

Copyright © 2004

What a wonderful *turbo* world ...

By

Dr. Sorin Adrian Barbulescu
abarbulescu@myoffice.net.au

Version 1.2

ISBN 0-9580520-0-X

Copyright © 2004

What a wonderful *turbo* world ...

To Cristina and Toni

Disclaimer

YOU EXPRESSLY ACKNOWLEDGE AND AGREE THAT USING THE INFORMATION AND SOFTWARE FROM THIS BOOK IS AT YOUR SOLE RISK AND THAT THE ENTIRE RISK AS TO SATISFACTORY QUALITY, PERFORMANCE, ACCURACY AND EFFORT IS WITH YOU.

IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DAMAGES WHATSOEVER, INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, LOSS OF DATA, BUSINESS INTERRUPTION OR ANY OTHER COMMERCIAL DAMAGES OR LOSSES, ARISING OUT OF OR RELATED TO YOUR USE OR INABILITY TO USE THE SOFTWARE PROVIDED IN THIS BOOK.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT ANY WARRANTY OF ANY KIND.

YOU MAY MAKE ONLY ONE COPY OF THIS BOOK AND SOFTWARE IN MACHINE-READABLE FORM FOR BACKUP PURPOSES ONLY.

YOU MAY NOT REPRODUCE, RENT, LEASE, LEND OR SUBLICENSE PART OR WHOLE OF THE INFORMATION AND SOFTWARE FROM THIS BOOK.

Table of Contents

1	Introduction	15
1.1	Communication System	16
1.2	Channel coding: a new turbo world	20
1.3	Convolutional encoding and Viterbi decoding	21
1.4	The paradigm shift: Turbo Codes	23
1.5	Brief History of the Iterative Decoding Process	24
1.6	In the Previous Millennium	26
1.7	Coding Gain vs Block Size	28
1.7.1	Capacity limits for unlimited information block size	28
1.7.2	Asymptotic bounds function of information block size	28
1.7.3	Performance of actual codes	29
2	Encoder Architectures	30
2.1	PCC architecture (Turbo Codes)	30
2.2	SCC architecture	32
2.3	HCC architecture	34
2.4	Block Turbo Codes (BTC)	35
2.4.1	Turbo Product Codes (TPC)	36
2.4.2	Satellite IP: Boeing (Connexion)	37
2.4.3	iPSTAR	38
2.5	Low-Density Parity-Check Codes	39
2.5.1	V-LDPC	41
2.5.2	Comtech LDPC/BCH	41
2.5.3	Tornado Codes	42
3	Interleaver Design	44
3.1	“Row-Column” interleaver	44
3.2	“Helical” interleaver	44
3.3	“Odd-even” interleaver	44
3.4	“Simile” interleaver	46
3.5	“Frame” interleaver	48
3.6	Pseudo-random interleavers	48
3.7	“S-type” interleavers	48

What a wonderful *turbo* world ...

3.8	“Uniform” interleavers	48
3.9	Convolutional interleavers.....	49
3.10	Code Matched Interleaver.....	49
3.11	Chaotic Interleaver.....	50
3.12	The 3G interleaver	50
3.13	Non-block interleavers	50
3.14	The best interleaver	51
4	Turbo Decoding	53
4.1	Receiver architecture.....	53
4.2	Iterative Decoding Principle	53
4.3	SOVA.....	57
4.4	MAP	58
4.4.1	Description of the MAP algorithm	58
4.4.2	Description of the Log-MAP algorithm	60
4.4.3	Sliding Window MAP Algorithm	61
4.5	Comparison between MAP and Viterbi decoders	63
5	Turbo Codes and Higher Order Modulation	65
5.1	Description of the LLR module.....	66
6	Performance Bounds	68
6.1	Union Bounds	68
6.2	PCCC union bound.....	68
6.3	SCCC union bound.....	69
6.4	HCCC union bound.....	70
6.5	EXIT charts.....	71
7	Misconceptions	73
7.1	SNR Estimation	73
7.2	Delay.....	73
7.2.1	SCC	74
7.2.2	PCC	77
7.3	Decoder Complexity	79
7.4	Error Floor.....	80
7.5	Memory Requirements.....	80
8	Applications of Turbo Codes.....	81
8.1	Inmarsat.....	81

What a wonderful *turbo* world ...

8.2	DVB	83
8.3	Deep-Space Communications	88
8.3.1	CCSDS	88
8.3.2	DSN	88
8.4	UMTS/3GPP	89
8.5	DAB	89
8.6	Magnetic Recording Channels	89
8.7	Decoder-assisted frame synchronisation	90
8.8	Turbo Codes for Frequency-Hopped Spread Spectrum	90
8.9	Turbo Codes for jammed channels	90
8.10	Chaotic turbo codes	91
8.11	Analog Decoders	93
8.12	INTELSAT Proof-of-concept	94
8.13	Turbo codes for ADSL systems	96
8.14	Space-time turbo coded systems	97
8.15	Iterative Satellite Transceiver	99
8.15.1	Coding Design	101
8.15.2	Applications	104
8.15.3	Higher data throughput	106
8.15.4	De-mapping and decoding	108
8.15.5	Performance in Nonlinear Channels	108
8.15.6	All Access Architecture	111
9	Improved Security in Satellite Communications	112
9.1	Secret Key Exchange	114
9.2	Possible Interleavers	114
10	Joint Source and Channel Coding	115
10.1.1	State Transition Probabilities of VLC-Trellis	115
10.1.2	Iterative Decoding of using RVLC	116
11	Conclusions	118
12	References	119
	Appendix A: "C" code version v13	126
	Appendix B: S256 Interleaver file	146

