiments and Results

Tests on $GANGs$ protocol are done based on the different selections of parameters P_{in}, P_{off}, $aPeriod$, and network size. Tests on $IEEE802.11$ are done based on different P_{in} and network size. In our tests for which results are shown below, $GANGs$ protocol takes a P^h_{in} value of 0.3 and a P_{in} value of 0.7, $IEEE802.11$ takes a P_{in} value of 0.5. In each of the following figures the performance of $IEEE802.11$ is represented with one curve specified by "$IEEE802.11$". The performance of $GANGs$ is represented with three curves, each of which represents results for a different P_{off} value.

4.3.1 Average Throughput



(a) System Throughputs



(b) Source Nodes Throughputs

Figure 4.4: $GANGs$ Protocol: Average System Throughput

Figure 4.4a) shows the system throughput and b) shows the throughput of the source nodes. $GANGs$ protocol has better overall throughput which means that the system can handle more traffic load than an $IEEE802.11$ system. The throughput of the source nodes in an $IEEE802.11$ network is better than that of $GANGs$. This is because those source nodes only send data part of the time. This result also indicates that the choice of parameter P_{in} is not good enough to allow the $GANGs$ system to show better throughput. This will be explained later in the "average queue length" section. As for $GANGs$ protocol, the system throughput increases as the proportion of the $TDMA$ time slots, represented by P_{off}, increases, because the contention free time is increased. Also, the source nodes' throughput decreases as P_{off} increases. This is easy to understand since the total time for source nodes to transmit data is shorter.

4.3.2 Average Queue Length

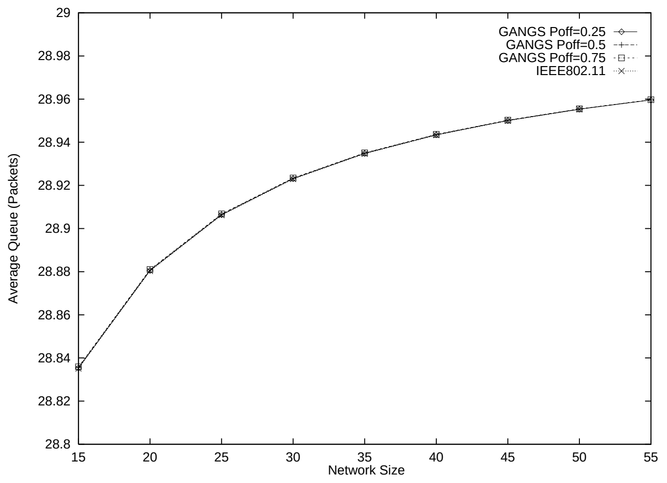
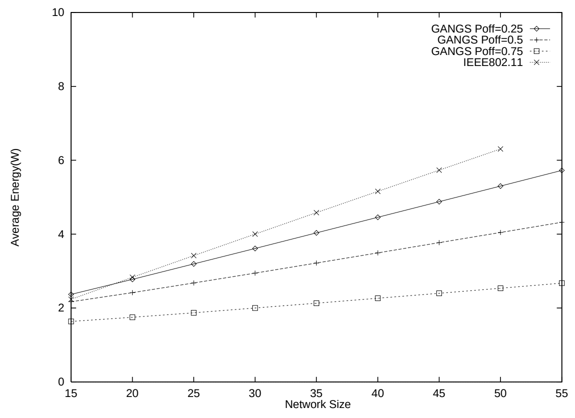


Figure 4.5: $GANGs$ Protocol: Average Queue Length

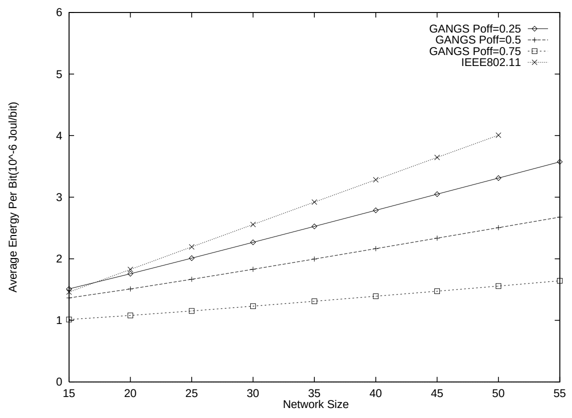
Figure 4.5 shows that the queue lengths are almost the same in $GANGs$ and $IEEE802.11$ systems. This is a result of the choice of parameter P_{in}. P_{in}'s have been chosen such that the whole system is fully loaded or overloaded based on the mathematical model used. Every node's queue is full most of the time. The difference in collision probability that should help source nodes to transmit more data in $GANGs$ protocol does not occur. More tests need to be done on different values of P_{in} and hdP_{in}. This is the reason that the $GANGs$ protocol does not show the same or better source node throughput.

4.3.3 Average Energy Consumption

Figure 4.6 shows the energy consumption of both protocols. Even though both protocols have similar overall throughput, the $GANGs$ protocol is much more energy efficient. As the network size increases, the number of collisions increase, causing more retransmission.



(a) Total energy Consumption



(b) energy Consumption Per Bit

Figure 4.6: *GANGs* Protocol: Average Energy Consumption

The energy spent on sending and receiving collided packets increases the total system energy consumption. For the *GANGs* protocol, the energy consumption decreases as the proportion of the $TDMA$ time slots, represented by P_{off}, increases. This is because the $TDMA$ time slot is a contention free time slot. No collisions occur during the $TDMA$ time slot. This not only helps to increase the system throughput, but also reduces energy consumption.

4.3.4 Average Message Delay

Figure 4.7 shows the one-hop message delay of both protocols. For *GANGs* protocol, the one-hop delay τ_{nn} between normal nodes is about the same as that for $IEEE802.11$. The one-hop delay τ_{hh} between cluster heads is much less. Consider a multi-hop message delivery with m hops, in $IEEE802.11$ the total delay will be the product $m\tau_{nn}$; in *GANGs*, the delay will be at most $2\tau_{nn}+(m-2)\tau_{hh}$, which is less than that in $IEEE802.11$ as the number m of hops increases. Also, as the proportion of $TDMA$ time slots, represented by P_{off}, increases, the message delay between cluster heads decreases. As mentioned before, the throughput of source nodes decreases in our current tests as the proportion of TDMA slots increases. If

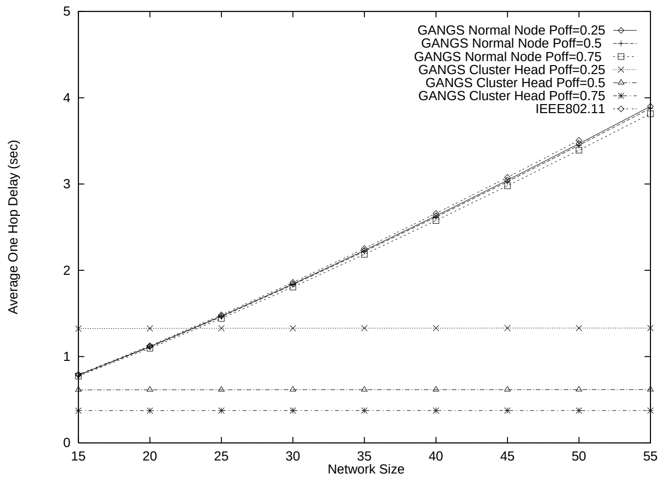


Figure 4.7: *GANGs* Protocol: Average One-Hop Delay

we consider the difference between the time a source starts to send a segment of data and the time a destination gets the last part of this data, we find that the shorter message delay between cluster heads will help *GANGs* protocol compensate for its smaller source node throughput.

From the performance figures, we see that *GANGs* protocol has better overall performance. It outperforms the *IEEE*802.11 protocol at system throughput, energy consumption and message delay. However, the results do not show good performance on source throughput, which we believe is caused by inappropriate choices of system parameters.

The performance shown in this section only represents the system performance when the system is stabilized, which means clusters are already established and cluster heads are already elected in the *GANGs* protocol. The time and energy consumption necessary to reach this stable state should also be taken into consideration when system performance is evaluated.

Chapter 5

Implementation of *GANGs* Protocol

5.1 Overview

NS2 is a well-known network simulation tool. Its MAC layer protocol simulations include both $IEEE802.11$ and $TDMA$ protocols. The implementation of $GANGs$ protocol is based on the existing implementation of those two protocols. For nodes in a specific *NS2* network, usually a single MAC protocol is adopted by all nodes. For each node in a $TDMA$ network, a specific time slot will be assigned if a service request is granted. For each node in an $IEEE802.11$ network, time segments will be granted through the DCF content procedure. For each node in the proposed $GANGs$ network, a hybrid MAC protocol is created based on the original $IEEE802.11$ protocol and $TDMA$ protocol.

During the lifetime of a $GANGs$ network, the network iterates through basic states such as a cluster establishment state, stable execution state, and cluster re-establishment state. Each node in the network can switch between the roles of cluster head and cluster node. When a node is working as a cluster head, its MAC protocol works the same way as for the $IEEE802.11$ protocol during system ON state, and $TDMA$ protocol during system OFF state. When a node works as a normal cluster node, during the system ON state it adopts

$IEEE$802.11 protocol, and during the system OFF state it shuts off.

