

Frequency Hopping Pattern Detection in Wireless Ad Hoc Networks

Min Song, Scott Wigginton

Department of Electrical and Computer Engineering

Old Dominion University

msong@odu.edu, swigginton@cox.net

Abstract

Frequency hopping is a technique that wireless devices communicate in a way that the transmission frequencies are alternated in a pre-determined ordered hopping pattern known only to the sending and receiving devices. In this paper, we develop an algorithm called Frequency Hopping Pattern Detection to detect the frequency hopping pattern of a wireless ad hoc network. The algorithm is performed in three stages. Stage 1 locates the first frequency in the pattern; stage 2 detects the remaining frequencies; and stage 3 calculates the entire hopping pattern. All three stages work together and guarantee the capture of all frequencies used, the pattern in which they occur, and the length of transmission for each frequency. The algorithm is simulated in a VHDL environment. Simulation results verify the correctness of the algorithm.

Key words – frequency hopping, wireless networks, spread spectrum, pattern detection

1. INTRODUCTION

Frequency hopping is a technique that wireless communicating devices use, which alternates transmission frequencies in a pre-determined ordered hopping pattern (HP) known only to the sending and receiving devices. The HP is pseudo-random across a set of known frequencies. After the current pattern is exhausted a new iteration occurs to allow the rest of the message to be transmitted. There are three components in the HP. The first one is the frequency of each hop. The communication is possible only if both the sender and the receiver are using the same frequency. The second one is the dwell time for each hop in the HP. The dwell time of a particular frequency is the time for which the frequency is being used for communication, i.e., the communicating devices must be well synchronized in the time domain. This allows the communicating devices to know how

long to transmit/receive before switching to the next frequency. The third component is the sequence of the frequencies used in the HP, i.e., the communicating devices must be well synchronized in the frequency domain. Finding the sequence refers to finding the starting frequency and the following frequencies in the correct order. By appearing to transmit randomly across a set of frequencies, for a random dwell time per frequency, the problem of interception and/or interference is non-trivial.

Frequency hopping is commonly considered as one of the security approaches at the physical layer because it increases security by employing the secret frequency hopping pattern and reduces jamming from outside devices [1]. Therefore, it has become one of the key security features of modern wireless networking standards such as IEEE 802.11 (Wireless Local Area Networks) [2] and IEEE 802.15 (Bluetooth) [3] protocols, in which the frequency hopping technique is implemented using spread spectrum technology. In North America and Europe, this standard uses 79 channels in the ISM (Industrial, Scientific, and Medical) band (2.4 GHz) each with bandwidth of 1 MHz. The standard also defines three basic hop patterns, each of which uses all 79 channels before repeating themselves, and establishes a minimum dwell time of 3.2 ms and a maximum dwell time of 400 ms. Other applications of frequency hopping include FHFSK mobile radios [4], FH-MFSK factory communication systems [5], CDMA systems [6], and FHMA [7]. Detectors that work under random noise conditions can be found in [8].

In this paper, we develop an algorithm to detect the hopping pattern without any prior knowledge. The rest of the paper is organized as follows. Section 2 formally specifies the system model and the problem statement. The operational behavior of the detection device is also discussed in Section 2. The algorithm to detect the frequency hopping pattern is presented in Section 3. The analytical and simulations results are provided in Sections 4 and 5, respectively. Section 6 concludes the paper.

2. SYSTEM MODEL AND PROBLEM STATEMENT

We consider a wireless ad hoc network that includes three types of devices: senders, receivers, and a detection device. Senders use a hopping pattern (HP) that consists of two to 79 randomly selected frequencies where each frequency may only exist once. The HP is labeled F' with frequencies $F_1, F_2 \dots F_{i-1}, F_i$. The number of frequencies in F' is denoted as $|F'|$. The dwell times exist in units of frames (3.2 ms are required to transmit each frame). Each frequency has a random dwell time assigned with a maximum of 125 frames (400 ms). Following the recommendation of IEEE 802.11 protocol [2], at the end of transmission on one frequency, 224 μs elapses before the next frequency begins transmission. During this period no frequency is capable of being detected. A receiver may test only one frequency at a time. The detection occurs by using the clear channel assessment (CCA) to determine if a frequency is currently being transmitted upon, this process requires 50 μs and a busy indicates a capture.

