A NEW MULTI-CHANNEL MAC PROTOCOL FOR WIRELESS AD HOC NETWORKS WITH SINGLE TRANSCEIVER

A Thesis by

Musa Aykut Canbolat

Bachelor of Science,

Middle East Technical University, Turkey, 2005

Submitted to The Department of Electrical and Computer Engineering and the faculty of Graduate School of Wichita State University in partial fulfillment of the requirements for the degree of Master of Science

July 2007
A NEW MULTI-CHANNEL MAC PROTOCOL FOR WIRELESS AD HOC NETWORKS WITH SINGLE TRANSCEIVER

I have examined the final copy of this thesis for form and content, and recommend that it be accepted in partial fulfillment of the requirement for the degree of Master of Science with a major in Electrical and Computer Engineering.

______________________________
Coskun Cetinkaya, Committee Chair

We have read this Thesis
and recommend its acceptance:

______________________________
Hyuck Kwon, Committee Member

______________________________
Mehmet Barut, Committee Member
DEDICATION

To,
my mother Döndü Hatun
my father Fahrettin
and my sister Hanife Hilal
"Before you criticize someone, walk a mile in his shoes. That way, when you criticize him, he'll be a mile away...barefoot."

Billy Connolly
ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Coskun Cetinkaya for his support and guidance throughout my studies at Wichita State University. I would also like to thank my family; my father Fahrettin, my mother Döndü and my sister Hilal for always being there for me. Without their support, this wouldn’t have been possible. I would also like to extend my gratitude to members of my committee Dr. Hyuck Kwon and Dr. Mehmet Barut. Thanks are also due to my neighbors Ali Ersin and Atilla and all my friends. Thank you all very much for being there whenever I needed.
ABSTRACT

Although IEEE 802.11a/b/g standards allow use of multiple channels, only a single channel is popularly used, due to the lack of efficient protocols that enable use of Multiple Channels. There are some papers challenging this problem. Some of them have requirements that will increase the cost, like requirement of multiple transceivers. Some others address the problem with single transceivers, but are very hard to be employed in highly mobile Ad Hoc networks due to network-wide synchronization requirements. In this Thesis, multiple channel use in a wireless network with single transceiver nodes is addressed, and attempted to be solved with a new efficient Ad Hoc network MAC protocol, which intends to remove the requirement of network-wide synchronization.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>INTRODUCTION</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>1.1. Overview</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1.2. IEEE 802.11 Standard’s Multi-Channel Capabilities</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1.3. Organization of Thesis</td>
<td>2</td>
</tr>
<tr>
<td>2.</td>
<td>LITERATURE REVIEW</td>
<td>3</td>
</tr>
<tr>
<td>3.</td>
<td>MULTI-HOP AD HOC NETWORKS IN MULTI-CHANNEL ENVIRONMENTS</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>3.1. Hidden Terminal Problem</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>3.2. Handling Hidden Terminal Problem: Four-Way Handshake</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>3.3. Multi-Channel Hidden Terminal problem</td>
<td>9</td>
</tr>
<tr>
<td>4.</td>
<td>PROPOSED MULTI-CHANNEL MAC PROTOCOL</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>4.1. Overview</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>4.2. Search Process</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>4.2.1. Channel Switching Patterns for nodes</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>4.2.2. Channel Switching Frequency</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>4.3. Scheduling Process</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>4.4. Assumptions and Requirements</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>4.4.1. Calculation of $T_S$ - Worst Case Scenarios</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>4.4.1.1. Wireless LAN Setting</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>4.4.1.2. Multi-Hop Setting</td>
<td>20</td>
</tr>
<tr>
<td>5.</td>
<td>EXPERIMENTAL JUSTIFICATION</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>5.1. Light Load Conditions</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>5.2. Heavy Load Conditions</td>
<td>29</td>
</tr>
<tr>
<td>6.</td>
<td>CONCLUSION</td>
<td>33</td>
</tr>
<tr>
<td>7.</td>
<td>FUTURE WORK</td>
<td>34</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>1.</td>
<td>Dynamic Channel Allocation with Power Control</td>
<td>3</td>
</tr>
<tr>
<td>2.</td>
<td>MMAC protocol</td>
<td>5</td>
</tr>
<tr>
<td>3.</td>
<td>Hidden Terminal Collision</td>
<td>7</td>
</tr>
<tr>
<td>4.</td>
<td>Four way hand-shake mechanism</td>
<td>8</td>
</tr>
<tr>
<td>5.</td>
<td>Multi-Channel Hidden Terminal problem</td>
<td>9</td>
</tr>
<tr>
<td>6.</td>
<td>Channel switching for data and searching nodes</td>
<td>13</td>
</tr>
<tr>
<td>7.</td>
<td>Repeating channel switching patterns for second round of search</td>
<td>14</td>
</tr>
<tr>
<td>8.</td>
<td>Changing channel switching pattern for second round of search</td>
<td>15</td>
</tr>
<tr>
<td>9.</td>
<td>Multiple hidden terminal transmissions</td>
<td>22</td>
</tr>
<tr>
<td>10.</td>
<td>Light load throughput</td>
<td>28</td>
</tr>
<tr>
<td>11.</td>
<td>Throughput for different number of RTSF attempts</td>
<td>29</td>
</tr>
<tr>
<td>12.</td>
<td>Throughput for different contention window sizes</td>
<td>30</td>
</tr>
<tr>
<td>13.</td>
<td>Average search time vs. number of flows</td>
<td>31</td>
</tr>
<tr>
<td>14.</td>
<td>Wireless LAN vs Multi hop throughputs</td>
<td>32</td>
</tr>
</tbody>
</table>
CHAPTER 1

