802.11A - OFDM PHY
CODING AND INTERLEAVING

Fernando H. Gregorio

Helsinki University of Technology
Signal Processing Laboratory,
POB 3000, FIN-02015 HUT, Finland
E-mail:gregorio@wooster.hut.fi

1. INTRODUCTION

In a communication system there are three fundamental resources : signal power, time and bandwidth. These resources can be traded against each other depending on the premium attached to each resource in a given situation. The general objective is achieve maximum data transfer, in a minimum bandwidth, while maintaining an acceptable quality of service.
In digital communication the quality of service is associated with the probability of bit error at the receiver. The Shannon-Hartley law give a theoretical limit for the transmission rate of data from a transmitter of given power, over a channel with a given bandwidth, operating in a given noise environment. In order to find this maximum value coding techniques was introduced.
This paper presents coding and interleaving techniques applied in WLAN systems, which are defined in IEEE 802.11a/g standards. An overwiev about convolutional codes and interleaving in 802.11a is presented in sections 2 and 3.
In the section 4, some simulations using interleaving and coding in a IEEE802.11a implementation are presented. Finally, some conclusions are presented in the last section.

2. CONVOLUTIONAL CODES

Convolutional codes are present in a great number of today's systems. This codes are applied in IS-95 and GSM (mobile phone) and WLAN systems. The standards IEEE 802.11a and Hiperlan/2 use convolutional codes and IEEE 802.11b includes an optional mode that uses them. The popularity of convolutional code is due to their good performance and flexibility to achieve different coding rates.

2.1. Coding with Convolutional Codes

Error control coding , or channel coding, is a method of adding redundancy to information so that it can be transmitted over a noisy channel to another party, and subsequently be checked and corrected for errors that occurred in transmission.
Convolutional codes are commonly specified by three parameters n, k, m. Where n is the number of output bits, k is the number of input bits and m is the number of memory registers.
Encoder for a convolutional code accepts k-bit blocks of information sequence and produces an encoded sequence (codeword) of n-bit blocks. However, each encoded block depends not only on the corresponding k-bit message block at the same time unit, but also on M previous blocks. Hence, the encoder has a memory length of m. Encoder operates on the incoming message sequence continuously in a serial manner.
The quantity k/n called the code rate, is a measure of code's efficiency . Other important parameter of convolutional code is the constraint length of the code and is defined by $L = k(m-1)$ The constraint length L represents the number of bits in the encoder memory that affect the generation of the n output bits. The error correction capacity is related with this value.
The number of bits' combinations in the registers is called the states of the code and are defined by number of states $N_s = 2L$, where L is the constraint length of the code.

The convolutional code structure is easy to draw from its parameters. First draw m boxes representing the m memory registers. Then n modulo-2 adders to represent the n output bits. Now connect the memory registers to the adders using the polynomial generator. The selection of which bits are going to be added to produce the output bit is called the polynomial generator g for that output bit.
For instance, in figure 1 is presented a small convolutional coder with code rate $k/n = 1/2$ and two memory elements. The *Output* 0 has a generator polynomial $g_0 = (111)$ and the *Output* 1 has a generator polynomial $g_1 = (101)$. The state transition and output values are presented in table 1.
The number of memory's registers determines the gain that the convolution code can achieve.

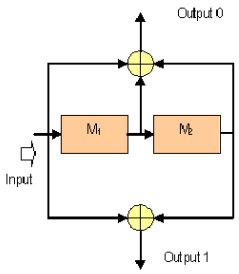

Fig. 1. Block diagram Convolutional Encoder $k/n = 1/2, m = 2$

Input bit	Current state	Next state	Output
0	0	0	0
1	0	1	3
0	1	2	2
1	1	3	1
0	2	0	3
1	2	1	0
0	3	2	1
1	3	3	2

Table 1. States transition and output value of convolutional encoder

However, the number of registers is limited by the decoding complexity of Viterbi algorithm. Because its complexity grows exponentially with the number of memory elements. IEEE 802.11a limited the number of elements to 6. The basic measure of channel coding performance is coding gain, which is usually measured in dB s as the reduction of required Signal To Noise Relation SNR to achieve a certain bit error rate (BER) in AWGN channel. The minimum free distance of the code determine the performance of the convolutional code. The coding gain is:

$$C_{gain} = 10log_{10}(CRd_{free}) \quad (1)$$

where CR is the coding rate and d_{free} is the free distance defined as the minimum Hamming distance between two different code words.

