

Bit-Flipping Decoding

Navid Azizi

March 20, 2003

1 Introduction

Product codes are codes, where the codewords are arranged in a matrix, where each row of the matrix is an element of linear code C_1 and each column of the matrix is an element of the linear code C_2 .

Decoding can be performed by bit-flipping, where each bit decides if it should be flipped given messages from its row/column decoders. The decoding iterates until all errors have been corrected or a maximum number of iterations is reached.

Error patterns which are spread along different rows and columns should be easily correctable using this type of decoder. Problems, however, arise when there are many errors along the same rows or columns; for example, take 4 errors that are in the same two rows and columns. In these cases it is hoped that the either through iteration, or through extra information provided by the decoders, the correct bits will be flipped.

In this report the error-correcting properties of product codes composed of two of the following codes are analyzed through computer simulation.

- Single-Parity-Check code of length 5 (S5)
- Single-Parity-Check code of length 10 (S10)
- The [7, 4] Hamming Code (H7)
- The [15, 11] Hamming Code (H15)
- The [8, 4] Extended Hamming Code (X8)
- The [16, 11] Extended Hamming Code (X16)

2 System

A C program was developed to simulate a communication system and measure the Frame-Error-Rate (FER) and Bit-Error-Rate (BER) of Hamming codes. The system was divided into four modules: the encoder, the channel, the decoder, of which only the decoder is interesting and will be discussed below.

2.1 Decoder

Four different decoders were designed to test the sensitivity of the BER and FER to the decoder. Of the four, the most complex is a generalization of the simpler three and thus the architecture for the complex decoder (D4) will be explained first.

Each row/column of the message matrix is sent to the appropriate decoder for C_1 or C_2 . If the decoder detects that there is an error along the row/column it sends back information to the bits of the respective row/column. What information is sent back depends on what type of code is being detected.

2.1.1 Single-Parity-Check Sub-Decoder

If the code being detected is a Single-Parity-Check (SPC) code, then the decoder can only determine if there is an error in the received word. In this case the decoder tells each bit involved in the row/column word that there exists a possibility of error. The bit, after receiving this message, will increment its ERROR counter.

2.1.2 Hamming Sub-Decoder

Hamming codes are single-error-correcting codes, so not only can they detect errors, but they can also decide which bit needs to be corrected. The corrected bit will be the wrong bit, however, if there were two or more errors in the received word. Thus, Hamming codes do not always correct the right bit. In that regard, the Hamming decoder sends back two messages to each bit involved in the decoding process.

If the Hamming decoder detects an error, just as with the SPC decoder, the decoder tells each bit involved in the row/column word that there exists a possibility of error. The bits, after receiving this information, will increment their ERROR counter. Additionally, the Hamming decoder, tells the bit that the decoder guesses should be corrected that it may be the one that needs to be corrected. In this case the bit increments its GUESS counter.

2.1.3 Extended Hamming Sub-Decoder

Extended Hamming codes are Hamming codes protected with an additional parity bit. Thus the decoder has two sources of information when decoding the code: it has the syndrome from the original Hamming code, and the parity check. We can enumerate the different possibilities:

1. The syndrome is zero and the parity check is passed.
There are no errors in the word
2. The syndrome is zero and the parity check fails.
An error has occurred in the word, but the decoder does not know where.

3. The syndrome is non-zero and the parity check is passed.
There is an even number of errors, and thus the bit that the Hamming word would correct is incorrect.
4. The syndrome is non-zero and the parity check fails.
There is an odd number of errors. There is strong possibility that there was a single-bit error because if there were three errors the syndrome would be zero ($d_{min} = 3$). Thus the bit that the Hamming word would correct is very likely to be the right bit to flip

The decoder will communicate differently with the bits involved in the row/column word for each of the above four cases.

1. No information is passed back to the bits.
2. The decoder tells each bit that there exists a possibility of error. The bits will increment their ERROR counter.
3. The decoder tells each bit that there exists a possibility of error. The bits will increment their ERROR counter. The Hamming correction information will be discarded.
4. The decoder tells each bit that there exists a possibility of error. The bits will increment their ERROR counter. Furthermore, the decoder will communicate to the bit that the Hamming code predicts is the error. The bit will increment its E_GUESS counter. Note the difference between this counter and the GUESS counter used with the normal Hamming decoder. The guess provided here is a much stronger guess than the one provided by the simple Hamming decoder.

2.1.4 Iterative Decoding

Once the bits have communicated with all decoders, they have values for their ERROR, GUESS and E_GUESS counters. Using these three counters, the bits can generate a score for themselves as follows:

$$\text{SCORE} = \text{GUESS} * \text{GUESS_POINTS} + \text{ERROR} * \text{ERROR_POINTS} + \text{E_GUESS} * \text{E_GUESS_POINTS}$$

There are a limited number of possible scores, and before the decoding process even starts the possible scores can be ordered from high to low. In the original decoding step the threshold score is chosen to be the largest score possible, and over every iteration the threshold score is chosen to be the next score in the ordered list.

Thus, in each iterating step, each bit decides if it should be "ipped depending on if its score is larger than or equal to the threshold score in that iteration. Hopefully this will lead to the received word being corrected.

2.1.5 Decoder 1 (D1)

This decoder uses the above framework but sets `GUESS_POINTS` and `E_GUESS_POINTS` to zero. Thus, the decoder effectively reduces to a simple decoder where each sub-decoder tells the row/column if there was an error and bits flip themselves if there are two or more errors, (and after a couple iterations, when there is one error).

2.1.6 Decoder 2 (D2)

This decoder uses the above framework but sets `GUESS_POINTS` to equal `E_GUESS_POINTS` which is much larger than `ERROR_POINTS`. Thus, the guesses provided by the Hamming and Extended Hamming decoders play a much larger role. This decoder can also be seen as approximately being the decoder where the Hamming decoders only relay a message to the bits that they think are flipped.

2.1.7 Decoder 3 (D3)

This decoder uses the above framework but sets `GUESS_POINTS` to one, `ERROR_POINTS` to two, and `E_GUESS_POINTS` to four¹.