INTRODUCTION

1.1. Overview

In today’s society, computer networks have a rapidly increasing importance due to the need for voice, image, video, document or more generally data transfer. Everyday costumers demand better service quality, more flexible mobility and portability, and higher cost efficiency. Especially in the area of wireless networks, there is a fast development, because the demand is increasing in the direction of higher mobility, not only for personal use, but also for military use. Especially Ad Hoc networks, in which clients can communicate among each other without an access point, are becoming more and more popular for data communications.

1.2. IEEE 802.11 Standard’s Multi-Channel Capabilities

IEEE 802.11 protocols are established to set a standard for wireless communications. Current popular use of these standards require single channel operation for simplicity, however they basically support multi-channel capabilities. For example physical layer of IEEE 802.11b standard has 14 channels, which are 5 MHz apart from each other. To be completely overlapping-free there must be a spacing of at least 30 MHz. Thus, 3 of these 14 channels can be used in a multi-channel operation. [1]
Single channel operation does not combine the channels together, instead uses just one of them. This means the resources that are actually available are not being used. To overcome this issue, powerful MAC protocols are needed to operate in multi-channel environments.

1.3. Organization of Thesis

In this Thesis work, first a few papers are reviewed to show what state-of-art is in the attempted problem. Then some background information about the proposed work is given. Later, proposed protocol, especially the behavior of it in different situations is explained in detail, necessary calculations are made. Finally there is an experimental justification part that empirically determines some network parameters, reveals the performance of the proposed network, and compares it to the protocols that have popular commercial use.
CHAPTER 2

LITERATURE REVIEW

As briefly mentioned in the introduction part, there are several recent papers proposing new protocols to operate in a multi-channel environment [2-5, 8-14]. In the Dynamic Channel Allocation with Power Control (DCA-PC) protocol [2], Wu et al. assumes nodes have two transceivers, one of which operates only on a dedicated control channel. According to the control messages a channel is assigned for transmission in an ‘on demand style’, and the other transceiver enables data transfer according to the channel assignment schedule.

Although this protocol does not require network wide synchronization, it has two major drawbacks. First of all it requires two transceivers that makes it impossible to be applied to IEEE 802.11 standards with only a basic software change. Another drawback is that it increases the overhead, because it assigns one of the channels only for control information. Especially when the number of channels is low, this overhead is very

Figure 1. Dynamic Channel Allocation with Power Control
significant, and degrades the performance of the network significantly. Another similar approach that uses two transceivers and one dedicated control channel is made by Wing et al. [3]

Another MAC protocol (MMAC) was proposed by So and Vaidya [4] that uses a single transceiver for Multi-Channel Ad Hoc Networks. In their work, they divide the time into beacon intervals, and they split these beacon intervals into two phases; a control phase (which is called ATIM period in reference to IEEE 802.11 power saving mechanism) and data exchange phase. During the control phase, all nodes in the network switch to a pre-selected channel in order to exchange control information. Using an RTS/CTS mechanism, nodes make an agreement on channels that they are going to use for data exchange. In the data exchange phase, every node switches to its assigned channel, and transfers its data. Authors also employ a power saving mechanism in their protocol; when a node does not have any data to send or receive after ATIM period, it does not switch to a different channel but enters the idle mode until it hears a beacon signal which implies another beacon interval starting. The idle node hearing beacon signal wakes up to exchange control information with other nodes.
Although this protocol removes the requirement of multiple transceivers, it places a major overhead to the system; during the ATIM period, since all nodes in the network switch to a common channel, there is no data transmission, and the bandwidth of unused channels is wasted. Moreover, if the network is highly congested, the exchange of control information will take excessive time, and a longer ATIM period will be required. In other words, the size of the ATIM window must be changed dynamically. MMAC does not offer a solution to this problem. Even in the case of dynamic variation of ATIM window, it would have left a very little time for data transfer in case of crowded networks. Another problem with this protocol is, it requires network-wide synchronization for beacon intervals and ATIM periods. In a highly mobile Ad hoc Network, this synchronization is extremely hard to achieve due to the dynamic behavior of the network.

