Performance Analysis of Ad Hoc Communication Over Multiple Frequency Hopping Channels

György Miklós, Ferenc Kubinszky, András Rácz, Zoltán Turányi, András Valkó
Traffic Analysis and Network Performance Lab, Ericsson Research, Hungary

Miklós Aurél Rónai, Sándor Molnár
High Speed Networks Laboratory, Budapest University of Technology and Economics

Abstract

Frequency hopping radios possess a number of attractive features, but their use in ad hoc communication is problematic because of the need to synchronize and coordinate the frequency hopping patterns and transmission attempts. We propose a mechanism that allows the use of several frequency hopping channels within a single ad hoc network. We investigate the performance of the proposed scheme using a combination of analytical and simulation tools. We achieve a significant improvement in aggregate throughput despite the additional performance penalty of switching between frequency hopping channels. We also show how this performance penalty can be decreased by grouping devices into the same channel based on the traffic pattern.

*Corresponding author. Address: H-1037 Budapest, Laborc u. 1, Hungary; Phone: +36 1 437 7633, FAX: +36 1 437 7767, Email: Gyorgy.Miklos@eth.ericsson.se
1 Introduction

Frequency hopping spread spectrum radio technology [9] possesses a number of advantages that has motivated its selection in many radio systems. These advantages include robustness against interference, fading and noise, simplicity and low cost of implementation. A key advantage is that a number of such systems can be independently operated in the same coverage area with limited interference. There is no hard capacity limit for the number of interferers. Increasing their number results in a graceful degradation of performance.

In an ad hoc networking context, a number of design trade-offs must be made to make use of the advantages of applying frequency hopping radios. A key question is the establishment of the frequency hopping channels. One possibility is to use a single frequency hopping channel in the entire ad hoc network. However, this solution does not make use of the multiplexing gain that can be achieved by transmitting traffic on many frequency hopping channels in parallel. On the other hand, if we decide to use more than one channel in the ad hoc network, then depending on the traffic pattern, some nodes may be forced to synchronize to and switch between several frequency hopping channels. This switching procedure itself may cause performance degradation.

In this paper we propose Multiple Frequency Hopping Channel communication (MFHC), a scheme that allows the use of several frequency hopping channels within a single network, and investigate the performance of MFHC ad hoc networks. Our approach is to use CSMA/CA (carrier sense multiple access with collision avoidance) random access scheme [7] for each channel, with the extension that we allow a device to switch to a new frequency hopping channel (FHC for short) before each packet transmission. Each node has an associated home FHC that it follows by default. If a source node needs to send a packet to a destination node on the same home FHC, it uses the basic random access scheme on the common hopping channel. If, on the other hand, a source node needs to send a packet to a destination node that has a different home FHC than the source, then it switches to the home FHC of the destination and applies the random access scheme on the destination node’s home FHC.

The MFHC solution makes it possible to send data to a node on any FHC, but it is clear that communication is more efficient when switching between FHCs is not needed. An FHC selection algorithm can be used to select the home FHC so that a node needs to switch to a different FHC as infrequently as possible.

The paper is organized as follows. Section 2 describes existing and proposed frequency hopping systems and their networking capabilities. Section 3 describes our MFHC solution that enables ad hoc networking...
using multiple frequency hopping channels. Section 4 presents three basic FHC configurations and compares them through a simple analytical model. Section 5 investigates the proposed MFHC solution via simulations. Section 6 concludes the paper.

2 Related Work

A number of existing and proposed systems use frequency hopping spread spectrum radios, providing a limited networking capability. Here we provide a brief overview of the systems summarized in Table 1.

Bluetooth [3] has been designed as a cable replacement radio technology. It is a short range (10m) radio interface working in the unlicensed 2.4Ghz band using frequency hopping spread spectrum. The frequency hopping rate is 1600/sec, where hopping takes place over 79 equally spaced 1 MHz wide hops. The frequency hopping radio in Bluetooth has been designed to facilitate the development of very small and cheap implementations. The frequency hopping technology makes the system robust against interference caused by other Bluetooth and non-Bluetooth interferers in the same band [13].

Bluetooth employs a connection-oriented approach where devices have to synchronize using a paging procedure to establish the common frequency hopping channel, referred to as a piconet. The node initiating the procedure becomes the master of the piconet. The formation of the piconet takes a relatively large overhead of several seconds, but makes data transmission straightforward once the piconet is established. This is in harmony with the requirements of cable replacement applications where a connection needs to be set up rarely, typically only once when the application is started or re-started. Once the piconet is established, the frequency hopping sequence is derived from the clock and address of the master node. The timing synchronization is defined by the transmissions of the master.