A $GANGs$ network starts with all nodes working in contention mode. In order to establish the cluster topology, each node has to exchange energy information and go through a local maximum procedure until it identifies itself as either a cluster head or a cluster node. The system is said to be in a cluster establishment state when the local maximum procedure is ongoing. After the local maximum procedure reaches a stable state in which each node is either a cluster head with an assigned $TDMA$ time slot or a normal node with an identified head, the network will enter a relatively stable working mode. A cluster head will not give up its status unless certain criteria are met and the system enters the cluster re-establishment state. The cluster re-establishment procedure is usually invoked when some cluster head nodes are no longer the "strongest" of their peers, and it is necessary to re-elect other nodes with more power to be the new cluster heads. The cluster re-establishment procedure can be invoked within parts of the network or on the entire network.

5.2 Cluster Establishment Procedure

The purpose of the $GANGs$ protocol cluster establishment procedure, called the local maximum procedure (LMX), is to virtually distribute all nodes into different groups, called clusters. Each cluster has a head node and all other nodes within the cluster are normal nodes. After clusters are established, normal nodes communicate through cluster heads. Cluster heads communicate with peer cluster heads to transmit the information throughout the network. The target of the LMX procedure is to guarantee that each node is either a head node or has a head node in range, and each head node has enough head node neighbors to maintain the connectivity of the network. Note that if there is certain disconnection in the existing network, which means there are at least two nodes that can not reach each other no matter what route is used, the LMX procedure will not solve the problem.

The intent of the LMX procedure is to elect the most powerful nodes to be the cluster heads, and the most powerful of the remaining nodes to be inter-heads that maintain connectivity. The LMX procedure is implemented as a state machine. A single node will go through several stages during the LMX procedure until a relatively stable topology is created.

5.2.1 LMX Procedure States

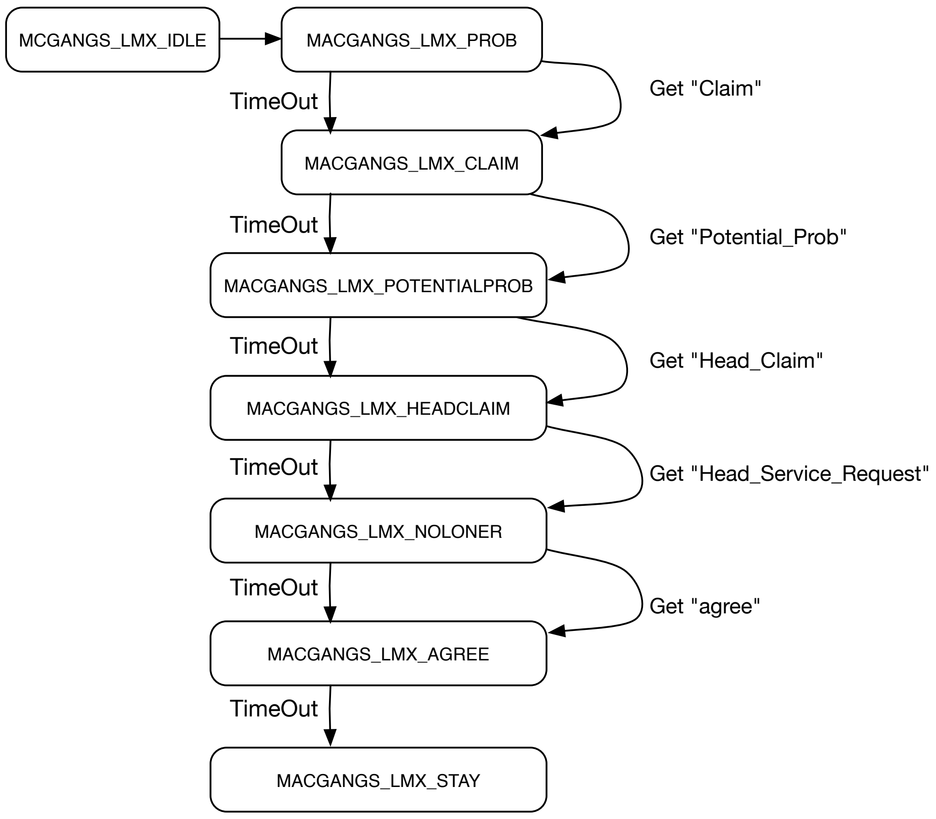


Figure 5.1: *GANGs* Protocol: LMX State Transitions

The LMX procedure works roughly as a state machine. Figure 5.1 shows the associated states and state transitions. A node state transition occurs either when time is up or a certain type of packet is received.

1. State MACGANGS_LMX_IDLE

   This is the starting state of all nodes. In this state, a node is neither a head node nor a normal node. The network topology has yet to be created. This is usually before the LMX procedure has started.

2. State MACGANGS_LMX_PROB

   Once a node's LMX procedure begins, it enters a MACGANGS_LMX_PROB state. In this state, a node will send energy information to its neighbors. It also collects energy information of all neighbors so that it will have the information to determine if it is a local maximum, the node that has the most energy among neighbors.

3. State MACGANGS_LMX_CLAIM

   In this state, any node that has determined that it is a local maximum will send out a cluster head *Claim* packet to its neighbors to claim to be a head. Note that at this stage, no two local maximum nodes can reach each other directly.

4. State MACGANGS_LMX_POTENTIALPROB

   In this state, there will be some nodes that have received cluster head *Claim* packets from more than one local maximum node. These nodes are good candidates for inter-head nodes, the heads that connects local maximum nodes. Each of these nodes will send out a potential head probe packet *Potential Probe* to notify the neighbors of its desire to be a cluster head.

Also, in this state, there might be nodes that are not local maximum nodes and yet have not received any head claim information. This situation exists when the local maximum nodes within the scope of those nodes are within the ranges of more powerful nodes. Those nodes can either send out a *Head Service Request* to their local maximum nodes or wait until a later stage of the LMX procedure.

5. State MACGANGS_LMX_HEADCLAIM

   In this state, all nodes that have sent out the potential head probe information will get similar information from the neighbor nodes that were in the same situation. One of these nodes will become the head node. There are certain metrics that the nodes evaluate to make this decision. First, the node that has the greatest number of orphan neighbors, neighbors that are not in the range of any current head, will win. Second, the node that has the greatest number of head neighbors will win. Finally, the node that has the most energy will win. This evaluation process balances connectivity and energy-awareness.

6. State MACGANGS_LMX_NOLONER

   In this state, the system will determine if there is any node that is left without a head in range, or if there is any head that does not have an outgoing link to other heads. If a normal node does not have a neighbor head at this stage, it will send a *Head Service Request* to its local maximum node. This also handles the unassigned nodes mentioned in state MACGANGS_LMX_POTENTIALPROB. If a head node does not have an outgoing link, it will send a *Head Service Request* to the neighbor that has the greatest number of cluster heads in its range. If none of its neighbors can reach a head node other than itself, then the "loner" head node will try to delegate the head service request to its neighbors.

7. State MACGANGS_LMX_AGREE

   In this state, all head service requests are answered. Ideally, every node has at least one neighbor head, every cluster head has at least one outgoing link to other cluster heads, and each node has identified itself as a normal node or a cluster head. A normal node will choose as a cluster head the most powerful among all neighbor heads, and send the agree message to its chosen head.

8. State MACGANGS_LMX_STAY

   In this state, the topology has been created and nodes are running normally until the next LMX procedure is invoked. Various metrics can be used to determine the start of the next LMX procedure. One choice is that it will occur when a cluster head's energy is decreased to a specific level, such as lower than 80 percent of the energy of its most powerful cluster node.

5.2.2 LMX Procedure State Transitions

The transitions among the LMX states are both time driven and event driven. Each state has a designated time out duration. Once the time is up, it will transit to the next possible state. Also, certain packets from the neighbors, identified as events, can expedite the transition when neighbor nodes' states are out-of-synchronization. The selection of the time out duration for each state can directly affect the performance of the LMX procedure and the $GANGs$ protocol.

In any state, if a node gets a $GANGs$ control packet from some node that is not in its neighbor list, it will bring up an "active node information" query. Suppose node A sent its information, including the energy information, to its neighbors by using a broadcast packet. Some neighbors got the packet; some neighbors did not. Suppose neighbor B did not get it. During the LMX procedure, node A will keep sending other packets accordingly. Unless all packets it sends are lost to B, B will get at least one packet from A. Node B notices that A is its neighbor but is not included in its neighbor list during the MACGANGS_LMX_PROB state. Node B then has a reason to believe there might be other neighbors that also missed the energy probe packet from A. Node B will send a "ReqProbe" packet to A with a unicast packet, which has the MAC layer delivery guarantee. When node A gets this packet, it will broadcast its energy information once again. Consider the possibility that more than one node will send "ProbeRequire" packet to A, in order to avoid excessive broadcast packets, node A will put a lower limit on the interval between two consecutive energy probe packets.

A node transits to state MACGANGS_LMX_PROB when the LMX procedure starts. The node collects neighbor energy information during a predefined length of time and transits to state MACGANGS_LMX_CLAIM when time is up. The waiting can be interrupted if a *Claim* packet is received, which means that one of its neighbor has already claimed to be a local maximum. It is then pushed into state MACGANGS_LMX_CLAIM.

After a node transits to state MACGANGS_LMX_CLAIM, it knows whether it is a local maximum node. A local maximum node will send out "Claim" packet to claim to be a head node, and push neighbor nodes into state MACGANGS_LMX_CLAIM if they are not already in it. A node will also start to collect all head node information for a predefined length of time during this state. If a node notices that it is qualified to be an inter-head node, it will send out a *Potential Probe* packet when time is up and push all neighbors into state MACGANGS_LMX_POTENTIALPROB.