To improve the encoder's operation knowledge, three different graphical representation are available.

- Tree Diagram: The tree diagram attempts to show the passage of time as we go deeper into the tree branches. It is somewhat better than a state diagram but still not the preferred approach for representing convolutional codes. Here instead of jumping from one state to another, we go down branches of the tree depending on whether a 1 or 0 is received.
  Figure 2 shows the Tree Diagram for the example code.

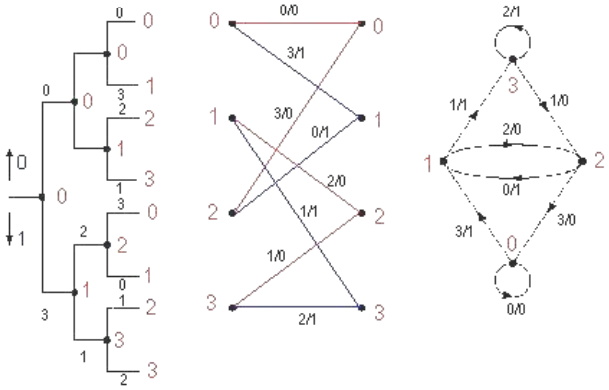

Fig. 2. Tree, Trellis and state diagram representation

- Trellis Diagram: Trellis diagrams are preferred over both the Tree and the State Diagrams because they represent linear time sequencing of events. The x-axis is discrete time and all possible states are shown on the y-axis. We move horizontally through the trellis with the passage of time. Each transition means that new bits have arrived. The Trellis Diagram is drawn by lining up all the possible states ($2L$) in the vertical axis. Then we connect each state to the next state by the allowable codewords for that state. There are only two choices possible at each state. These are determined by the arrival of either a 0 or a 1 bit. . The arrows going upwards represent a 0 bit and going downwards represent a 1 bit.
  We can draw the trellis for as many periods as we want. Each period repeats the possible transitions. We always begin at state 00. Starting from here, the Trellis expands and in L bits becomes fully populated such that all transitions are possible. The transitions then repeat from this point on.

- State Diagram: A state diagram for our example code is shown in the figure 2. Each node represents a state. At any time, the encoder resides in one of these states. The lines to and from it show state transitions that are possible as bits arrive. Only two events can happen at each time, arrival of a 1 bit or arrival of a 0 bit. Each of these two events allows the encoder to jump into a different state. The state diagram does not have time as a dimension and hence it tends to be not intuitive. The State Diagram contains the same information than this is in the table of states but it is a graphic representation

2.2. Decoding

There are several different approaches to decode convolutional codes. These are joined in two basic categories, Sequential decoding and Maximum Likelihood decoding (Viterbi decoding).
Both methods represent two different approaches of the same basic idea behind decoding.
Assuming that 3 bits were sent using a rate $1/2$ code, 6 bits were received. These six bits may or may not have errors. It is known from the encoding process that these bits map uniquely. So a 3 bits sequence will have a unique 6 bits output. But due to errors, any and all possible combinations of the 6 bits are received.
The permutation of 3 input bits results in 2^3 possible input sequences. Each of these has a unique mapping to a six bit output sequence by the code. These form the set of permissible sequences and the decoders task is to determine which one was sent. For example, if a wrong sequence is received, this received sequence can be compared with all permissible sequences and pick the one with the smallest Hamming distance(or bit disagreement).
If a message of s bits is received, then the possible number of codewords is 2^s.
Viterbi decoding is the best known implementation of the maximum likely-hood decoding.The principal used to reduce the choices is that the errors occur infrequently so that the probability of error is small and the probability of two errors in a row is much smaller than a single error, that is the errors are distributed randomly.
The Viterbi decoder examines an entire received sequence of a given length. The decoder computes a metric for each path and makes a decision based on this metric. All paths are followed until two paths converge on one node. Then the path with the higher metric is kept and the one with lower metric is discarded. The paths selected are called the survivors.
For an N bit sequence, the total number of possible received sequences is 2^N. Of only 2^{kL} these are valid. The Viterbi algorithm applies the Maximum Likelihood principles to limit the comparison to 2 to the power of kL surviving paths instead of checking all paths. The most common metric used is the Hamming distance metric, *Hard Decoding*. This is just the dot product between the received codeword and the allowable codeword.
Other option, is calculate the Eucledian distance, *Soft Decoding*. Soft decoding has lower probability of decoding error than hard decoding. But distance computation, for hard decoding, is easier, since it is based on bit-operations. There is a trade-off between probability of decoding error and computational complexity.
Viterbi algorithm is used in WLAN systems, where soft decoding is recommended because the performance improvement that it provides does not need any communications resources.
In OFDM systems, where the frequency response of the channel is known, the amplitude of individual carriers is incorporated into the Viterbi algorithm. The calculation of Euclidean distance is modified by this factor, as $p_n = |H_k|^2 \left|\hat{b}_n - b_n\right|^2$, meaningfully improving the performance in fading channel. This improve in WLAN applications is discussed in Section 4.

