2001

Hybrid ARQ with parallel and serial concatenated convolutional codes for next generation wireless communications

Naveen Chandran
West Virginia University

Hybrid ARQ with Parallel and Serial Concatenated Convolutional Codes for Next Generation Wireless Communications

Naveen Chandran

Thesis submitted to the College of Engineering and Mineral Resources at West Virginia University in partial fulfillment of the requirements for the degree of

Master of Science

in

Electrical Engineering

Approved By

Dr. Matthew. C. Valenti, Chair
Dr. Mark Jerabek
Dr. Roy Nutter

2001
Lane Department of Computer Science and Electrical Engineering
Morgantown, West Virginia

Keywords: Hybrid ARQ, Turbo Codes, Serial Concatenated Codes, Wireless Communications, Channel Coding, 3G.

ABSTRACT

Hybrid ARQ with Parallel and Serial Concatenated Convolutional Codes for Next Generation Wireless Communications
By: Naveen Chandran

Hybrid FEC-ARQ systems combine the advantage of ARQ systems of obtaining high reliability with that of forward error correcting (FEC) systems of providing high throughput. This research focuses on evaluating the currently used FEC encoding-decoding schemes (convolutional codes, turbo codes and serial concatenated codes) and improving the performance of error control systems by incorporating these schemes in a hybrid FEC-ARQ environment. The first section of the thesis gives a comprehensive discussion of the latest issues in wireless communications, nature of the wired and wireless channel, and various ARQ protocols.

The next section introduces convolutional codes and Viterbi decoding as a stepping-stone to the turbo coding scheme. Turbo codes are parallel concatenated convolutional codes (PCCCs) separated by an interleaver, and have shown to perform at very high-energy efficiencies - about 0.5 dB away from the Shannon capacity. An iterative soft-input soft-output (SISO) maximum-a-posteriori (MAP) decoding scheme is explained and its practical implementation is discussed. A type-II hybrid ARQ scheme with serial concatenated convolutional codes (SCCCs) is proposed for the first time and is a major contribution of this thesis.

It is seen that there is a vast improvement in BER performance of the successive individual FEC schemes discussed above. Also, very high throughputs can be achieved when these schemes are incorporated in an adaptive type-II hybrid ARQ system.

Finally, the third section of this thesis discusses the equivalence of the PCCCs and the SCCCs. Hybrid codes using a combination of both these schemes can be generated using the proposed technique and forms the second major contribution of this thesis. In conclusion, the thesis is summarized and ideas for future work are identified.

Acknowledgements

I wish to express my heartfelt gratitude to my advisor, Dr. M. C. Valenti, for the enduring guidance and encouragement that he has provided ever since I began my M.S. program. I have the privilege of being his first research assistant at West Virginia University and have truly enjoyed my association with him. The best part in being his student is the work-enhanced friendly relationship that I have got to share with him. He has always provided valuable insight into my research that has helped to surface the finer details involved. I really appreciate the amount of time and effort he has devoted to my work. Thank you, Dr.!

I would also like to convey special thanks to my committee members, Dr. Jerabek and Dr. Nutter, for reviewing my work and providing worthy feedback and tips for future work.

It has been a great pleasure working with the students at the Wireless Communication Research Laboratory (WCRL) and I would like to thank them for the vastly helpful exchange of technical information that we have had from time to time.

Graduate studies in the United States would have never been possible for me had it not been for the motivation and backing of my entire family and my parents in particular. My parents have always laid special emphasis on my taking up higher education and thereby gaining expertise in my field of interest. In parallel, they have also instilled in me the confidence, courage and determination it takes to accomplish every challenge that life brings with it. Thus, seeing me graduate with my Master's degree will definitely be a special moment for my parents and it is to them that I dedicate this thesis.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Over the past decade, wireless technology has undergone enormous growth. In particular, the cellular communications industry has been experiencing the same type of exponential market growth seen by the PC industry and the Internet. Recent surveys have shown that a new wireless subscriber signs up every 2.5 seconds and the number of cellular and personal communication system (PCS) users in the U.S. has recently surpassed 100 million. As the demand for wireless services is constantly rising, cellular and wireless technology is rapidly advancing to keep up to the demand. Third generation wireless systems, to be launched worldwide by the year 2003, are capable of providing voice, data, video, and multimedia services to the mobile handset at the fingertips of users.

The challenge facing today's engineer is to provide high speed, high bandwidth, and reliable end-to-end communication at low power and system complexity, thereby reducing the overall system cost. Speed of communication depends on the underlying means of transferring information from the transmitter to the receiver – either wired or wireless. The channel imparts a multitude of effects that may be detrimental to correct reception of the information at the receiver. However, high speeds at low powers can be achieved by efficiently encoding the information before passing it to the channel coupled with improving the decoder design at the receiver.

Turbo channel codes, invented by a group of researchers in France in the year 1993 [Ber 93], are a parallel concatenation of convolutional codes and have been shown to perform very close to Shannon's channel capacity bound, about 0.5 dB away from it. This breakthrough has brought about increased motivation to the coding community and since then, turbo codes have been proposed for a variety of communication applications, viz. next generation wireless systems, deep-space and satellite communications. Indeed, the turbo coding scheme is the backbone of this thesis.

This chapter provides an introduction to wireless personal communications and error control channel coding. The motivation to pursue this research and outline of the thesis follows the introduction.

1.1 Looking back: History of Wireless Communications

Wireless is not a recent technology[1]. As early as 1793, wireless messages were transmitted in France using the optical telegraph. Stations consisting of a telescope and a set of semaphore flags capable of encoding multiple messages were placed on adjacent hills. The whole of France was linked by 566 such stations, and a message could be sent from Paris to the border in about an hour. A major breakthrough occurred in 1844, when the first long distance communication over wires took place between Baltimore and Washington using the electrical telegraph, which transmitted Morse-coded signals.

The first wireless electrical communication devices emerged in the late 19th century. The original application of wireless was to communicate where wires could not go – with ships at sea. In 1895, Guglielmo Marconi, an Italian working in the U.K., invented radio [Goo 97], which originally used a spark-gap transmitter to transmit Morse-coded signals. The first wireless transatlantic communication occurred in 1901. Throughout the next century, great advances were made in wireless communication technology.

Up until 1906, wireless could only transmit Morse-coded signals. This changed when an American named Fessenden developed amplitude modulation (AM) for voice communications. The first wireless voice message using continuous wave AM was transmitted from Brant Rock, MA and could be received as far away as New York, NY.

During the First World War, the idea of broadcasting emerged, but broadcast stations were generally too cumbersome to be either mobile or portable. A major milestone was reached in the year 1920, when the world's first commercial AM radio broadcast took place in Pittsburgh, PA. The 100 watt transmitted signal could reach homes several hundred miles away.

The development of mobile radio paved the way for personal mobile communications. The first widespread non-military application of land mobile radio, police car dispatch, was pioneered in Detroit, Michigan in 1921. However, it only possessed a one-way communication link. Later in 1930, a push-to-talk mobile radio set was deployed in Bayonne, NJ. This had a half-duplex, two-way communication link where one channel was shared by both users. Subsequently, the Handie-Talkie was manufactured by Motorola in 1940 – a two way portable radio using AM that was used during World War II by the allied forces.

It was the concept of Frequency Modulation (FM), developed by Edwin Armstrong (U.S.) in 1935, that brought about efficient mobile and portable communications. Because the message is encoded into the frequency and not the amplitude, FM offers

[1] Portions of sections 1.1 to 1.3 have been published by the author in [Cha 01]

clearer reception, requires lower power, and is more robust against noise, interference and fading. FM was used by the famous Walkie-Talkie, a two-way portable radio that also saw extensive action during World War II.

1.1.1 Predecessors to Cellular

Until 1946, land mobile radio systems were unconnected to each other or to the public switched telephone network (PSTN). A major breakthrough was reached in 1946 with the development of the radiotelephone in the U.S. by AT&T, which allowed mobile users to be connected to the PSTN. This service, named the Mobile Telephone Service (MTS), used FM at a carrier frequency of 150 Mhz but offered only 1-3 channels per city at 120 Khz per channel. Initially, each market was served by a single high-power transmitter and a large tower, and only one half-duplex call could take place at a time per channel. Because it was a non-trunked system, each mobile unit was permanently assigned a specific frequency from the pool of possible frequencies. Thus, a mobile user could not place a call if another user was communicating at the same frequency, even if other channels were available. Although MTS was successful in providing basic wireless telephony, it had many shortcomings. Because it used a high power transmitter placed at high elevation to encompass a large area, it was prone to interference. Also, since MTS did not support handoff, calls were dropped when a mobile unit crossed the boundary of one service area to the next.

Some of these problems were addressed by the introduction of the Improved Mobile Telephone Service (IMTS) in 1965. IMTS supported full-duplex signaling, which allowed two-way communication over two separate frequencies. Technological improvements allowed the required channel bandwidth to decrease from 120Khz to 60Khz in 1950, and from 60Khz to 30Khz in 1965. Thus, the number of concurrent calls was doubled in 1950 and then doubled again in 1965. A further increase in efficiency was provided by the introduction of automatic trunking, which enabled a dynamic assignment of channel frequencies to each mobile unit. Now a mobile could place a call on any open channel. Thus, the blocking probability was reduced, and the number of supported subscribers could be increased.

Although IMTS was a significant improvement over MTS, it did not incorporate any handoff mechanism and experienced a 50% probability of blocked calls. The capacity, in terms of the ratio of the number of channels to subscribers, was not sufficient. For instance in 1976, only twelve trunked channels were available for the entire New York City market of approximately ten million people. The system could only support 543 paying customers, and the waiting list exceeded 3,700 people.

1.1.2 The Birth of Cellular

The congestion in the radio spectrum made it clear that a whole new approach to mobile telephony was necessary. During the 1960's the concept of cellular telephony was pioneered in the U.S. at AT&T Bell Laboratories. In 1968, AT&T proposed the concept of cellular telephony to the Federal Communications Commission (FCC). The idea behind cellular telephony is to break each market into small coverage regions or *cells,* as shown in Fig. 1.1 [Rap 96]. Each area is approximated with a hexagonal cell and a base station is located at the center of each cell. Each cell is assigned a fraction of the total available radio spectrum. Transmitted power drops with distance and hence cells that are located far apart can be assigned the same spectrum. Because of its ability to reuse spectrum, cellular accommodated many more customers than before. Moreover, since the coverage area of each base station is greatly reduced, low power transmitters at each base station are sufficient to cover the entire cell. The mechanism of handoff, in which a mobile unit is transferred between cells as it moves out of the coverage area of a particular cell, is incorporated. This takes place without the knowledge of the user, permitting long and non-interrupted calls to be placed. Fig. 1.1 illustrates clusters of cells with a frequency reuse factor (K) equal to 7 (i.e. the complete spectrum is shared by a cluster of K=7 cells). It also depicts a typical situation where handoff may be required.

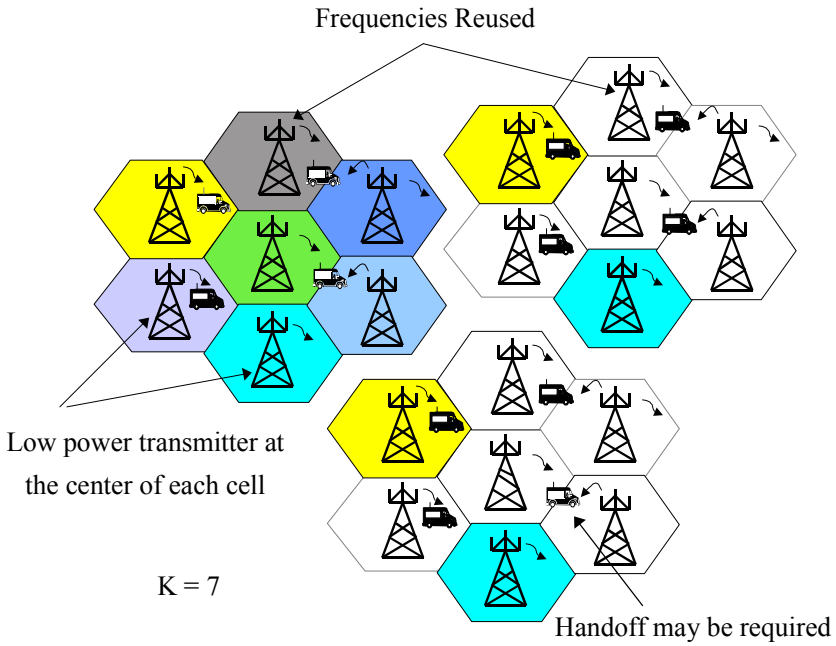


Fig. 1.1 Cellular Concept / Frequency Reuse.

Cellular service has proven to be extremely successful. Cellular phones have evolved from a niche item reserved for the rich to a consumer product. The exponential growth in

customer demand has caused cellular networks in large metropolitan areas to become extremely congested. In theory, cellular has the potential to provide service to all who desire it by making the cells smaller and smaller via a process called cell-splitting. However there are practical and economic factors that limit just how much a cell can be shrunk. Each cell must be serviced by a centrally located base station, which is expensive and utilizes unsightly antenna towers. Many local governments block the placement of base station towers. In addition, having more cells increases the rate at which calls are handed over from one cell to another, which in turn complicates the network architecture.

1.2 Perceiving the Present: Today's Cellular Systems

1.2.1 First Generation Cellular Systems

The first operational cellular system in the world was fielded in Tokyo, Japan, by Nippon Telephone and Telegraph (NTT), in 1979 [Pah 95]. It used 600 FM duplex channels in the 800 Mhz band with a channel bandwidth of 25 Khz. This was followed in 1981 by the Nordic Mobile Telephone (NMT 450) system, which was developed by Ericsson and began operation in Scandinavia in the 450 Mhz frequency band using 25 Khz channels. Total Access Communications System (TACS) was introduced in the United Kingdom in 1982 and the Extended Total Access Cellular System (ETACS) was deployed in 1985. The total number of ETACS channels was 1000 with a channel bandwidth of 25 Khz. Subsequently, in Germany, the C-450 (450 Mhz frequency band) cellular system was introduced in September 1985. Another system called Radicom 2000 was introduced in 1985 in France and operated at 200 Mhz. Thus, at the end of the 1980's, cellular service in Europe was characterized by a multitude of systems that were not interoperable.

The situation in the United States was different than in Europe. In the U.S., there was initially only a single analog cellular standard called Advanced Mobile Phone System (AMPS), which was first placed in service by Ameritech in Chicago in 1983 [Rap 96]. AMPS uses FM in the 800 MHz frequency band with each channel having a bandwidth of 30 KHz. The FCC initially allocated 40 MHz of bandwidth for AMPS cellular in 1981. To ensure some level of competition, the FCC insisted on a duopoly by mandating that two and only two cellular providers serve each market. Each provider was designated as either "A-side" or "B-side". A-side providers were upstart companies that did not originate in the traditional telephone business and were called nonwireline carriers, whereas B-side providers had a base in telecommunication services and were called wireline carriers. Each provider had access to 20 MHz of bandwidth offering 666 full duplex channels in each of 734 markets. In 1986, the FCC allocated another 10 MHz leading to an increase in the number of channels to 832.

1.2.2 Second Generation Cellular Systems

By the late 1980's, it was clear that the first generation cellular systems, which were based on analog signaling techniques, were becoming obsolete. Advances in integrated circuit (IC) technology had made digital communications not only practical, but actually more economical than analog technology. Digital communication enables advanced source coding techniques to be utilized, which allows the spectrum to be used much more efficiently, thereby reducing the amount of bandwidth required for voice and video. In addition, with digital communications, it is possible to use error correction coding to provide a degree of resistance to interference and fading that plagues analog systems, and to allow a lower transmit power. Also, with digital systems, control information is more efficiently handled, which facilitates network control.

Second generation digital systems can be classified by their multiple access techniques as either Frequency Division Multiple Access (FDMA), Time Division Multiple Access (TDMA) or Code Division Multiple Access (CDMA) as illustrated in Fig. 1.2. In FDMA, the radio spectrum is divided into a set of frequency slots and each user is assigned a separate frequency to transmit. In TDMA, several users transmit at the same frequency but in different time slots. CDMA uses the principle of direct sequence spread-spectrum in which the signals are modulated with high bandwidth spreading waveforms called signature waveforms or codes. Although the users transmit at both the same frequency and time, separation of signals is achieved because the signature waveforms have very low cross correlation. In practice, the TDMA and CDMA schemes are combined with FDMA. Thus the term 'TDMA' is used to describe systems that first divide the channel into frequency slots and then divide each frequency slot into multiple time slots. Similarly, CDMA is actually a hybrid of CDMA and FDMA where the channel is first divided into frequency slots and each slot is shared by multiple users that each use a different code.

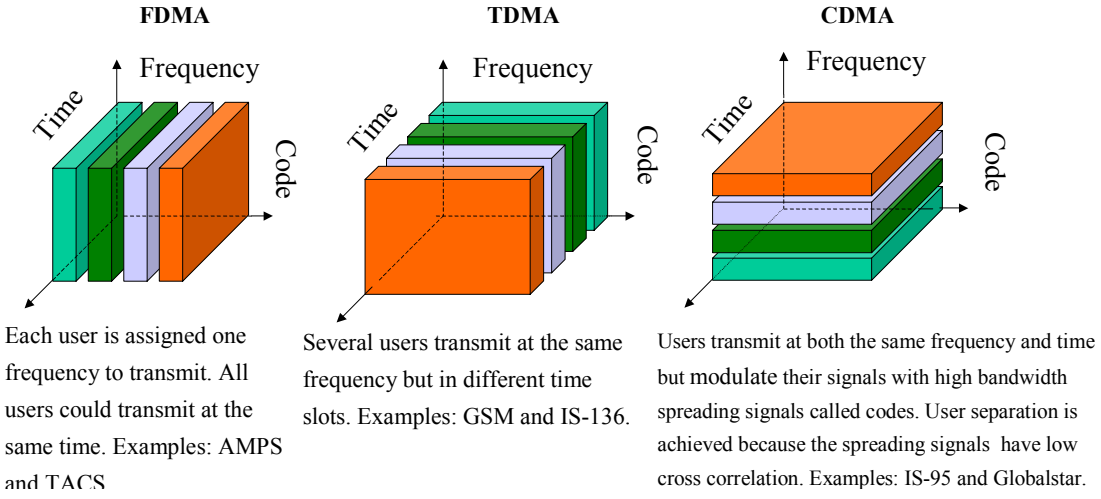


Fig. 1.2 Multiple Access Schemes.