2.1.8 Decoder 4 (D4)

This decoder uses the same point system as D3, but is fundamentally different. In fact it removes the paradigm that bits decide independently that they should flip or not. Instead of using a threshold score that changes from iteration to iteration, the threshold score is chosen to be the maximum score from within the bits. One other small change was used to make this decoding scheme more accurate: for the bits that have their `ERROR` counter set to two, and their `GUESS` counter set to one, only bits with this configuration can be flipped along one row at a time (instead of all bits with this configuration being flipped). This decoder, as will be seen in the results section, performs much better than the other decoders, and thus the small communication between bits to find the maximum score is seen as worthy.

3 Results and Discussion

Figure 1 shows the FER for Decoder 1 for a subset of the codes. It is seen that since the Hamming Codes and the Extended Hamming sub-decoders used with this decoder do not give any extra information compared to the SPC sub-decoders. The best codes are the ones that have the smaller block lengths. The H7H7, H7X8, H75S, X8S5, and S5S5 have FER that are very similar. The fact

¹Since we are using 2-dimensional product codes, `GUESS_POINTS` has been set to 2^0 , `ERROR_POINTS` has been set to 2^1 and `E_GUESS_POINTS` has been set to 2^2 . This guarantees that `ERRORs` are worth more than `GUESSES` and so forth. If higher dimension codes were used, then the base could be changed to the dimension to obtain the same ordering.

²Probabilities that give rise to FERs of 10^{-4} will be summarized below for all codes.

that the smaller block length codes have the best FER is consistent since we are not using the stronger error correction capability of the Hamming and Extended Hamming codes. By using a larger code we are making it possible to have more errors in the block when the block is traversing the channel and thus making it harder to correct.

Figure 2 shows the FER for Decoder 2 for a subset of the codes. Here we see that the best performing code is the X8X8 followed by the H7H7 and X8X16 code. The next best performing codes are the ones that combine Hamming codes with Extended Hamming codes, and then Extended Hamming codes with SPC codes. The worst codes are S10S10 and S5S5, but the H15S5 code has almost an identical FER as the S5S5 code showing that with such a long sub-codeword, the Hamming code can give the wrong guess reducing its FER.

Figure 3 shows the FER for Decoder 3 for a subset of the codes. Here we see that the best performing code is the X8X8 followed by the X16X16 and then the H7H7 code. The worst codes are S10S10 and S5S5. We see that by not weighing the guess from the Hamming code as much as that of the Extended Hamming code we are betting the FER of the codes that involve a Hamming decoder.

Finally, Figure 4 shows the FER for D4 for a subset of the codes. Looking at just this figure, there does not seem to be any different trend compared to that of D3. We will see below, however, that the FER for some codes is better when using D4.

3.1 Decoder Comparison

In Figures 5 through 9 a comparison of the H7H7, X8X8, H7X8, H7S5, X8S5 codes for all four decoders is shown. The FER of the S5S5 code is shown on each graph as a point of comparison.

It is seen that when using D1, none of the codes are any better than the S5S5 code. Thus it is important for the Hamming and Extended Hamming Codes to return more information than just that there is an error in the code.

Furthermore when comparing D2 and D3, we see that in the H7S5 case D2 performs much worse than D3. This is because in D2 the Hamming guesses are weighed a lot, and the guesses might be wrong. In D3, the Hamming guesses are weighed minimally, so the other perpendicular code can help in recovering from two errors on the same row. In the H7H7 graph the distinction between D2 and D3 is less pronounced because of the reduced possibility two errors in both a row and a column.

The X8S5 code sees no difference between D2 and D3 since Hamming codes, which are scored differently in D2 and D3, are not used in the product code. It would be expected that the X8X8 code also not be affected by the move from D2 to D3, however, there is a slight improvement. This improvement is the byproduct of extra iterations in D3³

³As a result of a higher number of possible scores.

The H7X8 code sees virtually no difference between D2 and D3. It is expected that the improvement of the Extended Hamming decoders and the degradation of the Hamming decoders in D3 cancel each other out.

Finally, when comparing D4 to D3 we see that there is no change seen when an SPC code is involved, there is an improvement when a Hamming code is involved and there is a degradation in the X8X8 case. The extra improvement in the H7H7 and H7X8 codes is believed to come, not from the dynamic maximum score determination, but rather by how the decoder handles multiple bits which have their ERROR counter set to two and their GUESS counter set to one. Without this special step, Hamming codes can loop between equivalent error patterns on each iteration, and thus not find a correct solution. The degradation in the X8X8 code is, however, surprising.

3.2 Numerical Results

Tables 1 and 2 show the interpolated channel probability that leads to an FER of 10^{-4} for the decoders. The rate of the code is also shown, and the theoretically largest value of cross-over probability that leads to zero error probability at the given code rate (the Shannon limit) as a point of comparison. The best codes in terms of FER are :

1. X8X8 with D3. FER of 10^{-4} at $p = 2.9 \times 10^{-2}$
2. X8X8 with D2. FER of 10^{-4} at $p = 2.2 \times 10^{-2}$
3. H7H7 with D4. FER of 10^{-4} at $p = 2.0 \times 10^{-2}$

Even these best codes are almost an order of magnitude worse than the Shannon Limit. Most codes are closer to two orders of magnitude worse than the Shannon Limit. The best codes near a rate of 1/2, which are an order of magnitude worse than the designed codes, are :

1. X16X16 with D3. Rate of 0.473. FER of 10^{-4} at $p = 1.3 \times 10^{-2}$
2. H7H15 with D2. Rate of 0.419. FER of 10^{-4} at $p = 9.0 \times 10^{-3}$
3. H15H15 with D2. Rate of 0.538. FER of 10^{-4} at $p = 5.9 \times 10^{-3}$

4 Conclusion

This report documents the simulation of various product codes with different bit-flipping decoding techniques. It was seen that Hamming decoders could be advantageous over SPC codes if they communicate their guess of which bit must be flipped to that single bit; if on the other hand Hamming decoder only detect which sub-words have errors then they perform worse than SPC codes. If Hamming decoders could send a non-binary message back to the bits of the sub-word, then they become even more advantageous. This non-binary message

to a single-bit would include if the bit is part of a sub-word that has an error and if the decoder believes that the bit is the error itself.