The goal of the detection device is to detect the HP given no knowledge about the HP used by the sender and the receiver. Our objective is to design a Frequency Hopping Pattern Detection (FHPD) algorithm such that the detection device with this algorithm can detect the HP expressed as

$$HP = \sum_{i=2}^K DT_i \cdot \vec{F}_i, K \leq 79.$$

where DT_i is the dwell time of frequency F_i . The arrow above the F_i indicates that the order of sequences matters.

The detection device is specified to detect exactly one frequency at a time. This detection occurs by calling the CCA procedure. In order for the CCA to determine that a specific frequency is currently being transmitted upon, the transmission must exist for the whole of the test period (in this case 50 μs). If a new transmission begins on a frequency during the test, or if an existing transmission on the tested frequency ceases prior to the test completion, the CCA is unable to capture regardless of the correct frequency having been tested. The other time the CCA is guaranteed to not capture a frequency is during the frequency switch when no transmission exists.

Fig. 1 shows an example that results in a failure to detect any frequency, regardless of which frequency is tested. It also depicts the maximum number of lost tests during the transition from one frequency to another.

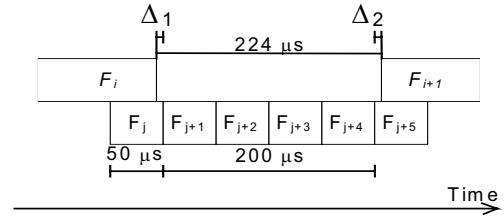


Fig. 1. Samples of non-detecting tests

The upper portion of Fig. 1 depicts the transmission frequencies, marked as F_i and F_{i+1} and the switching delay between them. The lower portion shows six iterations of the CCA test on frequencies F_j to F_{j+5} . F_i has ceased transmission a Δ_1 prior to F_j having completed its test. F_{i+1} began transmission Δ_2 after F_{j+5} began its test, where $\Delta_2 = 24 \mu\text{s} - \Delta_1$. When $\Delta_1 \geq 24 \mu\text{s}$, F_{j+5} can complete its test and only five tests are lost; this is the case with the fewest lost frequencies.

Each frequency's transmission allows for a set of n tests to occur. This number of tests is based on the length of the transmission and the length of the detection. This value must also account for lost captures due to partial tests due to the lack of synchronization. The following formula covers these cases.

$$n = \left\lfloor \frac{\text{Transmission length}}{\text{Detection length}} \right\rfloor - 1 \quad (1)$$

This equation can be used to determine the number of tests allowed per transmission. In the case where only one frame is transmitted, $n = 63$. When the maximum 125 frames are transmitted, 7,999 frequency tests may occur.

3. FHPD ALGORITHM DESIGN

Three key stages were recognized in the design of the algorithm. These stages are of a fashion that each stage can be considered an algorithm on its own accord. They are linked such that the results of the first (second) stage are taken as the initialization input to the second (third) stage. The stages can be summarized as the following:

- 1) Capture any frequency that is being transmitted on.
- 2) Using the first frequency that was found to locate the rest of the frequencies.
- 3) Using the data from all the frequencies found to properly order and determine the transmission length per frequency.

The objective of the first stage is to identify one particular frequency that is transmitting, regardless of its location in the HP. The second stage is to capture the rest of the frequencies and use a time-based marker to associate them with the first frequency. The final stage takes the frequencies and the time markers from the second stage and calculates the order of the frequencies and the individual dwell times.

3.1 First Frequency Capture Algorithm

The first frequency capture algorithm (FFCA) is used to identify the first frequency in F' , denoted as F'_0 . The method used to guarantee this is to scan through all the frequencies, and if no frequency is found the scan begins anew. The algorithm ends when F'_0 is located which is then passed to the second stage.