Bahl et al. proposed another protocol called Slotted Seeded Channel Hoping (SSCH) [5] in which nodes continuously switch channels using pseudorandom seeds. In
other words, every node has a number of patterns to switch through channels, via which load balancing is obtained. When a node wants to communicate with another one frequently, it acquires one or more of the target node’s seeds and follows target’s pattern of channel switch, thus they happen to be in the same channel frequently. SSCH requires each node to broadcast a packet informing other nodes in the channel from its presence in the channel. Since nodes change channel continuously, these information broadcasts will place a major overhead in the system, especially for networks with high number of nodes. Another drawback is that network-wide synchronization is needed for channel switch. When a node informs the nodes in a channel of its presence, every possible node in the channel should have heard of its presence to be able to discover if it has to schedule a communication.
CHAPTER 3

MULTI-HOP AD HOC NETWORKS IN MULTI-CHANNEL ENVIRONMENTS

3.1. Hidden Terminal Problem

Hidden terminal problem is one of the challenges that Multi-hop wireless networks face.

![Diagram of hidden terminal collision](image)

Figure 3. Hidden Terminal Collision

To visualize this problem, assume that nodes A, B, and C in figure 3 stand on a line where A and B are in one-hop distance to each other so as B and C. This means C and A can not communicate with each other directly. When A is sending packets to B, C, being unaware of this communication, may try to communicate with B. Since B can hear both A and C, but not at the same time, the packets sent from each of them collide.
3.2. Handling Hidden Terminal Problem: Four-Way Handshake

To overcome the hidden terminal problem, IEEE 802.11 standards have a feature called four way handshake.

![Diagram of Four-Way Handshake]

Figure 4. Four way hand-shake mechanism

As seen in figure 4 when A wants to communicate with B, it first sends a Request-to-send (RTS) packet. Upon the receipt of this RTS, B sends back a Clear-to-send (CTS) packet. Since node C hears this CTS message, it defers its possible transmission until the end of the communication between A and B ends. Since B sends an Acknowledgement packet (ACK) at the end of the data packet, C ends its deferral of communication.
3.3. Multi-Channel Hidden Terminal problem

In a multi-channel environment, the nodes in the network are free to switch their channels in order to be able to communicate with every possible node in the network. Even if the efficient RTS/CTS mechanism is used in a multi-channel network, it might not handle the hidden terminal problem, because a node can come into the channel after RTS/CTS transmission, but before the end of DATA transmission.

As seen in figure 5, nodes A and B first schedule a data transmission an RTS/CTS mechanism on Channel-1 while node C is on Channel-2. Right after the DATA transmission starts between A and B, node C switches to Channel-1. Even if it listens to the channel first, it can not hear A sending its data to B because it is not in one hop distance to A. Assuming Channel-1 is free for transmission, C sends an RTS packet, which causes collision at node B.

Figure 5. Multi-Channel Hidden Terminal problem

As seen in figure 5, nodes A and B first schedule a data transmission an RTS/CTS mechanism on Channel-1 while node C is on Channel-2. Right after the DATA transmission starts between A and B, node C switches to Channel-1. Even if it listens to the channel first, it can not hear A sending its data to B because it is not in one hop distance to A. Assuming Channel-1 is free for transmission, C sends an RTS packet, which causes collision at node B.
The proposed protocol solves this problem, by waiting for a duration of a maximum data packet transmission. This feature will be explained in detail in the later parts of this work.
CHAPTER 4

PROPOSED MULTI-CHANNEL MAC PROTOCOL

4.1. Overview

As explained in the Literature Review part in detail, existing Multi-Channel MAC protocols that can work with nodes having single transceivers require network-wide synchronization. In this Thesis, a new and efficient MAC protocol is proposed that does not have a network-wide synchronization requirement, and can work with single transceiver nodes. The common property of existing protocols that yield network-wide synchronization requirement is they combine the search and scheduling processes. However, in this protocol, search process is separated from scheduling. In other words, a node searches first for its destination node, and after it finds, it schedules its transmission via an RTS/CTS mechanism.