In the early 1990's, second generation (i.e. digital) cellular systems began to be deployed throughout the world. Europe led the way in 1990 with the development of GSM (Global System for Mobile communications), the Pan-European digital cellular standard. The purpose of GSM was twofold — to upgrade transmission technology and to provide a single, unified standard in Europe. Because of the presence of so many different first generation systems, it was necessary to allocate a new frequency band for GSM, over the range 890-960 Mhz. GSM uses the TDMA multiple access scheme.

In the U.S., there was but a single standard, AMPS. Since there was just one standard, there was no need to set aside new spectrum. However, the AMPS standard was becoming outdated and it was apparent that new technology would be required in congested markets. The industry's response was to introduce several new, but incompatible, bandwidth efficient standards. In 1992, the TR45.3 technical subcommittee of the Telecommunications Industry Association (TIA) introduced the IS-54 (North American TDMA Digital Cellular or United States Digital Cellular) standard based on TDMA. It provides a three-fold increase in the system capacity over AMPS. IS-136, a new version of IS-54, was introduced in 1996 and supported a set of additional services. The IS-95 CDMA standard, known as cdmaOne, was introduced by the TR45.5 subcommittee of the TIA in 1993. Since CDMA is used, the system is very flexible, supports wideband signaling, and offers increased capacity. Both IS-136 and IS-95 operate in the same band as AMPS and are specified to be dual-mode systems. The dual-mode nature of these systems allows for a gradual transition from AMPS to the newer digital systems.

At the same time the 900 Mhz band was being transitioned from AMPS to the new cellular standards, the FCC auctioned a new block of spectrum in the 1.9 Ghz band. The systems that occupy this band are collectively known as Personal Communications Systems (PCS), implying a slightly different range of coverage and service than cellular. Examples of PCS systems include J-STD-008 (upbanded CDMA), DCS-1900 (upbanded GSM) and J-STD-011 (upbanded USDC) [Zen 99].

1.2.3 Satellite Telephony

Satellite telephony is similar to cellular telephony with the exception that the base stations are satellites in orbit around the earth. The goal of satellite systems is to extend cellular access to people in remote and rural areas where conventional fixed phone or terrestrial cellular service do not exist. Global roaming enables users subscribed to a satellite system to communicate worldwide. Satellite telephony systems can be categorized according to the height of the orbit as either LEO (Low earth-orbit), MEO (Medium earth-orbit), or GEO (Geosynchronous earth-orbit). Typical heights above the earth's surface are 500 – 1500 Km for LEO satellites, 5000 – 12,000 Km for MEO satellites and 35,800 Km for GEO satellites [Mil 98].

GEO systems have been used for many years to relay television signals. GEO telephony systems, such as INMARSAT, allow communications to and from remote locations, with the primary application being ship-to-shore communications. The advantage of GEO systems is that each satellite has a large footprint, and global coverage up to 75 degrees latitude can be provided with just 3 satellites. The disadvantage of GEO systems is a long round-trip propagation delay of about 250 msec and a high transmit power required by both the mobile unit and the satellite.

With LEO systems, both the propagation time and the power requirements are greatly reduced, allowing for more cost effective satellites and mobile units. The main disadvantage of LEO systems stem from the smaller footprint and high speed of the satellites, which means that more satellites are required and handoff frequently occurs as satellites enter and leave the field of view. A secondary disadvantage of LEO systems is a shorter lifespan of 5-8 years (compared to 12-15 in GEO systems) due to the increased amount of radiation in low earth orbit. MEO systems represent a compromise between LEO systems and GEO systems, balancing the advantages and disadvantages of each. Examples of LEO systems include Iridium (66 satellites, 1998 startup), Globalstar (48 satellites, 1999 startup), Ellipso (14 satellites, 2000 start up), Teledesic (288 satellites, 2002 startup), and Orbcomm (35 satellites, 1999 startup). Examples of MEO systems include ICO (10 satellites, 2000 start up), and Odyssey (12 satellites, 1998 startup). Although LER systems have already come online, the market is slow to accept them, with Iridium, Orbcomm and Globalstar experiencing financial difficulties and bankruptcy.

1.3 Envisaging the Future: Third Generation Cellular Systems

Mobile communications at the end of the twentieth century was characterized by a diverse set of applications using many incompatible standards around the world. For today's mobile communications to become truly personal communications in the new century, it will be necessary to consolidate the standards and applications into a single unifying framework. The eventual goal is to define a global third generation mobile radio standard, initially called the Future Public Land Mobile Telecommunications System (FPLMTS) and later renamed IMT-2000 [Oli 99] for International Mobile Telecommunications by the year 2000, that would offer services like wireless voice, video, e-mail, web browsing, videoconferencing, multimedia, and e-commerce, at any time and from anywhere.

In view of achieving this goal, the International Telecommunications Union (ITU) is evaluating several proposals submitted by standards committees in Europe, the United States, Japan, South Korea, and China. The committees have, by and large, embraced the wide-bandwidth CDMA technology. However, each committee has proposed its own distinct standard. The European Telecommunications Standards Institute (ETSI) in Europe, the Association of Radio Industries and Business (ARIB) in Japan, and the

Telecommunications Technology Association (TTA) in South Korea have all developed standards based on a common technology known as Wideband CDMA (WCDMA), while the Telecommunications Industry Association (TIA) in the United States has proposed a standard called cdma2000. While WCDMA and cdma2000 have many similar features, a major difference is that WCDMA is backward compatible with GSM networks, while cdma2000 is backward compatible with IS-95 networks [Zen 99].

A recent agreement between Ericsson and Qualcomm, the main proponents of the WCDMA and cdma2000 standards respectively, has brought about an opportunity for the two standards to converge into a single third generation standard. The final standard or standards will take a multi-tiered approach, with cells of a variety of sizes. Small cells would be used for low mobility applications, such as cordless telephony, and for high data rate services, such as WLANs. Larger cells will overlay the small cells and serve high mobility applications such as the vehicular communications served by today's cellular systems. Ideally, all types of services would be unified into a single standard, and global roaming would finally become possible.

Advances in cellular communication technology have resulted in improved capacity, higher data rates, and better quality of service. The first generation systems introduced the concept of cellular telephony but, due to their analog nature, were expensive and had limited features. With the aid of digital technology, second generation systems provide greater capacity and reduced power requirements. Advanced multiple access schemes brought about efficient sharing of the available spectrum. The past few years have seen CDMA being increasingly viewed as the multiple access scheme of choice.

A multitude of first and second generation systems have been deployed and currently offer a range of predominantly voice-oriented services. Third generation cellular systems have been proposed with the goal of providing a seamless integration of mobile multimedia services into a single global network infrastructure. These systems are all summarized in Table 1.1. It is expected that this final standard would be flexible and adaptive while offering higher data rates and better spectrum efficiency. Satellites will provide coverage to areas that otherwise would not have terrestrial cellular service. Though a distinct possibility, it remains to be seen if this vision would be successful in the third generation systems. Perhaps a fourth generation of cellular systems may be required before seamless global roaming is finally possible.

NAME	AMPS	GSM / DCS-1900	IS-136 USDC	IS-95	cdma2000	WCDMA / UTRA
Generation	1	2	2	2	3	3
Year Of Introduction & Origin	1983 U.S.	1992 / 1994 Germany	1996 U.S.	1993 U.S.	2002 U.S.	2002 Europe
Region Of Coverage	U.S.	Europe, India, U.S. (PCS)	U.S.	U.S., Hong Kong, Middle-East, Korea.	U.S.	Europe
Frequency Band Uplink (Mhz) Downlink (Mhz)	 824-849 869-894	Cellular / PCS 890-915 / 1850-1910 935-960 / 1930-1990	Cellular / PCS 824-849 / 1850-1910 869-894 / 1930-1990	Cellular / PCS 824–849 / 1850-1910 869-894 / 1930-1990	PCS 1850– 1910 1930– 1990	 1920 – 1980 2110 – 2170
Multiple Access Scheme	FDMA	TDMA	TDMA	CDMA	CDMA	CDMA
Bandwidth per Channel	30 kHz	200 kHz	30 kHz	1.25 MHz	1.25, 3.75, 7.5, 11.25, 15 MHz	5, 10, 20 MHz
Modulation Type	FM	GMSK	$\pi/4$ - DPSK	QPSK and OQPSK	QPSK and BPSK	QPSK and BPSK
Max. Output Power Base: Mobile:	 20 W 4 W	 320 W 8 W	 20W 4 W	 1.64 kW** 6.3 W	 1.64 kW** 2 W	 Unspecified 1 W
Users / Channel	3	8	3	Up to 63	Up to 253	Up to 250
Data Rate	19.2 kbps*	22.8 kbps	13 kbps	19.2 kbps	1.5 kbps to 2.0736 Mbps	100 bps to 2.048 Mbps

* Using Cellular Digital Packet Data (CDPD).

** Total Effective Isotropic Radiated Power (EIRP) for all the carriers within the channel bandwidth.

Table 1.1: Comparison of various cellular standards.

1.4 A Generic Communication System

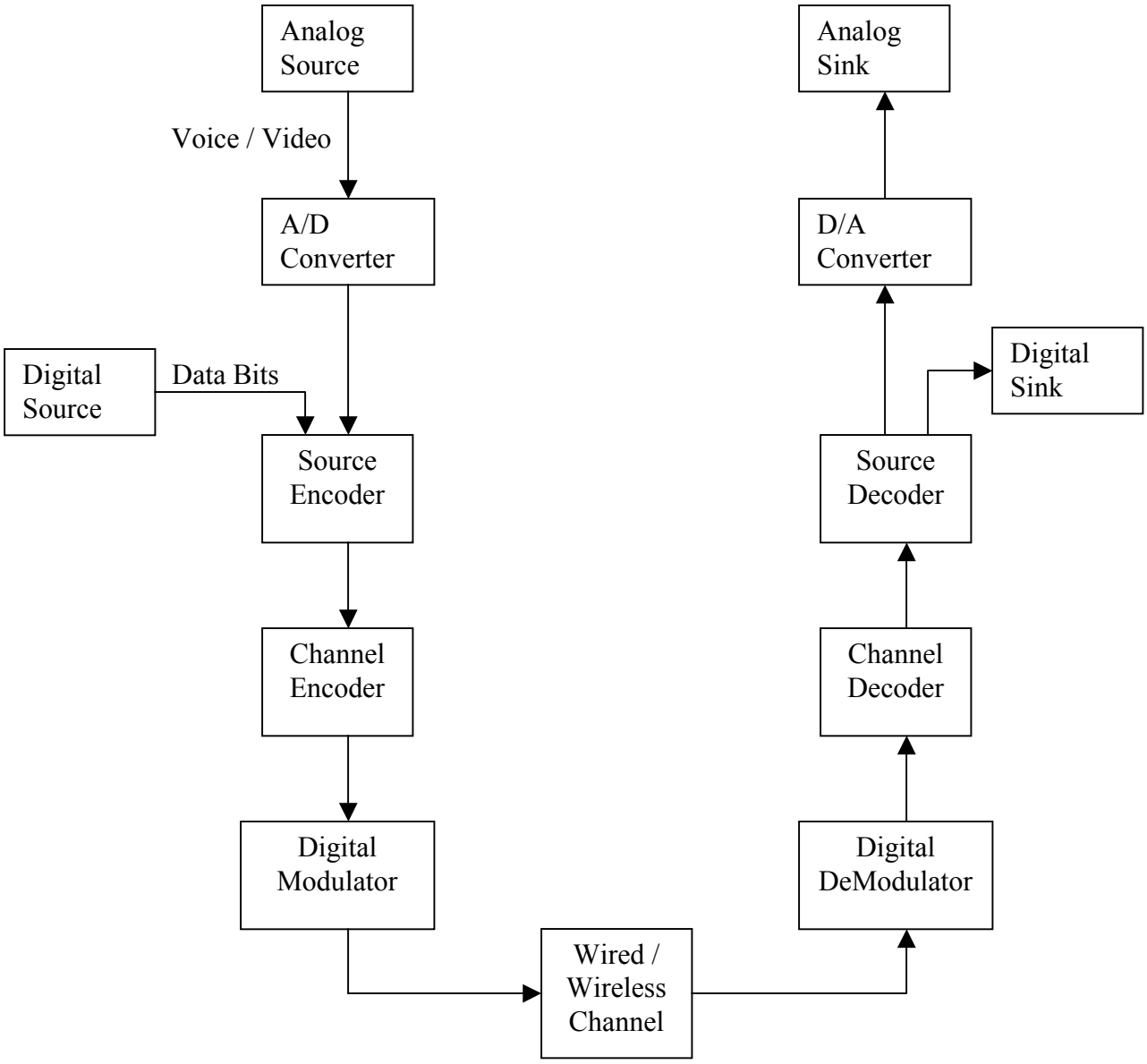


Fig. 1.3 Block diagram of a Generic Communication System.

The block diagram of a generic communication system is shown in Fig. 1.3. The source of a typical communication system could be either analog, for example voice, or digital, for example data from a computer. Digitalizing the output of the information source makes it possible to provide source and channel coding of the data. Thus, the output of the analog source is passed through an analog-to-digital converter whose output is fed into the source-coding block.

It is in the best interest of an efficient communication system to transmit few data symbols as possible. The output of a conventional source may contain a significant amount of redundancy in the data symbols produced for example successive symbols being correlated or non-uniform probability of each symbol. The process of source coding is to remove this redundancy to the extent possible by taking advantage of the structure of input data.

Channel coding selectively adds redundancy to the source encoded data to permit error detection / correction at the receiver. A mathematical combination of the input data bits is used to generate the redundant or parity bits. The channel encoding process could be considered as the reverse process of source encoding in the sense that source encoding compresses the input information by reducing redundancy in it while channel coding introduces redundancy in the source-encoded information. However, this redundancy added as a result of channel coding is more logical and controlled in nature than the inherent redundancy in the original information, and hence, can be effectively utilized by the receiver to detect as well as correct errors in the received sequence.

Since bits are unsuitable to be transmitted over a physical channel, a digital modulator modulates the channel-encoded data into a continuous-time waveform, which is then transmitted over a physical channel. As shown in Fig. 1.3, physical channels are usually of two generic types: wired and wireless. Examples of wired physical channels are coaxial cable, optical fiber, etc., while electromagnetic channels in free space are wireless channels. The source and channel encoded and modulated information signal is passed through an appropriate channel, which distorts the signal by imparting a multitude of effects that are random in nature.

The input to the receiver is the corrupted signal waveform. The digital demodulator in the receiver demodulates the continuous-time waveform and makes estimates of the code bits transmitted. The channel decoder works on the estimates of the code bits, and the redundancy information it possesses owing to the knowledge of the channel code to make estimates of the data bits and / or detect and correct as many errors as possible. The redundancy (i.e. the code bits that were added to the data bits while encoding) is removed at this stage.

The estimates of the data bits are input to the source decoder, which uncompresses the data bits and attempts to reconstruct the original user data by adding appropriate

redundancy to it. The data bits are then delivered to a digital sink or digital-to-analog converted before being delivered to an analog sink.

Efficient channel coding can provide increased power and bandwidth efficiency while controlling the speed of transmission. A channel code has two facets to it. Ideally, a channel code should be as random in nature as possible to combat the effects of the channel while at the same time, it must possess enough structure in it so as to develop a decoding algorithm to be employed at the receiver that detects and corrects errors caused by the channel. Channel encoding and decoding pose a lot of challenging problems for a good communication system design and this indeed will be the primary focus of this thesis.

1.5 Error Control Channel Coding

It was Shannon's channel capacity theorem back in 1948 [Sha 48] that laid the foundation for information theory and error control coding. With the help of concepts such as entropy and mutual information, the theorem set a theoretical upper bound on the capacity of a channel in a noisy environment – a performance bound that is still to be met by any communication system. Ever since, there has been an increased motivation to pursue research in the field of applied mathematics and coding theory. The ultimate goal is to design a communication system, more precisely an efficient channel coding scheme, which will practically realize Shannon's channel capacity bound.

1.5.1 Shannon's Channel Capacity Theorem

Consider an additive white Gaussian noise (AWGN) channel that has an output

$$Z = X + N \,, \tag{1.5.1}$$

where X is the discrete random input to the channel and N is a Gaussian random variable with zero mean and variance σ_N^2.

The channel capacity of an AWGN channel is defined as the maximization of the mutual information between X and Z and is given by

$$C = \max_{p(x)} \{I(X;Z)\} \,, \tag{1.5.2}$$

where the mutual information is given by

$$I(X;Z) = H(Z) - H(Z|X) \,, \tag{1.5.3}$$

where $H(Z)$ is the differential entropy of a random variable Z with variance σ^2 given by

$$H(Z) \leq \frac{1}{2}\log_2(2\pi e\sigma^2). \tag{1.5.4}$$

Therefore, in order to maximize C, $H(Z)$ has to be maximized. A Gaussian distribution with variance σ^2 provides the highest differential entropy. Hence, if $H(Z)$ is to be maximized, Z must be a Gaussian random variable with zero mean and variance σ_Z^2. Since the noise is Gaussian, the variance of the output Z is a sum of the individual variances of the input and the noise.

Thus,

$$H(Z)_{\max} = \frac{1}{2}\log_2(2\pi e(\sigma_X^2 + \sigma_N^2)). \tag{1.5.5}$$

The channel capacity therefore follows

$$C = \frac{1}{2}\log_2(1 + \frac{\sigma_x^2}{\sigma_N^2}), \tag{1.5.6}$$

where σ_x^2 depends on the modulation scheme, and σ_N^2, the variance of Gaussian noise, is equal to $\frac{N_o}{2}$.

For M-ary modulation, the variance σ_x^2 is equal to the average symbol energy E_s.
Channel capacity is, thus, given by

$$\therefore C = \frac{1}{2}\log_2(1 + \frac{2E_s}{N_o}) \qquad \text{bits/symbol,} \tag{1.5.7}$$

where $E_s = rE_b$, the product of the fraction r of data bits per symbol interval and the energy per symbol.

Claude Shannon's channel capacity theorem [Sha 48] thus states that it is possible to achieve reliable communication with as low an error probability as desired as long as the code rate r is less than the channel capacity C (r < C). Conversely, it is not possible to achieve a bit error probability that tends to zero if r > C. Moreover, Shannon has shown that the link between a source encoder and a channel encoder could be a bit stream regardless of the nature of the source and channel.

The channel coding theorem proves the existence of good codes that provide reliable communications with very low bit error probability but does not elaborate on how to design such codes.