During MACGANGS_LMX_POTENTIALPROB, all neighbor potential inter-head nodes will collect each other's *Potential Probe* packets. Each such node will apply the selecting algorithm to the choice of inter-head node. A node that wins out will send the *Head Claim* packets to its neighbors after a predefined length of time has passed, and push all neighbors into state MACGANGS_LMX_HEADCLAIM.

A node transits to state MACGANGS_LMX_HEADCLAIM either through timeout or receiving *Head Claim* packet. A normal node that transits to this state through timeout might be a "loner" node, which means that it is not within the reach of any head node. A head node in this state could also be a "loner", which means that it does not have any neighbor head nodes. Each node will collect all received *Head Claim* information and determine whether it is a "loner" node. After a predefined length of time has passed, a loner node will send out a *Head Service Request* to one of its neighbors according to certain criteria. The neighbor that gets the request will be pushed into state MACGANGS_LMX_NOLONER.

After a node transits to state MACGANGS_LMX_NOLONER, it will wait a predefined length of time and process possible *Head Service Request* information. A normal node in this state will make a decision as to which head node it will choose service from. It will then send an *Agree* packet to its chosen head node, and transit to state MACGANGS_LMX_AGREE. A node that receives an *Agree* packet in this state will be pushed into state MACGANGS_LMX_AGREE.

After a node transits to state MACGANGS_LMX_AGREE, it will wait a predefined length of time. This delay will give all the nodes in the system time to finish the head-choosing process. When the time is up, the node transits to state MACGANGS_LMX_STAY and one cycle of the LMX procedure is complete. The system will then enter a $TDMA$ time slot assignment procedure.

$IEEE$802.11 MAC protocol is deployed during the LMX procedure. The hybrid $GANGs$

protocol will not become active until the $TDMA$ slot assignment is complete.

5.3 TDMA Time Slot Assignment Procedure

The goal of $TDMA$ time slot assignment is to provide each cluster head a contention-free time slot for forwarding concentrated traffic. When the $GANGs$ system is in the OFF state, only cluster heads in the system are sharing the medium. The time segment of the OFF state can be treated as a multiple event time slot. Each cluster head will occupy one of the time slots, and no two neighbor cluster heads can share the same slot. As the breadth of the system increases, the assignment of the time slots among all cluster heads becomes a non-trivial task. Theoretically, the time segment can be divided into finer piece to accommodate as many cluster heads as possible in order to provide contention-free access. From a single cluster point of view, the maximum number of time slots is the number of neighbor cluster heads plus its own cluster head. But some "vacant" time slots are necessary considering that the choice of the slot location of a cluster head is actually constrained by the choices of all its neighbor cluster heads. This situation was briefly described in chapter 1. An excessive number of "vacant" slots will lead to inefficient use of the medium bandwidth. The goal of the time slot assignment algorithm is to achieve a fully assigned as well as sufficient system.

5.3.1 Feasibility of Time Slot Assignment

In this section, the feasibility of the $TDMA$ time slot assignment procedure will be proved, and the maximum number of time slots needed to achieve a fully assigned system will be determined.

Assume each node has $(M-1)$ neighbor nodes. Define problem $P(n,m,M)$ as the slot assignment procedure, in which n is the current total number of slots, and m is the current number of neighbors that have an assigned slot.

Assume there is an existing slot assignment according to node N_0,

$$[N_0, N_1, N_2, ..., N_{M-1}]$$

Then, according to node N_0, each neighbor has a slot. No collision exists.

Now consider node N_1. Suppose N_1 has neighbor list,

$$[N_0, N_1, ...N_i, N1_0, N1_1, ...N1_{M-1-i}]$$

in which $[N_2, N_3, ...N_i]$ are shared neighbors of N_0 and N_1. Then the procedure turns into problem $P(M, i+1, M)$.

For any node N in $[N1_0, N1_1, ...N1_{M-1-i}]$, there are three possible cases and associated resolutions:

1. N is the neighbor of all nodes in $[N_{i+1}, N_{i+2}, ...N_{M-1}]$

   N is assigned slot (M), then the problem turns into: $P(M+1, i+2, M)$

2. N is not a neighbor of any node in $[N_{i+1}, N_{i+2}, ...N_{M-1}]$

   N is assigned slot (i+1), then the problem turns into: $P(M, i+2, M)$

3. N is not the neighbor of at least one node in $[N_{i+1}, N_{i+2}, ...N_{M-1}]$, say, N_k,

   N is assigned slot (k), then the problem turns into: $P(M, i+2, M)$

So $P(M, i+1, M) = Union(P(M+1, i+2, M), P(M, i+2, M))$, and $P(n, M, M)$ is the final solution. Recursively,

$$P(M, i+1, M) = Union(P(M, i+2, M), P(M, i+3, M), ..., P(M, M, M), P(M+1, i+2, M), P(M+2, i+3, M)...)$$

The sub-procedure that affects the total number of slots is $P(M+1, i+2, M)$, or in general,

$$P(M + cnt_0, i + 1 + cnt_1, M) \quad and \quad cnt_1 \geq cnt_0$$

Since $(i + 1 + cnt_1) \leq M$, and $cnt_1 \leq (M - i - 1)$, so $cnt_0 \leq (M - i - 1)$. The maximum number of slots needed for a non-collision time slot assignment is:

$$MaxSlot = M + cnt_0 \leq (M + M - i - 1) = (2M - i - 1)$$

It is obvious that $i \geq 1$. So, $MaxSlot \leq 2(M - 1)$

For any node that follows $N_0, N_1, ...$, suppose the number of slots has already grown to $2(M-1)$, then any following scheduling procedure is a problem of $P(2(M-1), i, M), M \geq i \geq 2$, which means that there are $2(M-1) - i - 1$ unassigned slots for $(M - i - 1)$ nodes.

Since each node has at most $(M - 1)$ neighbors, as long as $2(M-1) - i - 1 \geq M - 2$, that is $M \geq i + 1$ which is equivalent to $M \geq 3$, the MaxSlot of 2(M-1) will serve all the nodes.

As a special case, for $M = 2$, which means each node has only one neighbor, the maximum number of two slots will serve all the nodes.

This proves that for the $GANGs$ protocol, the $TDMA$ time slot assignment procedure is feasible. Also, for a network in which the degree of connection is 6 (ideally, not practically), the maximum number of $TDMA$ slots needed is 10.

5.3.2 Heuristic Algorithm for Time Slot Assignment

The scheduling algorithm works on the assumption that the $GANGs$ protocol has passed the LMX procedure. Each node has its cluster head, and each cluster head is aware of its neighbor cluster heads. The purpose of the scheduling algorithm is to find the $TDMA$ slot (contention-free time slot) assignment for the neighbor cluster heads. The continuity of the time slots among neighbor cluster heads is not guaranteed, which means that there could be one or more vacant time slots between any two time slots that are actually occupied by neighbor cluster heads, and the $GANGs$ protocol requires that normal nodes keep quiet during all $TDMA$ slots that are within their transmission range. The ultimate goal is to use the least amount of total time slots to reach contention-free scheduling. A heuristic search approach is proposed for this task. Heuristic approaches are usually adopted for situations where there is not sufficient information, or no sufficient information can be accessed through a central location. A $GANGs$ network fits this situation due to its distributed nature. Information is maintained within each individual node. A node has no access to nodes other than its neighbor nodes without intensive message exchange. However, after the LMX procedure, there exists a pseudo "centralization" among all neighbor heads due to their energy ranking. This characteristic can be utilized in the heuristic search algorithm. The proposed time slot assignment algorithm works iteratively in five phases. Information is exchanged among neighbor heads to support and drive the search. For the sake of simplicity, the term *head* is used in the following description to refer to cluster head nodes.

First, at the beginning of scheduling, none of the heads have an assigned time slot. Using the same concept as the *local maximum* used in the cluster heads election stage, every connecting head should be aware of its *local maximum*, which is the neighbor head that has the maximum energy. If a head is such a local maximum, it will try to perform scheduling for all of the neighbor heads. It will put itself on the first time slot, then check each of its neighbors to find its energy rank. One fact worth mentioning is that connecting heads do not necessarily share the same set of neighbor heads. If in any case a head is not taken care of by any *local maximum*, it will assign itself a slot number according to its energy rank among all neighbors.

Second, now that each head should have an assigned time slot, an evaluation function is applied to each head. For each head, three kinds of metrics are used. The result from the evaluation function indicates whether a contention-free time slot assignment has been achieved.

1. The number of collisions between a head and all neighbor heads. This metric must reach zero at the end of the search.

2. The number of collisions among all neighbors of a head. This is necessary for the case where more than one neighbor of a head chooses the same time slot, and they are not within each other's transmission range. By choosing the same time slot, all those neighbors might not affect each other, but they will affect their shared neighbors. This metric must reach zero at the end of the search.