In the protocol, the nodes are grouped in two. First group is the searching nodes, which have data to transmit to a receiver whose channel schedule is unknown. The second group is data nodes, which are currently in regular data transmission or in idle mode.
4.2. Search Process

4.2.1. Channel Switching Patterns for nodes

In the later parts of this work, it is assumed that the number of channels available is \( N \), the searching node has a channel switching pattern defined by vector

\[
\text{CSP}_S = [\text{CHS}_1, \text{CHS}_2, \ldots, \text{CHS}_N],
\]

and the destination node has a switching pattern defined by vector

\[
\text{CSP}_D = [\text{CHD}_1, \text{CHD}_2, \ldots, \text{CHD}_N]
\]

where neither \( \text{CHS}_1 \) nor \( \text{CHD}_1 \) is necessarily the channel number 1, but it’s the first channel in corresponding nodes channel switching pattern.

This random assignment of channel switching pattern is needed for all nodes, because of load balancing issues. If all nodes in the network followed a predefined pattern to switch channels, there would be accumulation of nodes, and thus accumulation of transmission, at some particular channels.

All the nodes update their channel switching patterns after each completion, and select another random channel sequence. This feature will enforce the load balancing skills of the proposed protocol. It will also supply additional benefits which will be explained in later parts of this work.
We say the time that a searching node spends in a channel is $T_S$, where the time a data node spends in a channel is $T_D$.

### 4.2.2. Channel Switching Frequency

According to the proposed protocol, all nodes switch their channels asynchronously. However, the searching nodes must switch through channels at a much faster rate, in order to be able to find its destination. This conclusion follows from the fact that a searching node can find its destination if it visits all the channels before destination node changes its channel. But since the searching node has no idea about when the destination node made its last channel switch, it is a clear challenge.

![Diagram showing channel switching for data and searching nodes](image)

**Figure 6. Channel switching for data and searching nodes**

In figure 6, if $\text{CHD}_1=\text{CHS}_2$, and $\text{CHD}_2=\text{CHS}_1$, no matter how long the destination node stays at $\text{CHD}_2$, the searching node can not find it in one round of visit to all nodes. If we make two rounds of visits to all nodes in the channel, and put an upper limit to the frequency of channel switch for data nodes, we can guarantee a successful search process. If the destination node switches to a previously visited channel during searching
node’s first round of visit, the searching node will hit the destination node in its second round of visit.

For a worst case scenario, we can assume in figure 7 that CHD_1=CHS_2, and CHD_2=CHS_1. Since the destination node switched to a previously visited node, searching node can not make a hit in its first round. In the second round, if destination node stays long enough in its new channel, searching node will make a hit.

![Channel Switching Patterns](image)

Figure 7. Repeating channel switching patterns for second round of search

If we use the same channel sequence in both two rounds of searching, CHD_2=CHS_1 scenario search would be successful at t = (N+1) \cdot T_S, and the lower bound for T_D would be T_D=NT_S. However, when the searching node finishes its channel sequence once in a search process, we require it to determine another channel switching pattern randomly. The reason for this is, in case the searching node collides with another searching node in the first round of both; they will collide in their second round too. And if the destination node is in the channel that collisions occurred, search process will fail. This situation has a very small probability of occurrence, but determining a new channel sequence is not a big deal, and we do not want to reduce our success rate in searching.
When the channel sequence is updated for second round of visits, CHD$_2$ can be immediately equal to CHS$_1$’, or away at CHS$_N$’. Since we are dealing with the worst case scenarios, assume CHD$_2$=CHS$_N$’, which will yield the highest search time. In such a case, at t=2NT$_S$, search will succeed. The lower limit for T$_D$ will be (2N-1) · T$_S$. However because searching node and destination node are not synchronous at all, we set T$_D$=2NT$_S$.

4.3. Scheduling Process

In this protocol, when a searching node finds its destination, it schedules a transmission with that destination using the regular IEEE 802.11 MAC protocol’s RTS/CTS mechanism. Also searching nodes look for their destinations in each channel using RTS/CTS, with some little difference. To assign priority of access the channel to the searching nodes, they wait only for a Priority Inter-Frame Spacing (PIFS) amount of time before they access the channel. This PIFS is shorter than Distributed Inter-Frame Spacing (DIFS) that data nodes wait before they send their RTS. We also call the RTS/CTS messages used by searching nodes RTSF/CTSF (request-to-send-
find/confirmation-to-send-find) to distinguish it from regular data scheduling. The RTSF/CTSF messages include the channel switching pattern and time-clock of the corresponding sender, which will enable scheduling when destination node is found.