List of Figures

Figure 1-1 Conventional Block Diagram of a Communication System	16
Figure 1-2 Channel Capacity.....	17
Figure 1-3 Block Diagram of a Communication System	18
Figure 1-4 Information exchange between source (de)coding and channel (de)coding/(de)modulation.....	19
Figure 1-5: Competitive Advantages of Turbo Codes.....	21
Figure 1-6 Four state rate half convolutional encoder	21
Figure 1-7 Trellis diagram.....	22
Figure 1-8 50 Years of Coding (by courtesy of Alex Grant).....	26
Figure 2-1: PCC Scheme	30
Figure 2-2 Rate $\frac{1}{2}$ turbo encoder	31
Figure 2-3 Low Coding Rate Turbo Code.....	31
Figure 2-4 SCC Scheme	32
Figure 2-5: FlexiCode encoder architecture	33
Figure 2-6 FlexiCode decoder architecture	33
Figure 2-7 BER performance for FlexiCode	34
Figure 2-8 HCC Scheme	34
Figure 2-9 BTC Scheme.....	35
Figure 2-10 AHA4541 BER curves, QPSK.....	37
Figure 2-11 iPSTAR footprint	38
Figure 2-12 iPSTAR system architecture	39
Figure 2-13 LDPC vs Turbo code BER comparison.....	40
Figure 2-14 BER performance of V-LDPC codes.....	41
Figure 2-15 BER performance of Comtech LDPC/BCH codes.....	41
Figure 2-16 TPC vs LDPC performance.....	42
Figure 3-1 A four state systematic convolutional encoder.....	47
Figure 3-2 Convolutional interleaver architecture.....	49
Figure 3-3 Extrinsic Information.....	51
Figure 4-1: Communication System Block Diagram.....	53
Figure 4-2 Typical BER curve for turbo codes.....	54
Figure 4-3: Generic turbo decoder	55

What a wonderful *turbo* world ...

Figure 4-4: MAP Forward state metrics	59
Figure 4-5: MAP backward state metric	60
Figure 4-6: MAP decoding using overlapping windows	63
Figure 4-7: Interleaving constraint for sliding window decoding	63
Figure 4-8 Comparison between MAP and Viterbi decoders.....	64
Figure 5-1: Mapping in the I dimension	66
Figure 5-2: Mapping in the Q dimension	67
Figure 7-1: Rate 1/2 SCS with rate 2/3 outer code and rate 3/4 inner code	74
Figure 7-2: Rate 1/2 SCS with rate 2/3 outer code and rate 3/4 systematic inner code.....	75
Figure 7-3: Symbol transmission order for rate 1/2 SCC.....	75
Figure 7-4 Rate 1/2 PCC.....	77
Figure 7-5: Rate 1/2 PCC with B/2 transmitter delay.....	78
Figure 7-6: Rate 1/2 PCC with B/4 transmitter delay.....	78
Figure 8-1: Turbo Decoder Card	81
Figure 8-2 BGAN service area	82
Figure 8-3 BGAN services.....	82
Figure 8-4 CRSC encoder.....	83
Figure 8-5 Elementary Encoder for CCSDS turbo code	88
Figure 8-6 Chaotic Encoder.....	91
Figure 8-7 Autocorrelation functions.....	91
Figure 8-8 Histogram and FFT	92
Figure 8-9 Chaotic Turbo Code	92
Figure 8-10: <i>Turbo Codec – QPSK Modem</i>	94
Figure 8-11: BER performance of Rate 1/2 Turbo Codec – QPSK Modem	95
Figure 8-12: BER performance of Rate 3/4 Turbo Codec – QPSK Modem	95
Figure 8-13 BLAST system diagram	97
Figure 8-14 Turbo coded space-time scheme	98
Figure 8-15 Premier 5 satellite modem	99
Figure 8-16 Premier 5 BER curve for rate 1/2	100
Figure 8-17 Premier 5 BER curve for rate 3/4	100
Figure 8-18: E_b/N_0 requirements vs modulation scheme.	102
Figure 8-19: Performance of the SW algorithm.....	104
Figure 8-20: Performance of the punctured rate 1/2 code.....	105
Figure 8-21 Iterative Satellite Transceiver Design	106

What a wonderful *turbo* world ...