1.5.2 Error Control Coding Schemes

Several error detection and correction schemes have been proposed since Shannon published the channel capacity theorem in 1948. Block codes were the first error correcting codes ever discovered and are still very popular. Block codes get their name because the encoder takes as input a message block of finite length, selectively adds redundancy to it, and returns as output a code word that is a larger block as compared to the message. In the late 1940s, Hamming invented a block error correcting code that bore his name [Ham 50]. These codes were such that for every 4 data bits, 3 parity redundant bits would be transmitted. The codes were capable of correcting a single error in the 7 bit pattern. Block codes can be classified in a variety of ways: Linear, cyclic, and systematic. Linear codes are those in which addition of any two code words results in a valid codeword. In cyclic codes, a cyclic shift of any code word is another valid codeword. Systematic codes are those in which the input sequence appears directly in the output. Linear systematic codes have been used in a vast number of communication applications.

Despite its initial success, the Hamming code was very inefficient in correcting a large number of errors. Golay extended the Hamming code to p-ary codes of length $\frac{p^n - 1}{p - 1}$ where p is a prime number and n is the total number of code bits. The Golay codes that were based on Pascal's triangle of binomial coefficients were able to correct two to three errors [Wic 95]. The Golay code was used in Voyager I and II missions to Saturn [Val 99]. The first codes to provide variable error correction capability were Reed-Muller codes, which were invented by Reed and Muller in 1954 [Mul 54]. These codes were used in Mariner and Pioneer space probes. After that, major breakthroughs occurred with the invention of Reed-Solomon (RS) and Bose-Chaudhuri-Hocquenghem (BCH) codes in 1960 [Bos 60]. Both these codes possess variable error correction capability. RS codes are a special case of BCH codes, which possess a more rigid structure and better distance properties. RS codes can correct channel errors in bursts. These codes found applications in Voyager missions to Uranus and Neptune. RS codes are also used in CD players.

Convolutional codes were invented by Elias in 1955 [Eli 55]. These codes were the first to be comprised of memory elements in the form of a shift register. Unlike block codes that break up the message symbol into smaller blocks and encode the entire block at once, a convolutional code continuously encodes a stream of data bits and maps the result to a codeword. The shift register could have k inputs and n outputs. Hence, a convolutional code rate is equal to k/n. Most practical communication systems like GSM, IS-95, etc. use convolutional codes.

Turbo codes, invented in 1993 [Ber 93], perform very close to (about 0.5 dB away from) the Shannon capacity limit. The conventional turbo code is a parallel concatenation of two or more recursive systematic convolutional encoders separated by a uniform

interleaver. Serial concatenated coding schemes have also been proposed that have been shown to have better performance than the parallel case at high signal to noise ratios.

The first practical decoding algorithm to have been invented for block codes was threshold decoding. Many algorithms like algebraic decoding, sequential decoding, etc. were subsequently proposed. These algorithms were limited by the computational complexity. In 1967, Andrew Viterbi brought about a major breakthrough by discovering a maximum-likelihood decoding algorithm for convolutional codes [Vit 67]. This algorithm, named the Viterbi algorithm (VA), reduces the error probability of a sequence of received bits. The VA has a fixed number of computations per step and thus is not bounded by the computational cut off rate. Thus, the VA can easily be built in hardware. This decoding algorithm has been of widespread use in varied applications involving maximum-likelihood estimation of Markov processes.

The maximum-a-posteriori (MAP) algorithm was proposed by Bahl *et al* in 1974 to decode convolutional codes [Bah 74]. The algorithm minimizes bit error probability instead of sequence error probability as in the VA. However, it is more than twice as complex as the VA. The BCJR MAP algorithm, named after its founders, had not come into recognition until the discovery of turbo codes. While the MAP algorithm is the decoding algorithm used for turbo codes, the VA performs as well as the MAP algorithm for convolutional codes.

1.6 Motivation and Outline of Thesis

The purpose of this research is to design and evaluate the various existing forward error correcting channel encoding-decoding schemes (FEC), improve the performance of such schemes by incorporating channel state information making the system adapt to varying channel conditions (ARQ), and analyze the use of such an adaptive system with the latest error control schemes. As shown in the block diagram of the generic communication system in Fig. 1.3, the flow of information is from the transmitter to the receiver. The design of such a simplex system involves deciding a code rate of transmission and encoding the data with the appropriate channel code with this fixed code rate regardless of the nature of the channel at a particular time. The nature of the channel is random and cannot be accurately predicted in a one-way communication system. Moreover, a wireless channel is a hostile environment in which there is fading due to the motion of the mobile and path loss due to the distance between the transmitter and receiver. The various effects of fading in a wireless channel are studied in detail in Chapter 2 of the thesis. A comparison of the nature of the wireless channel with its less hostile wired counterpart is also made.

An objective of efficient channel coding is to constrain the process of information transmission in such a manner that the deleterious effects of the channel are minimized. A coding scheme adaptive to the nature of the channel can be realized only if the information regarding the state of the channel is provided to the transmitter so that it can

modify the rate of coding and transmission. A simplex communication system does not adapt to the state of the channel at a particular time. However, if a feedback channel is provided from the receiver back to the transmitter, various levels of design optimization can be performed and many benefits can be reaped. First, now the receiver can make the channel state information available to the transmitter through the feedback channel. Second, the transmitter can then react to hostile channel conditions by increasing the security provided with a frame of data. This increases the throughput of the system. Such a protocol in which the receiver can ask for a retransmission if the previous frames of data have been received in error and the transmitter can resend the requested frames is called as automatic repeat request (ARQ) protocol and is described in Chapter 3. The advantage of providing the system with an ARQ mechanism is increased system reliability in the sense that now the receiver does not deliver erroneous packets to the user. Instead, it asks for a retransmission of the data from the transmitter. Dedicated ARQ schemes, however, suffer from the drawback of rapidly decreasing throughput if the channel conditions are very hostile. Therefore, combining the forward error correcting (FEC) schemes with the ARQ schemes, called hybrid FEC-ARQ, can ensure reliable communication from end-to-end while maintaining high throughput. Chapter 3 throws light on the different types of hybrid ARQ schemes and the model that is used in this thesis.

Chapter 4 discusses the use of such a hybrid FEC-ARQ scheme with rate compatible punctured convolutional (RCPC) codes. Efficient retransmission is achieved in the proposed scheme by using the concepts of code combining and incremental redundancy. The chapter commences with an introduction to non-systematic convolutional codes. The structure and working of the convolutional encoder and the analytical performance of such codes are understood. An analytical and simulated performance comparison between hard and soft-decision Viterbi decoding is then made. The simulation results of BER and throughput vs. signal-to-noise ratio are shown. It is seen that there is an improvement in throughput performance of the combined hybrid RCPC-ARQ system over the conventional convolutional coded system without the use of ARQ.

Turbo codes, the backbone of this thesis, are the parallel concatenation of constituent convolutional codes (PCCC) separated by an interleaver. Turbo codes have been shown to perform about 0.5 dB away from the Shannon capacity. The bit error rate (BER) performance of turbo codes shows a waterfall region at low signal to noise ratios in which there is a dramatic fall in the bit error rate with slight increase in signal to noise ratio. A rate compatible punctured turbo (RCPT) coded system is simulated in Chapter 5. As seen with convolutional codes, the turbo coded system performance can also be improved vastly in a Hybrid ARQ environment. Such an RCPT – Hybrid ARQ system is simulated in Chapter 5.

Although parallel concatenated convolutional (turbo) codes perform very well at low SNR, they give rise to a fairly high BER floor at high SNR. In order to alleviate this problem, serial concatenated convolutional codes (SCCCs) were proposed and analyzed.

In chapter 6, a hybrid FEC-ARQ technique that is based on rate compatible punctured SCCC codes is proposed. Simulation results show performance over both AWGN and fading channels for both the SCCC-based system and a similar PCCC (turbo code)-based system. The work in this chapter constitutes the original contribution of this thesis and has been published in [Cha 01a].

There is a tight relationship between PCCC and SCCC codes, and in fact the PCCC code can be described as a particular type of punctured SCCC code. By understanding the relationship between SCCC and PCCC, it is possible to design hybrid-codes that benefit from the advantages of each. Chapter 7 throws light on the relationship between PCCC and SCCC and means to construct such hybrid codes.

Finally, chapter 8 lists the applications of this research in future generation communication systems, summarizes the thesis with an emphasis on the thesis's original contribution to the field and provides insight into possible extensions of this work in future.

Chapter 2

The Wireless Channel

The channel, or the path between the transmitter and the receiver, being random in nature, is the most uncontrollable parameter in an end-to-end communication system. A number of effects introduced by the channel cause distortion in the information transmitted making correct estimation of the data bits at the receiver rather difficult. These effects collectively called noise vary in nature based on the type of channel: wired or wireless. A wireless channel is much more complex than its wired counterpart since data transmitted can take multiple paths before it reaches the receiver. Motion of the wireless handset coupled with obstructions in the transmitted paths such as mountains, forests, trees, tall buildings, etc., cause fading in the signal waveform.

It is imperative to understand these effects caused by the unknown channel and model these effects in the design of a communication system. This chapter will first discuss the nature of noise in a wired channel before focusing on the wireless channel in detail.

2.1 Additive White Gaussian Noise (AWGN) Channel

Terrestrial wired channels are AWGN channels. White noise is noise whose power spectral density is uniform over the entire range of frequencies of interest. The term white is used in analogy with white light, which is a superposition of all visible spectral components. The voltage distribution of this noise follows a normal or a Gaussian distribution i.e. a bell shaped curve. Hence, it is called an additive white Gaussian noise channel. The mean of this noise distribution is zero while its variance is a function of the noise spectral density. Fig. 2.1 shows the spectrum of AWGN noise bandlimited between $-\frac{f_s}{2}$ to $\frac{f_s}{2}$ where f_s is the sampling rate of the transmitted signal and N_o is the noise power spectral density.

Let us assume binary phase shift keying (BPSK) modulation. Hence, the amplitude of the transmitted signal is $\pm\sqrt{E_s}$ depending on whether the bit is a 1 or a 0, where E_s is the energy per symbol that is equal to the energy per bit since BPSK modulation is used without coding.

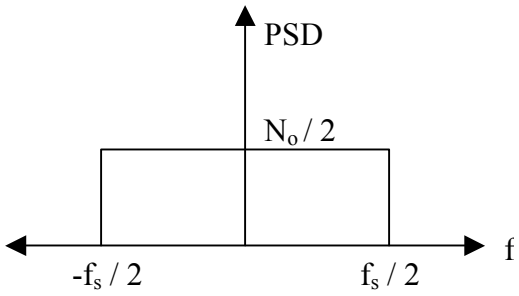


Fig. 2.1 AWGN Noise Spectrum.

The noise variance is given by

$$\sigma^2 = \frac{N_o f_s}{2} \tag{2.1.1}$$

Therefore, the normalized signal to noise ratio (SNR) is

$$\frac{E_b}{N_o} = \frac{(\pm 1)^2 T_b f_s}{2\sigma^2} \tag{2.1.2}$$

where T_b is the time period of a bit,
$T_b f_s$ is the number of samples per bit and
Energy per bit is normalized to unity.

If 1 sample per bit is used, the variance σ^2 can be represented as:

$$\sigma^2 = \frac{1}{2(E_b / N_o)} \tag{2.1.3}$$

This formula is used to compute the AWGN noise variance. The additive white Gaussian noise can be found by taking the product of the standard deviation of the noise and a normally distributed random number with unit variance.

2.2 Difference between Wired and Wireless Channels

A time-invariant white Gaussian noise channel as shown in the previous section models a wired channel. However, this is not the case when a wireless channel is considered. First, the distance between the transmitter and the receiver results in the signal experiencing a loss called path loss [Skl 97]. Second, the data transmitted over a wireless channel typically encounters various obstructions in its path like buildings, trees and forests. The

signal reflects after striking these obstructions and follows multiple paths as it arrives at the receiver at different times. This time spreading phenomenon of the wireless channel is called fading. A wireless channel is severely limited by the amount of fading present. Manifestations of fading are large-scale fading and small-scale fading. These will be discussed in detail in the next few sections of this chapter. Moreover, owing to the relative motion between the transmitter and receiver, a Doppler spreading effect takes place on the signal. Different frequencies of transmission could undergo different levels of fading if the channel is frequency selective. The speed of the mobile determines the amount of Doppler shift the signal will undergo. Thus, in summary, unlike the wired channel, the mobile-radio channel is time-variant because of the motion between the transmitter and receiver, and is much more complex in nature [Rap 96].

Each of the fading channel manifestations is considered in detail in the next few sections.

2.3 Fading Channels

Fading channels are of two types: large-scale fading and small-scale fading [Rap 96]. The block diagram in Fig. 2.2 illustrates the fading channel manifestations and the causes of each type of fading.

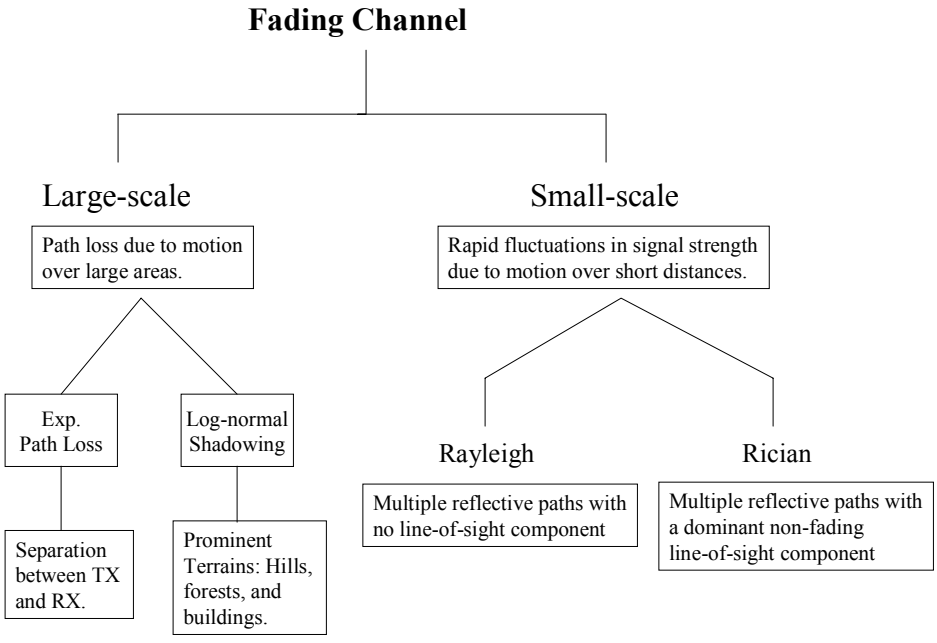


Fig. 2.2 Fading Channel Manifestations.

As shown in the block diagram, large-scale fading is caused due to the separation between the transmitter and the receiver. Motion over the area of separation gives rise to a path loss. Large-scale fading is modeled as a combination of free space path loss and log-normal shadowing.

On the other hand, small-scale fading occurs because rapid fluctuations in signal strength due to constant change in location of the mobile – motion over short distances. Dramatic changes in the signal amplitude and phase can be encountered by small position changes on the order of half a wavelength in the spatial separation between the transmitter and the receiver.

There are two types of small-scale fading called Rayleigh fading and Rician fading. If the multiple reflective paths are large in number and there is no line-of-sight path between the transmitter and the receiver, the envelope of the received signal can be statistically described by a Rayleigh pdf. In the case of there being a dominant non-fading component, such as a line-of-sight path between the transmitter and the receiver, the small-scale fading is described by a Rician pdf.

A mobile radio traversing a large area experiences both large-scale as well as small-scale fading. The resultant fading waveform could be viewed as a superposition of small-scale fading over large-scale fading.

2.3.1 Large-Scale Fading

Large-scale fading, as shown in Fig. 2.2, is caused by prominent terrain contours like hills, buildings, etc. between the transmitter and the receiver and by the separation between transmitter and receiver. The receiver is shadowed by such prominent structures. Large-scale fading is described in terms of a mean path loss due to path loss combined with a log-normal distributed variation about the mean. Due to shadowing, there are various large-scale fading models that compute the path loss as a function of distance. The most widely used models discussed below are the free space model, the exponential path loss model and the log-normal fading model.

2.3.1.1 Free Space Propagation Model

The free space model considers the following [Skl 97]:

- Region between the transmitter and the receiver is free of all objects that might absorb or reflect radio frequency energy.
- Within the separation between the transmitter and the receiver, the atmosphere acts as a perfectly uniform and non-absorbing medium.
- The earth is infinitely far away from the propagating signal.

In this idealized free space model, the attenuation or path loss between the transmitter and receiver is directly proportional to the nth power of the distance between the

transmitter and the receiver and is inversely proportional to the nth power of the wavelength of the propagating signal. For the free space model, n, the path loss exponent is equal to 2. The equation for path loss is given as [Rap 96]

$$L_s(d) = \left(\frac{4\pi d}{\lambda}\right)^n, \tag{2.3.1}$$

where $L_s(d)$ is the path loss with respect to distance, d is the distance between transmitter and receiver, λ is the wavelength of the propagating signal, and n (path loss exponent) is equal to 2.

The received power in dB is, therefore,

$$P_r(d) = P_t - L_s(d). \tag{2.3.2}$$

The power received is thus predicted by subtracting the path loss from the transmitted power in decibels. However, the free space model is impractical when a wireless mobile channel is considered where signal propagation takes place in the atmosphere and near the ground over multiple reflected paths.

There are three basic mechanisms that affect signal propagation in a communication system:

- *Reflection* occurs when a propagating electromagnetic wave impinges on a smooth surface with very large dimensions as compared to the RF signal wavelength.
- *Diffraction* occurs when a dense body obstructs the path between the transmitter and receiver with large dimensions compared to λ causing secondary waves to be formed.
- *Scattering* occurs when a radio wave impinges on a large rough surface or any surface whose dimensions are on the order of λ or less causing the reflected energy to spread out in all directions.

More practical large-scale fading models than the free space model incorporate these mechanisms while computing the received signal power or effectively, the path loss.
Furthermore, the free space propagation has a path loss exponent equal to 2 while when reflections and obstructions are present like in the case of a mobile radio channel, the path loss exponent is much higher. It is found to be ranging between 2 and 4 based on the propagation environment as shown in Table 2.1.

Environment	Path loss exponent, n
Free space	2
Urban area cellular	2.7 – 3.5
Shadowed urban cellular	3 – 5
Obstructed in building	4 – 6
In building line-of-sight	1.6 – 1.8
Obstructed in factories	2 – 3

Table 2.1 Path Loss Exponents based on Environment [Rap 96].