The following steps outline the specifics of the algorithm:

- 1) Set j to frequency 1.
- 2) Scan F_j for 50 μs .
- 3) If a channel is recognized then stop and record $F'_0 = F_j$.
- 4) If no channel is found and $j \leq 78$, increment j and go to step 2. When $j = 79$, go to step 1.

The length of the cycle for the sender varies based on the size of F' and is determined as follows,

$$F_{cycle} = \sum_{i=1}^{|F'|} (DT_i + 224 \mu s) \quad (2)$$

For the case $DT_i = 3.2$ ms (2) reduces to

$$F_{cycle} = |F'| \cdot 3.424 \text{ ms} \quad (3)$$

The length of the receiver's cycle was established earlier as the time to guarantee testing on all 79 frequencies, which takes 4 ms. Equation 4 is set to determine the number of iterations each cycle must complete until they are synchronized, where x represents the number for sender's cycle and y stands for the tester's cycle.

$$3.424 \text{ ms} \cdot |F'| \cdot x = 4 \text{ ms} \cdot y \quad (4)$$

It follows that $|F'| \cdot x = y \cdot (4/3.424)$ which then reduces to $|F'| \cdot x = y \cdot (125/107)$. The relationship between the y and $|F'| \cdot x$ terms determine the ratio of cycles from each end to establish synchronization. This can be found for any combination of $|F'| \cdot x = 107$. With the value 107 being prime, in order to prevent a different set of frequencies to be scanned x

must be set to 1. This would require 107 frequencies, which is outside the scope of this paper.

For the case where $|F'| = 2$, 107 iterations are necessary to establish synchronization of the cycles. Fig. 2 shows the capture during the second iteration of the cycle. During the first iteration if F_i is not found then it is shown that $F_i \geq 64$. During the next cycle the values in this range are tested, guaranteeing the capture. The upper region of Fig. 2 depicts the transmitted frequencies while the lower are represents the tested frequencies. Frequencies in the shaded area fall into the non-detectable region of operation (frequency switch). With this analysis it is now possible to assert that a frequency will always be captured which in turn certifies that the algorithm will terminate.

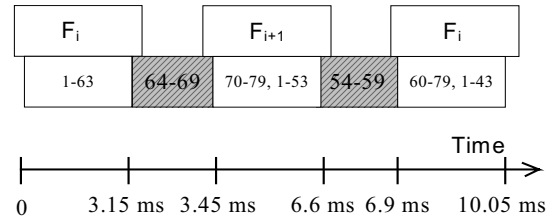


Fig. 2. FFCA scan where $|F'| = 2$

3.2 Remaining Frequencies Capture Algorithm

Once the FFCA is complete, it passes F'_0 to the Remaining Frequencies Capture Algorithm (RFCA). The RFCA makes note of when frequencies cease transmission in relationship to the end of the F'_0 signal. Upon initialization, the following two scanning arrays are created, F_{scan1} and F_{scan2} , where $F_{scan1} = \{F_1, F_2 \dots F_{38}, F'_0\}$ and $F_{scan2} = \{F_{39}, F_{40} \dots F_{78}, F_{79}\}$. The RFCA uses an array of *End Times*, ET_i , to store the timer in relation to the frequency that was found. The algorithm is described as follows:

- 1) Test F'_0 until it is not found. Initialize the timer to 0.
- 2) Begin scanning F_{scan1} . Increment the timer with each test.
- 3) If a frequency is found, continuously test it until it is not found.
- 4) If a found frequency is not equal to F'_0 , record the timer associated with the end of transmission in the appropriate ET_i position.
- 5) If F'_0 is found, continuously test it until it is not found. Record the *period* as the timer - 1 and reset the timer.

- 6) Begin scanning $F_{\text{scan}2}$. Increment the timer with each test. When the timer exceeds the period, exit algorithm (or if F_0' is found).
- 7) If a found frequency is not equal to F_0' , record the timer associated with the end of transmission in the appropriate ET_i position.

It was noticed that by only scanning for F_0' until it reappeared was wasting multiple test cycles. By testing for F_0' once out of a scan pattern, several other frequencies can be tested during this first pass through the period.