Figure 8-22 Turbo Decoder architecture for 2 Mbit/s	106
Figure 8-23 Turbo Decoder architecture for 48 Mbit/s	107
Figure 8-24 Iterative decoding loop	108
Figure 8-25 SSPA AM-AM transfer characteristic	109
Figure 8-26 SSPA AM-PM transfer characteristic	109
Figure 8-27 Distortion of rectangular 16QAM constellation at an IBO = 3 dB.....	109
Figure 8-28 BER in a linear channel after 8 iterations	110
Figure 8-29 Degradation of BER function of IBO.....	110
Figure 9-1 Armoured Cable Model	112
Figure 9-2 Transmitter block diagram.....	113
Figure 9-3 Receiver block diagram.....	113
Figure 10-1: Tree representation and VLCtrellis with parallel transitions.	116
Figure 10-2: Iterative source and channel decoder	116
Figure 10-3: BER curves for joint source channel decoding.....	117

List of Tables

Table 1-1 E_b/N_0 required for a BER = 10^{-6} for CC and TC.....	27
Table 1-2 Comparison between convolutional codes (CC) and turbo codes (TC).....	27
Table 1-3 Capacity limits	28
Table 1-4 Required E_b/N_0 function of block size and coding rate	29
Table 3-1 Writing data row-wise in memory.	44
Table 3-2 Reading data column-wise from memory.	44
Table 3-3: Reading data diagonal-wise from memory.	44
Table 3-4 Odd-positioned coded bits.....	45
Table 3-5 Even-positioned coded bits.	45
Table 3-6 The output for a pseudo-random interleaver.	45
Table 3-7 3x5 block interleaver.	45
Table 3-8 Even-positioned coded bits.	46
Table 3-9 Information bits and multiplexed coded bits for an “odd-even” interleaver.	46
Table 3-10 Final encoder state for $n = 2$ for N information bits.....	47
Table 3-11 Simile odd-even block helical interleaver.	47
Table 3-12 The output of a simile odd-even block helical interleaver.	48
Table 7-1: Delay reduction for a fixed information block in an SCC	76
Table 7-2: Increase in information block size for a given total delay D_{E-D} in a SCS...	77
Table 8-1 Circulation state correspondence table	84
Table 8-2 Permutation parameters.....	84
Table 8-3 Puncturing patterns (“1” = keep).....	85
Table 8-4: Delay Reduction for Sliding Window Algorithm	103

Acronyms

16QAM	16 quadrature amplitude modulation
256QAM	256 quadrature amplitude modulation
32PAM	32 state pulse amplitude modulation
3GPP	third generation partnership project
4AM	4-state amplitude modulation
64QAM	64 quadrature amplitude modulation
8PSK	eight phase shift keying
ACI	adjacent channel interference
ADSL	asymmetric digital subscriber lines
AM	amplitude modulation
APP	<i>a posteriori</i> probability
ARQ	automatic repeat request
ATM	asynchronous transfer mode
BCH	Bose-Chaudhuri-Hocquenghem
BER	bit error rate
BLAST	Bell laboratories layered space-time
BPSK	binary phase shift keying
BTC	block turbo code
C/I	carrier to interferer
C/N	carrier to noise ratio
CC	convolutional codes
CCI	co-channel interference
CCSDS	consultative committee for space data systems

What a wonderful *turbo* world ...

CDMA	code division multiple access
CRC	cyclic redundancy code
CRSC	circular recursive systematic convolutional
DAB	direct audio broadcasting
DAMA	demand assigned multiple access
DMT	discrete multitone
DSN	deep space network
DSNG	digital satellite news gathering
DSP	digital signal processing
DVB	digital video broadcasting
DVB-S	digital video broadcasting – satellite
E_b/N_0	energy per bit to noise density ratio
EIT or EXIT	extrinsic information transfer
ESA	european space agency
FEC	forward error correction
FFT	fast fourier transform
FH-SS	frequency-hopped spread spectrum
FPGA	field programmable gate arrays
FSP	finite state permuter
HCC	hybrid concatenated codes
HIHO	hard-in-hard-out
HPA	high power amplifier
IBS	Intelsat business services
IDR	intermediate data rates
IOWC	input-output weight coefficient
IOWEF	input-output weight enumerating function

What a wonderful *turbo* world ...

IP	Internet protocol
IRWEF	input-redundancy weight enumerating function
ITR	institute for telecommunications research
JSCC	joint source and channel coding
LDPC	low-density parity-check
LLR	log-likelihood ratios
LOVA	list output Viterbi algorithm
LUT	look-up-table
MAP	maximum <i>a posteriori</i>
MCPC	multi channel per carrier
MF-TDMA	multi-frequency time division multiple access
MIMO	multiple-input, multiple-output
MLDA	maximum likelihood decoding algorithms
MOD	media on demand
OFDM	orthogonal frequency division multiplexing
PCBC	parallel concatenated block code
PCC	parallel concatenated codes
PCCC	parallel concatenated convolutional code
PER	packet error rate
PLD	programmable logic device
PM	phase modulation
PSNR	peak signal to noise ratio
QAM	quadrature amplitude modulation
QPSK	quadrature phase shift keying
RAM	random access memory
RS	Reed-Solomon

What a wonderful *turbo* world ...

RSC	recursive systematic codes
SCC	serial concatenated codes
SCCC	serial concatenated convolutional code
SISO	soft-in-soft-out
SIT	satellite interactive terminal
SLVA	soft list viterbi algorithm
SNR	signal-to-noise ratio
SOVA	soft output viterbi algorithm
SW	sliding window
TC	turbo codes
TCM	trellis coded modulation
TCP	transport control protocol
TDMA	time division multiple access
TPC	turbo product codes
TTCM	turbo trellis coded modulation
TWTA	travelling wave tube amplifiers
UDP	user datagram protocol
UMTS	universal mobile telecommunications system
UW	unique words
VLSI	very large scale integrated
VSAT	very small aperture terminal

1 Introduction

This book is about how to solve a half-century old problem!

The solution is amazingly simple and universally valid: divide and conquer!