2.3.1.2 Log-normal shadowing

Okumura made path loss measurements for a wide range of antenna heights and coverage distances. Hata transformed Okumura's data into parametric formulas. The mean path loss, $\overline{L}_p(d)$, as a function of distance d between the transmitter and receiver is proportional to a n^{th} power of d relative to a reference distance d_o as is given as

$$\overline{L}_p(d) \propto \left(\frac{d}{d_o}\right)^n . \tag{2.3.3}$$

$\overline{L}_p(d)$ is stated in decibels as

$$\overline{L}_p(d)(dB) = L_s(d_o)(dB) + 10n\log\left(\frac{d}{d_o}\right). \tag{2.3.4}$$

The reference distance d_o is a point located on the far field of the antenna. The value of d_o is taken to be 1 km for large cells, 100 m for microcells and 1 m for indoor channels.

The path loss versus distance $\overline{L}_p(d)$ is an average and therefore, needs to be modified to provide for variations about the mean based on the environment of the different cites. Measurements of path loss over distance show that there is a significant deviation in the average path loss and is modeled as a log-normal random variable in addition to the mean.

Therefore,

$$L_p(d)(dB) = L_s(d_o)(dB) + 10n\log\left(\frac{d}{d_o}\right) + X_\sigma(dB), \tag{2.3.5}$$

where X_σ is a Gaussian random variable with a standard deviation σ (both in decibels). This random variable has a value that is based on measurements. Large-scale fading is, thus, modeled with the help of above equation.

2.3.2 Small-Scale Fading

As seen earlier, small-scale fading is of two types: Rayleigh and Rician. In the presence of a dominant non-fading specular component, the received signal envelope is described by a Rician pdf. As the amplitude of the specular component approaches zero, the Rician pdf approaches a Rayleigh pdf and is given as [Rap 96]

$$p(r) = \begin{cases} \dfrac{r}{\sigma^2}\exp\left(\dfrac{-r^2}{2\sigma^2}\right) & \text{for } r \geq 0 \\ 0 & \text{otherwise,} \end{cases} \qquad (2.3.6)$$

where r is the envelope amplitude of the received signal and σ is the rms value of the received voltage signal before envelope detection.

A typical signal envelope with Rayleigh fading pdf is shown in Fig. 2.3. The mobile speed is 120 Km/hr and the carrier frequency is 900 Mhz.

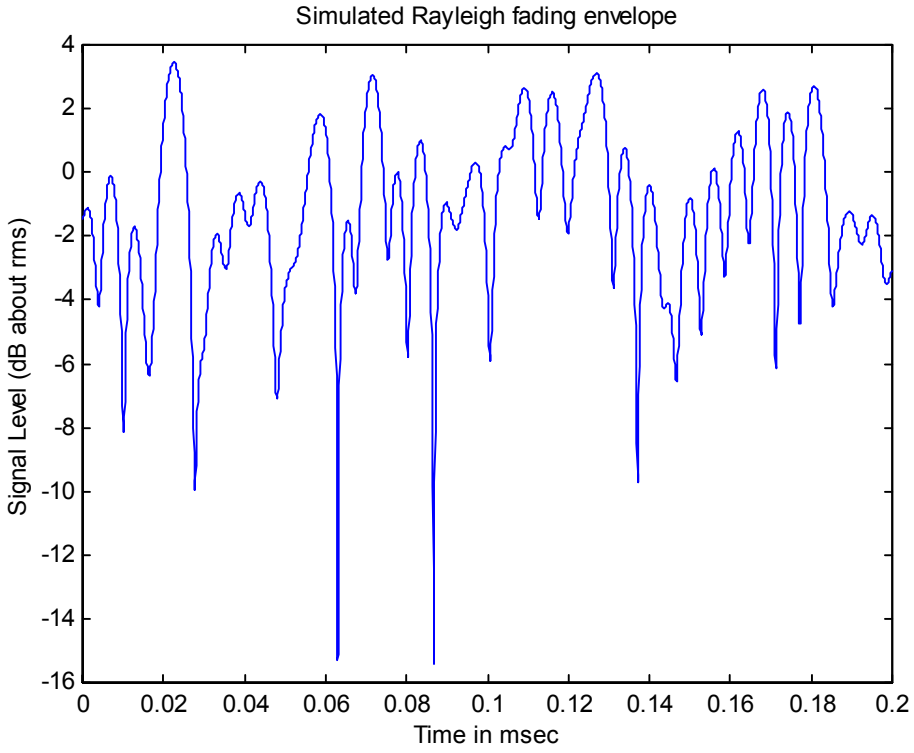


Fig. 2.3 A typical Rayleigh faded envelope experienced by a mobile traveling at 120 Km/hr and carrier frequency of 900 Mhz.

2.3.2.1 Types of Small-scale Fading

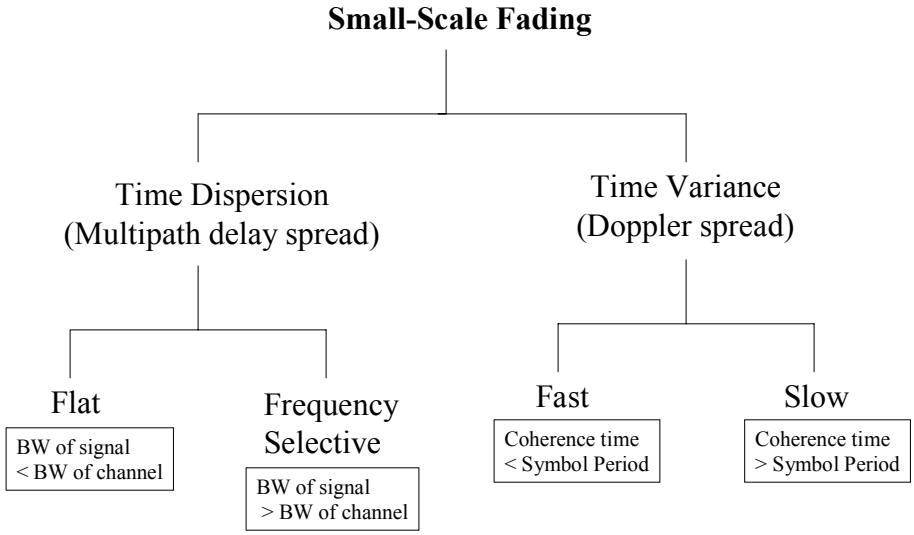


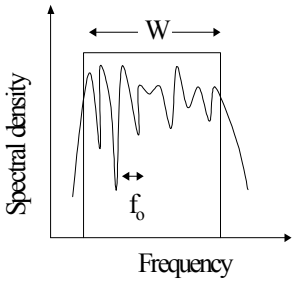
Fig. 2.4 Types of small-scale fading.

Multipath in the radio channel creates small-scale fading effects such as:

- Dramatic changes in signal strength due to motion over small distances.
- Random frequency modulation and broadening of the signals bandwidth due to varying Doppler shifts on different multipaths.
- Time dispersion caused by multipath delay.

The Doppler shift and multipath manifestations of small-scale fading are shown in Fig. 2.4. The two propagation mechanisms, time dispersion (multipath) and time variance (Doppler shift) are independent of one another. Four possible effects are caused by the time dispersion and frequency dispersion mechanisms.

The multipath delay spread causes the signal to undergo either flat or frequency selective fading. A channel will exhibit frequency-selective fading if the received multipath components (with time period T_m) extend beyond the symbol's time duration (T_s) i.e. $T_m > T_s$. Another name for frequency-selective fading is channel-induced inter-symbol interference. In this type of fading, the signal's spectral components are not all affected equally by the channel. Some components falling outside the coherence bandwidth (range of frequencies over which channel passes all the spectral components with approximately equal gain and phase) will be affected differently to those falling within the coherence bandwidth f_o.



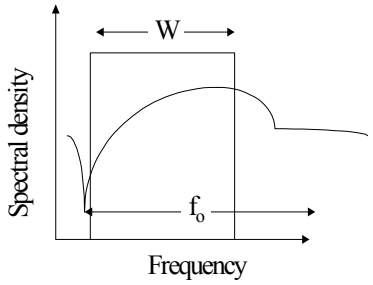


Fig. 2.5a Frequency-Selective Fading Channel. Fig. 2.5b Flat Fading Channel.

A channel is supposed to be a frequency non-selective or a flat fading channel if all the multipath components of the signal arrive within the symbol duration i.e. $T_m < T_s$. This also resembles the case when all the spectral components of a signal are affected in a similar manner by the channel i.e. $f_o > W$ where W is the bandwidth of the signal. Fig. 2.5a shows the power spectral density of a typical frequency selective fading channel while Fig. 2.5b shows that of a flat fading channel.

The time-variant nature or the Doppler spread manifestation of small-scale fading gives rise to two types of effects: fast fading and slow fading. A channel is a fast fading channel if $T_o < T_s$ i.e. the channel coherence time is smaller than the symbol duration. Therefore, the fading character of the channel will change several times within the symbol duration causing channel induced inter-symbol interference. A channel is referred to as a slow fading channel if $T_o > T_s$ i.e. the channel coherence time is greater than the symbol duration. Thus, the channel state will remain constant over the duration of the entire symbol.

2.4 Correlated and Fully Interleaved Fading

A correlated fading channel is one in which successive fading coefficients are correlated i.e. the successive fading coefficients are not dependent on each other. It can be generated by passing X and Y through a filter with an impulse response given by [Rap 96]

$$h[n] = J_o(f_d T_s n), \tag{2.4.2}$$

where f_d is the Doppler spread of the signal,

T_s is the symbol duration, and

J_o is a zero-order Bessel function of the first kind.

In general, all radio channels possess correlated fading. If the correlated fading channel is passed through a channel interleaver, a fully interleaved fading channel can be obtained.

A fully interleaved fading channel is one in which the fading coefficients are independent and identically distributed (i.i.d) complex Gaussian random variables[1]. A fully interleaved Rayleigh fading channel is the envelope of a complex Gaussian random process, which is made up of two independent Gaussian random variables as given by

$$Z = |X + jY|, \tag{2.4.1}$$

where X and Y are independent Gaussian random variables.

2.5 Practical Channel Model

There are a couple of channel models that will be used in the following chapters of the thesis:

- An Additive White Gaussian Noise (AWGN) Channel, as discussed in section 2.1, is the channel in which AWGN noise is added at the receiver front end. This noise models the thermal noise generated by reaction between electronic devices in the receiver with the radio frequency along with the cosmic noise picked up by the antenna and various other uniformly distributed noises.
- Fully interleaved Rayleigh flat fading channel as discussed in section 2.3 and 2.4.

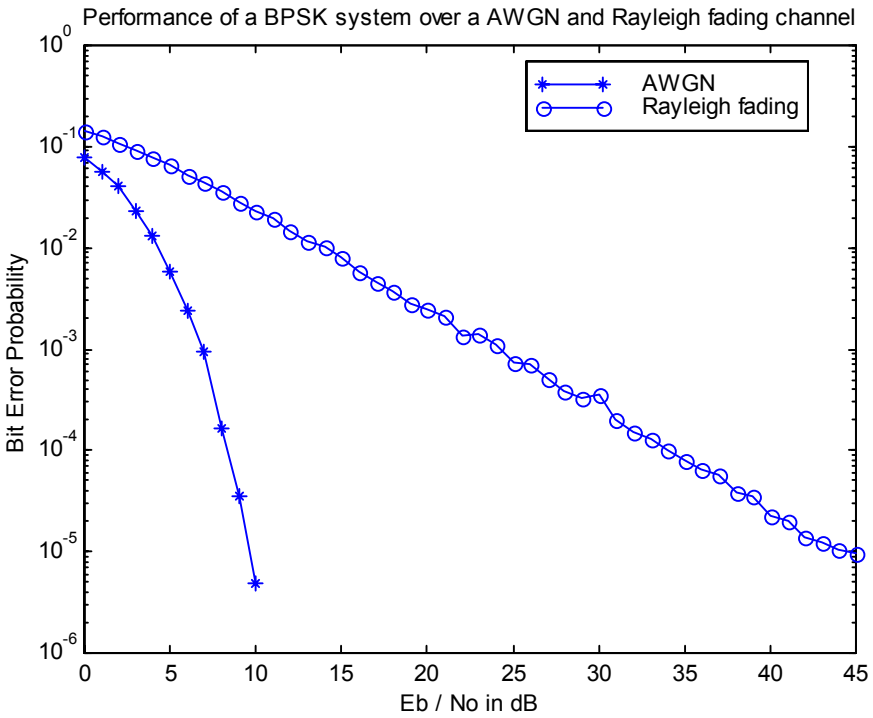
These are the two channel models that will be used in the remainder of the thesis.

2.6 Performance of a BPSK System in a Noisy Channel

The mobile radio channel is much more hostile in nature as compared to the conventional AWGN channel. Moreover, the Rayleigh fading channel is a worst-case scenario since it assumes no line-of-sight path between the transmitter and the receiver. In order to emphasize the amount of signal degradation undergone over a fading channel, a simple BPSK system is simulated over a AWGN and a Rayleigh flat fading channel and is shown in Fig. 2.6.

As seen from the plot, the performance of a BPSK system without channel coding is a lot worse in Rayleigh fading than it is in AWGN. The BPSK system in a Rayleigh fading channel requires ≈ 13 dB more power than that in an AWGN channel at BER of 10^{-3} (which is a benchmark for voice communications) while it requires ≈ 34.4 dB more power to achieve a BER of 10^{-5} (which is a benchmark for data communications)

[1] The envelope of a zero-mean complex Gaussian random variable is Rayleigh distributed, and the envelope of a complex Gaussian random variable with a real positive mean is Rician distributed.



.

Fig. 2.6 Performance of a BPSK system in an AWGN and Rayleigh flat fading channel.

2.7 Chapter Summary

The wireless channel is far more hostile in nature as compared to the wired channel. A multitude of channel effects discussed in the first few sections of this chapter cause severe degradation of the transmitted signal in such a hostile environment. Understanding the nature of the channel and introducing redundancy into the transmitted information (the process of channel coding) can drastically reduce the required signal to noise ratio to achieve the benchmark BER performances in both AWGN and fading channel environments. Every decibel of increase in required power greatly increases the complexity and cost of the system. Finally, with a practical system example, the detrimental effects introduced by the channel are explained and the need for efficient channel coding techniques is highlighted.

Chapter 3

Automatic Repeat Request Protocols

In a simplex data communication system, the flow of information is only in one direction – from transmitter to receiver. The error control provided in these systems is called forward error correction (FEC) owing to the forward nature of information flow [Cos 83]. The transmitter cannot adapt to channel conditions unless it receives feedback from the receiver about the channel state. If a feedback channel is provided from the receiver back to the transmitter, two-way communication can be employed, which opens up a lot of design options. On evaluating the previously received packet of data, the receiver can relay the channel state information back to the transmitter by means of a request for retransmission of the erroneous packet. This allows the transmitter to adjust the code / transmission rate and adapt to the nature of the channel accordingly.

The receiver in a simplex system has no choice but to deliver even erroneous packets to the sink or discard the packet completely. The receiver in a two-way system detects and corrects errors in a similar manner to that of the simplex system, but on detecting errors in the received packet, the receiver either discards it or stores it without delivering to the sink. The receiver then generates a request for retransmission, which is called **automatic repeat request (ARQ).**

The introduction of ARQ protocols dates back to the early days of digital computers. In 1960, a major development took place when Wozencraft and Horstein [Woz 60] described and analyzed a system that could detect as well as correct errors with retransmission requests. Their system provided significant improvement in performance over pure ARQ protocols. Benice and Frey [Fre 64], in 1964, introduced three basic types of ARQ protocols: stop and wait, go-back-N, and selective repeat protocols. In 1977, Sindhu [Sin 77] introduced the scheme in which erroneous packets are not discarded but stored and combined with additional copies of the packet generating a single packet with more reliability than the individual ones. Since then, a large number of systems have been proposed that involve some form of packet combining.

The next section discusses the three basic types of ARQ protocols and analyzes their performance. The concept of hybrid ARQ is introduced and its various types are then elaborated. Finally, the ARQ model used in this thesis is described.

3.1 Retransmission Protocols

There are three types of ARQ protocols: stop and wait, go-back-N, and selective repeat [Cos 83]. They differ in the following:

- Number of packets that the transmitter can transmit without accepting acknowledgements from the receiver for the previously transmitted packets, and
- Storage availability at the transmitter and receiver.

The performance of ARQ systems can be evaluated by two basic parameters:

- Reliability or packet error rate, which is the percentage of packets accepted by the receiver that contain one or more symbol/bit errors.
- Throughput of the system, which is the percentage ratio of successful number of data bits to successful number of code bits.

3.1.1 Stop and Wait ARQ Protocol

The mechanism of the stop and wait ARQ protocol is described by its name. In this type of ARQ system, the transmitter transmits a single packet and waits for the receiver to acknowledge the receipt of the packet. If the receiver finds the packet to contain errors, it asks for a retransmission of the packet. The transmitter does not transmit any further packets until the previous packet is correctly received. This is depicted in the following traffic diagram Fig. 3.1 [Wic 95].

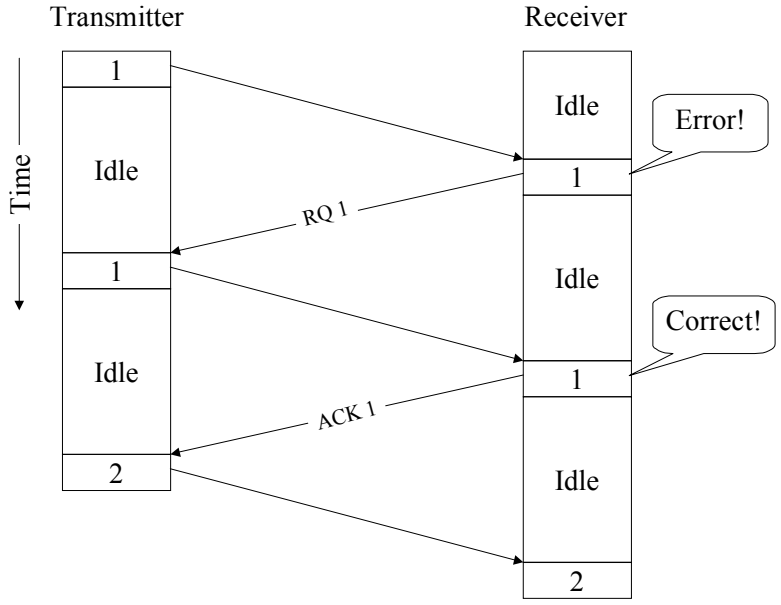


Fig. 3.1 A stop and wait ARQ protocol.