4.4. Assumptions and Requirements

1- A searching node changes its channel after consecutive unresponded RTSF messages.

In a multi-hop network, there is no guarantee that the destination node is in a one-hop distance. Since this protocol is designed to be used in multi-hop networks as well as single-hop networks, there is a chance that intended receiver is not reachable directly to the searching node. Due to this possibility of an unreachable receiver, we limit the number of attempts in a channel to a certain number. The number of attempts will be determined experimentally in later parts of this work.

2- A searching node changes its channel if it detects an RTSF message of another searching node.

If there are two searching nodes in a channel (and in each others range) there will be lots of collisions, because they both have the same priority to access the channel. This requirement is necessary to prevent such a situation as much as possible. The searching node which heard the RTSF message of the other node switches to the next channel of its corresponding channel switching pattern to allow the other node to make its search properly.
3. When a searching node detects a collision before its first RTSF, it waits a DIFS amount of time instead of a PIFS.

When a searching node detects a collision, it can not immediately conclude if any searching node is involved in this collision or not. In case the collision includes an RTSF packet of another searching node, the searching node should be able to know this to allow that colliding node to finish its search. When we wait a DIFS amount of time, the searching node involved in the collision will send its RTSF again before we send ours. Since we hear another RTSF before our first RTSF transmission, according to requirement-2 we change our channel. If the collision did not involve any searching nodes, we will be able to access the channel before the regular data nodes.

4. A searching node switches its channel if it detects a collision after its first transmission attempt.

The node knows that there are no hidden terminal transmissions (see requirement-6) or no collisions at the time that it sends its first RTSF. The assumption is based on the fact that since the channel is idle (and hidden terminal free) only for a PIFS amount of time, and during this time the channel could only be accessed by another searching node due to priority issues. A possible exception to this assumption could be the collision between our RTSF and CTSF of another successful search where the other searching node is at hidden terminal. But assumption still holds because the exceptions have very little probability.
5- A searching node switches its channel if it detects consecutive collisions.

The node assumes the consecutive collisions are between itself and another searching node, because seemingly the colliding nodes try to access the channel at same period. Although the collisions might have occurred because of different data nodes which don’t have any direct links between each other, this is a very reasonable assumption, because the mentioned exceptions are very rare.

6- If the searching node finds channel idle right after switching, it waits transmission duration of a maximum sized data packet before accessing the channel.

The idle channel can be sensed due to no transmission as well as a hidden terminal transmission, where receiver is in a single-hop distance. Since our RTSF signal will collide at the receiver of data transmission, we wait until the end of that data transmission. The searching node may hear an ACK signal pointing out the end of the data transmission at any time during this wait. However we do not send RTSF signal immediately after ACK because of the probability that there could have been multiple hidden terminal transmissions. If protocol is being used in a wireless LAN setting, i.e. there are no hidden terminals, this requirement is not needed.
4.4.1. Calculation of $T_S$: Worst Case Scenarios

According to this protocol, a searching node switches across the channels in order to find its destination to enable data transmission. The highest time that a searching node spends in a channel is needed, because the channel switching frequency of data nodes will be determined accordingly as explained above. Note that requirement-6 is not needed in a single-hop network, so we calculate worst case $T_S$ for two cases; a wireless LAN setting, and a multi-hop network setting. No matter what the setting is, the node switches the channels dynamically, and recent studies show that this switching delay $T_{sw}$ is approximately 80µsec[6,7]. Next we need to determine the access time $T_W$ and $T_M$, the access times for Wireless LAN and Multi-Hop settings, which results from the protocol requirements mentioned above.

4.4.1.1. Wireless LAN Setting

Since we are searching the worst case channel access time, $T_W$, we look at different cases.

1- *When the node switches into the channel, it finds channel busy due to a collision.*

In this case, the node waits for the collision to be cleared. This is first the RTS is sent, collision occurs and it takes Extended Inter-Frame Spacing (EIFS) time for channel to be cleared. Then node waits a DIFS amount of time according to requirement-3. Thus, in the worst case it takes:

$$T_{W1} = \text{RTS} + \text{EIFS} + \text{DIFS}$$
2- *When the channel is busy due to an ongoing transmission.*

In this case the searching node will wait until the end of the ongoing transmission. Since searching node waits only a PIFS amount of time to access the channel, it will have the priority, and get the channel. In the worst case node enters the channel while the data transmission is starting, and a maximum sized data packet is intended for transmission. Accessing the channel takes:

\[ T_{W2} = \text{RTS} + \text{SIFS} + \text{CTS} + \text{SIFS} + \text{DATA}_{\text{MAX}} + \text{SIFS} + \text{ACK} + \text{PIFS} \]

It can easily be seen from the equations that the latter is a worse case, and in fact the worst case scenario in a Wireless LAN setting.