The *End Times* array will be passed to the Calculate Hopping Patterns Algorithm (CHPA) when RFCA is completed.

3.3 Calculate Hopping Pattern Algorithm (CHPA)

The CHPA receives the array ET_i from the RFCA as input and calculates the order of the HP and the dwell times of those frequencies. The algorithm works as follows:

- 1) Sort the input array ET_i
- 2) Calculate the dwell time for F_0'
- 3) Calculate the rest of the dwell times.

The first issue for CHPA is to sort the array ET_i based on the timers. Several known algorithm exist that operate on the order $n \log n$. Quicksort is one such algorithm that is widely known and benefits from randomness of data [9]. After the frequencies are sorted, the dwell times must be calculated from the known timers. In order to calculate any dwell time, the timer of the previous frequency must be known, and in the case of F_0' , the previous frequency is the last one in the pattern. Equation 5 calculates the dwell time for the first frequency while equation 6 covers the other frequencies.

$$DT_0 = \left\lceil \frac{\text{period} - ET_{|F|} - 224\mu\text{s}}{3200\mu\text{s}} \right\rceil \quad (5)$$

$$DT_i = \left\lceil \frac{ET_i - ET_{i-1} - 224\mu\text{s}}{3200\mu\text{s}} \right\rceil \quad (6)$$

where the *period* in (5) represents the number of elements in order to indicate the last one. The ceiling function is applied because the timer used to calculate the period and the ET s is based on the $50 \mu\text{s}$ test, causing the measurement to be under by as much as the test period. This is corrected by readjusting the calculation over the measured value.

4. ANALYTICAL RESULTS

The analytical results are organized in the following manner. Each stage is analyzed individually in Sections 4.1, 4.2, and 4.3. Due to the space limit, the best and worst cases are the only items analyzed. In Section 4.4 the whole of the FHPD is analyzed.

4.1 FFCA Analysis

The best case for the FFCA is when all the frequencies used in HP have a dwell time larger than one frame. This guarantees that the first frequency transmitted will be captured in one scan cycle. The worst case exists when all the frequencies are used and they are each transmitted for one frame. If the frequencies are ordered in the worst possible combination when the FFCA begins, it is necessary to wait for the second passing of the first frequency transmitted to guarantee a capture. The results from these cases are in Table 1.

Table 1: Theoretical upper and lower bounds for the FFCA.

Condition	Best Case	Worst Case
Frequencies used	Any	79
Dwell time (frames)	≥ 2	1
Total time	$50 \mu\text{s}$	270.72 ms

4.2 RFCA Analysis

The RFCA passes through the transmission cycle twice. The RFCA is also responsible for determining when F_0' ceases transmission as soon as the FFCA captures it. The best case is found when only two frequencies exist in the HP and transmission lasts for only one frame each. The worst case is when all 79 frequencies are used, each transmits for the maximum of 125 frames, and the FFCA passes F_0' after only one test has passed into its transmission (requiring an additional 3.15 ms of transmission until it ends). The numeric values for these boundaries are found in Table 2.

Table 2: Theoretical upper and lower bounds from the RFCA

Condition	Best Case	Worst Case
Frequencies used	2	79
Dwell time (frames)	1	125
Total time	13.656 ms	63.635 s

4.3 CHPA Analysis

The issue of sorting *End Times* and calculating dwell times are based on the number of comparisons required. These values are dependent on the number of frequencies used, with the best case occurring when only two frequencies exist and the worst case when all 79 frequencies exist. Within quicksort there exist a best and worst case, these will be counted for as well. Table 3 shows the theoretical upper and lower bounds for the CHPA analysis.

Table 3: Theoretical upper and lower bounds for the CHPA.

Condition	Best Case	Worst Case
Frequencies used	2	79
Total time	4 ms	6320 ms

4.4 FHPD Analysis

The overall best and worst cases from the FHPD vary across the cases for the separate stages. The overall best case uses common best case between the RFCA and the CHPA where there are two frequencies existing for only one frame. The worst case for FHPD actually includes the best case for FFCA but also includes the worst cases for RFCA and CHPA. These selections can be found because the overall weight of each stage is not proportional to each other, thus the worst case for RFCA is 233.5 times larger than that of FFCA. The overall best and worst cases are shown in Table 4.