The benefit of this protocol is that there is no buffer space required either at the transmitter or the receiver. The erroneously received packets are discarded immediately. The disadvantage is that the channel must sit idle while the transmitter awaits an ACK / RQ.

3.1.1.1 Throughput analysis of Stop and Wait ARQ protocol

It is seen from Fig. 3.1 that the transmitter is idle when it waits for an acknowledgement from the receiver. This idle time is a function of the round-trip delay(λ), which is the sum of forward propagation delay(λ_1), feedback propagation delay(λ_2), and the processing time of the receiver(δ).

If the transmission rate is D bits/sec, then the idle time can be expressed in terms of the number of bits Γ that could have been transmitted during that time as

$$\Gamma = D(\lambda_1 + \lambda_2 + \delta) \quad \text{bits.} \tag{3.1.1}$$

Let T_r transmissions on average be necessary before the receiver accepts the packet without error. Each transmission involves the transmission of an n-bit (code) packet followed by an idle period. Therefore, the stop and wait ARQ requires the transmission of $T_r(n+\Gamma)$ bits to move a k-bit (information) packet from end-to-end. Thus, the throughput of such a system is [Wic 95]

$$\begin{aligned} \eta_{sw} &= \frac{k}{T_r(n+\Gamma)} \\ &= R\left(\frac{1-P_r}{1+\Gamma/n}\right), \end{aligned} \tag{3.1.2}$$

where P_r is the probability that a retransmission request is generated (frame / packet error rate), R is the rate of the error detecting code = k / n, k is the number of data bits and r is the number of code bits.

3.1.2 Go-Back-N ARQ Protocol

The stop and wait protocol's major disadvantage is that its throughput performance is very poor because the transmitter must sit idle as it waits for the acknowledgement from the receiver. Again, the advantage is that no buffering is required either at the transmitter or the receiver.

Buffering at the transmitter permits it to send more than one packet while it stores the previously transmitted packets so that it can resend the packets when requested. If this kind of buffering can be provided to an ARQ system, a go-back-N ARQ protocol can be

used. Basically, in the go-back-N protocol, the transmitter sends a continuous stream of packets without waiting for acknowledgements from the receiver until a pre-determined window size N is reached. Once the window size is reached, the transmitter stops transmitting and waits for the receiver to acknowledge a transmitted packet. This protocol is described with a traffic diagram in Fig. 3.2.

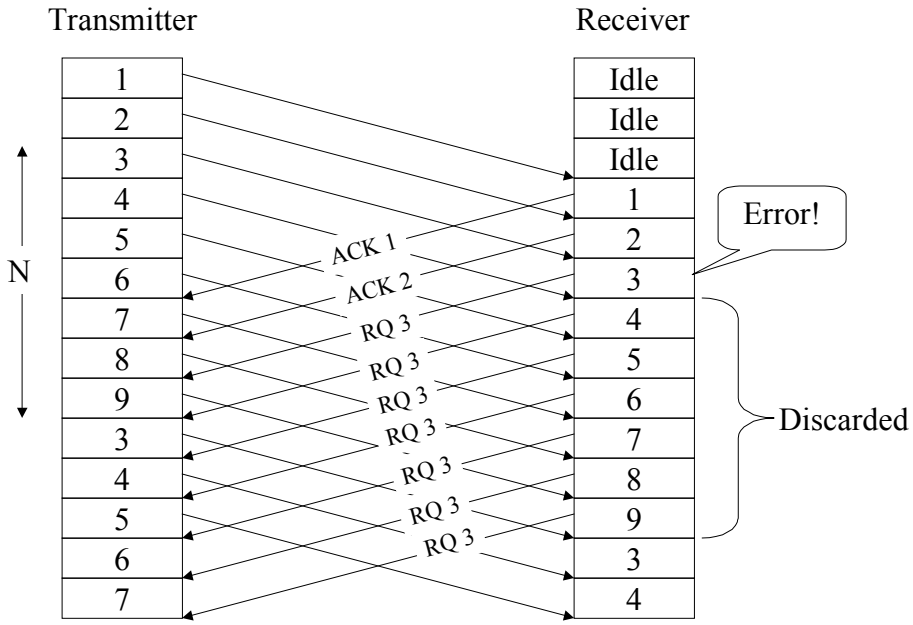


Fig. 3.2 A Go-Back-N ARQ protocol.

Say the window size used at the transmitter is 7. The transmitter then transmits 7 packets continuously and simultaneously waits for acknowledgements from the receiver. If acknowledgements arrive while the transmitter is transmitting packets within the window size, the transmitter does not stop when it reaches the window size but transmits an additional number of packets equal to the number of acknowledgements received. If no acknowledgements have been received until transmitter transmits 7 packets, then it waits after transmitting 7 packets for an acknowledgement from the receiver. As soon as the acknowledgement arrives, it starts transmitting the next packet and transmits as many packets as the number of acknowledgements received.

In Fig. 3.2, the transmitter receives acknowledgements of the first two packets even before 7 packets are transmitted and hence it continues to transmit an additional two packets and waits for the acknowledgement of the third packet from the receiver. Since the third packet is in error, the receiver asks for a retransmission of the third packet. The receiver discards all the packets that followed packet no. 3 and keeps requesting for packet no. 3 to be retransmitted. This is because the receiver cannot release unordered

packets to the sink. The transmitter completes transmission of 9 packets and sees that the receiver is requesting for the third packet. Since there is no buffer space at the receiver, the transmitter now has to go back N = 7 packets and start transmitting from the next packet (packet no. 3) onwards once again.

Therefore, the disadvantage of the go-back-N scheme is that there is a waste of system resources when the transmitter has to retransmit all the packets after the packet that was received in error. However, since idle time is greatly reduced, the throughput of a go-back-N ARQ protocol is far better than the stop and wait protocol for reasonable values of P_r. In practice, systems could have an adaptive window size depending on the packet error rate. In fact, the go-back-N protocol with a window size equal to one is same as the stop and wait ARQ protocol.

3.1.2.1 Throughput analysis of Go-Back-N protocol

The parameter N in go-back-N is the smallest number of packets that contain at least Γ bits assuming each packet is an n-bit packet. N is a function of the forward and feedback propagation delays, and the receiver processing time. Therefore,

$$N = \frac{D(\lambda_1 + \lambda_2 + \delta)}{n} = \frac{\Gamma}{n}. \tag{3.1.3}$$

Each retransmission request causes the retransmission of a total of N packets. Therefore, the throughput for the go-back-N scheme is given by

$$\begin{aligned} \eta_{GBN} &= \left(\frac{k}{n}\right)\left(\frac{1}{1+(T_r-1)N}\right) \\ &= R\left(\frac{1-P_r}{1+P_r(N-1)}\right). \end{aligned} \tag{3.1.4}$$

The Go-Back-N protocol is less sensitive to propagation delays and this reduction in sensitivity is obtained at the expense of increasing the buffer size at the transmitter.

3.1.3 Selective Repeat ARQ Protocol

The selective repeat ARQ protocol is an extension of the go-back-N protocol in which buffering is provided at both the transmitter and receiver. The functioning of the transmitter is similar to that in the go-back-N scheme. The transmitter continuously sends a stream of packets while simultaneously receiving the acknowledgements from the receiver. In case the receiver sends a negative acknowledgement or a request for retransmission, the transmitter goes back to the packet that was received in error and retransmits only that packet. It then resumes its transmission from where it stopped

before. The receiver stores the successive packets received after the erroneous packet and asks for the retransmission of only the erroneous packet.

Therefore, owing to the buffering provided, the receiver is able to selectively ask for a retransmission from the transmitter, which retransmits only the erroneous packet. This is possible only because of the tradeoff of providing buffering at both ends of the system. The selective repeat protocol is described in Fig. 3.3.

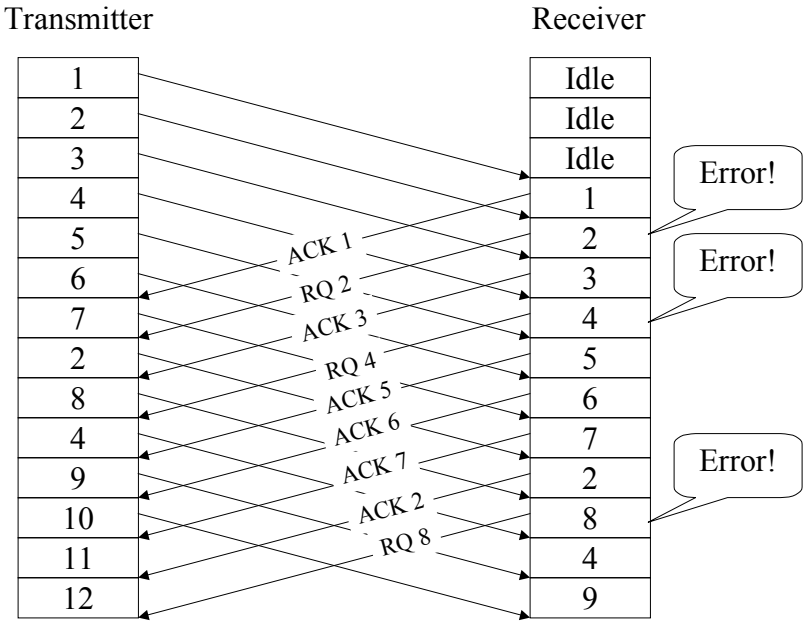


Fig. 3.3 A Selective Repeat ARQ protocol.

As seen in the diagram, the second, fourth and eighth packets are received in error. The receiver asks for retransmission of these packets and stores the successively received packets in the buffer. The transmitter resends only those packets that the receiver asked for and resumes its continuous stream of packet transmission.

3.1.3.1 Throughput of Selective Repeat ARQ Protocol

Since each retransmission results in the transmission of only one packet, the throughput is easily derived as [Wic 95]

$$\eta_{SR} = \left(\frac{k}{n}\right)\left(\frac{1}{T_r}\right) = R(1-P_r)\,. \tag{3.1.5}$$

The throughput of selective repeat ARQ protocol is the best of the three protocols discussed. However, it is only achieved with a tradeoff of having to provide buffering at both the transmitter and the receiver.

3.2 Hybrid ARQ Protocols

Although ARQ systems are simple, easy to implement and provide high system reliability, they suffer a severe drawback of the throughput decreasing rapidly with increased channel error rates. Forward error correction (FEC) systems maintain constant throughput (equal to the code rate R) irrespective of the channel error rates. However, FEC systems have two major drawbacks. First, when a received sequence is detected in error, the sequence has to be decoded and the decoder output has to be delivered to the user regardless of whether it is correct or incorrect. Since the probability of decoding error is usually greater than the probability of an undetected error, FEC systems are not highly reliable. Second, in order to achieve high system reliability, a long powerful code must be used, which can correct a large number of error patterns. This makes the decoder hard to implement and expensive.

The advantage of ARQ systems of obtaining high reliability can be coupled with the advantage of FEC systems to provide constant throughput even in poor channel conditions. Such a system, which is a combination of the two basic error control schemes – FEC and ARQ, is called a hybrid ARQ system.

The FEC block embedded in an ARQ system constitutes the hybrid ARQ system. The functioning of the FEC and ARQ blocks go hand in hand. The FEC block reduces the frequency of retransmission by correcting the commonly occurring error patterns. This increases the throughput of the system. When an error pattern is detected but cannot be corrected, the receiver requests a retransmission instead of delivering the erroneous packet to the user. This increases system reliability. Thus, the hybrid ARQ system exploits the advantages of the conventional FEC and ARQ systems by combining them effectively.

Hybrid ARQ systems are classified into two generic types differing in the manner in which the receiver treats erroneous packets. The types are

- Type-I hybrid FEC-ARQ.
- Type-II hybrid FEC-ARQ.

The following discussion throws light on the two generic types of hybrid ARQ.

3.2.1 Type-I Hybrid FEC-ARQ

In this scheme, the receiver discards the packet upon erroneous reception (i.e when errors remain after FEC decoding) at a particular code rate and asks for an entirely new

retransmission. Retransmissions take place at either the same or lower code rate until the packet is correctly received or until a pre-set number of retransmissions have taken place. Although this method does not require a large buffer at the receiver, it is a very inefficient method of implementing ARQ.

3.2.2 Type-II Hybrid FEC-ARQ

In this scheme, the receiver stores erroneous packets in a buffer so that they can be reused after subsequent retransmissions. Storing the previously received data allows the concept of incremental redundancy to be exploited. With incremental redundancy, the system begins by transmitting at the highest possible FEC code rate (this rate could be 1 if no parity bits are used). If a retransmission is requested, then only some of the previously unused parity bits are transmitted. With each retransmission, more and more of the parity bits are transmitted and the codeword is strengthened until eventually the receiver has all of the parity bits. Incremental redundancy allows the effective code rate to be gradually lowered until the packet can be successfully decoded. As can be expected, the throughput of type-II hybrid FEC-ARQ is significantly higher than it is for type-I hybrid FEC-ARQ, especially when multiple copies of the received erroneous packets are combined.

3.2.3 Code Combining and Diversity Combining

Code combining and diversity combining are forms of packet combining schemes. In packet combining, each received packet is combined with its predecessors until the combined packet is correctly decoded. Performance vastly improves if a large number of code rates are used and the receiver combines the code words at various rates that caused retransmission requests.

The two forms of packet combining – code and diversity combining, however, are slightly different from each other. In code combining, the packets are concatenated to form noise corrupted code words from increasingly long and lower-rate codes. The individual transmissions are encoded at same code rate R. Assuming N packets have caused retransmission requests at the receiver, these packets are concatenated to form a single packet. The new packet is now of code rate R/N. As the number of retransmission requests increase, more and more packets are combined together and the strength of the code word at the receiver is increased until it is correctly decoded. David Chase introduced the concept of code combining back in 1985 [Cha 85].

In diversity combining systems, multiple but identical copies of a packet are concatenated to create a packet whose individual symbols are more reliable than those in the individual packets. Examples of diversity combining schemes are symbol voting scheme for hard-decision decoding and symbol averaging for soft-decision decoding.

3.3 Hybrid FEC-ARQ Model Used

The type-II hybrid FEC-ARQ model described above is proposed for use in this thesis. The model incorporates incremental redundancy and code combining. An assumption of a noise-free feedback channel relaying channel state information from the receiver to the transmitter is made. Also, perfect error detection is assumed at the receiver. This means that all errors that occur will be detected. However, the number of errors corrected will depend on the channel coding scheme used. A selective repeat ARQ protocol is the inherent repeat request scheme while convolutional coding, parallel concatenated convolutional coding (turbo coding) and serial concatenated convolutional coding are used in this thesis as the inherent FEC schemes in the type-II hybrid FEC-ARQ system. It can be noted from the throughput equations for stop and wait, and selective repeat protocols that if no idle times are assumed, the performance of the stop and wait ARQ scheme is identical to that of the selective repeat ARQ scheme. Thus, a stop and wait ARQ protocol is used in the thesis owing to its ease of analysis. Because this type-II hybrid ARQ system can adapt to the varying channel conditions, it allows for the efficient utilization of channel resources.

3.4 Chapter Summary

This chapter introduced the basic concepts of automatic repeat request (ARQ) protocols and elaborated on the various kinds of ARQ protocols. The combination of FEC and ARQ called hybrid ARQ provides vast performance improvement in the system. The FEC plays the role of cleaning up the channel and if errors still remain, a retransmission request is issued. Various types of hybrid ARQ are discussed based on how a retransmission request is handled. Finally, an efficient type-II hybrid ARQ model that will be used in this thesis is described.

Chapter 4

Hybrid RCPC-ARQ Codes

Convolutional codes were invented by Elias in 1955 [Eli 55]. As introduced in Chapter 1, convolutional codes continuously encode an input stream of bits. The output of the encoder is a convolution of the current input data bit (s) with some or all of the m previous input bit (s) that are stored in the encoder's shift register memory where m is the number of memory elements in the register. In general, the rate (r) of a convolutional code is a ratio of the number of inputs (k) to the encoder and the number of outputs (n) from the encoder. The output code bits are then modulated and transmitted over the channel.

This code rate established by the channel encoder must be made adaptive to the channel conditions. Rather than using a preset code rate, the transmitter must be able to transmit at the highest code rate (only information bits are transmitted if the code is systematic) when the channel is sensed to be clean. If at any time, the channel begins to exhibit hostile properties, the transmitter reduces the code rate until it reaches the lowest encoder code rate.

Puncturing, first introduced by Mandelbaum [Man 74], is a process that helps to achieve this variation in the rate of transmission of the code. Puncturing a code is the process of selectively deleting a subset of bits from a frame of encoded data before transmitting the frame. If the code is systematic (i.e. the input data bits appear unaltered in the output code word), the parity bits generated by the encoder are usually punctured while the systematic bits are transmitted. On the other hand, if the encoder is non-systematic, then any of the parity bits generated by the encoder can be punctured. Since the punctured bits are not transmitted, the effective code rate is increased.

The r*ate compatibility* criterion was introduced for punctured convolutional codes by Hagenauer [Hag 88]. This restriction implies an incremental redundancy scheme in which a low rate codeword will utilize all the bits that were transmitted previously at a higher rate. Incremental redundancy means that the transmitter, while transmitting at a lower rate, will only need to transmit the additional bits that were punctured earlier. This criterion reduces the amount of overhead in transmission and utilizes the channel resources efficiently.

Rate Compatible Punctured Convolutional (RCPC) codes combine the principles of puncturing and rate compatibility with convolutional codes. These codes are rate compatible and can effectively adapt to the source and channel requirements of a system when combined with an ARQ system described in chapter 3. RCPC codes provide an efficient means of implementing a variable rate error control system using a single encoder / decoder pair. Since rate is directly proportional to throughput, an increase in the rate of transmission brings about increased throughput provided the code word is received correctly (i.e. the system is reliable). This reliability is achieved by using an ARQ system in conjunction with RCPC codes. The overall hybrid RCPC-ARQ system works very well even at high channel error rates and vast improvement in system performance is seen over conventional convolutional code performance.

This chapter gives a comprehensive description of RCPC codes and a hybrid RCPC-ARQ system. The first few sections provide an understanding of convolutional encoding, Viterbi decoding and the theoretical performance of these codes. A simulation-based performance comparison between hard and soft decision Viterbi decoding follows next. The concepts of puncturing and rate compatibility are then elaborated in detail and applied to convolutional codes. The next few sections describe the hybrid RCPC-ARQ system model used in this thesis and the simulated performance curves of bit error rate (BER) and throughput vs. signal to noise ratio are plotted and reviewed.