Since the hopping sequence is dependent on the master, a number of piconets may co-exist in close proximity, each piconet using a different hopping channel. Although there can be a certain amount of interference, this provides a good separation of the radio channels of different piconets. This also provides a logical separation since devices in different piconets do not even have to know about each other at all. Devices in the same piconet, on the other hand, need to be co-ordinated. In Bluetooth, this is performed by the master node using a centralized polling-based scheduling mechanism.

Even though Bluetooth has been optimized for cable replacement scenarios, it is nevertheless possible to use it in a networking context. The specification allows a device to be a member in multiple piconets and
several piconets can be connected into a so-called scatternet. However, the specification does not provide the mechanisms to form scatternets, and a number of important issues remain unresolved, such as how to decide about piconet membership and master roles (i.e., connection setup), how to schedule the presence of a node in multiple piconets, and how to discover and manage neighbours. These problems have to be resolved in extension protocols to the core Bluetooth specification. Research (see for example [8, 10, 11]) and specification work [4] is ongoing to address these issues.

Currently the most widely used ad hoc networking platforms are based on the IEEE 802.11 wireless LAN standard [7] which defines a number of physical layers, frequency hopping spread spectrum being one of them. However, communication is possible only in a single channel (between nodes in the same Basic Service Set in the 802.11 terminology). To use multiple channels, we have to have an infrastructure of connected access points. Without any infrastructure, it could be possible to use several independent hopping channels on the same coverage area to share the available spectrum, but only nodes on the same channel can communicate with each other. (Note that existing products that use the frequency hopping physical layer do not support fully distributed ad hoc operation even at a single channel, despite the fact that the standard allows this and defines a distributed time synchronization method. Instead, ad hoc operation is supported by products based on the direct sequence spread spectrum physical layer.) At the 802.11 MAC layer, multiplexing of traffic on a single channel is achieved by a CSMA/CA (carrier-sense multiple access with collision avoidance). An RTS (request to send) - CTS (clear to send) - data - ACK four-way handshaking mechanism is defined. The RTS-CTS message exchange decreases the overhead of collision (when packets are long) and solves the hidden terminal problem [7].

The Hop-Reservation Multiple Access (HRMA) protocol is introduced in [12] for frequency hopping spread spectrum packet radios. The protocol uses a hop reservation and RTS-CTS handshake mechanism to guarantee collision-free operation even in the presence of hidden terminals. The protocol uses a designated frequency for control message exchange and requires timing synchronization over the whole network. By relying on this common channel that every node listens to, collision avoidance and hop reservation for data transmission can be achieved, so that multiple data transmissions use different frequencies. However, the requirement of synchronizing the whole network in time and using a single common signalling channel may imply performance and robustness bottlenecks.

The design concepts used in the High Frequency (HF) Intra Task Force (ITF) Communication Network
Table 1: Summary of related work and MFHC with ad hoc frequency hopping systems

are discussed in [5] employing frequency hopping spread spectrum radios. The proposal incorporates the
Linked Cluster Algorithm that structures nodes into disjoint clusters making use of two TDMA frames that
are synchronized over the whole network. Once the clusters are formed, a second procedure called Link
Activation Algorithm controls how slots are allocated on the links. The available frequency band is divided
into several sub-bands, and an independent network is formed in each sub-band. This makes it feasible to
perform re-configuration of the network in one sub-band while communication can still continue in other
sub-bands. However, the complexity and performance implications of re-configuration of the clusters and
schedules are unclear.

3 Multiple Frequency Hopping Channel Communication (MFHC)

Our architecture (referred to here as MFHC) combines many of the advantages of frequency hopping sys-
tems presented in Section 2. We allow many frequency hopping channels (FHCs) to co-exist on the same
coverage area. Communication is possible both within an FHC, and across FHCs as well, based on an
adapted CSMA/CA scheme. This architecture is motivated by the advantage of using low cost frequency hopping radios as in Bluetooth, based on a simple connection-less approach with on-demand resource allocation scheme as in the case of IEEE 802.11, which enables networking between all devices as in HRMA and HF ITF, but without the need for a network-wide synchronization mechanism. None of the earlier systems possesses all these properties.

Channel access within a FHC is based on the CSMA/CA approach used by the IEEE 802.11 protocol [7]. This means that a node that has a packet to send on the FHC first waits until the channel becomes free for at least a minimum period of time, which we refer to as GS (guard space). Communication may begin at fixed slot boundaries. (We do not specify the length of a slot here, and simply use slots as the unit time on a FHC.) To resolve collisions due to more than one stations sending at the same time, a contention mechanism is applied as follows. Each station has a contention window, $CW$, and chooses a random backoff value $B$ from the interval $[0, CW - 1]$. In each slot when the channel is sensed free, the value of $B$ is decreased if it is above zero. A node may transmit when the value of $B$ reaches zero. If the transmission is successful, the value of $CW$ is initialized to $CW_{min}$. If the transmission is unsuccessful, the value of $CW$ is doubled unless it reaches $CW_{max}$. This scheme ensures that collisions will be resolved after one or more stages of contention.