Around the mid XX century, Claude Shannon defined the basis of the Information Theory, showing that it is possible to transmit information error free as long as it occurs at a rate below the channel capacity. How to do it, remained an unsolved problem until a few years ago.

Claude Shannon died at the dawn of the 3rd millennium, shortly after the answer was found: turbo codes. This book is dedicated to Claude Shannon, to this small closed circle in the history of mankind which started with a question about what is possible and ended with an answer for how to do it.

Turbo coding represents a new and very powerful error control technique, which has started to have a significant impact in the late 90s, allowing communication very close to the channel capacity. The powerful error correction capability of turbo codes was recognised and accepted for almost all types of channels leading to increased data rates and improved Quality of Service. Turbo codes can operate at 0.1 dB from the Shannon capacity limit outperforming any other coding technique known today [1].

Turbo codes were introduced in 1993 [2] and the first modem designed for a commercial application of turbo codes was tested in 1997 ([3], [4], [5], [6]).

A lot of research has been done in the application of turbo codes in deep space communications, mobile satellite/cellular communications, microwave links, paging, in OFDM and CDMA architectures. Turbo codes outperformed all previously known coding schemes regardless of the targeted channel. The extra coding gain offered by turbo codes can be used either to save bandwidth or reduce power requirements in the link budget.

Many standards based on turbo codes have already been defined or are currently under investigation. This book attempts to introduce turbo codes, to explain why they perform so well and to summarise current applications. It also discusses a few misconceptions related to the complexity, delay, sensitivity to signal-to-noise ratio (SNR), error floor and memory requirements of turbo codes. The references direct the reader to more detailed technical papers.

The book is organised in the following chapters. Chapter 2 introduces the architecture of a turbo encoder and very close relatives of turbo codes. Chapter 3 describes techniques to design a good interleaver, the key component of the turbo encoder. Chapter 4 describes the turbo decoder and the iterative decoding principle. Turbo codes used with higher order modulation are described in Chapter 5. The performance bounds are given in Chapter 6. In Chapter 7 some misconceptions are clarified and proven to be wrong by a multitude of applications of turbo codes described in Chapter 8. Chapter 9 shows how turbo-like codes can be used in secure communications. Combined source and channel coding is presented in Chapter 10. Chapter 11 concludes this book with a glimpse at new directions for further research into this wonderful *turbo* world.

What a wonderful *turbo* world ...

1.1 Communication System

Increasing demand for information exchange is a characteristic of modern civilisation. The transfer of information from the source to its destination has to be done in such a way that the quality of the received information should be as close as possible to the quality of the transmitted information.

A typical communication system may be represented by the block diagram shown in Figure 1-1.

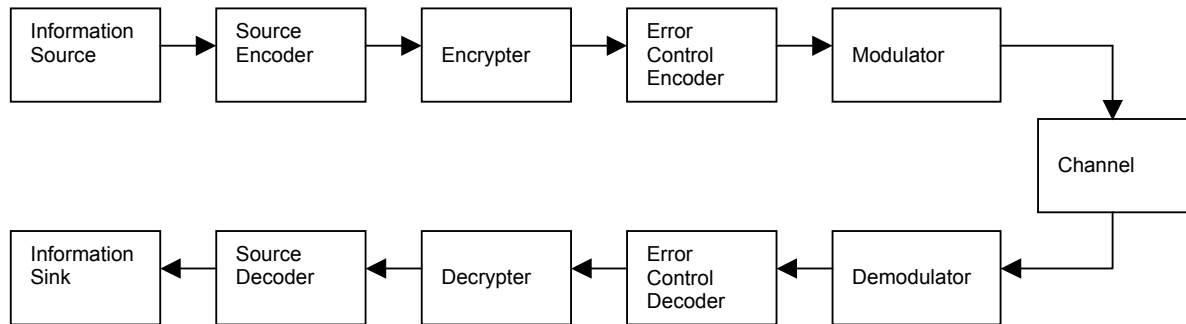


Figure 1-1 Conventional Block Diagram of a Communication System

The information to be transmitted can be machine generated (e.g., images, computer data) or human generated (e.g., speech). Regardless of its source, the information must be translated into a set of signals optimised for the channel over which we want to send it. The first step is to eliminate the redundant part in order to maximise the information transmission rate. This is achieved by the source encoder block in Figure 1-1. In order to ensure the secrecy of the transmitted information, an encryption scheme must be used. The data must also be protected against perturbations introduced by the communication channel which could lead to misinterpretation of the transmitted message at the receiving end. This protection can be achieved through error control strategies: forward error correction (FEC), i.e., using error correcting codes that are able to correct errors at the receiving end, or automatic repeat request (ARQ) systems. The modulator block generates a signal suitable for the transmission channel.

At this moment an important question arises: is this optimum transmitter architecture? Does it make sense to reduce redundancy in the source encoder only to increase it later on in the channel encoder?

Shannon did the hard work to find the answer to this question, which is yes: an optimum transmitter can be designed using completely separate techniques for digital source compression from those for channel transmission, even though the first removes redundancy and the second inserts it.

At the receiver, the reversed sequence of blocks performing the inverse functions can be used in order to recover the transmitted information. Now, although it looks like an obvious approach, is this an optimum receiver?

The answer is again given by Shannon: never discard information prematurely that may be useful in making a decision until after all decisions related to that information have been completed.

What a wonderful *turbo* world ...