Table 4: Theoretical best and worst cases for the FHPD.

Condition	Best Case		Worst Case	
	Frequencies used	79	2	79
Dwell time	1	1	125	125
Total time	21.92 ms	823.16 ms	2.01 s	63.96 s

5. SIMULATION RESULTS

VHDL was selected as the programming language to simulate the FHPD because of its ability to handle time-driven devices and the ease of implementing concurrent processes. For the first test, 25 test cases were generated with a random number of frequencies, frequency order and dwell times via a C++ program implemented in the GNU G++ 2.95.2 UNIX

environment. In addition to these test cases, four cases were generated with random frequencies but with a forced number of frequencies and dwell times in order to test key best and worst cases. The actual cases generated have the attributes as displayed in Table 5.

Table 5: Comparison of ideal test-case attributes and generated ones.

Attributes	Ideal	Generated
Average number of channels	40.5	40.64
Average number of frames per channel	63	62.79

The first test cases were put through the simulation with the four special cases. The environment was set with an initial delay of 9 μ s. The only value considered for overall analysis was the time when the done value was set to high, minus the delay. The results were plotted on a graph against the best and worst cases as shown in Fig. 3, where the x-axis represents the length of HP. The additional special test cases cannot be seen because they fall near the boundaries. Thus, those results are found in Table 6.

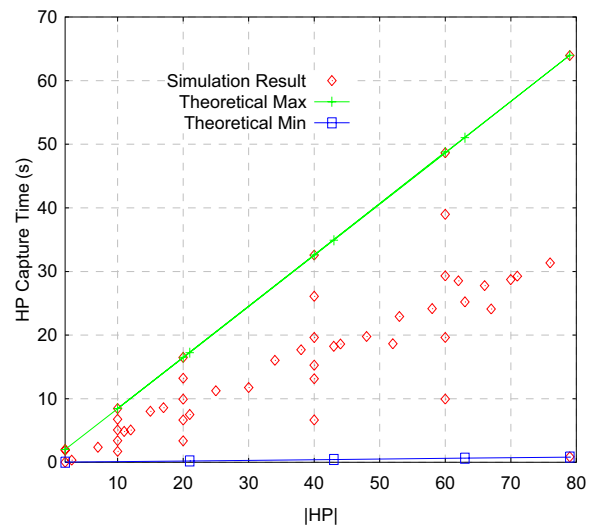


Fig. 3. The HP capture times vs. the pattern length

Table 6: Special test case results.

Test Case	Expected	Actual
HP = 2 DT= 3.2 ms	≥ 21.918 ms	24.81 ms
HP = 2 DT= 400 ms	≤ 2008.796 ms	1891.86 ms
HP = 79 DT= 3.2 ms	≥ 823.16 ms	852.56 ms
HP = 79 DT= 400 ms	≤ 63.955 s	63.944 s

The second test cases were created with a known $|HP|$ and the dwell time, which were then analyzed theoretically. The HP lengths selected were 10, 20, 40, and 60. The dwell times to be tested for each $|HP|$ was 80 ms, 160 ms, 240 ms, 320 ms, and 400 ms. Only one HP was created for each length, thus the HP with length 10 was tested for each of the dwell times. Figs. 4 and 5 show the results from these test cases.