We precede each packet transmission by an RTS-CTS message exchange, as in the 802.11 protocol. This handles the hidden terminal problem (the destination receives packets from a station that the source cannot receive from), and also decreases the overhead of contention in the case of long packets. In addition, the RTS-CTS message exchange provides a way for negotiation of parameters for the subsequent data transmission.

This scheme can be extended for multiple FHCs, as shown in the example of Figure 1. Even though it is allowed for a node to switch from one FHC to another, we associate a default FHC with each node, which we refer to as the home FHC of the node. The figure shows two FHCs, where FHC 1 is the home of nodes A and B, FHC 2 is the home of nodes C, D and E. A node may temporarily leave its home FHC, as node B does to visit FHC 2 (B’), but it returns to its home FHC as soon as it has finished contention or transmission. To initiate a data transmission to a node, we need to switch to the destination node’s home FHC and wait until the node is available and the channel is free.

When the destination node’s FHC is different from the source node’s home, then the source node has to switch between the source and destination FHCs during contention. This is illustrated in the figure, where
node B wants to send a packet to node C in FHC 2. First, it switches to FHC 2 (B’) and listens on the channel for at least a fixed amount of time (denoted by LN in the figure). This is needed to synchronize to the channel and determine if there is an ongoing data transmission in the FHC or not. If there is an ongoing data transmission, as in the example, then B must wait until this transmission is over (and observe the guard space, GS) before sending an RTS. In the figure, node D also wants to send to node C, and after colliding with B at the first RTS transmission, it wins the contention in the second stage. B notices this when it hears the RTS from node D and waits until this data transmission is over. For this period of time, it switches back to its home FHC. To determine when it can try again with a new RTS, node B uses its estimate of the length of the data transmission given in the RTS packet (this information is also given in CTS packets). Node B switches back to FHC 2 such that it spends the period of LN (listen) before its backoff counter reaches zero. In the figure, node D wins the contention once again, and B switches back to FHC 1. In the meantime, node A initiates a data transmission to node B which is unsuccessful because node B is away at that time. The RTS is retransmitted later, and the subsequent data transmission is started to node B. This delays node B switching to FHC 2 once again. However, when the transmission in FHC 1 is over, node B can immediately switch to FHC 2. After a period of LN has passed and FHC 2 is sensed free, node B sends its RTS which is successfully received this time, allowing the consequent data packet transmission. Once this is over, node B switches back to its home FHC.

Even though the multiple FHC approach presented above allows nodes to switch to a new FHC to send data, it is clear that communication within the same channel is much more efficient. This is why the home
FHC memberships should be determined such that most of the communication takes place within a FHC. To enable this, a dynamic home FHC selection scheme could be used. This means that a node measures its traffic in the different FHCs and selects its home FHC to be the one where it has the most measured average traffic. This reduces the overhead of switching between FHCs.

The address and home FHC of a neighbouring node is known from a neighbour discovery mechanism. This is either based on a static configuration, or on beacon packets sent by the nodes. Beacon packets can be sent at a dedicated frequency, or on a special frequency hopping sequence. In addition, beacon packets are sent on each FHC in order to synchronize the channel timing [7]. Here we do not analyze this question in detail, but we will consider the overhead of beacon packets used for channel synchronization in the analysis of Section 4.

### 4 Analysis of FHC Configurations

We now investigate the question of selecting the FHCs so that the performance of the communication is maximized. For this, we introduce three FHC configurations and compare their performance based on a simple analytical model. To enable the analysis, we first introduce a model for the contention mechanism. This will be followed by a system model that will be used for the subsequent performance comparison. The comparison in this section is implementation independent, that is, we concentrate on the FHC configurations and neglect the details of the backoff mechanism, packet types and local retransmissions. Later in Section 5 we repeat and elaborate the analysis based on simulations of an implementation of the architecture.

#### 4.1 Modeling of Contention

For our analysis we use a very simple performance model of the contention mechanism that captures the impact of the number of competing nodes on the time needed to resolve contention. We neglect most of the details so that our performance model remains analytically tractable. Our analysis is based on [1] and [2] where the authors aim at modelling the behaviour of the IEEE 802.11 contention mechanism as closely as possible, but here we make some additional simplifying assumptions.