4.1 Encoding of Convolutional Codes

Fig. 4.1 shows a typical rate 1/2 linear convolutional encoder. As shown in the figure, an input binary data stream x consisting of a series of bits $(x_0, x_1, x_2, \ldots)$, is fed into a shift register having m memory elements. The output streams $y^{(1)} = (y_0^{(1)}, y_1^{(1)}, \ldots)$ and $y^{(2)} = (y_0^{(2)}, y_1^{(2)}, \ldots)$ each contain an encoded output bit corresponding to every input bit and hence the code rate of this encoder is 1/2. Each encoded output stream is formed by modulo-2 addition of an input bit with tapped values from memory elements in the shift register according to a fixed pattern. This fixed pattern of the tap positions is called the generator sequence of the encoder. The generator sequence $g_i^{(j)}$ is the impulse response obtained at the jth output of the encoder by applying a single 1 at the ith input followed by a string of zeros. Since the encoder in Fig. 4.1 has only one input, the generator sequences of this encoder are given as

$$\begin{aligned} g^{(1)} &= (111) \\ g^{(2)} &= (101). \end{aligned} \tag{4.1.1}$$

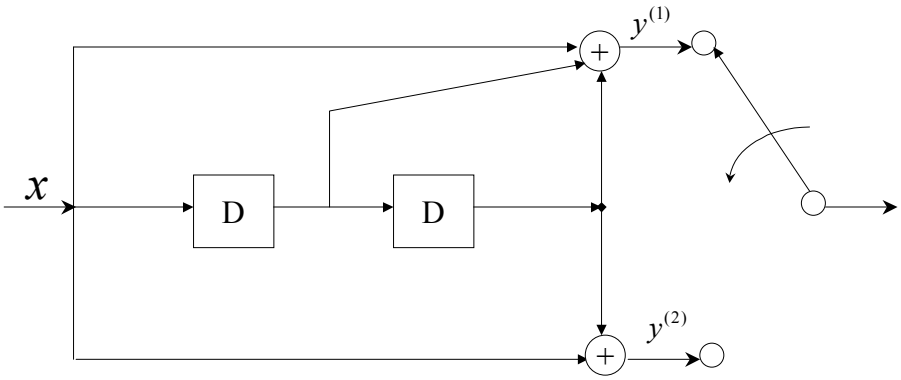


Fig. 4.1 A typical rate 1/2 linear non-systematic convolutional encoder.

The sequences have been terminated when the output streams contain only zeros. These generator sequences, thus, can be read from the encoder diagram through an examination of the tap positions.

The overall generator comprising of both the feed-forward generators is usually represented in octal. The octal generator for the encoder in Fig. 4.1 is (7,5). The generator, when represented in binary, forms a matrix having as many rows as there are outputs from the encoder and K columns. K is the constraint length of the encoder and is defined as the maximum number of bits in a single output stream that can be affected by any input bit. It is, thus, the maximum number of taps from the shift registers in the encoder. Practically, the constraint length is the length of the longest input shift register plus one,

$$K = 1 + \max_i m_i \,, \tag{4.1.2}$$

where m_i is the number of memory elements in the ith shift register.

The output of the single-input encoder in terms of the generator sequences is given by [Wic 95]

$$y_i^{(j)} = \sum_{l=0}^{m} x_{i-l} g_l^{(j)} \,. \tag{4.1.3}$$

Thus, the output is a discrete convolution of the input and the impulse response g, hence, the name "convolutional codes". This can be expressed as

$$y^{(j)} = x * g^{(j)}. \tag{4.1.4}$$

Equation (4.1.3) can be generalized for a k-input encoder and is given as

$$y_i^{(j)} = \sum_{t=0}^{k-1} \left(\sum_{l=0}^{m} x_{i-l}^{(t)} g_{t,l}^{(j)} \right). \tag{4.1.5}$$

A convolutional code is systematic when the input bits appear unaltered in the output sequence. This is not the case in Fig. 4.1 and the encoder in the figure is thus a non-systematic convolutional encoder. It is seen that the best convolutional codes are non-systematic ones. However, implementation of the decoding algorithm is slightly easier with systematic codes.

4.1.1 Analysis of Convolutional Codes

Convolutional encoders can be viewed as finite impulse response (FIR) filters or as finite-state automata. In both cases, the encoder consists of a fixed number of memory elements. Therefore, the encoder can assume any of the fixed number of total states at a particular time. The state transitions are specific in the sense that an input to a particular state leads to a fixed next state. Thus, the convolutional encoder has a time-invariant Markov chain structure. The finite state machine shown in Fig. 4.1 has an initial state 0 and is brought back to its original starting state by padding the input sequence with m zero bits called tail bits. The state transitions are depicted by the state diagram in Fig. 4.2.

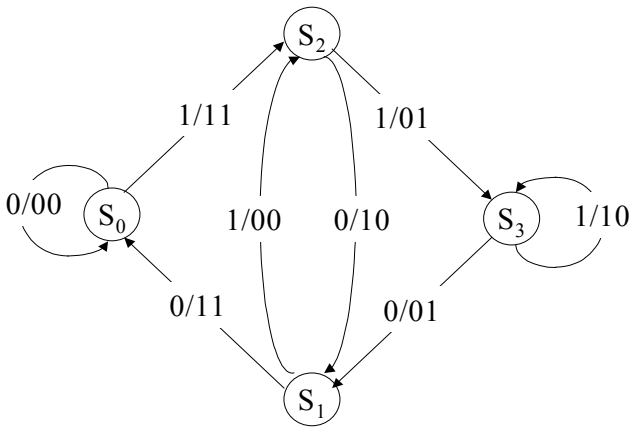


Fig. 4.2 State diagram for encoder in Fig. 4.1.

The state diagram in Fig. 4.2 is for the encoder shown in Fig. 4.1. The encoder has two memory elements and hence, it can be in any of the four different states at a particular time. The diagram also shows the transitions from each state to the others. The transitions are marked as X/YY where X is the input bit that causes the transition and YY are the two bits produced at the output as a result of encoding the input bit.

The performance of a convolutional code is usually measured in terms of the code's minimum free distance. The minimum free distance, d_{free}, of a convolutional code is the minimum Hamming distance between all pairs of complete convolutional code words. The minimum free distance for the encoder in Fig. 4.1 can easily be evaluated from the state diagram in Fig. 4.2. Starting from state 0, d_{free} is the minimum total Hamming weight over all the paths traced by the encoder until it comes back to state 0. The Hamming weight of a particular path is the number of ones in the output associated with the path. Therefore, the path traced by the encoder leaving state 0 and coming back to state 0 such that the Hamming weight is minimum is $S_0 \rightarrow S_2 \rightarrow S_1 \rightarrow S_0$ and the d_{free}=5.

State diagrams are also useful in finding if the code is catastrophic. A catastrophic code is one whose corresponding state diagram contains a circuit in which a non-zero input sequence produces an all-zero output sequence [Wic 95].

The analysis of convolutional codes can also be done by an extension of the state diagram that explicitly shows the passage of time, called a trellis diagram. Such a diagram for the encoder in Fig. 4.1 is shown in Fig. 4.3.

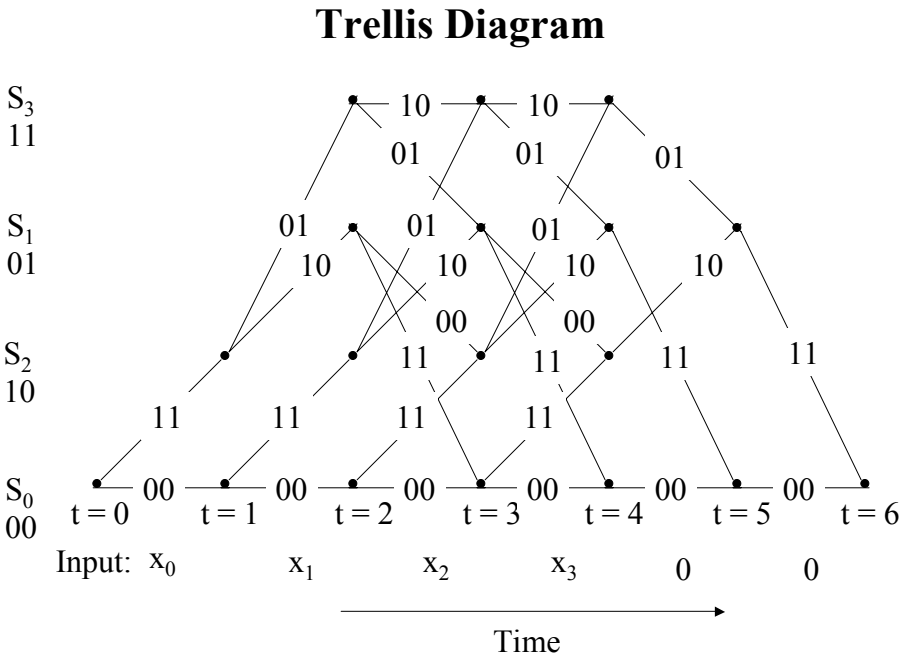


Fig. 4.3 Trellis diagram for encoder in Fig. 4.1.

The trellis diagram starts at time 0 and state 0. It has a total of (L+m) stages where L is the length of the input sequence and m is the number of tail bits added to bring the encoder back to state 0. In Fig. 4.3, L = 4 is assumed. Since m = 2 in Fig. 4.1, there are 6 stages and 2^m = 4 nodes at each stage in the trellis diagram. It is noted that if there are k inputs to the encoder, 2^k branches leave each node in the trellis diagram. There are also 2^k branches entering each node from time t=m until t=L. The diagram shows the output bits along each state transition. Starting at state 0 at time t=0, the trellis shows that there are only two states that the encoder can possibly be in at time t=1: state S_0 if input $x_0 = 0$ at t=0 producing output bits 00 or in state S_2 if input $x_0 = 1$ producing output bits 11. Also, it shows that the encoder can first reach states S_1 and S_3 only at time t=2. Similarly, other trellis sections for increasing times are shown. From time t=4 onwards, the inputs are tail bits. Hence, from each state, the trellis follows only those paths that are associated with input 0. Finally, all paths end at state 0.

There are 2^{kL} distinct paths through the trellis, each corresponding to a convolutional code word of length $n(L+m)$ where n is the number of outputs from the encoder. Thus, there are 12 output bits corresponding to 4 input bits as shown in Fig. 4.3. A decoding algorithm for convolutional codes called the Viterbi algorithm takes advantage of this trellis structure to reduce the complexity of the decoder.

4.2 Decoding of convolutional codes: Viterbi Algorithm

The encoder transforms the information sequence x into a convolutional code word y, which is modulated and transmitted across the channel. The corrupted version r of the transmitted sequence is received at the receiver. The convolutional decoder at the receiver has to produce an estimate y' of the transmitted code word by evaluating the noisy received vector r.

The Viterbi algorithm (VA) is a maximum-likelihood (ML) decoding algorithm, which produces an estimate y' that maximizes the probability $p(r|y')$. Let the input block to a rate 1/n convolutional encoder be of length L. The corresponding output code word is of length $n(L+m)$ where m is the number of memory elements in the encoder. The decoder receives the noisy code word of length $n(L+m)$ and has to produce an estimate y' of each of these code bits. The channel is assumed to be memoryless. This means that the noise affecting one particular bit is independent of the noise affecting other received bits. This assumption simplifies the expression of the probability $p(r|y')$ from a joint probability of all events to the product of the probabilities of individual events since they are independent [Wic 95].

$$p(r|y') = \prod_{i=0}^{L+m-1} \left[p(r_i^{(0)}|y_i^{'(0)}) p(r_i^{(1)}|y_i^{'(1)}) ..... p(r_i^{(n-1)}|y_i^{'(n-1)}) \right]$$

$$= \prod_{i=0}^{L+m-1} \left(\prod_{j=0}^{n-1} p(r_i^{(j)} | y_i^{'(j)}) \right). \tag{4.2.1}$$

The inner product in (4.2.1) is for the n output blocks while the outer product corresponds to the individual bits within a block. Equation (4.2.1) is called the likelihood function of y'. It is seen that the likelihood function comprises of a series of multiplications whose evaluation could be computationally complex. Logarithms are monotonically increasing functions and hence the estimate that maximizes $p(r|y')$ also maximizes $\log\ p(r|y')$. Moreover, by expressing (4.2.1) as a logarithm, the multiplications are converted into less complex additions, thereby simplifying the computation. Taking the logarithm of each side of (4.2.1) gives rise to the following log-likelihood function

$$\log p(r|y') = \prod_{i=0}^{L+m-1} \left(\prod_{j=0}^{n-1} \log p(r_i^{(j)} | y_i^{'(j)}) \right). \tag{4.2.2}$$

The summands in (4.2.2) are further converted to a form that can be easily manipulated called the bit metrics, which are given by [Wic 95]

$$M(r_i^{(j)} | y_i^{'(j)}) = a\left[\log p(r_i^{(j)} | y_i^{'(j)}) + b\right], \tag{4.2.3}$$

where a and b are chosen such that the bit metrics are small negative integers that can be manipulated by digital logic circuits.

Basically, bit metrics are Hamming distances between each output bit on a particular trellis branch and the corresponding received code bit for each trellis stage. The sum of these bit metrics on each branch are called branch metrics corresponding to a trellis stage. The path metric for a code word y' is then computed based on (4.2.3) as follows

$$M(r|y') = \sum_{i=0}^{L+m-1} \left(\sum_{j=0}^{n-1} M(r_i^{(j)} | y_i^{'(j)}) \right). \tag{4.2.4}$$

If a in (4.2.3) is real and positive, while b is simply real, then the code word y' that maximizes $p(r|y')$ also maximizes $M(r|y')$. The k^{th} partial path metric is computed by summing the branch metrics of the first k branches that the path traverses.

The VA commences with the trellis diagram shown in Fig. 4.3 and starts assigning values to each node in the trellis. These values are the partial path metrics of all the paths traversed by the trellis until the particular node is reached. If more than a single path enters a node in the trellis, then the value at that node is the smallest partial path metric among the metrics of all the entering paths. This path is called the survivor while the

other competing paths are called non-survivors. If the metrics of all the entering paths are equal, then any of them can be selected as the survivor (usually, the selection is random).

Similarly, each of the nodes in the trellis is labeled with its path metrics and the survivors at each node in a trellis stage are marked. This process is called trace-forward. The trace-back process commences once the last trellis stage (t=L+m) is reached. Starting from state 0 at time t=L+m, trace-back is done by tracing the nodes backward following only the survivor paths. Since each node has only one survivor, the trace-back process will yield a unique ML path. The final estimate of the data can then be taken from this path.

4.2.1 Implementation of Viterbi Algorithm

The Viterbi algorithm can be described by a repetitive add-compare-select (ACS) operation. The steps in the implementation of the Viterbi algorithm are summarized as follows. $P_{i,t}$ denotes the partial path metric at state i of the trellis at a particular time t.

1. Place n received code word bits below each trellis stage.
2. Initial Conditions: At time t=0, set $P_{0,t} = 0$ and $P_{i,t} = \infty$ (a very large number for practical implementations) for all $i \neq 0$.
3. Move to the next trellis stage (increment time t).
4. For each state i, compute individual branch metrics entering the node corresponding to the state. The branch metric is the Hamming distance between the output bits on each branch of the trellis and the n received code word bits.
5. ACS Operation:
    a. Add the branch metric to the partial path metric (computed during the previous trellis stage) of the state the branch is leaving from.
    b. Since every node will have two branches originating from different previous states, compare the partial path metrics of the two branches entering each stage.
    c. Select the survivor branch, which is the one with the least partial path metric.
6. If t < L+m, repeat step 5 for each state and return to step 3 after all states are exhausted.
7. If t = L+m, start from state 0 and trace-back through the trellis's survivor paths entering each state. The trace-back operation gives rise to a unique ML path and the inputs at each of these trellis sections of the unique path corresponds to a ML input word.

Fig. 4.4 shows the trellis diagram of the encoder in Fig. 4.1 after Viterbi decoding is done. The generator is (7,5) in octal and the rate is 1/2. The corresponding sequences are as follows:

Input sequence (L=3 + 2 tail bits) = [1 1 0 0 0]

Encoded sequence (2*(L+m) = 10 bits) = [1 1 0 1 0 1 1 1 0 0]

The encoded sequence above is BPSK modulated and transmitted. The received sequence is a noisy version of the transmitted sequence. Hard decision is performed on the received bits using the following scheme: if received value > 0, r = 1 else r = 0.

Suppose that r is given by the sequence [1 1 1 1 0 1 1 1 1 0]. It can be seen that there is 20% probability of error in the sense that two out of the 10 bits are received in error. However, the Viterbi decoder is still able to correct the errors and output the estimated input sequence of [1 1 0].

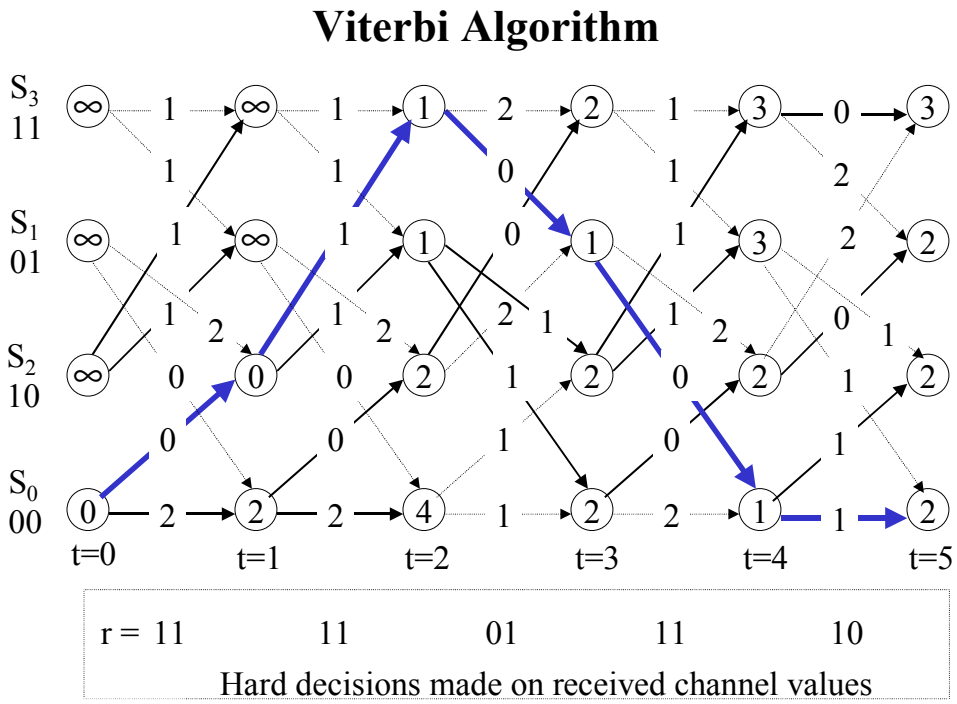


Fig. 4.4 Viterbi decoding for encoder in Fig. 4.1.