Extended Hamming decoders should use the same non-binary message to obtain the full advantage of using them. Extended Hamming decoders are more powerful than Hamming decoders because their guess about which bit should be corrected has a higher probability of being correct.

Furthermore this report presented four different ways that a bit can decide to flip or not. Different codes were optimal under different decoding strategies.

Table 1: Channel Probability that leads to FER of 10^{-4} for D1 and D2

Decoder	Code	Rate	Channel Probability	Probability at Shannon Limit
D1	H7H7	0.327	3.5E-4	1.77E-1
	H7H15	0.419	1.0E-4	1.39E-1
	H7X8	0.286	8.4E-5	1.96E-1
	H7X16	0.393	9.5E-5	1.49E-1
	H7S5	0.457	3.7E-4	1.25E-1
	H7S10	0.514	1.2E-4	1.05E-1
	H15H15	0.538	1.9E-4	9.78E-2
	H15X8	0.367	7.4E-5	1.59E-1
	H15X16	0.504	2.2E-5	1.09E-1
	H15S5	0.587	6.7E-5	8.30E-2
	H15S10	0.660	4.7E-5	6.31E-2
	X8X8	0.250	2.8E-4	2.15E-1
	X8X16	0.343	8.1E-5	1.70E-1
	X8S5	0.400	3.1E-4	1.46E-1
	X8S10	0.450	1.4E-4	1.27E-1
	X16X16	0.473	2.1E-5	1.19E-1
	X16S5	0.550	9.9E-5	9.40E-2
	X16S10	0.619	4.7E-5	7.41E-2
	S5S5	0.640	4.7E-4	6.82E-2
	S5S10	0.720	2.2E-4	4.85E-2
S10S10	0.810	1.2E-4	2.91E-2	
D2	H7H7	0.327	1.5E-2	1.77E-1
	H7H15	0.419	9.0E-3	1.39E-1
	H7X8	0.286	8.9E-3	1.96E-1
	H7X16	0.393	1.0E-2	1.49E-1
	H7S5	0.457	1.0E-3	1.25E-1
	H7S10	0.514	7.7E-4	1.05E-1
	H15H15	0.538	5.9E-3	9.78E-2
	H15X8	0.367	9.7E-3	1.59E-1
	H15X16	0.504	6.6E-3	1.09E-1
	H15S5	0.587	4.0E-4	8.30E-2
	H15S10	0.660	3.2E-4	6.31E-2
	X8X8	0.250	2.2E-2	2.15E-1
	X8X16	0.343	1.4E-2	1.70E-1
	X8S5	0.400	5.3E-3	1.46E-1
	X8S10	0.450	3.2E-3	1.27E-1
	X16X16	0.473	8.2E-3	1.19E-1
	X16S5	0.550	2.6E-3	9.40E-2
	X16S10	0.619	1.8E-3	7.41E-2
	S5S5	0.640	4.7E-4	6.82E-2
	S5S10	0.720	2.2E-4	4.85E-2
S10S10	0.810	1.2E-4	2.91E-2	

Table 2: Channel Probability that leads to FER of 10^{-4} for D3 and D4

Decoder	Code	Rate	Channel Probability	Probability at Shannon Limit
D3	H7H7	0.327	1.6E-2	1.77E-1
	H7H15	0.419	8.7E-3	1.39E-1
	H7X8	0.286	9.6E-3	1.96E-1
	H7X16	0.393	1.0E-2	1.49E-1
	H7S5	0.457	3.7E-3	1.25E-1
	H7S10	0.514	2.3E-3	1.05E-1
	H15H15	0.538	5.6E-3	9.78E-2
	H15X8	0.367	1.1E-2	1.59E-1
	H15X16	0.504	5.9E-3	1.09E-1
	H15S5	0.587	2.1E-3	8.30E-2
	H15S10	0.660	1.2E-3	6.31E-2
	X8X8	0.250	2.9E-2	2.15E-1
	X8X16	0.343	1.9E-2	1.70E-1
	X8S5	0.400	4.9E-3	1.46E-1
	X8S10	0.450	3.2E-3	1.27E-1
	X16X16	0.473	1.3E-2	1.19E-1
	X16S5	0.550	2.7E-3	9.40E-2
	X16S10	0.619	1.9E-3	7.41E-2
	S5S5	0.640	3.9E-4	6.82E-2
	S5S10	0.720	2.5E-4	4.85E-2
S10S10	0.810	1.2E-4	2.91E-2	
D4	H7H7	0.327	2.0E-2	1.77E-1
	H7H15	0.419	1.3E-2	1.39E-1
	H7X8	0.286	1.3E-2	1.96E-1
	H7X16	0.393	1.2E-2	1.49E-1
	H7S5	0.457	4.0E-3	1.25E-1
	H7S10	0.514	2.5E-3	1.05E-1
	H15H15	0.538	8.8E-3	9.78E-2
	H15X8	0.367	1.3E-2	1.59E-1
	H15X16	0.504	7.1E-3	1.09E-1
	H15S5	0.587	2.6E-3	8.30E-2
	H15S10	0.660	1.6E-3	6.31E-2
	X8X8	0.250	1.9E-2	2.15E-1
	X8X16	0.343	8.6E-3	1.70E-1
	X8S5	0.400	5.3E-3	1.46E-1
	X8S10	0.450	3.2E-3	1.27E-1
	X16X16	0.473	6.3E-3	1.19E-1
	X16S5	0.550	2.8E-3	9.40E-2
	X16S10	0.619	1.7E-3	7.74E-2
	S5S5	0.640	3.8E-4	6.82E-2
	S5S10	0.720	2.5E-4	4.85E-2
S10S10	0.810	1.2E-4	2.91E-2	