In our model, there is one round of contention in each slot. This means that we assume that the nodes sending the RTS packets get immediate feedback on the success or failure of the contention, and we neglect the possible loss and associated delay with the CTS packet. We also assume that each contending node is
aware of an ongoing transmission on the channel and does not attempt to send an RTS during this period.

It has been observed (see [1] and [2] and the references therein) that the initial value of the contention window, \( CW_{\text{min}} \), may impact the overhead of the contention and its optimal value is dependent on the number of competing nodes. In our analysis we do not consider the question of setting the \( CW_{\text{min}} \) constant, instead we use an adaptive setting of the contention window based on the number of contenders and always use the optimal setting. Note that this issue is considered in detail in [2] where an adaptation mechanism is proposed and it is shown that the performance is close to the optimal settings.

Let the number of contending devices be denoted by \( k \), and let the size of the contention window at each node be \( W \). Using a simple Markovian model, it can be shown [2] that a single node transmits an RTS in a given slot with a probability of \( \tau = 2/(W + 1) \). In a given slot a new packet transmission is initiated when exactly one node transmits an RTS. Assuming independence between nodes, its probability is

\[
P_{tx} = k(1 - \tau)^{k-1}\tau(1 - p).
\]

where \( p \) is the probability of non-collision error (interference, fading, noise). Our purpose here is to find the value of \( \tau \) (and \( W \)) that maximizes \( P_{tx} \). Taking the derivative of the expression and solving it for zero, we get

\[
\tau = \frac{1}{k}
\]

and consequently \( W = 2k - 1 \). The probability of successful RTS transmission is then \( P_{tx} = (1 - 1/k)^{k-1}(1 - p) \). It is well known that the first factor in the formula goes to \( 1/e \) as \( k \) increases. We can thus approximate the probability as

\[
P^*_{tx} = \frac{1}{e}(1 - p)
\]

In Figure 2 we show the probability of successful contention in a slot as a function of the number of competing nodes. The figure shows the results of simulation of the backoff procedure with a fixed value of \( CW_{\text{min}} = 8 \), the corresponding analytical model based on [1], the value of \( P_{tx} \) for the adaptive case as computed above, and the approximation \( P^*_{tx} \). (In this case, \( p = 0 \) was used.) The figure shows that the analytical model in [1] gives a very good approximation for the simulated exponential backoff performance. This performance curve shows a slight increase which is due to the fixed initial contention window setting. The adaptive window performance \( P_{tx} \) gives an upper bound that tends to the simulated values of the backoff procedure as the number of nodes increase, similarly to the approximation \( P^*_{tx} \).
In the following, we will use the approximation $P_{tx}^*$ since it is close to the simulated backoff results especially as the number of nodes increases, and it gives an analytically tractable approximation which is independent of the implementation details and parameters of the contention procedure. (For simplicity, we will extend this approximation even for the $k = 1$ case.) With this approximation, the average number of slots it takes for a given node to win the contention, not counting slots where data is transmitted, is $k/P_{tx}^* = ke/(1 - p)$.

![Backoff procedure](image)

Figure 2: Simulated and analytical performance of contention

4.2 System Model

To model system performance, we introduce a network and traffic model, and compare a number of FHC configurations. Our primary performance metric will be the total system throughput. We will compare the throughput performance of three different FHC configurations.

To model a number of different application groups used over the same coverage area in an ad hoc networking scenario, we use a group-based traffic model: devices send most of their data to other members of the same group. The total of $N$ nodes are divided into groups of size $G$. In our numerical analysis, we consider the extreme case where nodes within a group send packets to the members of the same group only. (Later in Section 5 we will investigate the effect of inter-group communication.) Sources are assumed to be greedy, which means that sources always have a packet to send. Before each packet transmission, the destination is chosen randomly and independently according to a uniform distribution from the other nodes in the same group. Each of the $N$ nodes are within transmission range of each other, so transmissions in different groups at the same time and same frequency collide. We assume that transmission errors can be
detected and an ARQ (Automatic Repeat reQuest) protocol retransmits the errored segments. In this implementation independent analysis the details of the ARQ protocol and the associated potential segmentation and reassembly mechanism are not considered, but we model the additional load caused by retransmissions through a factor of $1/(1 - p)$, where $p$ is the transmission failure probability for a packet.

We distinguish three different FHC configurations based on the set of nodes that use a common home FHC for contention resolution and communication. These configurations are as follows. In the *common* FHC case the same single channel is used by all of the $N$ nodes. This will be our reference case where devices do not need to switch to a different FHC. In the *device* FHC case there is a separate FHC for contention and data transfer for each device. In this case, for each destination a node has to switch to a new FHC. The third FHC configuration that we investigate represents a compromise between the two extremes. In the *group* FHC case, every group of $G$ nodes has its own FHC for contention and data transmission. Since in our traffic model of this section packets are sent only within the group, therefore nodes do not have to switch to a different FHC in this case, either.