The survivor path at every state in the trellis is shown with a dark arrow. The branch metrics are labeled at each branch and the path metrics are listed at each node. The ML path is the path marked by the thickest line in the diagram.

4.3 Performance of Convolutional Codes

4.3.1 Hard decision decoding

As seen in the Viterbi decoding example of the previous section, the received code bits are evaluated and a decision is made on each bit to be either 0 or 1. If the received value of the bit is above a particular threshold, then the bit is a 1 and if the received value is below the threshold, it is 0. This threshold logic can be viewed as a 1-bit quantizer. The hard-decided bits are then input to the Viterbi decoder for estimating the original input sequence. This is known as hard-decision decoding.

Instead of code word error probability, the performance of convolutional codes is measured by the *first error event probability* P_e. P_e is the probability that an error event (when the decoder chooses a path from the trellis diverging from the correct path) begins during the current time interval. Another probability that is an important performance measure for convolutional codes is called the *pairwise error event probability* $P_2(d)$. $P_2(d)$ is the probability that the decoder will choose a path that is Hamming weight d away from the correct path and is given as

$$P_2(d) = \sum_{j=\frac{d}{2}+1}^{d} \binom{d}{j} p^j (1-p)^{d-j} + \frac{1}{2}\binom{d}{d/2} p^{\frac{d}{2}} (1-p)^{\frac{d}{2}} \quad \text{for even d}$$

$$= \sum_{j=\frac{d}{2}+1}^{d} \binom{d}{j} p^j (1-p)^{d-j} \quad \text{for odd d,} \tag{4.3.1}$$

where p is the probability of error in a binary symmetric channel.

The first error event probability P_e is related to $P_2(d)$ as follows

$$P_e \leq \sum_{d=d_{free}}^{\infty} a_d P_2(d), \tag{4.3.2}$$

where a_d is the number of paths through the trellis with output weight d and d_{free} is the minimum distance of the code as described in section 4.1. The first term of (4.3.2) is called the free distance asymptote and serves as a good approximation to the first error event probability at high SNR, i.e. $P_e \approx a_{d_{free}} P_2(d_{free})$.

The bit error probability of convolutional codes with hard-decisions over a binary symmetric channel is given as

$$P_b \leq \sum_{d=d_{free}}^{\infty} c_d P_2(d), \tag{4.3.3}$$

where c_d is the information weight of all the paths through the trellis with output weight d.

For BPSK modulation over an AWGN channel, the probability of code bit error p is given as

$$p = Q\left(\sqrt{\frac{2rE_b}{N_o}}\right). \tag{4.3.4}$$

For example, p = $Q\left(\sqrt{\frac{E_b}{N_o}}\right)$ for the encoder shown in Fig. 4.1 with rate r= 1/2.

4.3.2 Soft decision decoding

In hard-decision decoding, the received channel bits are hard-limited to be a 1 or 0. This 1-bit quantization of the received channel values results in the loss of important information. Instead, the received channel values can be fed directly to the Viterbi decoder. This can be viewed as infinite bit quantization (practically, a very large number of bits of quantization). In hard-decision decoding, the Viterbi decoder minimizes the Hamming distance between the output bits of a particular branch in the trellis and the n-bit received code word block while computing the branch metric. Rather than the Hamming distance, the soft-decision decoding scheme minimizes the squared Euclidian distance between the two code words. Thus, this type of decoding is dependent on the type of modulation and channel. Soft-decision decoding is similar to hard-decision decoding except in the computation of the branch metric. One of the biggest benefits of convolutional codes is that soft-decision decoding can be implemented very easily and typically results in a coding gain of about 2 dB.

Squared Euclidian distance is given by

$$d_e(c,r) = \sum (r_i - c_i)^2 . \tag{4.3.5}$$

For BPSK over an AWGN channel, the pairwise error event probability is given as

$$P_2(d) = Q\left(\frac{d_e}{\sqrt{2N_o}}\right) = Q\left(\sqrt{\frac{2E_b rd}{N_o}}\right), \tag{4.3.6}$$

where p = $Q\left(\sqrt{\frac{E_b d}{N_o}}\right)$ for the encoder in Fig. 4.1 with rate r=1/2.

Thus the only difference in the analysis is that soft-decision decoding uses pairwise error probability given in (4.3.6) in place of the pairwise error probability given by (4.3.1). The rest of the analysis for performance of convolutional code with soft-decision decoding is the same as that for hard-decision decoding.

4.3.3 Soft-decision vs Hard-decision: Performance Comparison

The performance of hard-decision and soft-decision decoding schemes is simulated for the code parameters mentioned below and the resulting plot of bit error rate vs. signal to noise ratio in dB is shown in Fig. 4.5.

- Constraint Length: K = 4.
- Generator Polynomial: (15, 17, 13) in octal.
- Code rate = 1/3.
- Input frame size L = 253.
- Modulation: BPSK.
- Channel: AWGN.

It is seen from Fig. 4.5 that with all the code parameters remaining the same, the soft-decision decoding scheme has about 2.1 dB of coding gain over the hard-decision decoding scheme at a bit error rate of 10^{-5}. These schemes are simulated over an AWGN channel. The performance betterment by using the soft-decision decoding scheme is even more prominent in Rayleigh or Rician fading channels.

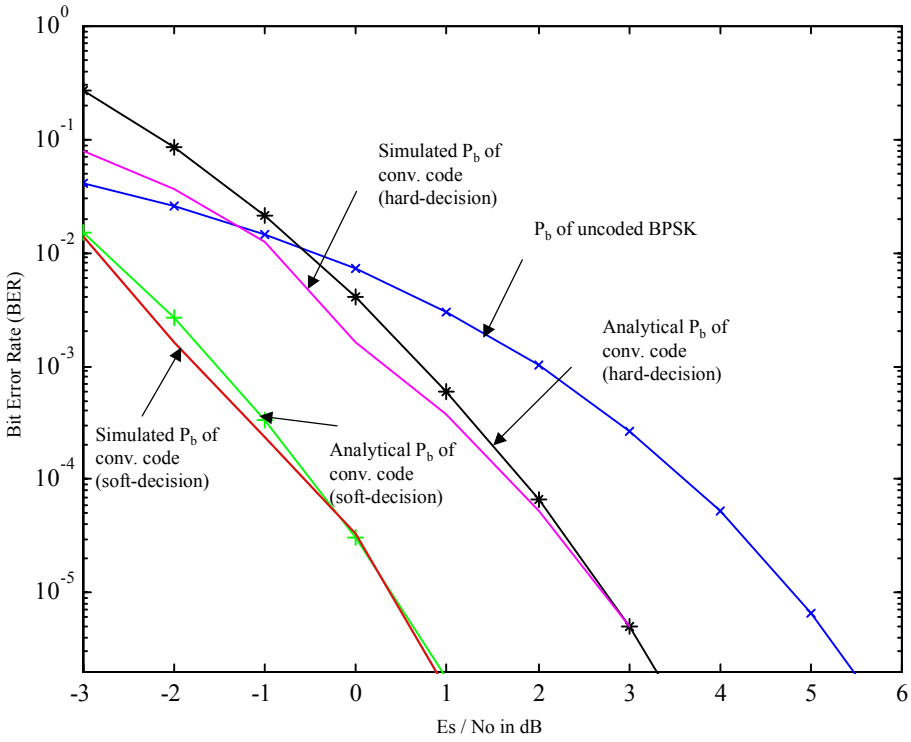


Fig. 4.5 Performance of Hard-decision vs. Soft-decision decoding of a rate 1/3, K = 4 convolutional code.

4.4 Puncturing of Convolutional Codes

Many applications can benefit from the use of high rate convolutional codes. These applications also include those that require only moderate error correction but demand high throughput. The highest rate that can be achieved by a convolutional code of rate k/n with one input stream is 1/2 (two output bits for every input bit). There are two means to increase the code rate. One is to increase the number of inputs such that an encoder with rate k/n can achieve rates such as 2/3, and 3/4. However, an increase in the number of inputs corresponds to an exponential increase in the complexity of the decoding algorithm. In general, the number of branches entering each node of a trellis is 2^k. Hence, instead of two branches entering each node as seen in the k=1 case, the number of branches / trellis section will exponentially increase with the number of inputs. Thus, the ACS operation in the Viterbi algorithm can become highly complex.

Puncturing the code is another means of implementing high rate codes. This process increases the transmission rate of the code by periodically deleting bits from the output of the encoder according to a specific format called the puncturing pattern. For example, let the puncturing period be 4. In a rate 1/2 convolutionally encoded system, every input bit

gives rise to two output bits. Now, from a frame of 4 encoded output bits, one bit is marked and deleted from the sequence as follows:

Period = 4, Input Sequence = $[x_0, x_1]$,
Encoded output sequence of a rate ½ encoder = $[y_{0,} y_1, y_2, y_3]$
Let the puncturing pattern be [1 1 0 1]. This means that the third bit in the output sequence is deleted. Hence, the transmitted sequence is $[y_{0,} y_1, y_3]$.

Now, for every 2 input bits, there are only three encoded output bits. Thus, the code rate 2/3 is achieved. Similarly, rate 3/4 can be achieved when 2 out of six bits are deleted in the output sequence. Therefore, by increasing the puncturing period, a large number of distinct high rate codes can be generated from a mother 1/n code. These code rates can vary from the mother code rate 1/n (lowest when no bits are punctured) to the highest punctured rate used.

A family of codes can be generated from a mother code of rate 1/n with rates given by [Hag 88]

$$R = \frac{P}{P+l} \quad l = 1,....(n-1)P, \tag{4.4.1}$$

where R is the rate of each code, P is the puncturing period, and n is the number of outputs from the encoder.

The punctured bits are first marked and then deleted. A puncturing pattern is used to select the bits to be punctured and the same pattern is made available to the receiver. At the receiver, zeros are inserted into the received sequence at the punctured positions or erasures.

Let the noisy received sequence from the channel be R = $[\bar{y}_0, \bar{y}_1, \bar{y}_3]$. Therefore, the received sequence after zero-insertion is $R = [\bar{y}_0, \bar{y}_1, 0, \bar{y}_3]$. This sequence is then fed into the Viterbi decoder. Puncturing the code does not increase the complexity of the Viterbi algorithm and hence can be used for decoding of punctured convolutional codes. Furthermore, puncturing the code helps in increasing the transmission rate when the source demands so or when the channel is sensed to be clean. Thus, the process can make the system adapt to the source and channel conditions at a particular time.

4.4.1 Rate Compatibility Criterion

A rate compatibility criterion was applied to the set of punctured convolutional codes by Hagenauer [Hag 88]. This implies that all the code bits of a high rate punctured code are used by the successive lower rate code words or in other words, the high rate code words are embedded in the low rate code words. Thus, while decreasing the code rate and increasing the error protection to the code, the transmitter only needs to transmit the additional bits that were punctured during the previous transmission of the high rate code word. This is the concept of providing incremental redundancy. The rate compatibility criterion can be expressed in equations as follows [Hag 88]

$$\begin{aligned} &\text{if} \quad a_{ij}(l_0) = 1 \quad \text{then} \quad a_{ij}(l) = 1 \quad \text{for all} \quad l \geq l_0 \geq 1 \\ &\text{if} \quad a_{ij}(l_0) = 0 \quad \text{then} \quad a_{ij}(l) = 0 \quad \text{for all} \quad l \leq l_0 \leq (n-1)P - 1, \end{aligned} \tag{4.4.2}$$

where $a(l)$ is the puncturing matrix at a particular code rate and the other parameters are as defined in (4.4.1).

This reduction in the amount of redundant information causes a significant increase in throughput. Good rate compatible punctured convolutional (RCPC) codes are found by a computer search method. Especially for $d \geq d_{free}$, most of the RCPC codes are good since they are derived from a mother code of rate 1/n. Moreover, RCPC codes provide an efficient means of implementing a variable rate error control system using a single encoder / decoder pair.

4.5 A Hybrid RCPC-ARQ System

RCPC codes adapt very closely to the channel conditions by providing variable rate error control. Since the rate is directly proportional to throughput, an increase in the rate of transmission owing to puncturing the code increases the throughput if the code words are all received correctly at this high rate. Reliability of the system (delivering packets with no errors or with fewer errors than the correcting capability of the code) is increased by providing the RCPC system with an ARQ protocol. A type-II hybrid FEC-ARQ scheme, as described in the chapter 3, is used with this RCPC system. The overall system achieves good throughput performance at reasonably high channel error rates. The parameters used in the simulation of this hybrid RCPC-ARQ model are as follows:

- Generator Polynomial of RCPC encoder = (15, 17, 13) in octal.
- Constraint Length = 4.
- Mother code rate r = 1/n = 1/3.
- Input frame size L = 253.
- Puncturing Period P = 8.

- Family of punctured rates: $R = \frac{P}{P+l}$, $l = [2,4,6,8,10,12,14,16]$.
- The puncturing rates range from 4/5 (highest) to 1/3 (lowest).
- Channel Type: AWGN.
- RCPC Decoder: Soft-decision Viterbi decoder.

Puncturing patterns used are as given in [Hag 88] and shown in Table 4.1.

	Rates							
	4/5	2/3	4/7	1/2	4/9	2/5	4/11	1/3
Matrix	356 231 0	356 335 0	377 335 0	377 377 0	377 377 210	377 377 252	377 377 356	377 377 377

Table 4.1 Octal puncturing matrix for each code rate.

4.6 Throughput Efficiency

Simulations of a type-II hybrid ARQ scheme with RCPC codes are run for different individual rates and puncturing schemes as shown in Table 4.1. Since the system is rate compatible and involves transmission of information at various rates from highest to lowest, the throughput efficiency of such a system is defined to be the expectation of the code rate as a function of frame error rate (FER) at each particular value of E_s/N_o.

Therefore, the throughput efficiency of the type-II hybrid RCPC-ARQ system is given by

$$T_{av}\left(\frac{\varepsilon_s}{N_o}\right) = E\left\{r\left(FER\left(\frac{\varepsilon_s}{N_o}\right)\right)\right\} \tag{4.6.1}$$

$$= \sum_{r_i} r_i P[r = r_i],$$

where r_i is a particular code rate and $P[r = r_i]$ is the probability mass function of the rate.

The probability that the system is transmitting at a particular rate is the product of the probability of there being frame errors in transmissions at higher rates and the probability of success in the transmission at the current rate. Therefore the probability mass function in (4.6.1) is given by

$$P[r = r_i] = \left(1 - FER\left(r_i, \frac{\varepsilon_s}{N_o}\right)\right)\left(\prod_{r_j > r_i} FER\left(r_j, \frac{\varepsilon_s}{N_o}\right)\right). \quad (4.6.2)$$

The throughput efficiency of the system over a channel is measured by first generating frame error rate (FER) curves through simulation for each of the possible code rates, and then using (4.6.2) and (4.6.1) to arrive at a numerical value.

4.7 Simulation Results

Simulations of the RCPC–ARQ system described by the parameters mentioned in the previous section are run for each of the different punctured code rates. The following results are obtained.

- Bit Error Probability (BER) vs. Signal to Noise Ratio (Es / No in dB).
- Throughput Efficiency vs. Es / No in dB.

Fig. 4.6 and Fig. 4.7 show the BER vs. Es / No in dB for different puncturing rates and patterns used. Since variable rates are used, the performance plots are parameterized by energy per symbol rather than energy per bit. Rates ranging from 4/5 to 1/2 are used in simulating the plot in Fig. 4.6 while rates ranging from 4/9 to 1/3 are shown in Fig. 4.7.

The BER performance of RCPC codes shows that the codes perform worse as the code rate increases and the highest code rate performs the worst. In Fig. 4.6, rate 4/5 is the highest code rate and it requires about 5.75 dB to reach a bit error probability of 10^{-5}. There is almost a 1.5 dB performance difference between the two highest rates (rate 4/5 and rate 2/3) and about a 2.5 dB coding gain from rate 4/5 to 4/7 at a bit error rate of 10^{-3}. Performance gets better as the code rate decreases and the lowest code rate in Fig. 4.6 (rate 1/2) requires only 0.2 dB to reach a bit error rate of 10^{-3}.

Fig. 4.7 exhibits a similar nature of BER performance only at lower signal to noise ratios since the rates used are lower than 1/2. However, the improvement in the BER performance at each code rate reduces as the code rate decreases. It is seen that the coding gain at a bit error rate of 10^{-3} between successive code rates is within 0.5 dB.

Throughput is a key measure to evaluate the performance of an ARQ system. The throughput efficiency vs. Es / No in dB plot is shown in Fig. 4.8.

Fig. 4.8 is plotted by first generating frame error rate (FER) curves for each individual punctured rate and then using (4.6.1) and (4.6.2) to determine the throughput efficiency.