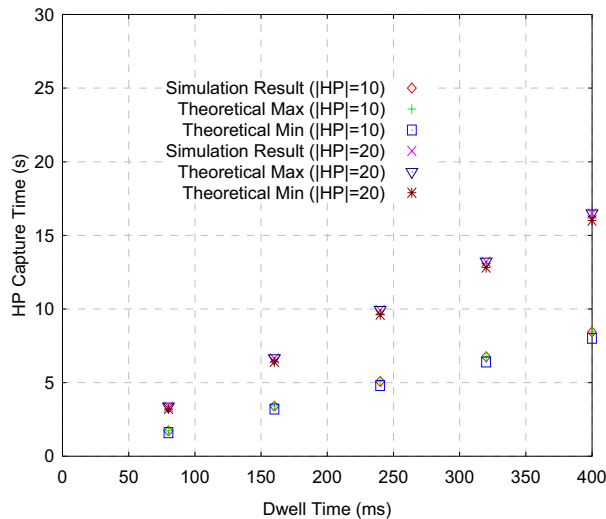


Fig. 4. The HP capture time vs. the dwell time

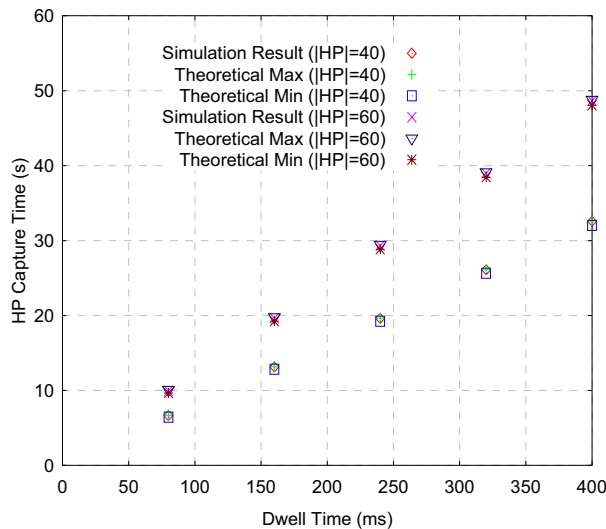


Fig. 5. The HP capture time vs. the dwell time (cont.)

6. CONCLUSIONS

In this paper, we have presented the frequency hopping pattern detection algorithm in wireless ad hoc networks. The algorithm has three stages: First

Frequency Capture Algorithm (FFCA), Remaining Frequencies Capture Algorithm (EFCA), and Calculate Hopping Pattern Algorithm (CHPA). Each stage of the algorithm was analyzed for its run time boundaries. The overall algorithm was then analyzed, assisted by the previous results to produce the upper and lower bounds of 21.918 ms and 63.955 seconds.

The simulations in VHDL were successful. It executed properly and all results fell within the expected ranges. The randomly generated test cases were near the ideal cases and the special cases adequately tested the upper and lower boundaries of the algorithm.

REFERENCES

- [1] A. D. Wood, and J. A. Stankovic, "Denial of Service in Sensor Networks," *IEEE Computer*, Oct. 2002, pp. 54–62.
- [2] <http://standards.ieee.org/getieee802/802.11.html>.
- [3] The official Bluetooth Web site, <http://www.bluetooth.com>.
- [4] D. J. Goodman, P. S. Henry and V. K. Prabhu, "Frequency-Hopped Multilevel FSK for Mobile Radio," *Bell System Technical Journal*, Vol. 59, No. 7, Sep. 1980, pp. 1257–1275.
- [5] P. Yegani and C. D. McGuillem, "FH-MFSK Multiple Access Communication Systems Performance in the Factory Environment," *IEEE Trans. Veh. Technol.* Vol. 42, Feb. 1993, pp. 333–344.
- [6] L. Yang and L. Hanzo, "Blind Joint Soft-Detection Assisted Slow Frequency-Hopping Multicarrier DS-CDMA," *IEEE Transactions on Communications*, Vol. 48, No. 9, Sep. 2000, pp. 1520–1529.
- [7] Y. T. Su, Y. Shen and C. Hsiao, "On the Detection of a Class of Fast Frequency-Hopped Multiple Access Signals," *IEEE Journal on Selected Areas of Communications*, Vol. 19, No. 11, Nov. 2001, pp. 2151–2164.
- [8] R. A. Dillard and G. M. Dillard, "Likelihood-Ratio Detection of Frequency-Hop Signals," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 32, No. 2, April 1996, pp. 543–553.
- [9] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, *Introduction to Algorithms*, 2nd ed., Cambridge, MA, The MIT Press, 2001, pp. 145–164.