Figures 3 - 5 illustrate the three cases. The dark rectangles represent the data packets sent on a given hopping channel, while the lightly shaded rectangles represent contention on a given channel, with the arrows showing the direction of the data transmission and the contention. In the following we will analyze each of these configurations separately.

### 4.3 Common FHC

In the common FHC case each device uses the same frequency hopping channel for both contention resolution and data transmission (Figure 3). The channel is occupied by alternating transmission and contention periods.

To find an approximation for the system throughput, we have to characterize both the length of the data transmissions and the length of the contention periods. To characterize the length of the data transmissions, we assume that each packet transmission takes $L$ slots, where the amount of data transmitted corresponds to $L_0$ slots, which is taken to be a constant. $L > L_0$ because there may be errors on the channel causing retransmissions. We only consider the errors caused by interference and use an independent and identically distributed error model with an error probability of $p_c$. To model the retransmissions of the ARQ protocol we use $L = L_0/(1 - p_c)$ since $1 - p_c$ is the success rate of transmissions.
To find an approximation for the time spent with contention, we use the results of Section 4.1. The approximate average time until one of the nodes wins the contention is \( C_e = e/(1 - p_c) \). From this, the utilization of the common frequency hopping channel is

\[
U_c = \frac{L}{L + C_e} = \frac{L_0}{L_0 + e}.
\]

The utilization of a single node (i.e., the fraction of time spent with transmission) is therefore:

\[
u_c = \frac{U_c}{N} = \frac{1}{N} \left( \frac{L_0}{L_0 + e} \right).
\]

To find the total throughput, we also take into account that the channel synchronization must be maintained. This requires the exchange of packets that consume overhead. Here we consider that synchronization is maintained by the transmission of special single-slot beacon packets with a base period of \( T_b \) slots. This decreases the capacity of the channel by a factor of \( 1 - 1/T_b \). The total throughput is then

\[
\Theta_c = U_c (1 - 1/T_b)
\]

measured in the unit of the capacity of a single frequency hopping channel.
4.4 Group FHC

The group FHC case is characterized by each group of $G$ devices using a common frequency hopping channel for both contention resolution and data transmission (Figure 4).

We use a similar approach to find an approximation for the system throughput as in the previous subsection. The utilization of a single FHC can be computed in the same way:

$$U_g = \frac{L_0}{L_0 + e}.$$

The utilization of a single node then becomes

$$u_g = \frac{U_g}{G} = \frac{1}{G} \left( \frac{L_0}{L_0 + e} \right).$$

We now approximate the probability of interference error, $p_g$. A single frequency hopping channel is disturbed by $N/G - 1$ other similar channels. Each channel hops on $K$ different carriers independently in
a pseudo-random manner. Since the channels are not synchronized to each other, a transmission in a single slot in one channel may disturb two slots in a different channel if the carriers collide. Therefore the error probability caused by data transmissions (neglecting the interference caused by RTS and CTS packets) can be approximated by

\[ p_g = 1 - \left( 1 - U_g \frac{2^K}{K} \right)^{N/G-1}. \]

Then the total throughput is obtained by summing the traffic in each channel, taking into account the synchronization overhead and that data transmission has an efficiency of \( 1 - p_g \) due to errors:

\[ \Theta_g = \frac{N}{G} U_g (1 - p_g)(1 - 1/T_b). \]

### 4.5 Device FHC

The device FHC is characterized by each node having a separate channel for both contention and data transfer (Figure 5). This means that each time a source node sends a data packet to any destination node, the source first has to switch to the FHC of the destination.

We approximate the utilization of a single frequency hopping channel belonging to a given node by approximating the average time taken with data reception as follows. Similarly as above, a single packet reception takes \( L = L_0/(1 - p_d) \) slots, where \( p_d \) is the error probability. A packet reception is preceded by a contention period. This would take on average \( e/(1 - p_d) \) slots in general.

However, contention is prolonged in this case for the following reasons. While waiting for incoming RTS packets initiating a data transfer, each node also has a packet to send at the same time. This means that a node has to switch between its own frequency hopping channel and that of the frequency hopping channel of its destination, as described in Section 3. The nodes participates in two contentions simultaneously, once as a potential transmitter and once as a potential receiver. Even if this could be done with 100% efficiency, this would double the time of the average contention. However, switching between the channels necessarily implies inefficiency. In addition, the contention window of source nodes are increased due to the fact that a destination node does not respond to an RTS when it has switched to a different FHC. The extent of these effects depends on many implementation dependent factors, such as the time needed to switch to a different FHC, and the setting of the maximum contention window. We approximate these effects by assuming that contention is prolonged by a factor of \( \beta \) due to the inefficiency incurred by switching between different channels. Our approximation for the contention period is therefore \( C_d = \beta e/(1 - p_d) \). We have \( \beta > 2 \).
since the length of contention is at least doubled. (We will investigate the value of $\beta$ through simulations in Section 5.)