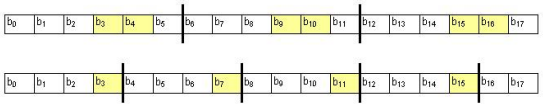

Fig. 3. Puncturing patterns of IEEE802.11a, $3/4$ and $2/3$ code rate.

2.3. Puncturing Convolutional Codes

The characteristics of a wireless channel typically vary with time, and therefore to obtain optimal performance it is necessary to adapt the error coding scheme to the changing channel characteristics. Code puncturing allows an encoder / decoder pair to change code rates, i.e., code error correction capabilities, without changing their basic structure.
Code puncturing involves not transmitting certain code bits. The encoder for a punctured code can be fabricated using the original low-rate convolutional encoder followed by a bit selector which deletes specific code bits according to a given puncturing rule. Only the bit selection rule is changed to generate different rates of codes. At the receiver side, a Viterbi decoder based on the mother code decoder is used for decoding the punctured codes of the family.
To decode different rate codes, only metrics are changed according to the same puncturing rule used by the encoder (the deleted bits are not counted when calculating the path metrics). Figure 3 shows the two different puncturing patterns of IEEE802.11a used to generate $3/4$ and $2/3$ code rate, both coming from the mother code of $1/2$ rate.

3. INTERLEAVING

An alternative to combat the effect of burst errors is interleaving. Interleaving simply involves interleaving symbols from two or more codewords before transmission on the channel. The number of codewords that are interleaved is referred to as the $depth$ of the interleaver, figure 4 shows an interleaver with an interleaving depth of $m = 6$ and a codeword length of $N = 8$. The data is written row-by-row into a $m \times N$ matrix and read out column-by-column by the interleaver before sending it over the channel. The reverse process is performed at the deinterleaver. Therefore, between successive symbols of any given codeword there are

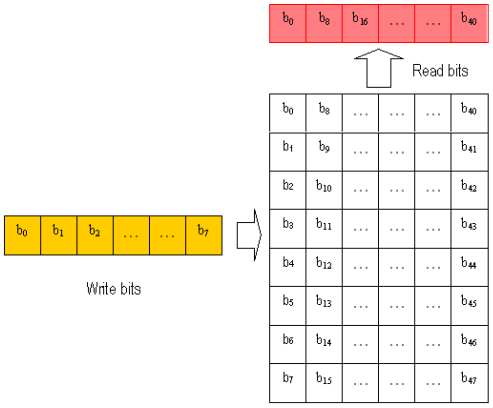

Fig. 4. Block Interleaver 8x6

$m-1$ symbols that belong to the $m-1$ other codewords being interleaved. If the interleaver has sufficient depth the fading processes that affect successive symbols belonging to the same codeword will be uncorrelated. Therefore, from the perspective of any single codeword, interleaving makes a burst error channel appear as one which has only random errors.
The use of interleaving results in extra delay because deinterleaving can be started only after all the interleaved data is received.
The interleaving chosen is function of the type of the channel and the coding technique used. In Gaussian channels, the error distribution can not be changed by relocating the bits so that interleaving is not useful.