Hmmm, is this a yes or a no? Before deciding either way, a better understanding of each block in the communication system is required.

In the traditional approach, the demodulator block from Figure 1-1 makes a “hard” decision for the received symbol and passes it to the error control decoder block. This is equivalent, in the case of a two level modulation scheme, to decide which of two logical values, say -1 and +1, was transmitted. No information is passed on about how reliable the hard decision is. For example, when a +1 is output by the demodulator, it is impossible to say if it was received as a 0.2 or a 0.99 or a 1.56 value at the input to the demodulator block. Therefore, the information concerning the confidence into the demodulated output is lost in the case of a “hard” decision demodulator. This is exactly the opposite advice that Shannon gave.

As shown in Figure 1-2, the channel capacity of a discrete-input real-output memoryless channel (C_{soft}) is greater than that for a discrete-input discrete-output (hard output) memoryless channel (C_{hard}) ([7], [8]).

The most important conclusion from Figure 1-2 is that the C_{soft} is greater than the C_{hard} by approximately 2 dB at low signal to noise ratios. This is the main reason for using soft output algorithms.

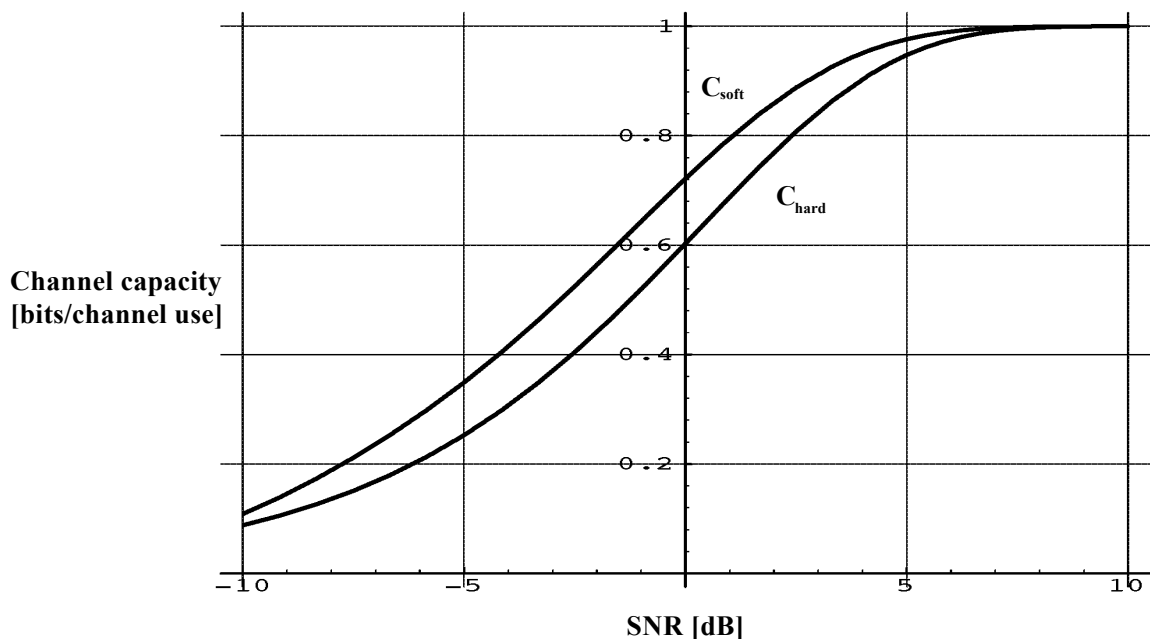


Figure 1-2 Channel Capacity

Therefore, soft output demodulators which produce a quantised output improve the receiver performance. The finer the quantization, the better.

The next block in the receiver structure is the error control decoder which needs to be discussed in conjunction with the error control encoder block.

What a wonderful *turbo* world ...

From coding theory [9], it is known that by increasing the codeword length or the encoder memory, greater protection, or coding gain, can be achieved. At the same time the complexity of typical decoding algorithm such as maximum likelihood decoding algorithms (MLDA) increases exponentially with the encoder memory and the algorithms become difficult to implement.

The increased error correction capability of long codes requires a very high computational effort at the decoder. This has led to research for new coding schemes which could replace the MLDA with simpler decoding strategies. These strategies combine simpler codes in such a way that allows each code to be decoded separately with less complex decoders. By using soft-in-soft-out (SISO) decoders, information can be passed from one decoder to the next in an iterative fashion. This is a “divide-and-conquer” strategy that in an iterative process can approach the performance of the MLDA.

By now it became obvious that the loop of this iterative process could be enlarged such that to include the demodulator block. The corrected data produced by the error control decoder could be used to improve the demodulator performance. This could be used for example in improving the performance of the equalization or synchronization stages in the demodulator. Therefore, a feedback from the decoder block to the demodulator block seems like a good idea.

On the same lines, the decoder can make some errors when the noise is too high. The source decoder however, based on the residual redundancy in the transmitted signal could help improve the decoder performance. In the end, a better practical receiver architecture is shown in Figure 1-3. This is because, in practice, the individual blocks in the communication system are far from ideal.

The only differences from Figure 1-1 are the feedback loops, which help to improve the total system performance.