Due to symmetry between nodes and roles, each node spends the same amount of time with transmitting and receiving, and consequently transmits on average one packet for each packet reception. This follows that the fraction of time spent with reception at a given node is

$$u_d = \frac{L}{2L + C_d} = \frac{L_0}{2L_0 + \beta e}.$$  

Due to symmetry of the traffic model, $u_d$ is also the time spent with transmission by a given node.

The error probability is given by

$$p_d = 1 - \left( 1 - u_d \frac{2}{K} \right)^{N-1}.$$  

To find the total throughput, we have to take into account the synchronization overhead. Each node in a group has to synchronize to all other nodes in the group, giving a factor of $1 - G/T_b$. We can write the total system throughput as

$$\Theta_d = Nu_d(1 - p_d)(1 - G/T_b).$$

### 4.6 Performance Comparison

We now evaluate the performance of the FHC configurations based on the analytical model of the previous subsections. First, we plot the total throughput as a function of the total number of nodes $N$, see Figure 6. In the figure we use tentative parameter settings: the group size was fixed to 10 nodes, packet length was 12 slots, number of hop frequencies was set to 79, we used $\beta = 4$, and the synchronization overhead was not included. In the upper left box, we plot the node utilization, that is, the fraction of time spent with data transmission at a node. First of all we can observe that this is constant for the group and device FHC configurations. To explain this, notice that the groups are logically independent and do not depend on each other except for the interference. The increase in the interference is shown in the upper right. Interference causes data transmissions to be longer, but it also prolongs the contention period by the same factor explaining why the utilization remains constant. (Note that in a given implementation the effect of packet losses may cause a different factor of increase for the data transmission time and for the contention. This may result in slight changes in the utilization as will be visible in the simulation results of the next
In the common FHC case, nodes share the same channel which causes the per node utilization to decrease as the number of nodes increases.

![Graphs showing system performance](image)

**Figure 6: System performance,** $G = 10, L_0 = 12, T_b = \infty$

The figure in the lower left box shows the total throughput of the system. This is constant for the common FHC case since the total capacity of a single channel is used, and it is not affected by interference. In the device and group cases, the total throughput increases with increasing number of nodes. This is because the number of FHCs are increased providing multiplexing gain. The slope of the curves decrease though because of the increased interference. The device FHC case allows for a greater number of parallel data transmissions to be multiplexed than the group FHC case which allows only a single transmission per group. This explains the significantly higher total throughput of the device FHC configuration.

Also plotted in the lower right box is a measure of the spectral efficiency. We obtained this measure by dividing the total throughput by the number of hop carriers, $K$. If all carriers were continuously transferring data, this measure would yield 1; its value therefore represents the efficiency of utilizing the available spectrum. We made an exception in the common FHC case, where we did not divide the total throughput by $K$, since only a single common channel is used in this case, which could - hypothetically - span even the whole available spectrum without causing any interference. In the rest of the cases, this is not possible since many channels need to be multiplexed that could interfere with each other.
The results show that the spectral efficiency is highest in the *common* FHC case, and it is lower for the other configurations. This observation can be interpreted as follows. If a single common high-speed channel can be used by all devices on an on-demand time-division basis, it can give a much more efficient usage of the available spectrum than dividing it into many uncoordinated low-speed channels. We have to keep this in mind when considering the other configurations employing multiple uncoordinated frequency hopping channels. However, a high-speed common channel may be difficult to realize in practice. A full comparison of these two cases, involving other aspects such as hardware limitations, cost, radio propagation and error characteristics, are out of the scope of this paper.

Figure 7 shows the dependence of the node utilization on the group size and packet length, with only a single group present. (Note that the node utilization determines the total throughput.) We can see that the *device* FHC case offers a constant per node utilization, while the *group* and *common* FHC cases (which are identical in this scenario) yield a decreasing per node utilization. The reason for this is that the *device* FHC configuration allows multiplexing of data transmissions within a group. To compare the two curves at $G = 2$, notice that we have $\beta > 2$, which follows that $u_d < u_g$. This means that for a group of two nodes, the *device* FHC is necessarily less efficient than the *group* FHC. Depending on implementation dependent value of $\beta$, the two curves must intersect each other, marking the group size where the *device* FHC configuration is equally efficient as the *group* FHC. By comparing the two graphs for long and short packet size, we can observe that the intersection point is also dependent on the packet size. When packets are shorter, the effect of backoff overhead is increased, therefore the per node utilization (and the total throughput) is lower.