3.1. Frequency Interleaving

Frequency interleaving is used to exploit the frequency diversity in wide-band transmissions. After frequency interleaving, the local deep fading is averaged over the whole bandwidth of the system. The frequency interleaving should be implemented for all the data symbols in a single OFDM symbol. This means, that the data symbols of two neighbouring OFDM symbols should not be interleaved in one iteration. For this reason, the dimension of the frequency interleaver should be equal to the number of data symbols in a single OFDM symbol, which means $N_c = N_{carrier} \times bits_{carrier}$. Frequency Interleaving is used in IEEE 802.11a standard, where the depth has been defined to be equal to one OFDM symbol providing an important improvement of the system. The combined effect of interleaving and convolutional channel coding takes advantage of the frequency diversity provided by the wideband nature of the transmitted signal.

Data Rate	6, 9, 12, 18, 24, 36, 48, 54 Mbps
Code	Convolutional Code
Code rate	$1/2$, $2/3$, $3/4$
No. Subcarrier	52
No. pilot tones	4
OFDM symbol duration	$4\mu s$

Table 2. Simulation parameters IEEE802.11a

3.2. Time Interleaving

Time interleaving is used to exploit the time diversity of the channel. After the time interleaving, the local time deep fading in some OFDM symbols is averaged over all OFDM symbols. The time interleaving depth should be larger than the maximum burst-error in time domain. Time interleaving is not applied in WLAN systems, because the slowly fading characteristics of the channel.

4. SIMULATIONS - CODING AND INTERLEAVING APPLIED IN IEEE802.11A

Using the Simulation Software, WLAN Simulator, provided by [1], the performance of coding and interleavers were evaluated.

Some important parameters of IEEE802.11a, which are included in the simulations, are presented in the table 2.

In IEEE802.11a, 8 different data rate are defined. Data Rate is a function of the modulation (BPSK, QPSK, 16-QAM and 64-QAM) and the code rate. The data rate is calculated using,

$$\text{Data rate} = \frac{bits_{carrier} N_{carrier} CR}{T_{OFDM}} \qquad (2)$$

where $bits_{carrier}$ is the number of bits per carrier, i.e 1 for BPSK, $N_{carrier}$ is the number of subcarriers with information (48 in IEEE802.11a), CR is the code rate and T_{OFDM} is the OFDM symbol duration.
In the first simulation, the Bit Error Rate, BER, is evaluated for BPSK systems without coding and QPSK system with convolutional coding. In both cases interleaver of 1 OFDM symbol is considered. The data rate is defined in 12 Mbps. This value is obtained without coding in BPSK modulation, and with $1/2$ code rate if QPSK is considered. The figure 5 present the BER for both systems in Gaussian for different signal to noise relation (SNR). From the simulation it is possible to reach the conclusion that with the same data rate and the same BER, the system which use coding needs 3 dB of energy less than the system without coding.
In other point of view, transmitting the same power, the distance between the coded system transmitter and receiver can be incremented. For example, in an indoor environment,

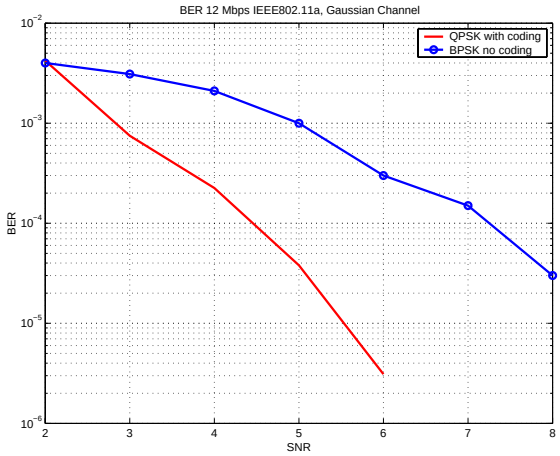

Fig. 5. BER in Gaussian channel

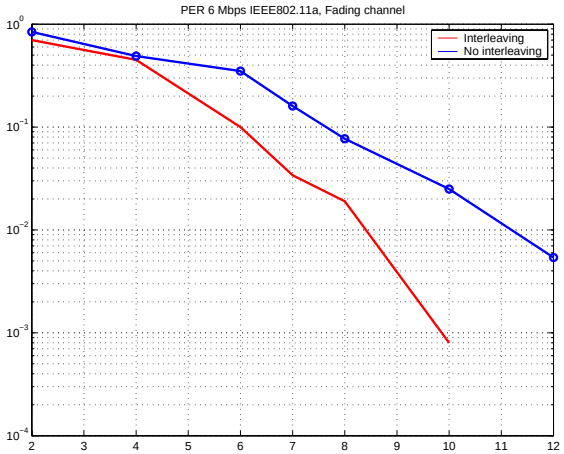

Fig. 6. PER with and without interleaving in spread Rayleigh fading channel

where the path loss is $L_p \propto 1/d^3$, the transmitter power decrease to $1/8$ or by $9dB$ for every doubling the distance. In our system, at BER=3.10^{-2}, $2.9dB$ is the gain, the distance can be incremented in an 25% in one indoor enviroment.