3. The difference between the time slot number that is assigned to a head and the total number of its neighbor heads. For example, if a head has four neighbors, then the time slot that is assigned to it shall ideally not exceed four. Usually it will be assigned a value that is larger than the number of total neighbors. This minimization of this metric represents the goal of *using the least number of total time slots.*

Third, ideally the evaluation function value for all nodes should reach zero, which is unlikely according to the third metric. The goal of the search is then to find the minimal evaluation value with the constraints that metric one and metric two have to be zero. A threshold is defined as the *dynamic* factor. It represents how *likely* a head is to change its time slot choices during the search procedure. After every iteration, the threshold is adjusted according to how far the evaluation is from the minimum. If the evaluation goes further, the threshold will be increased, indicating the head has the tendency to change its current choice; if the evaluation gets closer to the minimum, the threshold will be decreased, indicating that the head has a tendency to keep its current choice.

Fourth, according to the previous proof of the maximum number of total time slots, the maximum number of time slots needed to allocate contention free TDMA slots for a set of neighbors is the double of the total number of neighbors. So if a node decides to change its current choice, it will choose a random number in the range [0, 2*(number of neighbors)] which has no conflict with the current choices of the neighbors. It also checks for conflicts with the neighbors of its neighbors. If more than one neighbor shares the same time slot, the one with the least energy will have to pick a new slot until there is no conflict.

And last, a metric has to be chosen to decide when to end the search. In the current implementation, once the first and the second metrics are met, the search ends.

5.4 Experiments and Results

In this section, the implementation details of the proposed *GANGs* protocol will be described.

5.4.1 Implementation Structure

The proposed *GANGs* protocol is implemented on the base of the widely used *NS2* network simulator. *GANGs* protocol serves at the *MAC* layer of the simulation system. The structure of the *GANGs* component and its connection to the network simulator are illustrated in figure 5.2.

The purpose of *GANGs* protocol is to collect local traffic during the contention period, and forward the local traffic to the destinations on contention-free paths that are constructed by *GANGs* cluster heads. In order to achieve this goal, cluster nodes should use their cluster

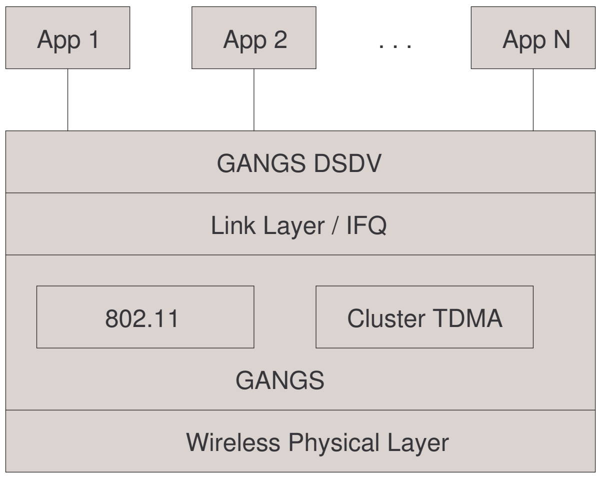


Figure 5.2: *GANGs* Protocol Implementation in *NS2*

heads as the cluster gateways. Both *MAC* layer and network layer routing protocol need to be modified for *GANGs* protocol.

In the original ad-hoc wireless network where *DSDV* routing protocol is deployed, each network node advertises its routing table periodically or on-demand. After collecting routing table information from all neighbor nodes, a new local routing table is created or updated. Route choice to any destination is based on the metrics defined in the routing table. Usually, the shortest path that has the least hops will be saved and used later. In *GANGs* protocol implementation, though the same principle is maintained, one additional rule is applied to cluster nodes. A cluster node always chooses it cluster head as the next hop for any destination that is not a direct neighbor. *DSDV* routing protocol is modified to force cluster nodes to ignore advertised routing information that is not from their cluster heads.

The *MAC* layer is constructed as a combination of 802.11 protocol and cluster *TDMA* protocol. Both the 802.11 component and cluster *TDMA* component in a network node will share the underlying wireless physical interfaces and the upper link layer and interface queue components. A timer in the $GANGs$ protocol component will switch the network node states between ON, a.k.a. contention period and OFF, a.k.a. contention-free period. 802.11 scheme is applied during contention periods and cluster $TDMA$ scheme is applied during contention-free periods. Cluster nodes all enter sleep mode during contention-free periods. In the original $TDMA$ protocol, time is divided into segments, and each segment is divided into multiple slots. Each node is allowed to send data to one destination in one time slot and receive data from one source in another time slot. This requires that both sender and receiver are aware of the common time slots. A node can enter sleep mode when it is neither sending nor receiving. $GANGs$ cluster $TDMA$ adopts a similar approach. Each cluster head can send data to all its neighbor cluster heads within its time slot, and each cluster head stays awake during all time slots in order to receive the packets that are sent to it.

5.4.2 Experiment Configuration

The experiment is based on the network topology shown in figure 5.3. Suppose there are a total of N nodes in one test, of which n are source nodes, and nd are relay nodes. nd CBR traffic sources are evenly attached to the n source nodes. Each CBR source is connected to one relay node. In order to simulate the forwarding actions of the relay nodes, a "null" sink node is used. nd sink agents are attached to the "null" sink node, and each relay node is connected to one of the sink agents.

Source nodes accept data from the upper CBR traffic sources and send the data to the destination relay nodes. Relay nodes accept the data and forward a copy to the sink agents. For a relay node, the ratio of the amount of data it receives to the total amount of data that is received by all relay nodes and all sink agents represents the parameter P_{in} that was

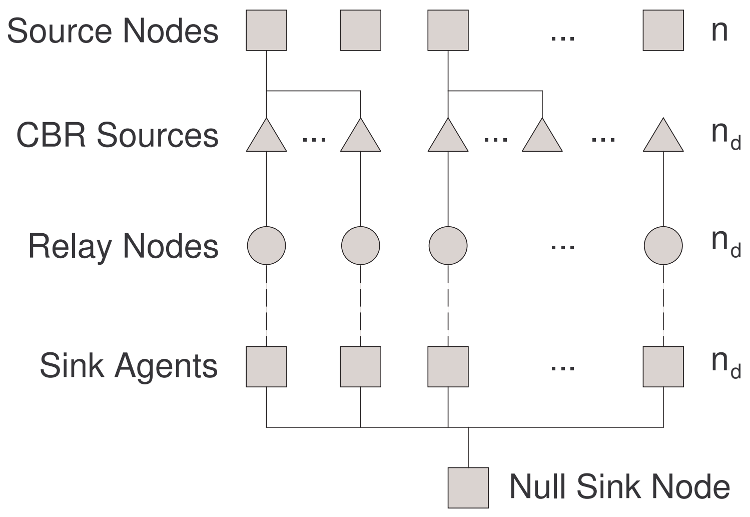


Figure 5.3: *GANGs* Protocol Experiment Configuration in *NS2*

described in the previous *GANGs* model chapter.

Network Node Configuration

Each node in the test has a transmission range of 250 meters. There are four kinds of 2-dimensional simulation layouts: 250 by 250 with 5 nodes, 500 by 250 with 10 nodes, 500 by 500 with 20 nodes, and 1000 by 250 with 20 nodes. Node locations follow a uniform distribution. The different layouts will introduce various numbers of hops between a data source and a data destination.

In all the tests, data packet size is fixed at 1500 bytes and the network interface bandwidth is set to 2Mbps. For 802.11 protocol, minimal backoff window size is set to 31 and the maximum backoff window size is set to 127. For cluster $TDMA$ protocol, the number of time slots is set to the number of total cluster heads, and the length of each time slot is the total contention-free period divided by the number of slots. The link layer interface queue limit is set to 50. To simplify the tests, $TDMA$ slot assignment is statically calculated outside the simulation and fed back to the simulation after the LMX procedure is complete and all cluster heads are elected. This means that overhead caused by the $TDMA$ slot assignment is not included in the tests.

Throughput & Queue Length Data Collection

Network throughput is collected in two ways. First, all relay nodes and sink agents count the number of arriving packets and record the delay of each packet. Application level throughput and average delay are calculated using that data. Second, all nodes' MAC layers collect information on every successful and failed packet transmission and idle state in the medium. MAC level throughput, packet collision probability, and packet transmission probability are calculated using that data. Total throughput for the system is the summation of the throughputs of all relay nodes and sink agents.

Each node calculates its average queue length at the link layer each time a packet enters or leaves the link layer interface queue. At the end of the test, queue length information is collected from all nodes. System average queue length is the average of the queue length over all nodes.

Energy Consumption Data Collection

Energy consumption data is also collected in two ways. First, Feeney and Nilson's energy model [18] described in the previous section is applied at the MAC layer. Given the fixed data packet size of 1500 bytes, for 802.11, the derived packet energy consumptions under different situations are:

eTS energy to transmit a successful packet($\mu W \cdot sec$) 0.0034408

eTF energy to transmit of a failed packet($\mu W \cdot sec$) 0.0005376 tTS time to transmit a successful packet(sec) 0.00698

tTF time to transmit a failed packet(sec) 0.000356

eRS energy to receive a successful packet($\mu W \cdot sec$) 0.001142

eRF energy to receive a failed packet($\mu W \cdot sec$) 0.000397

tRS time to receive a successful packet(sec) 0.00698

tRF time to receive a failed packet(sec) 0.000484

eIdle energy consumed in an idle slot($\mu W \cdot sec$) 0.00004215

tIdle time length of an idle slot(sec) 0.000050

eOvhdS energy consumed overhearing a successful packet($\mu W \cdot sec$) 0.00075308

eOvhdF energy consumed overhearing a failed packet($\mu W \cdot sec$) 0.00001999

For cluster heads in contention-free periods, there should not be any collisions. Energy consumed in contention-free periods should be the energy to transmit and receive the packets, plus energy to overhear the packets since all heads are awake during contention-free periods.