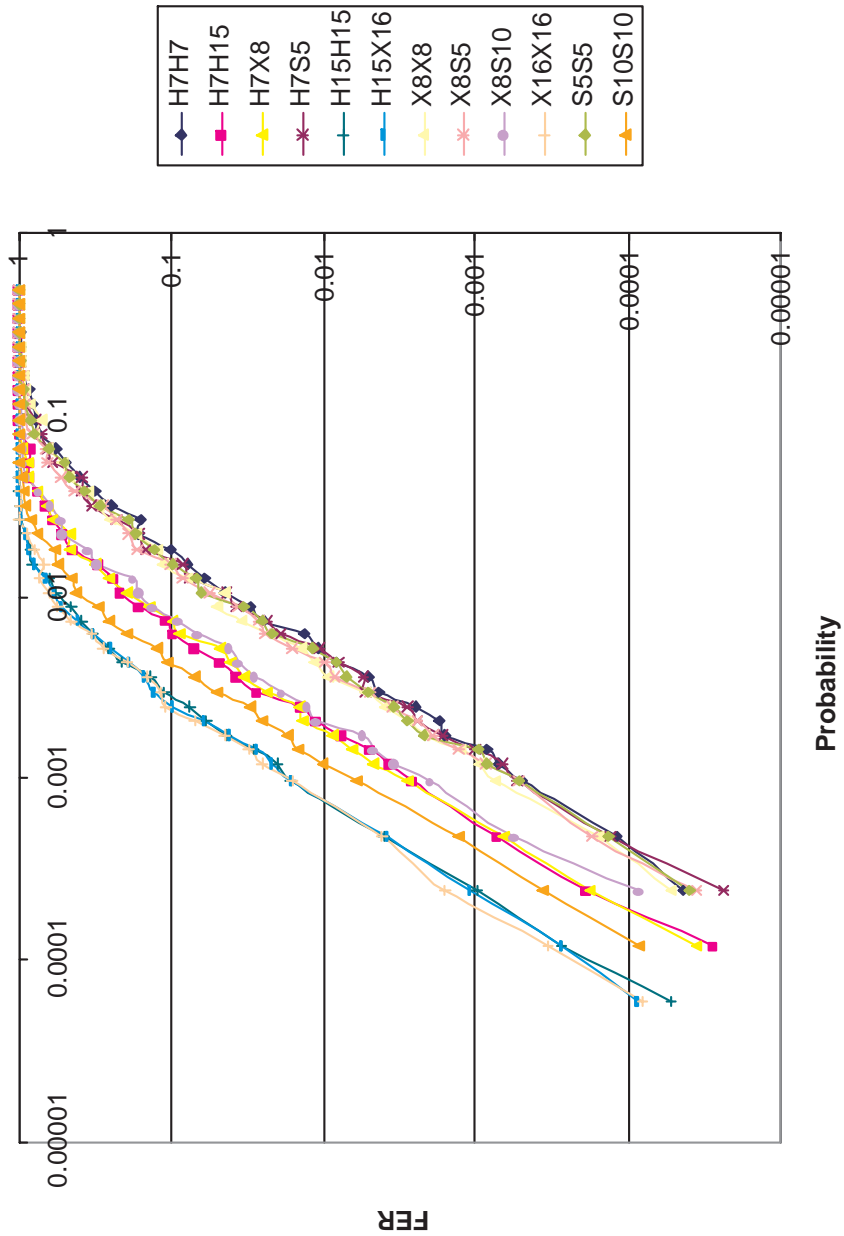


Figure 1: FER for the Decoder 1

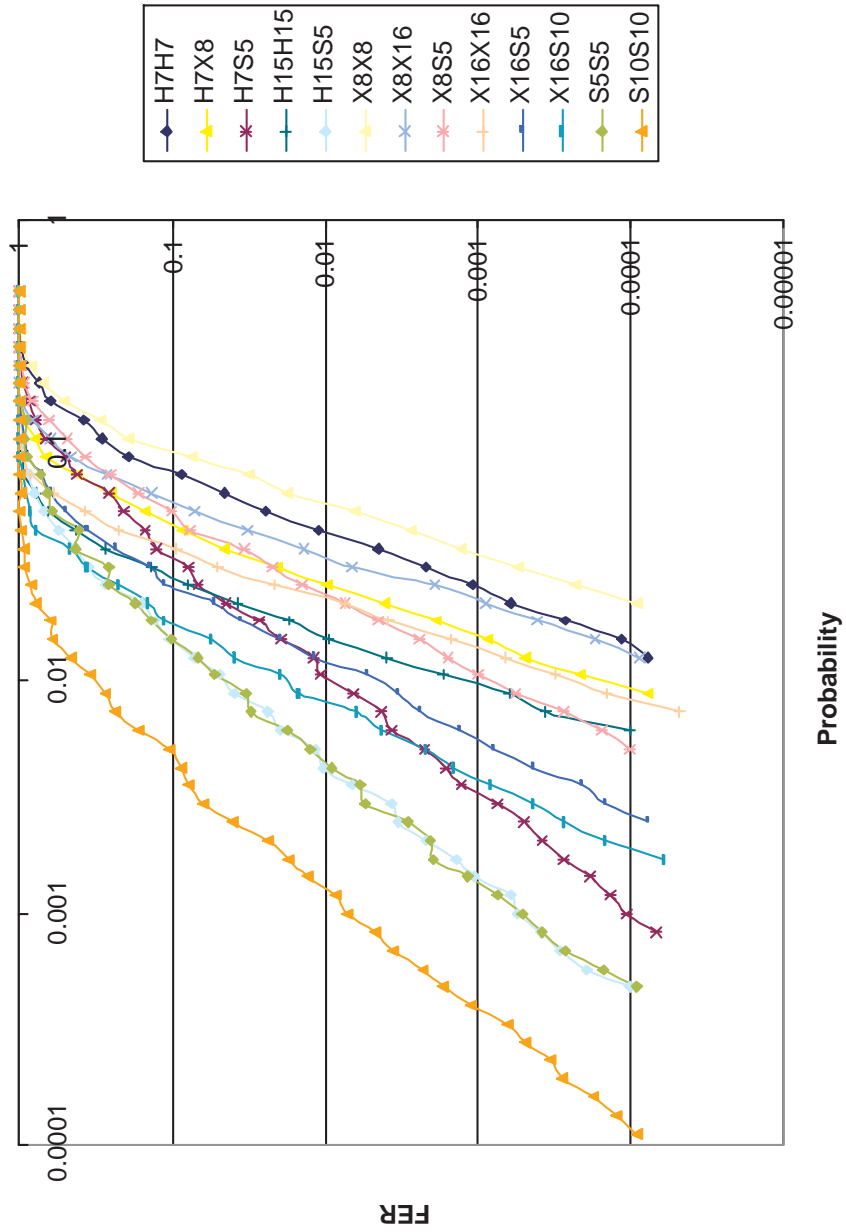


Figure 2: FER for the Decoder 2