### 4.4.1.2. Multi-Hop Setting

Recall that the last requirement of the protocol was only applicable in a multi-hop setting. Therefore the worst case searching time will be different from a Wireless LAN setting. According to requirement-6, in a multi-hop network, when a searching node enters a channel, and finds it idle, it should wait to make sure there is no hidden terminal data transmission at that moment. The amount of time that searching node should wait is

\[ T_1 = \text{DATA}_{\text{MAX}} + \text{SIFS}. \]
During this time, if there is a hidden terminal transmission, it should hear an ACK packet if it has not already heard the CTS of corresponding transmission. In case it does not hear any control information, it continues its searching process by sending an RTSF packet.

1- When there is a real hidden terminal transmission.

The node will hear an ACK packet during $T_I$ (as the worst case at the end of $T_I$), and wait for a PIFS amount of time to access the channel. Therefore;

$$T_{M1} = T_I + \text{ACK} + \text{PIFS} = \text{DATA}_{MAX} + \text{SIFS} + \text{ACK} + \text{PIFS}$$

2- If there was no hidden terminal transmission when channel was changed, but during $T_I$ a transmission has started.

Since searching node always waits to see whether there is a hidden terminal transmission or not when it enters the channel, there is always an opportunity for a data node to start a new transmission. When such a situation occurs, the searching node waits for the newly started data transmission to be completed. Therefore the waiting time before searching node accesses the channel is;

$$T_{M2} = T_I + \text{RTS} + \text{SIFS} + \text{CTS} + \text{SIFS} + \text{DATA}_{MAX} + \text{SIFS} + \text{ACK} + \text{PIFS}$$

$$= \text{RTS} + \text{CTS} + \text{ACK} + 2 \cdot \text{DATA}_{MAX} + 4 \cdot \text{SIFS} + \text{PIFS}$$
Of course another transmission may start during this TM2 period that is hidden to the ongoing transmission, but hearable to the searching node. Such a situation is visualized in the figure 9.

![Figure 9. Multiple hidden terminal transmissions](image)

In the figure consecutive nodes are assumed to be in one hop distance to each other. While node S (the searching node) was waiting for the data transmission between A and B to be completed, C and D may start a new transmission, that will either be heard by S, or will be a hidden terminal transmission to S. If node S sends an RTSF message after waiting $T_{M2}$ amount of time, there will be collision, and S will have to change its channel according to the protocol requirements. On the other hand if we require S to wait for the communication between C and D completed, A and B may start a new transmission upon completion of ongoing transmission as well.

This deadlock can be solved by simply switching channels. Because if we wait for all transmissions cleared, it may take excessive time for a search process. Or if we send an RTSF message ignoring the transmission between C and D, we will hear a collision which according to requirement-4 will lead to an immediate channel switch anyway.
As a result of the protocol requirements and calculations above, $T_{W2}$ is the time that a node will wait before accessing the channel in a Wireless LAN setting. Likewise $T_{M2}$ is the time that a searching node will wait before accessing the channel in a multi-hop setting.

Recall that our ultimate goal in this section was to determine the highest time that a node spends in a channel. Besides channel switch time and waiting time, node will spend some time accessing the channel, in other words searching for the node. Next we will determine this time for the worst case, and finalize our calculation for $T_S$.

After the searching node waits for the necessary amount of time, it will send its first RTSF message to find its corresponding receiver. If the intended receiver is in the same channel, and receives RTSF signal correctly, it will respond with a CTSF. In such a case, the searching node sends an ACK message to finalize the search process. The exchanged RTSF/CTSF signals include the channel sequence and time clock, so that they can schedule their next transmission.

There might be a couple of reasons if the searching node does not receive a CTSF response, like a possible collision, the absence of the intended receiver, or silence of the intended receiver due to an ongoing hidden terminal transmission. Recall that this hidden terminal transmission does not necessarily include the intended receiver, but any transmission that it can hear. There is no way to determine the reason for the searching node because it has a single transceiver, and can not listen and send at the same time. For
this reason, if searching node does not receive a CTSF response, it will send more RTSF messages, the number of which will be determined experimentally.