Second, the *NS2* simulator is equipped with a physical layer energy model. The model assumes that the node consumes a different level of power at transmission, receiving, idle, and sleep time. The model automatically calculates the energy consumed at transmission and idle time. The upper MAC layer will cause the model to enter sleep mode if the node is equipped with such functionality. In $GANGs$ protocol, cluster nodes can choose to enter sleep mode when the system is in a contention-free period.

Total energy consumption is computed at the end of the test. Average system power consumption is the total energy consumed over the total test duration. Power consumption per bit is the result of average power over average throughput.

5.4.3 Experiment Steps

Tests have been performed to verify the implementation of $GANGs$ protocol feature by feature. In each test, $GANGs$ protocol and $IEEE802.11$ protocol are used alternately as the MAC protocol. Data is collected for comparison for throughput, delay, queue length, and energy consumption.

LMX Procedure

In this set of tests, full $GANGs$ protocol is not used. Only the LMX procedure is applied. Timeout durations of all LMX states are set to one second. The LMX procedure is performed once during each test. Each test lasts 250 seconds. Each test only contains one source node, to which saturated traffic is applied.

For the 250 by 250 layout with 5 nodes, one cluster head is elected. For the 500 by 250 layout with 10 nodes, 4 cluster heads are elected. For the 500 by 500 layout with 20 nodes, 5 cluster heads are elected. For the 1000 by 250 layout with 20 nodes, 8 cluster heads are elected. In all tests, a cluster node is able to choose a cluster head, and connectivity among cluster heads is 100 percent.

As the network size increases, the impact of the LMX procedure on ongoing traffic increases, though not to a significant level. Compared with 802.11, average throughput decreases and energy consumption increases as the network size increases. The average queue length decreases since less data gets from source node to relay nodes.

Modified *DSDV* Protocol

In this set of tests, full *GANGs* protocol is not used. The *DSDV* adaptation to *GANGs* protocol is applied together with the *LMX* procedure. For all four test layouts, the resulting routing table for cluster heads follows the rule that they have one hop to direct neighbor nodes and always use some cluster head as the next hop to distant nodes. The resulting routing table for cluster nodes follows the rule that they have one hop to direct neighbor nodes and always use their own cluster heads as the next hop to distant nodes.

Cluster *TDMA* Slot Assignment

The implementation of the heuristic slot assignment algorithm is not included in the *NS2* simulator. It was implemented independently. Tests have been done on different numbers of nodes and different topologies. The slot assignment goal is usually achieved.

Complete *GANGs* Protocol

Numerous tests have been done on the attempted implementation of the full *GANGs* protocol. No positive results have yet been discovered and more experimentation is necessary.

5.5 Discussion & Further Consideration

A conclusion on the performance of *GANGs* protocol can not be reached without further investigation. There are some important issues regarding the implementation and testing of the *GANGs* protocol that require attention.

The *LMX* procedure and cluster *TDMA* slot assignment procedure of *GANGs* protocol add overhead to system performance including throughput and energy consumption. The promise is that after the system stabilizes, the energy and delay improvements for contention-free periods will make up for those losses. The complexities of both procedures may mean that this is unlikely for large networks.

The choice of the ratio of the contention interval and contention-free interval (P_{off}), and the total length of one full (ON/OFF) period ($aPeriod$) is critical. Ideally, the length of the contention-free period should take the current traffic load into account. If not enough packets accumulate at cluster heads to fill the contention-free slot time, bandwidth is wasted. If too many packets accumulate at cluster heads, then some packets will have to wait more than one unit of $aPeriod$ time, and packet delay will increase. The order of the cluster head time slot assignments can also cause delays. Suppose cluster heads A and B have slot assignment 1 and 2. The data from B to A will have to wait for the next contention-free period to be forwarded to the next cluster head.

The advantage of using the cluster $TDMA$ slot lies in the difference between the access delay of a packet under $TDMA$ and under 802.11. The access delay is fixed in $TDMA$ networks as it is the packet transmission time. The access delay in an 802.11 network heavily depends on the network size, density, and traffic load. The impact of access delay will be increased as the path from a data source to destination increases. $GANGs$ protocol may provide performance improvement in this case.

The number of elected cluster heads after the LMX procedure affects the performance of $GANGs$ protocol. In the current implementation, all nodes have the same transmission and receiving power. The range a cluster head covers is no different from that of a cluster node. This means an intermediate node needs to be elected as cluster head in order to provide connection between two local maximal nodes. If a high percentage of the nodes are elected as cluster heads, it will add complexity to $TDMA$ slot assignment, and defeat the purpose of establishing a *GANGs* network backbone. There is related work in network clustering that suggest using higher transmission and receiving power for cluster head nodes so that the number of cluster heads can be greatly reduced. This approach can also be adopted for *GANGs* protocol. One downside of this approach is that cluster heads will consume more power and their power level will decrease more rapidly. This will trigger more cycles of the *LMX* procedure.

References

[1] I. Akyildiz and W. Su, "A survey on sensor networks," Aug. 2002.

[2] A. Amis and R. Prakash, "Load-balancing clusters in wireless ad hoc networks," in *Proceedings of ASSET 2000, Richardson, TX, USA*, pp. 25–32, Mar. 2000.

[3] D. J. Baker and A. Ephremides, "The architecture organization of a mobile radio network via a distributed algorithm," *IEEE Transactions on Communications*, pp. 1694–1701, Nov. 1981.

[4] D. J. Baker and A. Ephremides, "A distributed algorithm for organizing mobile radio telecommunication networks," in *Proceedings of Second International Conference on Distributed Computer Systems*, pp. 476–483, Apr. 1981.

[5] S. Basagni, "Distributed and mobility-adaptive clustering for multimedia support in multi-hop wireless networks," in *Proceedings of Vehicular Technology Conference*, vol. 2, pp. 889–893, 1999.

[6] S. Basagni, "Distributed clustering for ad hoc networks," in *Proceedings of International Symposium on Parallel Architectures, Algorithms and Networks*, pp. 310–315, June 1999.

[7] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW: A media access protocol for wireless LANs," in *ACM SIGCOMM, London, U.K*, pp. 212–225, Oct. 1994.

[8] G. Bianchi, "Performance analysis of the IEEE802.11 distributed coordination function," *IEEE Journal in Selected Areas: Communication*, vol. 18, pp. 535–547, March 2000.

[9] L. Bononi, M. Conti, and L. Donatiello, "A distributed mechanism for power saving in IEEE 802.11 wireless lans," *Mobile Networks and Applications*, vol. 6, no. 3, pp. 211–222, 2001.

[10] D. Braginsky and D. Estrin, "Rumor routing algorithm for sensor networks," 2002.

[11] F. Cali, M. Conti, and Gregori, "IEEE802.11 wireless lan: Capacity analysis and protocol enhancement," in *INFOCOM '98 Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 1998.

[12] M. M. Carvalho and J. J. Garcia-Luna-Aceves, "Delay analysis of the IEEE802.11 in single-hop networks," in *11th IEEE International Conference on Network Protocols (ICNP'03), Atlanta, Georgia, USA*, Nov. 2003.

[13] A. Cerpa and D. Estrin, "Ascent: Adaptive self-configuring sensor networks topologies," in *21st International Annual Joint Conference*, IEEE Computer and Communication Societies (INFOCOM), June 2002.

[14] M. Chatterjee, S. Das, and D. Turgut, "An on-demand weighted clustering algorithm (wca) for ad hoc networks," in *Proceedings of IEEE GLOBECOM 2000, San Francisco, CA, USA*, pp. 1697–1701, Nov. 2000.

[15] H. S. Chhaya and S. Gupta, "Performance modeling of asynchronous data transfer methods of IEEE802.11 mac protocol," *Wirel. Netw.*, vol. 3, no. 3, pp. 217–234, 1997.

[16] D. S. J. De Couto, D. Aguayo, B. A. Chambers, and R. Morris, "Performance of multihop wireless networks: Shortest path is not enough," in *Proceedings of the First Workshop on Hot Topics in Networks (HotNets-I)*, (Princeton, New Jersey), ACM SIGCOMM, October 2002.

[17] D. Estrin and M. Srivastava, "Wireless sensor networks, mobicom 2002 tutorial," 2001.

[18] L. M. Feeney and M. Nilsson, "Investigating the energy consumption of a wireless network interface in an ad hoc networking environment," in *IEEE INFOCOM, Anchorage, AK, USA*, 2001.

[19] M. Gastpar and M. Vetterli, "the capacity of wireless networks: The relay case," in *Proceedings of INFOCOM'02*, IEEE, June 2002.

[20] M. Gerla and J. Tsai, "Multicluster, mobile, multimedia radio network," *Wireless Networks*, pp. 255–265, 1995.

[21] M. Grossglauser and D. Tse, "Mobility increases the capacity of ad-hoc wireless networks," in *IEEE Infocom'01, Anchorage, AL, USA*, Apr. 2001.

[22] P. Gupta and P. Kumar, "Capacity of wireless networks," tech. rep., University of Illinois, Urbana-Champaign, 1999.

[23] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *HICSS*, 2000.

[24] L. Huang and T.-H. Lai, "On the scalability of IEEE802.11 ad hoc networks," in *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pp. 173–182, ACM Press, 2002.