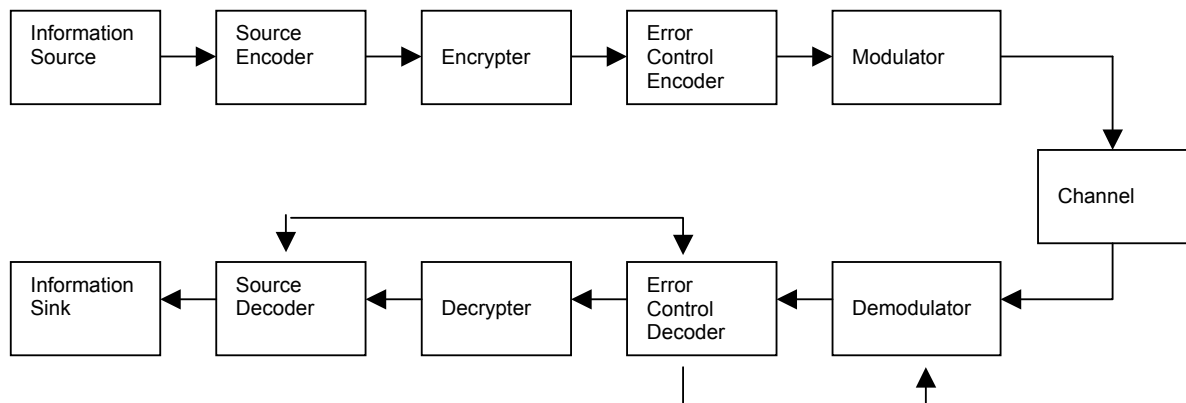


Figure 1-3 Block Diagram of a Communication System

After arguing in favour of this new architecture, we take a step back and ask again: is this really the best solution? Well...

From the physical layer point of view and for the purpose of this book, yes it is. But is this the whole story? The answer is no!

What a wonderful *turbo* world ...

What I really want is for the application running on my terminal (PC, mobile phone, etc) to communicate optimally with the other end. Since the above architecture doesn't care about the network layers, this is not an optimum solution.

A lot of effort is currently put into the optimization of the whole architecture of the communication system, defined from one application layer to another application layer.

One example is the "JOint source-channel CODing-driven digital baseband design for 4G multimedia streaming (JOCO)" project sponsored by the European Union [10] which tries to achieve an optimal allocation of user and system resources by a "co-operative" optimisation of communication system components. The basic idea is the joint determination of the source coding, channel coding, modulation, and network parameters to yield the best end-to-end system performance. A specific objective in this case is to build a global network architecture based on joint source channel (de)coding for future 4G systems.

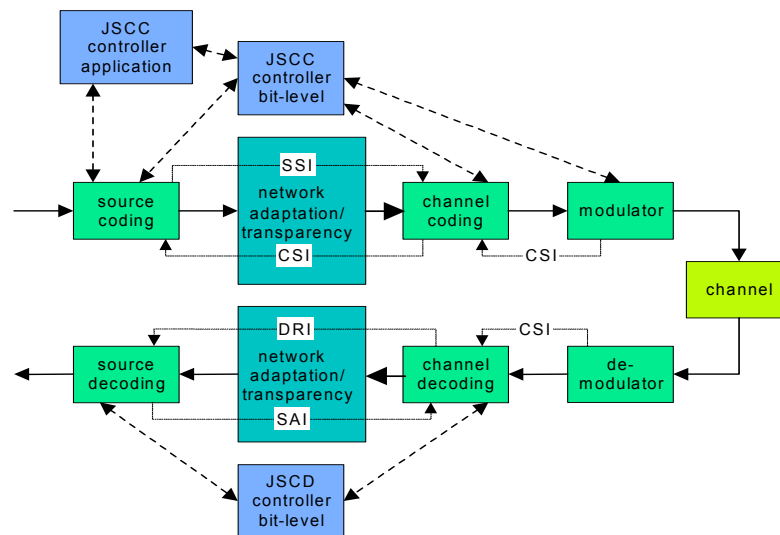


Figure 1-4 Information exchange between source (de)coding and channel (de)coding/(de)modulation

Targeting the fully IP-based structure of 4G, JOCO aims at making the source encoder running at the application layer to communicate with the channel (de)coder/(de)modulator at the physical layer through the IP protocol. In terms of performance, preliminary simulations of prototype systems adopting some basic JSCC techniques for the transmission of coded video over wireless channels show peak video quality gains of about at least 3 dB in terms of PSNR, or alternatively, the possibility to double the spectral efficiency (see Chapter 10).

It looks like now we have really found the final answer. We know how to estimate the channel capacity and we know how to transmit information close to the channel capacity rate... the half a century mystery was solved!

However, for the true believers in the never-ending story there is a sequel: push the channel capacity! All the investigations so far were focused on a single channel per user. Multiple transmit and receive antennas in a rich scattering fading channel promises capacities of tens and hundreds of bits per channel use. This technique called *space-time coding* briefly discussed in Section 8.14 represents the most promising area for future research. But this is another story...

What a wonderful *turbo* world ...

1.2 Channel coding: a new turbo world

The information of this world has a tendency to get corrupted, whether by gossip, by misinterpretation or by the Chinese Whispers effect. Have you ever wondered why a historical event may be seen completely differently in the US, compared to Thailand or Ireland? It seems that although somewhere something did really happen, the facts were warped by various sources of “noise”. Each of us is able to read between the lines and hence correct some of the distortions. However, even then, our interpretations may still be riddled with “errors”.