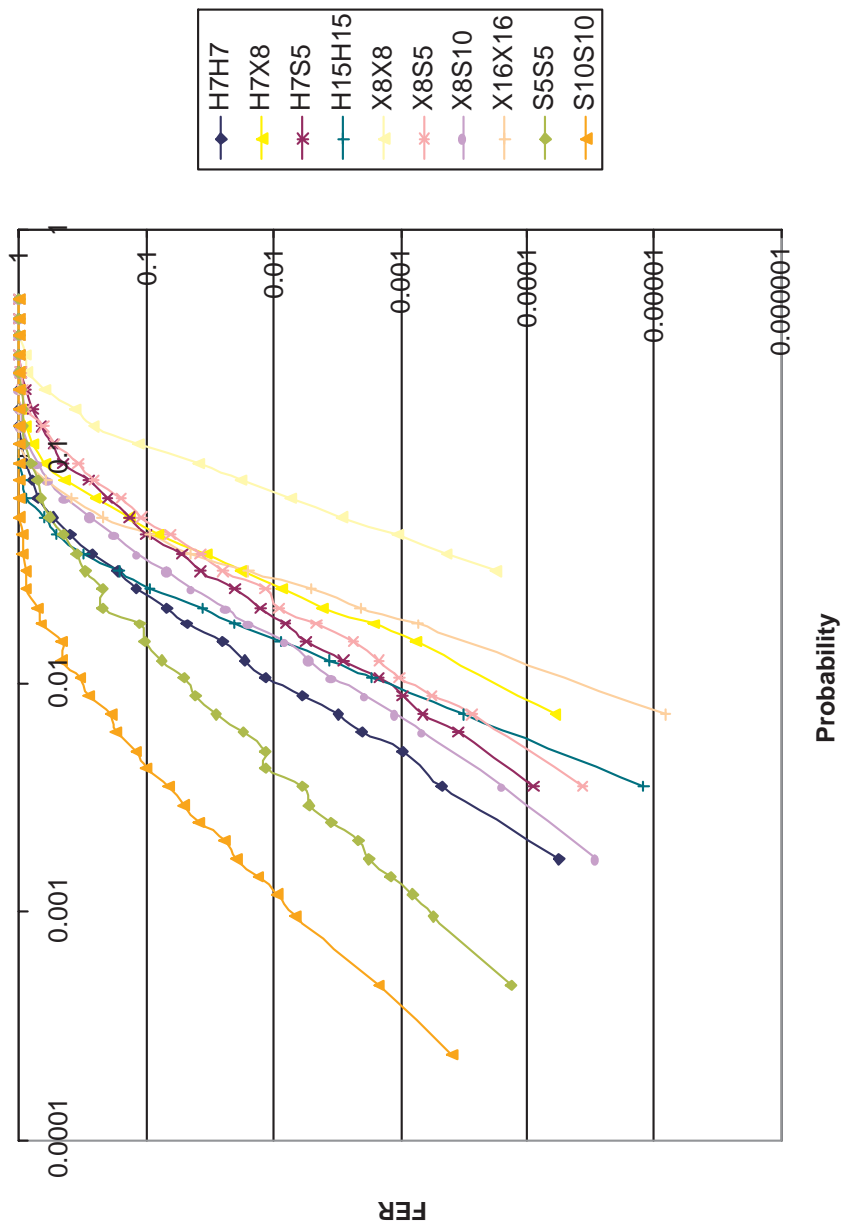


Figure 3: FER for the Decoder 3

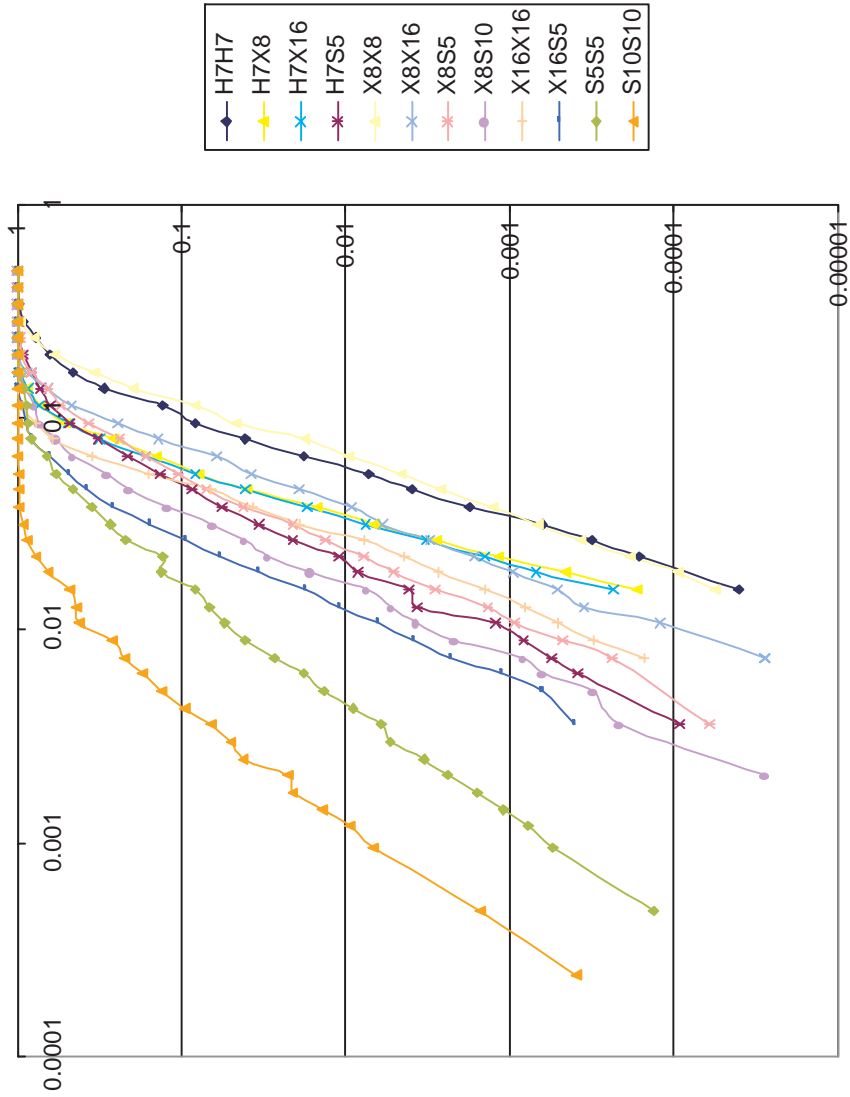


Figure 4: FER for the Decoder 4