[25] IEEE Computer Society LAN MAN Standards Committee, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Std 802.11-1997," 1997.

[26] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," in *Sixth Annual International Conference on Mobile Computing and Networking, Boston, MA*, pp. 56–67, Aug. 2000.

[27] P. Karn, "MACA- a new channel access method for packet radio," in *ARRL/CRRL Amateur Radio 9th Computer Networking*, pp. 134–140, 1990.

[28] B. Karp and H. T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks," in *Mobile Computing and Networking*, pp. 243–254, 2000.

[29] S. A. Khayam and H. Radha, "Markov-based modeling of wireless local area networks," in *Proceedings of the 6th international workshop on Modeling analysis and simulation of wireless and mobile systems*, pp. 100–107, ACM Press, 2003.

[30] J. H. Kim and J. K. Lee, "Capture effects of wireless CSMA/CA protocols in rayleigh and shadow fading channels," *IEEE Transactions on Vehicular Technology*, vol. 48, pp. 1277–1286, July 1999.

[31] L. Kleinrock and J. Silver, "optimum transmission radii for packet radio networks and why six is a magic number," in *National Telecommunications Conference. Birmingham, Alabama*, pp. 4.3.1–4.3.5, oct 1978.

[32] P.Chatzimisios and V.Vitsas, "Throughput and delay analysis of IEEE802.11 protocol," in *IEEE International Workshop on Network Appliances, (IWNA), Liverpool, U.K*, Oct. 2002.

[33] J. W. Robinson and T. S. Randhawa, "Saturation throughput analysis of IEEE802.11e enhanced distributed coordination function," *IEEE Journal On Selected Areas In Communications*, vol. 22, June 2004.

[34] S. M. Ross, *Stochastic Processes.* Wiley & Sons. Inc., 1983.

[35] I. C. S. Basagni and A. Farago, "A generalized clustering algorithm for peer-to-peer networks," in *Proceedings of Workshop on Algorithmic Aspects of Communication*, July 1997.

[36] E. Shih, S.-H. Cho, N. Ickes, R. Min, A. Sinha, A. Wang, and A. Chandrakasan, "Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks," in *Mobile Computing and Networking*, pp. 272–287, 2001.

[37] S. Singh and C. Raghavendra, "Pamas: Power aware multi-access protocol with signalling for ad hoc networks," 1999.

[38] H. Takagi and L. Kleinrock, "Throughput analysis for persistent CSMA systems," *IEEE Transactions on Communications*, vol. 33, pp. 627–638, July 1985.

[39] O. Tickoo and B. Sikdar, "Queueing analysis and delay mitigation in IEEE 802.11 random access mac based wireless networks," in *INFOCOM 2004, HongKong, China*, March 2004.

[40] W. Ye, J. Heidemann, and D. Estrin, “An energy-efficient mac protocol for wireless sensor networks,” 2002.

[41] E. Ziouva and T. Antonakopoulos, “CSMA/CA performance under high traffic conditions: Throughput and delay analysis,” *Computer Communications*, pp. 313–321, 2002.

[42] E. Ziouva and T. Antonakopoulos, “The effect of finite population on IEEE802.11 wireless lans throughput/delay performance,” *IEEE MELECON, Cairo, EGYPT*, May 2002.

Appendix

A.1 Basics of the Mathematical Model

Unpredictability and non-determinism are all around us. The behavior of any system may follow a number of different paths. Some of the paths may be more likely than others, but none are absolutely certain. This is called "randomness". It is interesting to know how those systems perform under different conditions. Instead of gaining such information by experiment on real systems, one can construct a mathematical model of the system and obtain the desired information by analysis. A mathematical model can capture all the essential features of a system, display underlying trends, and provide quantitative relations between input parameters and performance characteristics. Computer systems, communication systems, and other systems have two fundamental properties. First, they are dynamic. They go through different states as time progresses, and which state they will enter is affected by random phenomena. Thus we can use two sets of random variables to describe the behavior of the systems. One set is used to describe the state that the systems stay in, called state space. Another is called parameter space, and usually represents the time of each state. These two sets of random variables construct a stochastic process which is the appropriate tool for modeling and studying the random behavior of these systems.

A.1.1 Markov Chains

"If the state of the process at a given moment in time is known, then its subsequent behavior is independent of its past history". This is called the $MarkovProperty$. A stochastic process that has the $MarkovProperty$ is called a Markov chain in discrete time space and a Markov process in continuous time space. Markov processes provide us with an important modeling tool. [34]

Let $\{X(n) : n \geq 0\}$ be a Markov chain with finite state space $\mathcal{R}$, then,

$$P\{X(n+1) = j | X(i); i \leq n\} = P\{X(n+1) = j | X(n)\} = P_{ij}$$

$$P_{ij} \geq 0, \sum_j P_{ij} = 1$$

for every $n \geq 0$ and $j \in \mathcal{R}$

An irreducible aperiodic and positive recurrent Markov chain has a unique stationary distribution. In other words, for each state j, denote $P_j = P\{X_n = j\}$, if

$$P_j = \sum_{i=0}^{\infty} P_i P_{ij}$$

then the Markov chain has a stationary distribution, X_n will have the same distribution for all n, and the distribution can be represented as $\{\pi_j, j = 0, 1, 2, \ldots\}$,

$$\pi_j = \sum_{i=0}^{n} \pi_i P_{ij} \qquad \sum_{i=0}^{n} \pi_i = 1$$

π_i can be interpreted as the proportion of time that the Markov chain is in state i.

A.1.2 Semi-Markov Processes

A semi-Markov process is one that changes states in accordance with a Markov chain but takes a random amount of time between changes. Consider a stochastic process with state {0,1,2, …}, such that, whenever it enters state i, the probability that the next state it enters is j is P_{ij}, and the time until the transition from i to j occurs has distribution F_{ij}. If we let X_n denote the nth state visited, then $\{X_n, n \geq 0\}$ is a Markov chain. If X_n has the stationary distribution π, then π_j will be interpreted as the proportion of the X_n's that equal j, which is the frequency probability of the system staying in state j. Let's denote,

$$\mu_i = \sum_j P_{ij} \int_0^\infty tF_{ij}(t)dt$$

μ_i is the mean time that the system stays in state i once it enters state i. Then,

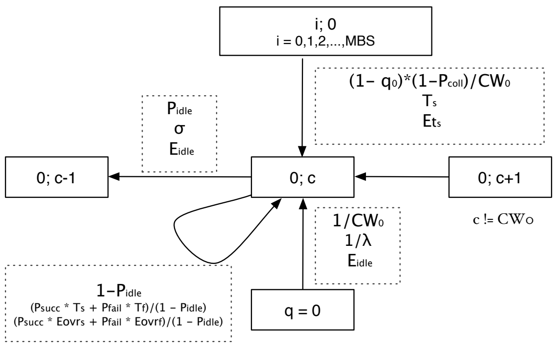
$$P_i = \frac{\pi_i \mu_i}{\sum_j \pi_j \mu_j}$$

P_i is the proportion of time that the system stays in state i.

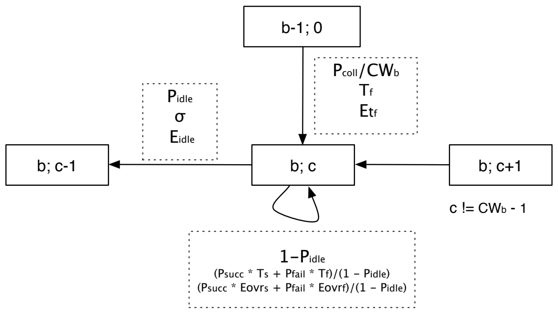
A.2 Source Node Model Details

Different assumptions are applied for source nodes and relay nodes. Source nodes listen, generate, and send data, and relay nodes listen, receive and forward data. Though our source node model takes queue empty state into consideration, when we put the source node model and relay model together to model the whole network, we intend to set up saturated traffic at source nodes such that the queue length information of the source nodes are not important any more. We use $(b; c)$ to describe the behavior of the source node, in which b is the backoff stage and c is the backoff count.

All the states that a source node goes through fall into seven types described below. In all figures, the three parameters that are shown on the top of the state transition arrow are transition probability, transition time duration, and transition energy consumption.



(a) Source Node State Type 1



b-1; 0
Pcoll/CWb
Tf
Etf
Pidle
σ
Eidle
b; c-1
b; c
b; c+1
c != CWb - 1
1-Pidle
(Psucc * Ts + Pfail * Tf)/(1 - Pidle)
(Psucc * Eovrs + Pfail * Eovrf)/(1 - Pidle)

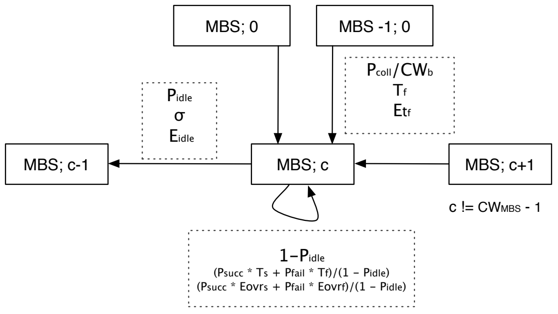
(b) Source Node State Type 2

Figure A.4: Source Node State Type 1 & 2

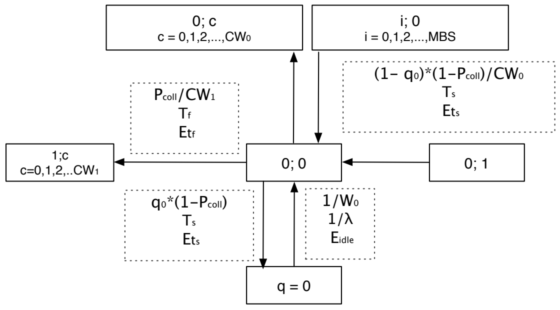
The type 1 source node state is shown in figure A.4.a), in which the backoff stage is zero and the backoff count is not zero. Since a source node will not accept packets in this state, the only action it can take is to decrement the backoff count while the medium is idle, for which the probability is P_{idle}. Otherwise, it will stay in the current state with probability $(1-P_{idle})$. Any state in which there is the possibility of a successful sent packet could transit to the current state with probability $(1-q_0)*(1-P_{coll})/CW_0$. The empty queue state could transit to the current state with probability $1/CW_0$.