If RTSF message of a searching node is left unresponded, the protocol assumes the situation is due to a collision, and generates a random back off counter between 0 and $CW_S - 1$, where $CW_S$ stands for contention window size for a searching node. The back off counter is decremented for every time slot that the channel remains idle. During this back off time, if the node hears a collision or RTSF signal of another node, it changes channels immediately according to the requirement-4. After back off counter vanishes it waits for PIFS amount of time and attempt for an RTFS signal again.

According to the protocol, a searching node must always have the priority to access the channel. To satisfy this even in the back off situations, we set;

$$DIFS = PIFS + CW_S \cdot st$$

where $st$ is slot time. This way even when the searching node is backing off, it is not giving the channel access priority to data nodes, but other possible searching nodes.

Let $T_F$ be the time lost in an unsuccessful RTSF attempt. In such a case, the node will send the RTSF message, wait for EIFS amount of time for channel to be cleared, back off for $CW_S - 1$ time slots, and finally wait PIFS amount of time to access the channel. Thus;
\( T_F = \text{RTSF} + \text{EIFS} + (\text{CW}_S - 1) \cdot \text{st} + \text{EIFS} \)

Recall that we are looking for the worst case times, thus it will take EIFS amount of time for channel to be cleared and node will back off for the maximum allowed time \((\text{CWS} - 1) \cdot \text{st}\). Except for the last attempt, a node will spend a maximum \(T_F\) amount of time for every failed RTSF attempt. For the last attempt, it switches channels immediately after EIFS amount of time, the time that is necessary for the channel to be cleared.

Notice that when the absence of a CTSF response is due to silence of the intended receiver, and if the intended receiver received the RTSF message correctly, it adopts the searching nodes time clock and channel sequence, so that it can meet the searching node at a later time.

To finalize our calculations for \(T_S\) we add all the worst case times for Wireless LAN and multi-hop scenarios. For wireless LAN scenario:

\[
T_{S,\text{LAN}} = T_{sw} + T_{W2} + (L-1) \cdot T_F + \text{RTSF} + \text{EIFS} \\
= T_{sw} + \text{RTS} + \text{CTS} + \text{DATA}_{\text{MAX}} + 3 \cdot \text{SIFS} + \text{ACK} + \text{EIFS} \\
+ (L-1) \cdot [\text{RTSF} + \text{EIFS} + (\text{CW}_S - 1) \cdot \text{st} + \text{EIFS}] \\
+ \text{RTSF} + \text{EIFS} \\
= T_{sw} + \text{DATA}_{\text{MAX}} + \text{RTS} + \text{CTS} + \text{ACK} + 3 \cdot \text{SIFS} \\
+ L \cdot (\text{EIFS} + \text{RTSF} + \text{EIFS}) + (L-1) \cdot (\text{CW}_S - 1) \cdot \text{st}
\]
where L stands for number of RTSF attempts before switching the channel. Similarly for a multi-hop scenario;

\[ T_{S,M} = T_{sw} + T_{M2} + (L-1) \cdot T_F + RTSF + EIFS \]

\[ = T_{sw} \]

\[ + RTS + CTS + ACK + 2 \cdot DATA_{MAX} + 4 \cdot SIFS + PIFS \]

\[ + (L-1) \cdot [RTSF + EIFS + (CW_S - 1) \cdot st + PIFS] \]

\[ + RTSF + EIFS \]

\[ = T_{sw} + 2 \cdot DATA_{MAX} + RTS + CTS + ACK + 4 \cdot SIFS \]

\[ + L \cdot (PIFS + RTSF + EIFS) + (L-1) \cdot (CW_S - 1) \cdot st \]

Recall that the switching time for the data nodes was determined simply by formulas:

\[ T_{D,LAN} = 2 \cdot N \cdot T_{S,LAN} \text{ and } T_{D,M} = 2 \cdot N \cdot T_{S,M} \]

where N is the number of channels.

When the search algorithm is completed but the corresponding node was unable to be located, the searching node repeats the search process up to 7 times. The searching node starts the search process again immediately with probability \( \frac{1}{2} \), or selects a random back off counter and starts after back off is complete.
CHAPTER 5

EXPERIMENTAL JUSTIFICATION

Experiments for the proposed algorithms are performed in an event driven simulation which uses the DSSS specification of IEEE 802.11a with 13 channels each of which have 54 Mbps capacity and 512 byte packet size. For simulations, a Wireless LAN setting is used, whereas multi-hop evaluations are left for future work. For simplicity, it is assumed that nodes have disjoint flows, in other words each flow has a unique sender receiver pair.