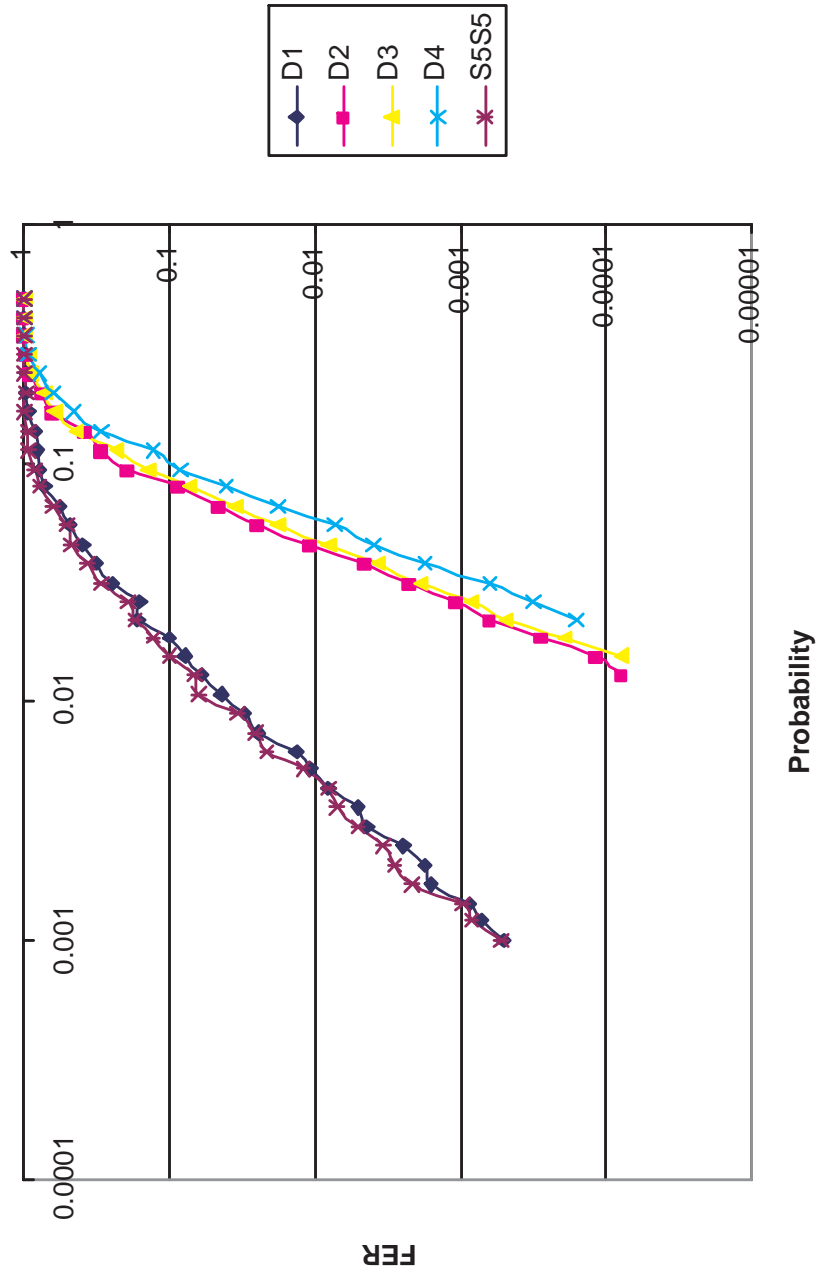


Figure 5: Comparison of Decoders for H7H7

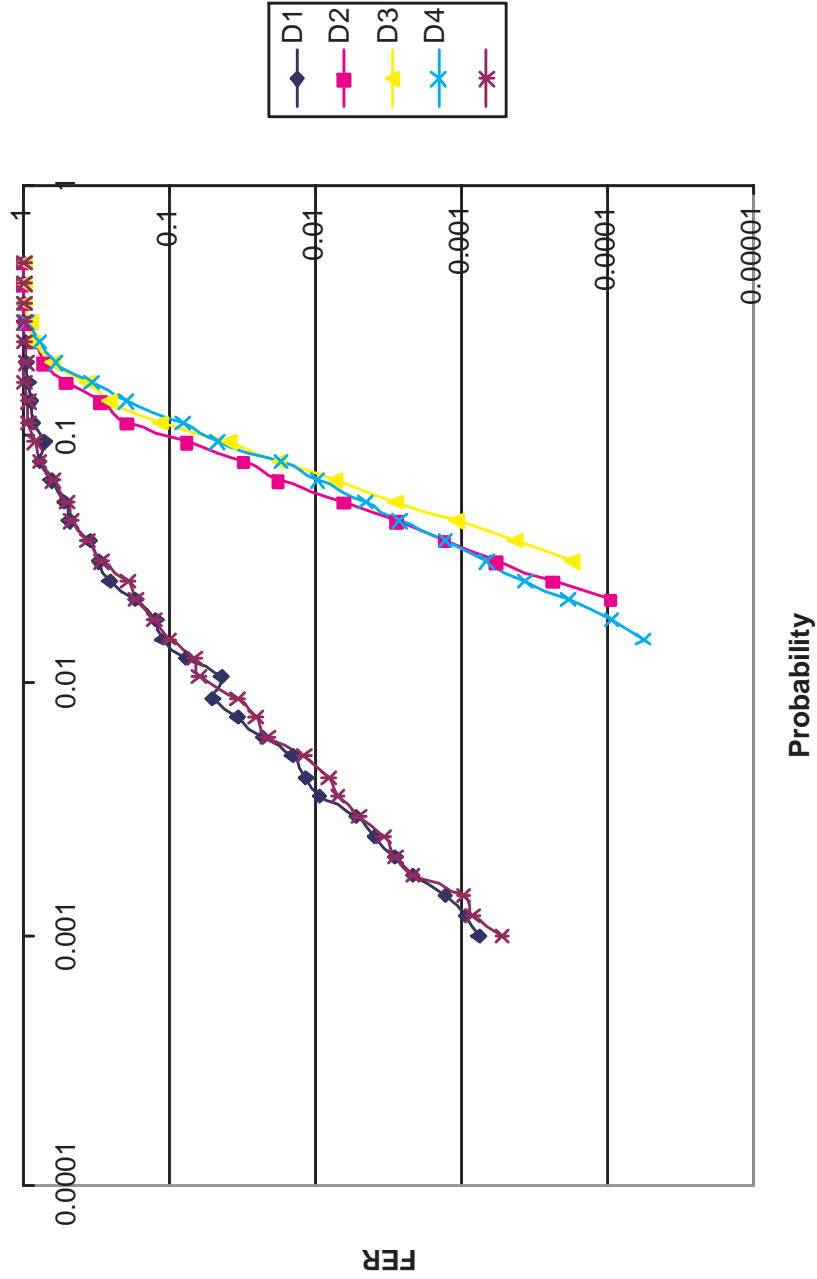