There seems to be no absolute truth, but how can we get as close to it as possible? The solution is to approach the task using an iterative process and using as much independent information as possible, with each different perspective bringing new “weight” to the final answer. Although this example refers to the semantics of information, a similar approach can be used to correct errors introduced when transmitting over a physical channel.

There are many interesting examples of successful attempts to correct corrupted data. Error correction codes work behind the scenes every time you watch a DVD movie or listen to a CD. The so-called Reed-Solomon (RS) codes are used to fix errors produced by a scratch on your CD or traces of coffee spilt over your favorite DVD. All these examples have a common thread: they use encoding, which is a process that adds extra structure to the original information. When “noise” corrupts the transmitted signal, the added structure is used to recover the source signal in a process called decoding.

In the late 1940s, Claude Shannon set the groundwork in the field of Information Theory. His greatest achievement in this area was to find the maximum capacity, or rate, at which information can be transmitted error-free. This was a great theoretical discovery; however no one really knew how to achieve this maximum transmission rate in practice because of the signal corruption due to noise.

The way to achieve this limit was discovered only recently, in 1993, by a team of French scientists who developed the so called “turbo codes”. Turbo coding represents a new and very powerful error control technique, which has started to have a significant impact in the late 90s, allowing communication very close to the channel capacity. The powerful error correction capability of turbo codes is recognized and accepted for almost all types of channels leading to increased data rates, cheaper service and ultimately, improved Quality of Service.

Turbo codes outperform any other coding technique known today. The three key areas of performance improvement that this technology provides shown in Figure 1-5 are: capacity (turbo codes achieve near theoretical capacity performance, with improvement up to 60% over existing standards), system cost efficiency (a user is able to send the same amount of data using only half the bandwidth), number of users (a satellite service provider is able to double the number of users without increasing satellite capacity).

Due to their outstanding performance and the state of the microelectronics industry today, Turbo Coding techniques are replacing the conventional codes used so far in almost all communication systems. From satellite communications and wireless applications to magnetic storage, turbo coding is poised to become the de facto coding technology of the new millennium.

What a wonderful *turbo* world ...

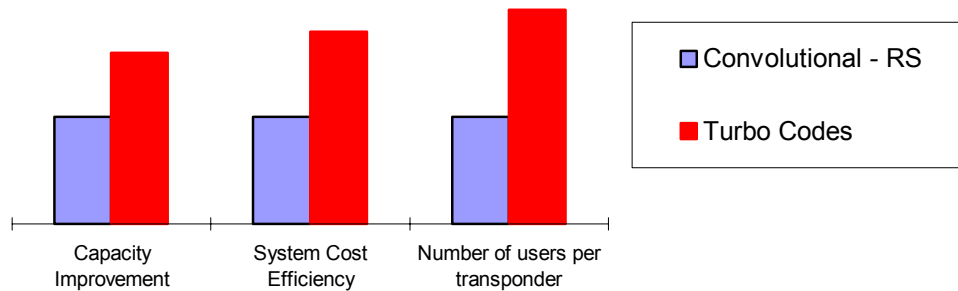


Figure 1-5: Competitive Advantages of Turbo Codes

Our focus here is on the encoder/decoder blocks. The conventional approach in the design of error control codes was to introduce a lot of structure. For example, shifting information bits serially through a register that has some connections between its cells, creates a new sequence (the so called parity bits), which depends on the information bits used and the connections made. The more parity bits we create, the higher the level of dependency introduced and the better the chance of detection and correction of errors. One could rightly assume that increasing the length of the shift register and the number of parity bits, therefore obtaining a more complex structure, leads to an increase in the number of dependencies.

1.3 Convolutional encoding and Viterbi decoding

The status of the shift register is called the state of the encoder. The example shown in Figure 1-6 is a four state binary encoder implemented by a shift register made of two memory cells (D). The outputs from the two modulo-2 adders represent the coded bits.

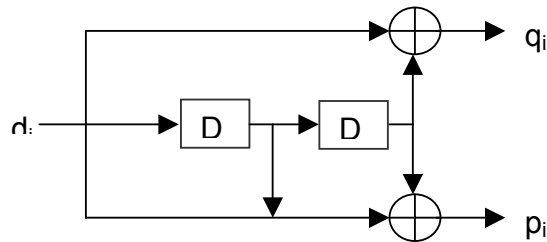


Figure 1-6 Four state rate half convolutional encoder

The encoder can be described as a state machine which changes its state from one clock to the other according to the trellis diagram shown in Figure 1-7. For example, if the current state of the encoder is “11” and the input bit d is “1”, the two outputs will be: $q = “0”$ and $p = “1”$. The important thing to observe is that only some transitions are allowed from one state to another. If the trellis is extended for all transmitted data, one can follow a “path” through the trellis.

In a noiseless system the transmitted “path” through the trellis is received without any errors. When noise is present, some coded bits can be reversed, fact that leads to a broken “path”: some combinations of coded bits are impossible to be connected on a continuous “path”. The best the receiver can do is to find the most likely “path” in the trellis that is the closest to the received

What a wonderful *turbo* world ...

sequence of bits. In this way the coded bits affected by noise are corrected by taking into account the history embedded in the “path”. The tool to achieve this is the Viterbi decoder.