There are two sets of experiments performed in this work. First one is heavy load conditions in which all the nodes that have packets to send start searching their corresponding receivers at the same time. This experiment will allow us to determine the behavior of the protocol when large number of searches are being made, which will be the case for highly dynamic networks. Also the effect of number of RTSF attempts and size of contention window will be determined under high load conditions to optimize network performance. In light load conditions experiment, the average number of flows that happen at the same time is fixed at some value. However, the flows start at different times, which will distribute the concentration of high priority RTFS/CTSF messages over time.
5.1. Light Load Conditions

First of all, light load condition experiments are performed to observe the network throughput, and compare it to the conventional single channel IEEE 802.11a throughput. The IEEE 802.11a protocol has 54 Mbps capacity in a single channel, and gives a throughput of about 11.3 Mbps [1].

![Light load throughput graph](image)

**Figure 10. Light load throughput**

As seen in figure 10, as the number of flows increase, MC MAC protocol outperforms the single channel IEEE 802.11a protocol. Note that the first data point for MC MAC was taken with a single flow, and because of the searching process, throughput degrades and MC MAC’s throughput does not reach IEEE 802.11a’s.
5.2. Heavy Load Conditions

To determine the vacant parameters we use a scenario that has heavy load conditions. The maximum number of nodes is set to 650 which is set at much lower numbers in similar works [4,5]. Recall that the parameter ‘L’ was previously assigned to the maximum number of RTSF attempts before changing the channel, and CWS stood for the contention window size for searching nodes.

Figure 11. Throughput for different number of RTSF attempts

Figure 11 shows throughput graphs for different values of L. Although there is not much difference, the highest throughput is given by L=2 by a slight difference as the network gets more crowded. This result is expected, because when number of attempts
for RTSF increase, search process takes more time, which will degrade the throughput. The reason for multiple RTSF attempts was to reduce the number of unsuccessful searches, but according to the experiment results no unsuccessful searches occurred. The main reason for the results being very close to each other is, when $L$ is low, sometimes the search process fails at first trial. Since we repeat the search up to seven times, the time gained at less RTSF attempts is lost at repeated search processes. Thus we set the number of RTSF attempts before switching channels to $L = 3$ as a moderate value.

![Throughput for different contention window sizes](image)

Figure 12. Throughput for different contention window sizes

In Figure 12 we see three different throughput graphs for different contention window size values. The highest throughput is yielded by $CW_S=3$. This result is expected too, because when a collision is detected by a searching node, it selects a back off counter
associated with CW_s, and higher back off counter means longer search process, which will reduce the throughput. When the network is extremely loaded, there may be too many high priority RTSF/CTS packets roaming around, which will increase the number of collisions. A small contention window may cause some problems in such a situation because it increases the probability that two collided searching nodes may select same back off counter, and collide again. However a contention window size of 3 is enough for even very high loads, and does not cause any search process fail.

Figure 13 shows the average time delay caused by the search process. The search process is expected to take some time, and this delay is expected to increase with the network load, because as the number of search processes in a network increase, the
number of collisions will increase as well, and this will cause repeated searches, extensive back offs etc. However the average delay seems perfectly reasonable even in very crowded networks.

To determine the performance of this MAC protocol in Multi Hop Ad Hoc Networks, we did an experiment comparing its throughput to the throughput in Wireless LAN scenario. Recall that to be able to observe the multi hop performance completely, a routing protocol will be needed, however preliminary experiments for multi hop requirements are done.

Figure 14. Wireless LAN vs Multi hop throughputs
CHAPTER 6

CONCLUSION

This Thesis work proposed a novel and efficient MAC protocol for multi-channel wireless networks. In the algorithm, nodes never have to listen and transmit at the same time, thus it works with nodes that have single transceivers. This feature makes the protocol perfectly applicable to today’s hardware equipment. The protocol basically separates the search and scheduling processes. Because of this, the protocol does not have a requirement for network-wide synchronization and it is easily applicable to multi-hop networks. According to the experiments performed, the protocol significantly improves the throughput in the network, and does not put too much overhead in terms of throughput and delay.
CHAPTER 7

FUTURE WORK

This Thesis work covers a MAC protocol for multi-channel Ad Hoc networks. However the experiments are performed in a Wireless LAN environment due to the lack of routing protocols. As future work, an efficient routing protocol will be designed, and experiments including this routing protocol and multi-hop capabilities will be done.
REFERENCES


<table>
<thead>
<tr>
<th></th>
<th>Authors</th>
<th>Title</th>
<th>Conference/Proceedings</th>
</tr>
</thead>
</table>