Figure 6: Comparison of Decoders for X8X8

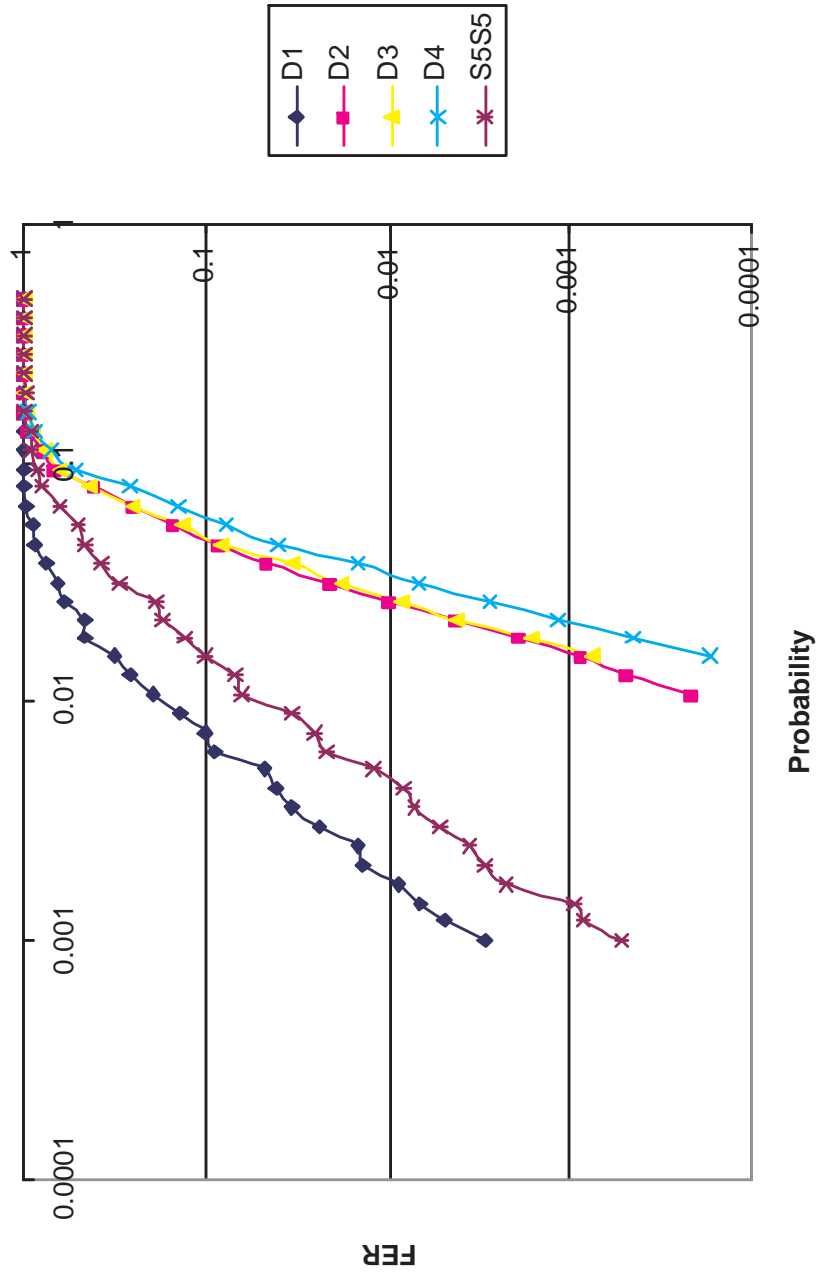


Figure 7: Comparison of Decoders for H7X8

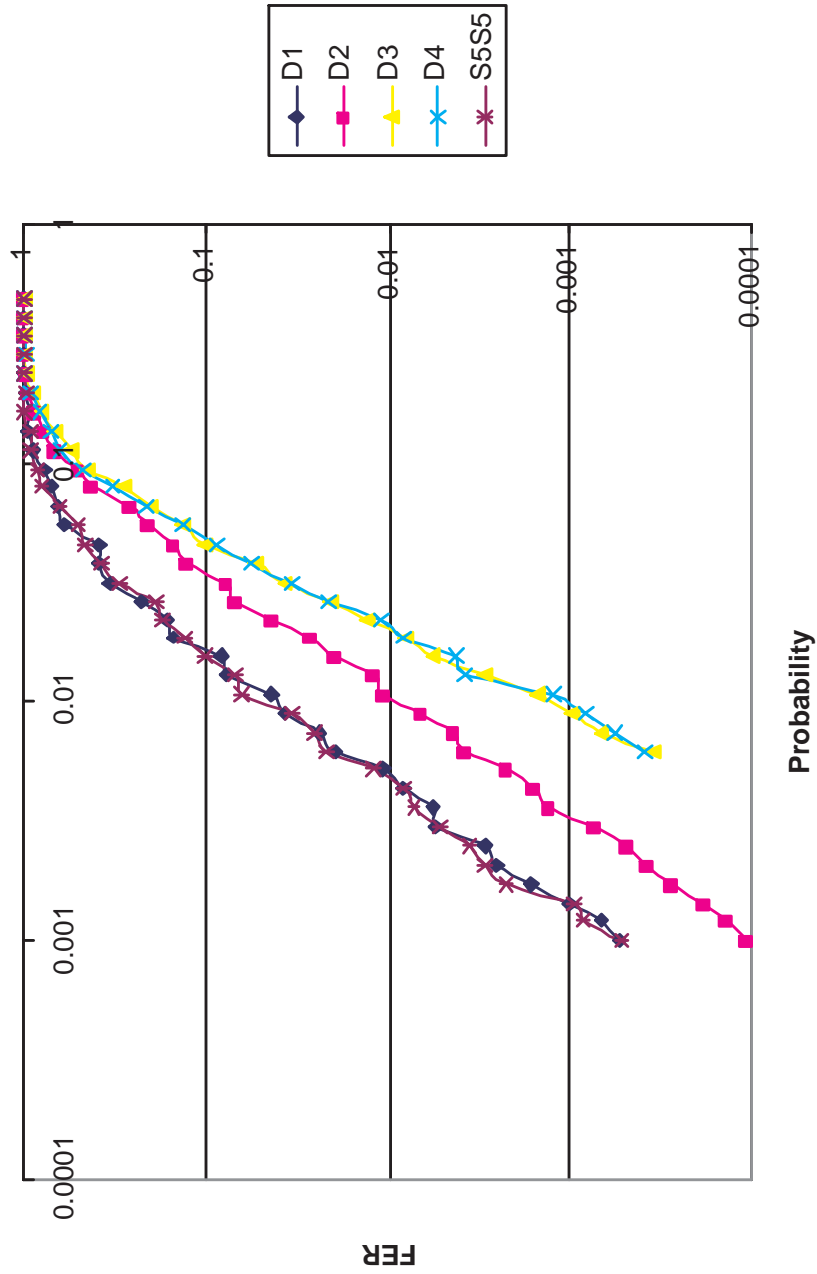