The type 2 source node state is shown in figure A.4.b), in which the backoff stage is neither zero nor MBS and the backoff count is not zero. In this state, a node can only decrement the backoff count as it does in the type 1 state. Also, all states that are at backoff stage $(b-1)$ and where there is the possibility of a sent packet could transit to the current state with a probability P_{coll}/CW_b.

One important probability property the model needs to comply with is that the summation of all the next state transition probabilities of a state must be 1.



(a) Source Node State Type 3



(b) Source Node State Type 4

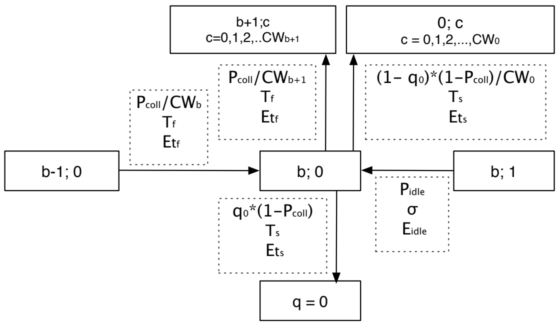
Figure A.5: Source node State Type 3 & 4

The type 3 source node state, in which the backoff stage is the MBS and the backoff count is not zero, is shown in figure A.5.a). Type 3 is the same as type 2 except that the node $(MBS; 0)$ could transit to the current state with probability P_{coll}/CW_{MBS}.

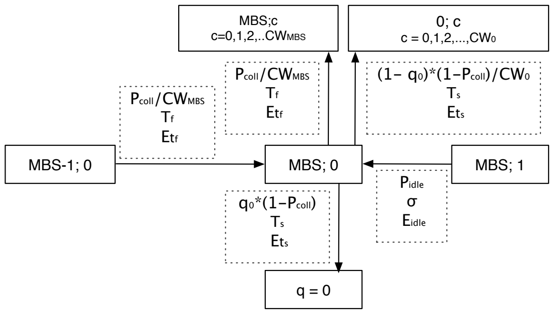
The type 4 source node state is shown in figure A.5.b). In this case, the backoff stage and backoff count are zero, which means that a node in this state will send out a packet. Depending on whether the packet is successfully sent or not, the node will jump into states $(0; c)$, c in the range $[0, CW_0]$, or states $(1, c)$, c in the range $[0, CW_1]$. Also, any state in which the node may send a packet could transit to the current state with a probability of $(1 - q_0) * (1 - P_{coll})/CW_0$.

The type 5 source node state is shown in figure A.6.a). The backoff count is zero, a node will send out a packet in this state then transit to states $(0; c)$, c in the range $[0, CW_0]$, or states $(b + 1, c)$, c in the range $[0, CW_(b + 1)]$. Also, the node could transit to the empty state with probability $q_0 * (1 - P_{coll})$.

The type 6 source node state, in which the backoff stage is the MBS and the backoff



(a) Source Node State Type 5



(b) Source Node State Type 6

Figure A.6: Source node State Type 5 & 6

count is zero, is shown in figure A.6.b). Type 6 is the same as type 5 except that a node in the current state could transit to the state $(MBS; c)$, c in the range $[0, CW_{MBS}]$, with probability P_{coll}/CW_{MBS}.

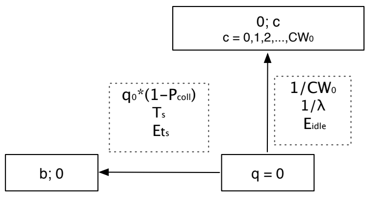


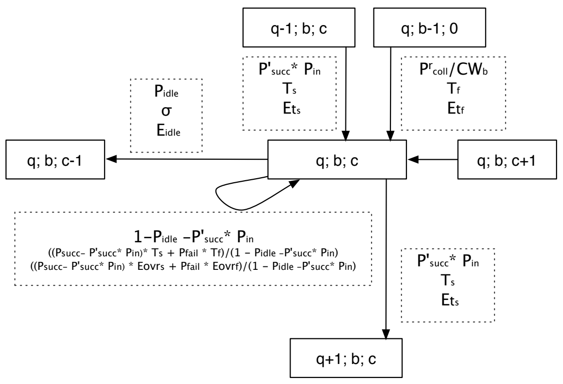
Figure A.7: Source Node State Type 7

The type 7 source node state is the empty queue state. From the empty queue state, a source node can only transit to the state $(0, c)$, c in the range $[0, CW_0]$. Any state in which the node may send a packet could transit to the empty queue state with probability $q_0 * (1 - P_{coll})$.

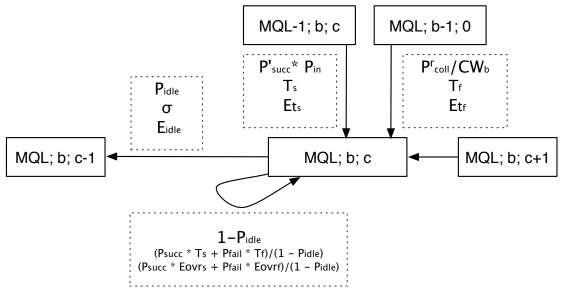
A.3 Relay Node Model Details

Relay nodes listen to the medium, get packets from the medium, then forward the packets. The number of packets a relay node receives depends on the upper layer routing protocol. In this mathematical model, we assume that the probability that a node will accept a successful packet is P_{in}, which means that this node accepts $(100 * P_{in})$ percent of the successful packets in the network.

All the states that a relay node goes through fall into ten types described below. Each state is represented as $(q; b; c)$, in which q is the queue length, b is the backoff stage, and c is the backoff count.


q-1; b; c
q; b-1; 0
P'succ* Pin
Ts
Ets
Prcoll/CWb
Tf
Etf
Pidle
σ
Eidle
q; b; c-1
q; b; c
q; b; c+1
1−Pidle −P'succ* Pin
((Psucc− P'succ* Pin)* Ts + Pfail * Tf)/(1 − Pidle −P'succ* Pin)
((Psucc− P'succ* Pin) * Eovrs + Pfail * Eovrf)/(1 − Pidle −P'succ* Pin)
P'succ* Pin
Ts
Ets
q+1; b; c

(a) Relay Node State Type 1



(b) Relay Node State Type 2

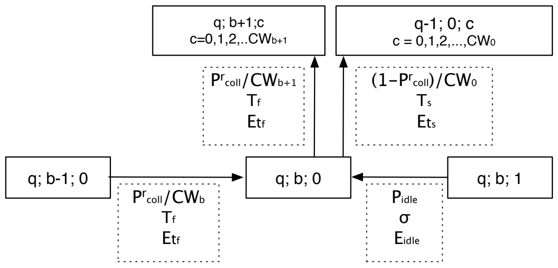
Figure A.8: Relay Node State Type 1 & 2

The type 1 state is shown in figure A.8.a). In this state, queue length q is neither 0 nor MQL, backoff stage b is neither 0 nor MBS, and the backoff count is not zero. Since queue length is not zero and the backoff count is not zero, a node in this state has packets to send but will not send while in this state. Several actions can lead the node from this state to others. First, it will decrement the backoff count if the medium is idle and go to state

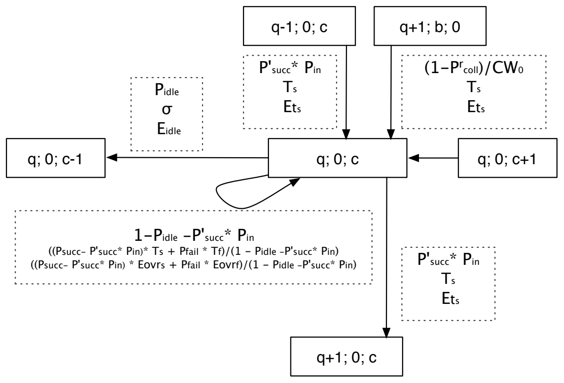
$(q; b; c-1)$, for which the probability is P_{idle}. Second, it may accept a successful packet sent by other nodes from the medium and go to state $(q+1; b; c)$, for which the probability is $P'_{succ} * P_{in}$. The last possible action is that the backoff counter will be suspended because the medium is not idle and the node will stay in the same state. We must keep in mind that the summation of outgoing transition probabilities must be 1. Accordingly, for the states that could transit to the current state, state $(q; b; c+1)$ could transit to the current state with probability P_{idle}, and state $(q-1; b; c)$ with probability $P'_{succ} * P_{in}$. If the previous state of a node was $(q; b-1; 0)$, the node would have sent out a packet. If the packet collides, then the exponential backoff mechanism will be invoked. The backoff stage will increment by 1, and a random backoff count will be chosen from zero to CW_b. So the node will transit from the previous state to the current state $(q; b; c)$ with probability P^r_{coll}/CW_b.