![Figure 7: Dependence of node utilization on the group size and packet length. $N = G, T_b = \infty$. On the left $L_0 = 12$, on the right $L_0 = 2$.](image)

Figure 8 shows the dependence of the total throughput on the synchronization overhead. This overhead depends on the accuracy of the clocks that are used: the less accurate they are, the more frequently we need
to send beacon packets to keep the synchronization. The figures plot the base beacon sending period. The figures show that the device FHC case is the most sensitive to synchronization overhead, especially for higher group size. This is attributed to the fact that in this case, a node in a group has to synchronize to all other nodes in its group to be able to send data, while in the other cases nodes have to synchronize to one channel only. Note also that we took a very conservative computation for the synchronization overhead, since only a single slot was wasted for a beacon packet. In a practical implementation, however, this overhead might be much higher, which further emphasizes that the device FHC configuration is very sensitive to accurate synchronization.

Figure 8: Dependence of system throughput on the beacon period and group size. \( N = G, L_0 = 12 \). On the left, \( G = 2 \), on the right, \( G = 10 \).

5 Simulation Study

To investigate the performance of the implementation of different FHC configurations, we have implemented the MFHC scheme in a packet level simulator [6]. Figure 9 shows the architecture of the simulator. The physical layer consists of a packet collision detector which determines the reception status of every individual packet. Each node has an associated FHC object in the physical layer. The link layer representation of each node connects to exactly one FHC in the physical layer at a time, the one that it follows at the given moment as determined by the MFHC protocol implementation in the link layer.

We consider scenarios where all nodes are within radio range of each other, which represents the worst case in terms of interference. In the physical layer model, packets are either lost due to collision (interference), or they are delivered correctly. In the link layer, we model the contention mechanism as described in Section 3. FHCs are independent of each other using a pseudo-random frequency hopping pattern. We
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel capacity</td>
<td>1 Mbit/sec</td>
</tr>
<tr>
<td>Slot length</td>
<td>1 ms</td>
</tr>
<tr>
<td>Hop frequencies</td>
<td>79</td>
</tr>
<tr>
<td>Segment length</td>
<td>1, 2, 3, 4 or 5 slots</td>
</tr>
<tr>
<td>Segment header length</td>
<td>164 bits</td>
</tr>
<tr>
<td>Guard time for frequency hopping</td>
<td>0.1 ms per slot</td>
</tr>
<tr>
<td>Length of RTS, CTS, ACK packets</td>
<td>1 slot</td>
</tr>
<tr>
<td>Minimal contention window</td>
<td>8 slots</td>
</tr>
<tr>
<td>Maximal contention window</td>
<td>64 slots</td>
</tr>
<tr>
<td>Synchronization overhead</td>
<td>0 (not considered)</td>
</tr>
<tr>
<td>Listen time on new FHC</td>
<td>6 slots</td>
</tr>
</tbody>
</table>

Table 2: Simulation parameters

![Simulator architecture](image)

Figure 9: Simulator architecture
have implemented a segmentation and reassembly mechanism, and an ARQ protocol that gives feedback on the reception status after each segment. Lost segments are retransmitted immediately. Packets can be sent at the beginning of a slot. The slot timing is aligned to frequency hopping: there is a guard time at the end of each slot to allow devices to tune to a new frequency. Table 2 lists the parameters used in the simulations. Note that the channel capacity and the number of hop frequencies were selected to reflect the constraints of the 2.4GHz ISM band, and the other parameters were selected to reflect the current capabilities of typical Bluetooth implementations.

First we investigate the extent of multiplexing that can be achieved by using the group and device FHC configurations. Figure 10 shows the node utilization and total system throughput as the number of groups, each with ten member nodes, are increased. The results are in accordance with the analysis of Section 4, showing that the device FHC configuration increases the total system throughput by a factor of two. However, the multiplexing gain that is achieved by the device FHC is only present with large group sizes. Figure 11 shows that with a group size of two nodes, the device FHC case actually performs worse by about 30%, because it is less efficient in contention and can not make use of multiplexing.