The second simulation studies the interleaving performance in a fading channel. The data rate used is 6 Mbps with BPSK modulation ($1/2$ code rate). The BER and PER, figures 7 and 6, are evaluated with and without interleaver for different SNR.

The improvement of interleaver is more evident in the Packet Error Rate (PER) curve, figure 6. For example, at equal PER, 3.10^{-2}, the system with interleaver need $3dB$ less than the other. Finally, the effect of metric weighting in the soft decoding algorithm, $p_n = |H_k|^2 \left|\hat{b}_n - b_n\right|^2$, is evaluated. In a fading channel, the improve using metric weighting is significant. For example, a $SNR = 11dB$ the $PER = 0.02$ using weighting and $PER = 0.4$ using conventional soft decoding. The effect of weighting is to reduce the impact of the bits that are transmitted in bad carriers (low $|H_k|^2$) on the decision that the decoder makes, improving the detection capacity.

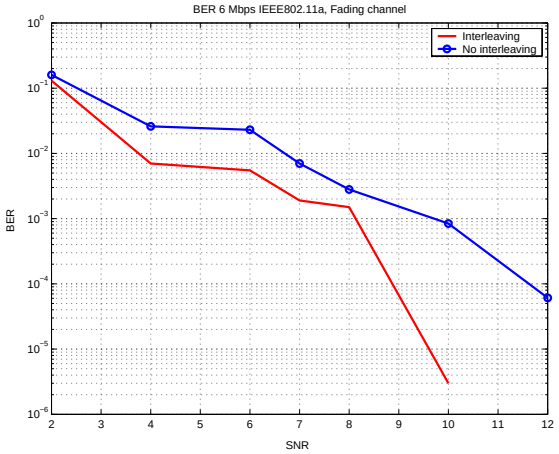

Fig. 7. BER with and without interleaving in spread Rayleigh fading channel

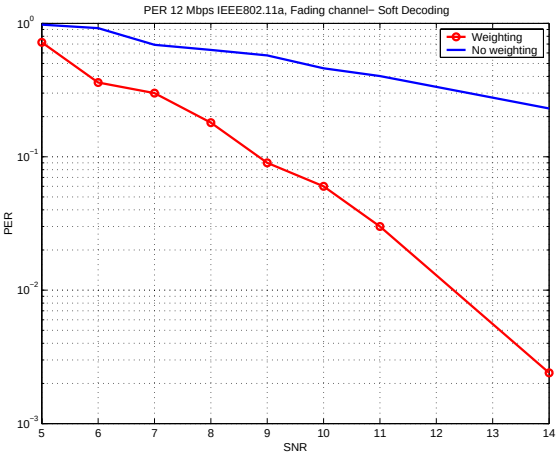

Fig. 8. Packet error rate effect of metric weighting in Viterbi decoding

5. CONCLUSIONS

In this paper coding and interleaving techniques was studied and their performance were evaluated in a WLAN environment. This work shows the great importance of this techniques to improve the quality of service in a IEEE802.11 service.

It is possible that other techniques, as Turbo Coding [3], Trellis Modulation [4] and Concatenated codes, can be incorporated in future WLAN standards but it is necessary to evaluate the improvement and the increment in the complexity that their incorporation produces.

6. REFERENCES

[1] Heiskala J. and Terry J., *OFDM Wireless LANs: A theoretical and Practical Guide*, Sams Publishing, 2002.

[2] Glover I. and Grant P., *Digital Communication*, Prentice Hall, 2004.

[3] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error-Correcting Code: Turbo Code," Proceedings of the 1993 IEEE International Conference on Communications, 10641070 (Geneva, Switzerland, May 1993).

[4] Biglieri E., Divsalar D., McLane P., Simon M, *Introduction to Trellis-Coded Modulation with Applications*,, Macmillan, 1991