The type 2 state is shown in figure A.8.b). It is the same as the type 1 state except that the queue length is the MQL. Since the MAC layer queue size reaches its maximum, the node will not accept packets before there is room in the queue. So the node can not transit to state $(MQL+1; b; c)$ and the probability that the node stays in the current state changes to $(1 - P_{idle})$, the duration that the node stays in the current state and the energy spent in this state change as well.

The type 3 state is shown in figure A.9.a). In this state the queue length q is not zero, the backoff stage is neither zero nor MBS, and the backoff count is zero. As mentioned above, a node always sends out a packet when the counter reaches zero. Whether the sent packet collides or is successful, the node will enter a different state. If the packet collides, the queue size will not be changed, the backoff stage will be incremented to $b+1$, a random backoff count c within range CW_{b+1} will be chosen, and the node will transit to state $(q; b+1; i)$. If



(a) Relay Node State Type 3



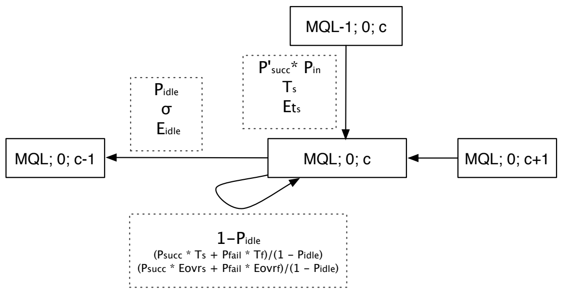
q-1; 0; c
q+1; b; 0
Pidle
σ
Eidle
P'succ* Pin
Ts
Ets
(1−Prcoll)/CW0
Ts
Ets
q; 0; c-1
q; 0; c
q; 0; c+1
1−Pidle −P'succ* Pin
((Psucc− P'succ* Pin)* Ts + Pfail * Tf)/(1 − Pidle −P'succ* Pin)
((Psucc− P'succ* Pin) * Eovrs + Pfail * Eovrf)/(1 − Pidle −P'succ* Pin)
P'succ* Pin
Ts
Ets
q+1; 0; c

(b) Relay Node State Type 4

Figure A.9: Relay Node State Type 3 & 4

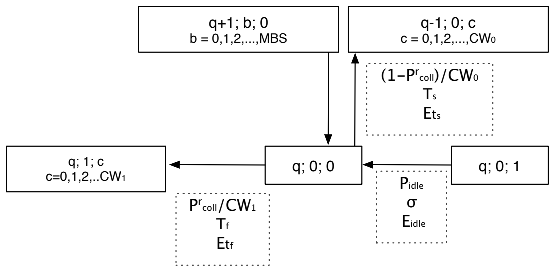
the packet does not collide, then the queue size will be decreased by 1, the backoff stage will be reset to zero, a random backoff count c within range CW_0 will be chosen, and the node will transit to state $(q-1; 0; i)$.

The type 4 state is shown in figure A.9.b). It is the same as the type 1 state except that the backoff stage is zero. Since every successful sending of a packet will reset the backoff stage no matter what it was, all states in which the node may send a successful packet could transit to the current state. Those states are $(q+1; b; 0)$ where b is any value with the range from zero to MBS.



MQL-1; 0; c
P'succ* Pin
Ts
Ets
Pidle
σ
Eidle
MQL; 0; c-1
MQL; 0; c
MQL; 0; c+1
1−Pidle
(Psucc * Ts + Pfail * Tf)/(1 − Pidle)
(Psucc * Eovrs + Pfail * Eovrf)/(1 − Pidle)

(a) Relay Node State Type 5

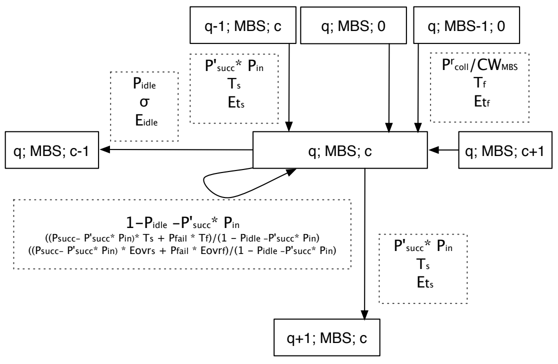


(b) Relay Node State Type 6

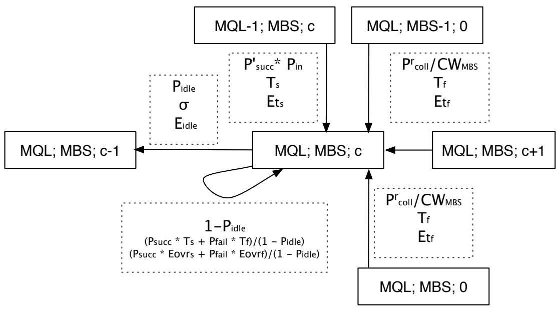
Figure A.10: Relay Node State Type 5 & 6

The type 5 state, in which the queue length is MQL and the backoff stage is zero, is shown in figure A.10.a). Since the node can not accept a successful packet, it can only transit to state $(MQL; 0; c-1)$ when the medium is idle. The states from which the node can enter the current state are state $(MQL-1; 0; c)$ with probability $P^{'}_{succ} * P_{in}$ and state $(MQL; 0; c+1)$ with probability P_{idle}.

Type 6 state is shown in figure A.10.b), in which both backoff stage and backoff count are zero and the queue size is not zero. A node in this state will either send out a packet successfully and get in states $(q-1; 0; c)$, c in the range $[0, CW_0]$, or send out a packet unsuccessfully and get in states $(q; 1; c)$, c within range $[0, CW_1]$. Also, any states that has the possibility to successfully send out a packet could transit to the current state. Those states are $(q+1; b; 0)$ with b in the range $[0, MBS]$.



(a) Relay Node State Type 7

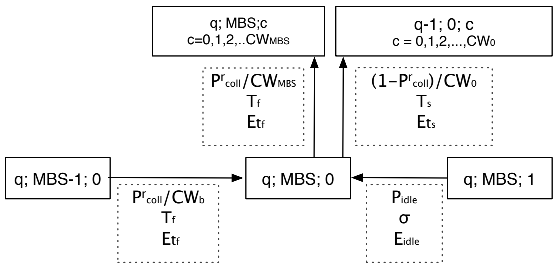


(b) Relay Node State Type 8

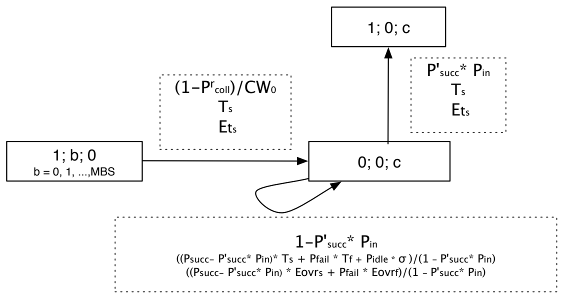
Figure A.11: Relay Node State Type 7 & 8

The type 7 state is shown in figure A.11.a). It is the same as the type 1 state except that the backoff stage is the MBS. This difference will make it possible for a node to transit to the current state from one more state than for the type 1 state. That is state $(q; MBS; 0)$, from which a node can transit to the current state with probability P^{r}_{coll}/CW_{MBS} when a sent packet collides.

The type 8 state is shown in figure A.11.b). It is similar to the type 5 state except that the backoff stage is MBS instead of 0. In this state, a node will not be able to accept a packet, so the states it could transit to are state $(MQL; MBS; c - 1)$ and itself. Also, if a node that was in state $(MQL; MBS; 0)$ sends out a packet that collides, the backoff stage can not be incremented by 1. It will still stay at stage MBS but possibly with a different backoff count. That means state $(MQL; MBS; 0)$ has probability P^r_{coll}/CW_{MBS} to transit to the current state.



(a) Relay Node State Type 9


1; 0; c
(1-Prcoll)/CW0
Ts
Ets
P'succ* Pin
Ts
Ets
1; b; 0
b = 0, 1, ...,MBS
0; 0; c
1-P'succ* Pin
((Psucc- P'succ* Pin)* Ts + Pfail * Tf + Pidle * σ)/(1 - P'succ* Pin)
((Psucc- P'succ* Pin) * Eovrs + Pfail * Eovrf)/(1 - P'succ* Pin)

(b) Relay Node State Type 10

Figure A.12: Relay Node State Type 9 & 10

The type 9 state is shown in figure A.12.a). It is similar to the type 3 state except that the backoff stage is MBS. The only difference caused by this is that when a node in this state sends out a packet that collides, it will stay at stage MBS with a possibly different backoff count.

The type 10 state is shown in figure A.12.b). In this state, a node does not have any packets in the queue, so it will stay in the current state until one packet enters. If one packet enters, the node will transit to state $(1; 0; c)$. Also, the states from which a node can enter the current state are states $(1; b; 0)$ with b in the range $[0, MBS]$.

The last state type is type 11, $(0; b; c)$, in which b is not zero. This is an impossible state. The frequency probability and time proportion of this state are always zero.