![Figure 10: Node utilization and total throughput as the number of groups are increased. Group size is fixed to 10.](image)

Figure 12 investigates the differences between small and large groups. The results follow the same trend as the analytical model shown in Figure 7. Fitting the formulas of Section 4 to the simulation results, we can approximate the value of $\beta$, which shows the inefficiency of contention in the device FHC case. We get a value of $\beta = 16$ in the case of 1500 byte packets and $\beta = 8$ in the case of 250 byte packets. These values are much greater than the minimal value of 2, showing that the switching of FHC during contention introduces a significant amount of extra overhead. Note also that this factor is not constant: in the case of long (1500 byte) packets and minimal group size of 2 nodes, the device FHC becomes more efficient ($\beta$ decreases to 8 in
Figure 11: Node utilization and total throughput as the number of groups are increased. Group size is fixed to 2.

This case). When there are only two nodes in the group, the intended destination node does not communicate with other nodes. That would cause failed RTS attempts whose number is much bigger in the case of long packets, which explains why we see this effect to a much greater extent in the case of long packets.

Figure 12: Utilization in the case of a single group. On the left packet length is 1500 byte, on the right packet length is 250 byte.

So far we have allowed traffic only within a group in our model. We now extend our traffic model to investigate the effect of traffic between the groups as well. In the extended model, with a probability \( p_{ng} \) a nodes chooses its destination from all the other nodes in the network, not just its own group. Figure 13 shows the throughput performance as a function of the probability \( p_{ng} \) which determines the non-group traffic. The device FHC configuration is not sensitive to this change since it does not depend on the formation of groups. It only shows a slight throughput decrease in the case when the group size is two, which is explained by the reasoning above in the previous paragraph. The group FHC shows a decrease in both small and large group sizes, but the decrease is much more significant when the group size is small. The reason for this is that when then group size is high, there is a higher chance that at least one of the potential destinations is available, and
so the channel can be utilized. In both cases, the results show that the device FHC configuration gives higher performance in the case of heterogeneous traffic, that is, when there is significant traffic between the groups.

![Figure 13](image1.png)

*Figure 13: The effect of non-group traffic on the total throughput. On the left group size is 2, on the right group size is 10. The number of nodes is fixed to 50.*

Finally we observe the effect of changing the traffic pattern within a group to model a client-server application (with no traffic between the groups). In this case we designate one node in all groups to be a server and the other nodes in the group to be clients. All nodes remain greedy as before in that they always have a data to send, but with a constant probability $p_s$, the clients choose the server as their destination. Figure 14 shows the total throughput as the constant $p_s$ is increased from 0 to 1 (server-client traffic only). In this experiment the total throughput of the group FHC configuration remains constant since this is determined by the capacity of the group channel. On the other hand the performance of the device FHC configuration decreases to that below the group case. When there is only server-client traffic, the device FHC case can not achieve multiplexing gain, and it is uses a less efficient contention scheme than the group FHC which explains its lower performance.

![Figure 14](image2.png)

*Figure 14: The effect of server-client traffic on the total throughput. Group size is fixed to 10, number of nodes is 50.*
6 Conclusion

We have proposed Multiple Frequency Hopping Channel communication (MFHC), a scheme that allows an ad hoc network to achieve high throughput through the use of multiple frequency hopping channels (FHCs) in parallel. Our scheme relies on the notion of home FHC. Each device participating in an ad hoc network has a home FHC which determines the frequency hopping scheme it follows whenever it is not transmitting at another FHC. To transmit to a particular device, it is necessary to switch to that particular device’s home FHC, listen to the channel and resolve contention. The difference from a traditional random access scheme is that besides the possibilities of success or collision, a third option is that the destination is “away” at another FHC.

This scheme allows ad hoc networks to benefit from the advantages of frequency hopping and increase their throughput, but it requires additional coordination compared to an ad hoc network using a single radio channel. We have investigated the impact of this additional coordination on the system’s performance using analytical and simulation tools. In particular, we have compared the extreme case of MFHC, where each device has its own distinct FHC, to a reference case where the entire ad hoc network uses the same FHC. The results show that the former case (device FHC) provides significantly higher total throughput than the reference case (common FHC).

We have also analyzed a case where subsets of an ad hoc network form a partially closed communication group in the sense that members of one group communicate mostly with other members of the same group and rarely with other nodes of the ad hoc network. This scenario may be typical in some realistic ad hoc networks. We have shown how MFHC can adapt to this case such that members of one group share the same home FHC. This case, referred to as group FHC, represents a compromise between the device FHC and common FHC cases. We have shown that it is especially well suited to server-client type traffic patterns, but it is ill-suited for heterogeneous traffic patterns. The group FHC configuration makes the contention mechanism more efficient and it requires less overhead for channel synchronization, in exchange for lower multiplexing gain and consequently lower total throughput.

As a continuation of the work presented here, we intend to design the protocols needed to govern the operation of a MFHC system. The necessary protocols include neighbour discovery and synchronization protocols, and an FHC selection algorithm that allows us to optimize the performance in an ad hoc networking scenario.
References