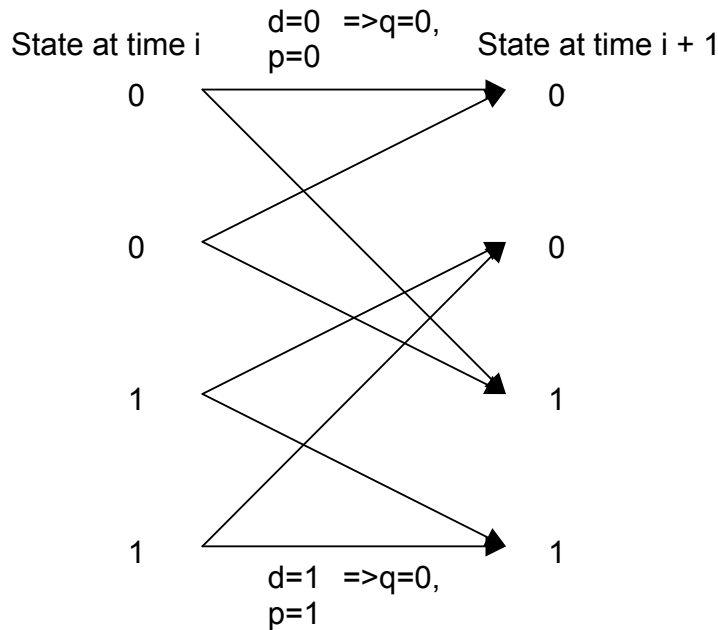


Figure 1-7 Trellis diagram

For each state at time i , the Viterbi decoder compares the path metrics entering that state and keeps the one with the smallest metric, called the survivor path. The rest of the paths entering that state are discarded. In the end all path metrics for all the states are compared, the one with the smallest path metric is selected for the transmitted codeword. The following steps describe the Viterbi algorithm:

- Assume that the transmission started with the encoder in a well defined state, say “00” in our example
- Initialise all state metrics to “1” less the state metric associated to state “00”, $S(0) = “0”$
- Calculate all possible branch metrics relative to the received coded bits at time i . This is done by calculating the Euclidian or Hamming distance between the received coded bits and all the possible combination, e.g. “00”, “01”, “10” and “11”
- For each state at time $i+1$ compute the path metrics of the path entering that state: add the state metric at time i for that path with the corresponding branch metric for that particular transition.
- For each state at time $i+1$ select the path with the best path metric and ignore the rest; this path metric becomes the new state metric for that state at time $i+1$
- Traceback: the survivor path is traced back and using the encoder inverse function, the information bits are produced

What a wonderful *turbo* world ...

The Viterbi algorithm produces the maximum likelihood path. The decoder can produce an error only when the noise creates a vector that is closer to another path in the trellis than the transmitted path. To reduce this possibility, therefore to reduce the bit error rate, the trellis must have a larger number of states.

The problem with this approach is the enormous complexity of the implementation of the state machine for the decoder. Say for example the shift register has 10 cells (equivalent to $2^{10} = 1024$ states). It follows that for every single bit received, the decoder has to compare ALL possible 1024 states in order to make a decision. Now consider that in order to get close to channel capacity one needs not 10 but 1000 cells in the shift register. This structure would have required investigating a huge number ($2^{1000} =$ a one followed by 301 zeros) of possible states. It is obvious that this technique becomes impractical with the increase in the number of dependencies.

This approach dominated the error control coding field for almost half a century. Researchers and engineers used special techniques, like for example sequential decoding for less than 50 cells, to implement the ever-increasing complexity needed for good signal recovery.

1.4 The paradigm shift: Turbo Codes

In his book “The Structure of Scientific Revolutions”, Thomas S. Kuhn argues that scientific progress occurs not in a linear and cumulative fashion but through discontinuities or leaps, which he calls paradigm shifts. This paradigm shift is often found to be based on “imports” from other fields of science.

Turbo Coding is the paradigm shift that occurred in the discipline of channel coding. Turbo Codes is not just a new set of codes, but a new way of thinking about channel coding, hence a new paradigm. Some of the concepts at the foundation of turbo codes are indeed “imported” from a variety of other fields, like feedback systems for electronic amplifiers or turbo engines.

The operation of a turbo codec relies on some basic ideas: using uncorrelated inputs, divide and conquer and processing information iteratively. The information to be transmitted is stored in a memory in order to be scrambled (interleaved) to produce two uncorrelated sequences that are then encoded and transmitted. This concept is the key to the exceptional performance of turbo codes. The type of interleaver and its size plays a significant role in the performance that can be achieved.

Given that the interleaver used is large, the actual encoders can be kept simple and still have a very large number of states associated with the trellis of the turbo code. Therefore, the turbo decoder can be implemented using two simple state machines that operate on each sequence. In this way, the decoding becomes an iterative process, with each one of the smaller decoders passing some information to the other one.

The information produced by the decoders contains not only the decoded message (hard output) but also the degree of confidence in the decision (soft output). For example, binary hard outputs are replaced by real number soft outputs: a high positive number indicates a higher confidence in decoding a ‘1’, whereas a high negative number indicates a higher confidence in decoding a ‘0’. In order to achieve this, a soft Viterbi decoder has to be used or even better, a *maximum a posteriori*(MAP) decoder. A MAP decoder maximizes the likelihood of each transmitted bit as opposed to a Viterbi decoder that produces the maximum likelihood path.