Figure 8: Comparison of Decoders for H7S5

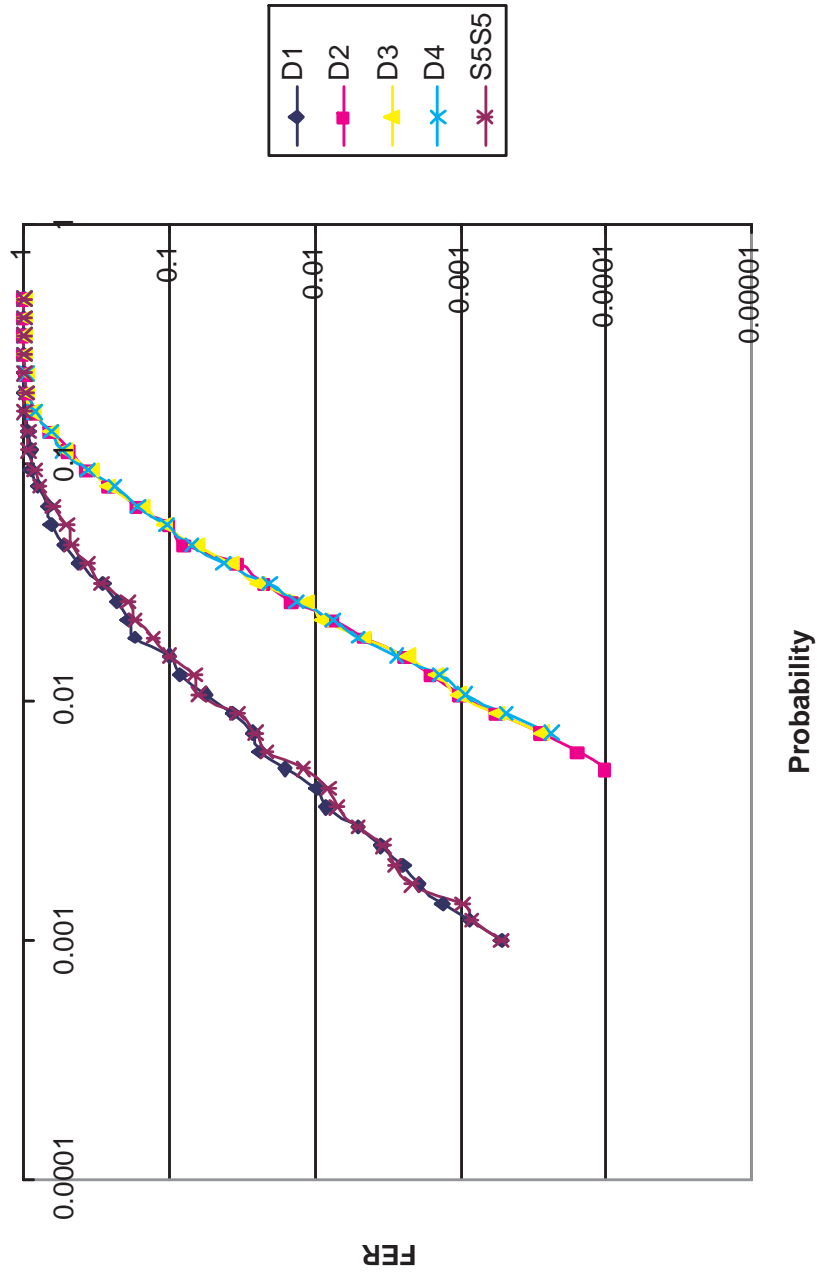


Figure 9: Comparison of Decoders for X8S5