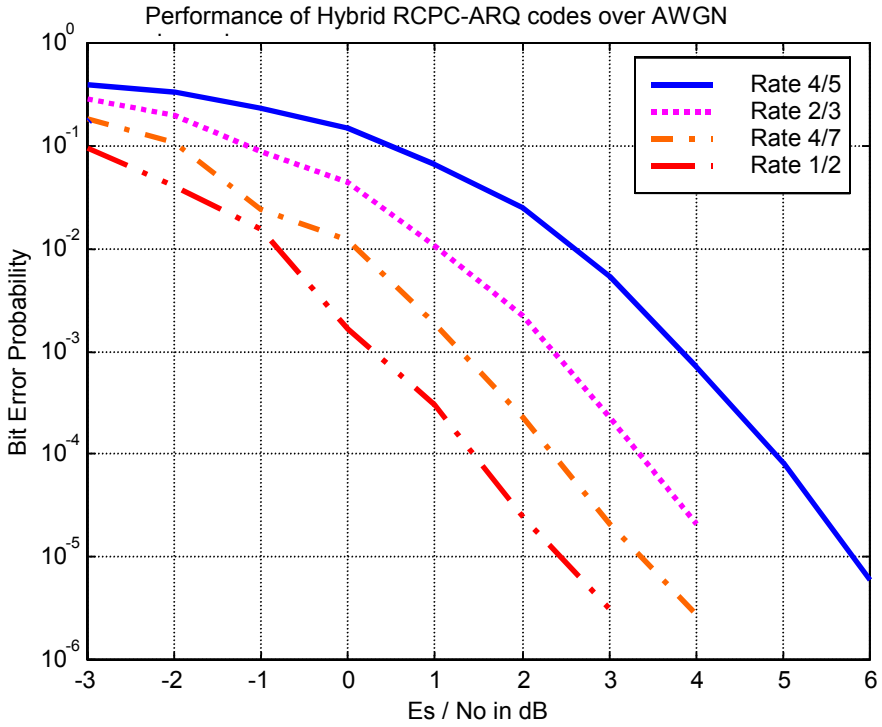


Fig. 4.6 BER Performance of a hybrid RCPC-ARQ System (Rates 4/5 to 1/2).

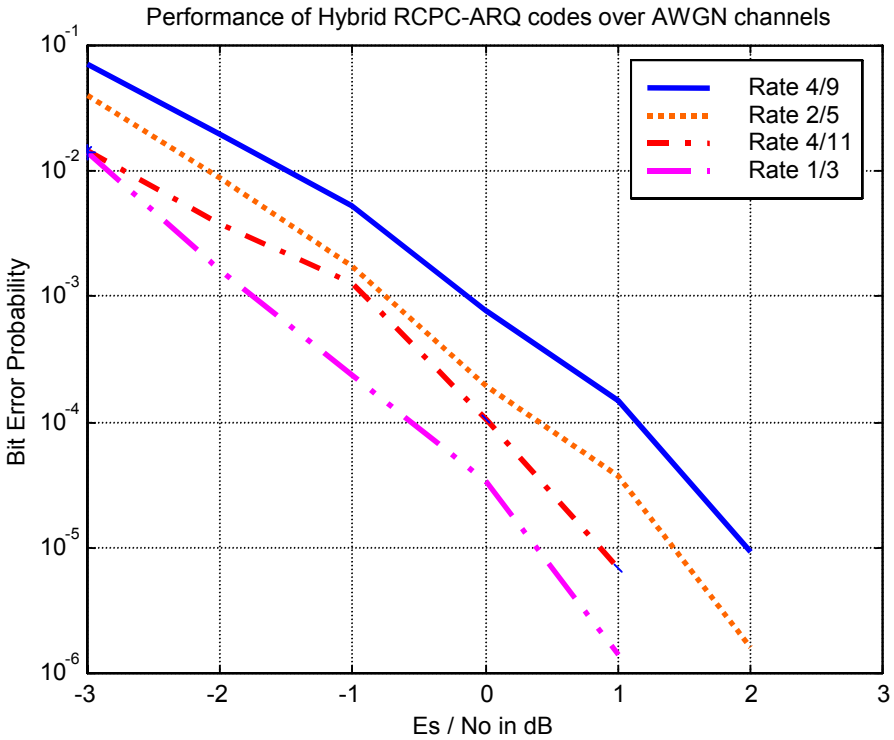


Fig. 4.7 BER Performance of a hybrid RCPC-ARQ System (Rates 4/9 to 1/3).

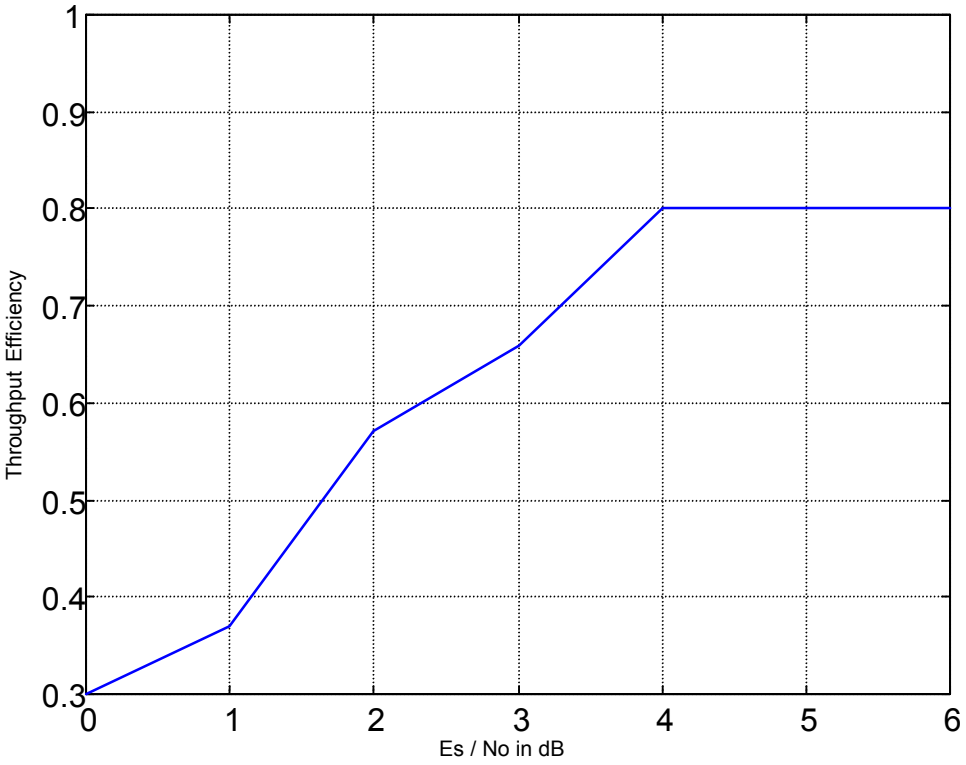


Fig. 4.8 Throughput Efficiency of RCPC codes vs. Es / No in dB.

As Fig. 4.8 shows, the throughput efficiency of the code is low at low signal to noise ratios and is equal to the lowest code rate (1/3) at Es / No = 0 dB. The throughput increases with increasing signal to noise ratio and it can be seen that at higher signal to noise ratios, the throughput performance of the hybrid RCPC-ARQ system is close to 4/5 (the highest code rate that is used with the code).

4.8 Chapter Summary

An NSC forward error correcting code is introduced in this chapter as a component of a hybrid FEC-ARQ system. The working of the NSC encoder and the soft and hard–decision Viterbi decoder (used to decode NSC codes) are detailed and implemented. It is seen that there is a performance improvement of about 2 dB while using soft-decision Viterbi decoding over hard-decision decoding. Puncturing increases the transmission rate of the code while the rate compatibility criterion reduces the amount of overhead information. Therefore, puncturing and rate compatibility provide an efficient means of implementing a variable rate error control system. Further, a type II hybrid RCPC-ARQ system is developed and simulated over AWGN channels. The BER and throughput

performance of such a system are plotted. The hybrid RCPC-ARQ system is seen to have a throughput performance far better than that of a conventional convolutional code system. On the whole, this chapter gives a comprehensive understanding of NSC codes as a stepping-stone to the concept of turbo codes that follows in the next chapter.

Chapter 5

Turbo Codes

Turbo codes, introduced by Berrou *et al* [Ber 93] in 1993, were a major breakthrough towards realizing Shannon's channel capacity limit. The performance of these codes achieves low bit error rates (10^{-5} – 10^{-6}) at very low signal to noise ratio, about 0.5 dB away from the theoretical capacity limit.

Turbo codes are a parallel concatenation of two or more convolutional codes, each separated by a non-uniform interleaver as shown in Fig. 5.1. The constituent encoders used in the turbo encoder are typically recursive systematic convolutional (RSC) encoders. However, any other type of convolutional or block code could also be used.

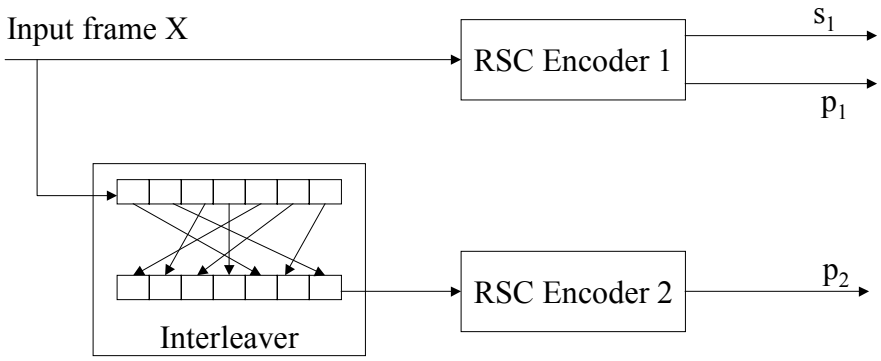


Fig. 5.1 A Fundamental Turbo Encoder.

The same input data stream X feeds both RSC encoders but the second encoder receives this input frame in a permuted order owing to the presence of the interleaver. Both the RSC encoders are rate 1/2 encoders since they have one input and two outputs (one systematic and the other parity). Since both encoders provide the same systematic information, the systematic output of only the top RSC encoder (s_1) is transmitted along with its parity (p_1) and the parity output (p_2) of the bottom RSC encoder.

The striking features of conventional turbo codes are parallel concatenation of convolutional codes, recursive systematic convolutional component codes, pseudo-random interleaving, and an iterative decoding algorithm. All these features will be briefly investigated in this chapter of the thesis. Finally, a rate compatible punctured

turbo-coded type II hybrid ARQ model is developed and performance results of such a system are presented.

The first few sections of this chapter explain the working and performance of RSC codes along with the advantages of using these codes as constituent blocks of the turbo code. Concatenation of such RSC codes is motivated with an emphasis placed on the functioning of the turbo encoder, issues such as interleaving and trellis termination, and the analytical performance of the concatenated code. The next few sections detail the iterative soft-input soft-output (SISO) maximum a posteriori (MAP) decoding algorithm. However, a detailed derivation of the MAP algorithm is not provided in this chapter but can be found in Appendix A of this thesis. This chapter then discusses practical implementation issues of the MAP algorithm. Finally, these blocks are integrated into an end-to-end communication system and a hybrid RCPT (rate compatible punctured turbo) -ARQ model is developed and simulated.

5.1 Recursive Systematic Convolutional Codes

As mentioned in the previous chapter, the bit error rate (BER) of non-recursive non-systematic convolutional (NSC) codes is lower than that of the non-recursive systematic convolutional codes (henceforth called NRSC codes) with the same constraint length at large SNR [Ber 93]. This is because the free distance of the NSC codes is larger than that of the NRSC codes. However, at low SNR, the NRSC codes perform better than the NSC codes.

A recursive systematic convolutional (RSC) code can be constructed from a NSC code by feeding back one of its outputs and making the encoder systematic (setting one of the output streams equal to the input stream). RSC codes combine the properties of NSC and NRSC codes. They perform better than their equivalent NSC codes at all SNRs for high code rates (rates > 2/3). Moreover, even for low code rates, the RSC codes have lower bit error rates than the NSC codes at low SNRs but the NSC codes perform better at high SNRs.

Fig. 5.2 is the schematic of the NSC code introduced in chapter 4 and Fig. 5.3 is the related RSC code. As shown in Fig. 5.3, one of the outputs from Fig. 5.2 is fed back to the input and the encoder is made systematic by setting output stream $Y^{(1)}$ to be equal to the input stream X.

Just as NSC codes can be viewed as finite impulse response (FIR) filters, RSC codes can be viewed as infinite impulse response (IIR) filters. Owing to the feedback from the memory locations, the impulse input (i.e. 1 followed by a stream of zeros) does not produce a finite length output. Only a particular combination of the encoder being in state zero and a stream of input zeros can ensure that the output is always zero beyond a finite length.

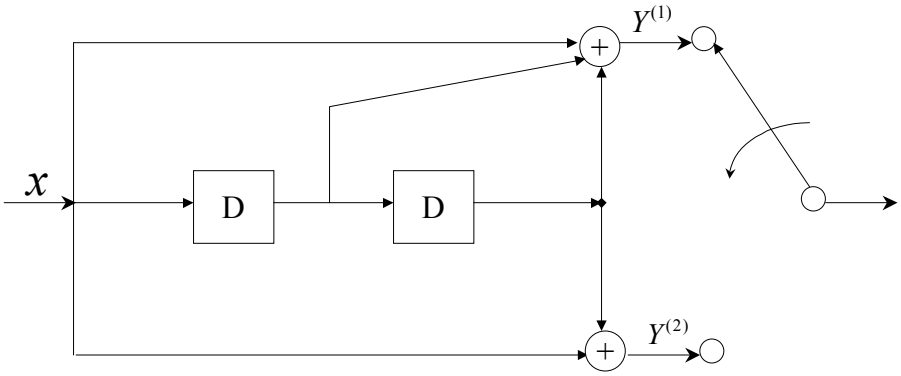


Fig. 5.2 A rate 1/2 NSC encoder shown in Chapter 4 with g = (7,5).

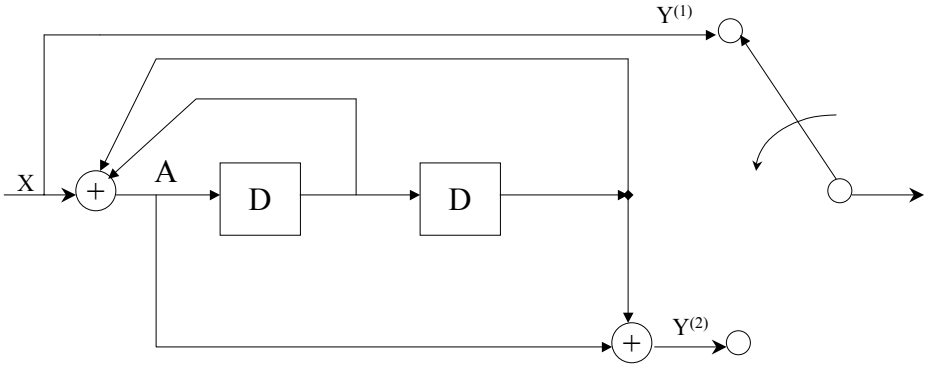


Fig. 5.3 A RSC encoder equivalent to the NSC encoder in Fig. 5.1 with g = (1 7/5).

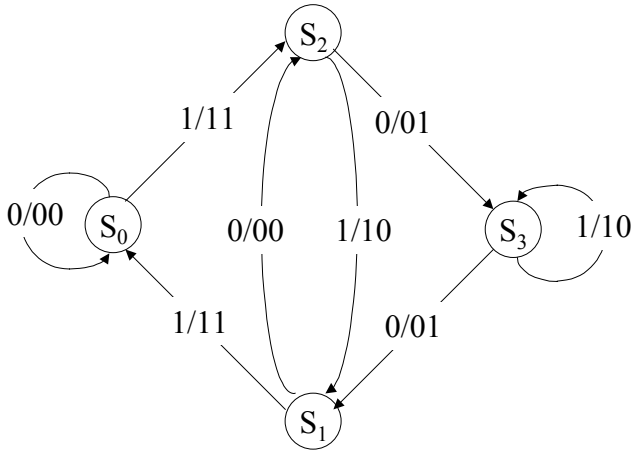


Fig. 5.4 State diagram for RSC encoder in Fig. 5.2.

In the above example, even with an input of all zeros following a 1, the encoder will loop around states two, three and one, thereby producing an infinite length output. This can be easily comprehended with a state diagram representation of the encoder in Fig. 5.3 shown in Fig. 5.4.

As seen in Fig. 5.3, the shift register input is no longer the input bit (as in NSC codes) but is the modulo-2 sum of the input bit with some or all of the memory contents; a pattern that is determined by the generator. The generator g for NSC codes is of the form $g = (g_1\ g_2)$ where g_1 and g_2 are both feedforward generators determining the tap positions for each output sequence. Since the RSC encoder is systematic, the generator is given by $g = \left(1 \quad \frac{g_2}{g_1}\right)$ where g_2 is the feedforward generator while g_1 is the feedback generator. However, the usual notation that is used and which will be followed in the remainder of this thesis will be $g = (g_1\ g_2)$ where the individual generators have the same definition as mentioned above in the RSC case.

The input to the shift register A can be expressed as

$$A_k = X_k + \sum_{i=1}^{m} g_{1i} A_{k-i}\ , \tag{5.1.1}$$

where k is the k^{th} input bit, g_{1i} is the value of the i^{th} tap position in the feedback generator, and m is the total number of memory elements in the shift register.

The two outputs of the encoder are the systematic output stream $Y^{(1)}$ and the parity output stream $Y^{(2)}$. The expressions for the output streams of the RSC encoder are as follows

$$\begin{aligned} Y_k^{(1)} &= X_k \\ Y_k^{(2)} &= \sum_{i=1}^{m} g_{2i} A_{k-i}\ , \end{aligned} \tag{5.1.2}$$

where g_2 is the feedforward generator.

5.1.1 Performance of RSC codes

The expressions for performance of RSC codes are the same as those of the NSC codes stated in Chapter 4. The first error event probability and the bit error rate probability performance measures are as given below

$$P_e \le \sum_{d=d_{free}}^{\infty} a_d P_2(d)$$

$$P_b \le \sum_{d=d_{free}}^{\infty} c_d P_2(d), \tag{5.1.3}$$

where a_d is the number of paths through the trellis with output weight d, c_d is the information weight of all the paths through the trellis with output weight d, and $P_2(d)$, the pairwise error probability, is given as

$$P_2(d) = Q\left(\sqrt{\frac{drE_b}{N_o}}\right). \tag{5.1.4}$$

The coefficients a_d of an RSC code is same as that of the related NSC code, while the coefficients c_d of an RSC code have a tendency to increase as a function of d but slower than the c_d coefficients of the related NSC code irrespective of the rate and constraint length. Therefore, at low SNRs, the BER of RSC codes is lower than that of the NSC codes. However, for rates $r \le \frac{2}{3}$, the first two coefficients $c(d_{free})$ and $c(d_{free}+1)$ are larger in RSC codes and thus, the NSC codes perform better than RSC codes at high SNR. For rates $r > \frac{2}{3}$, the RSC codes perform better than the NSC codes irrespective of the SNR. Simulation results in [Ber 93] show a coding gain of 0.7 dB in the performance of RSC codes as compared to NSC codes with rate $r = \frac{2}{3}$ and a coding gain of 1.8 dB at rate $r = \frac{3}{4}$.

5.2 Concatenated Convolutional Codes: Turbo Codes

A parallel concatenation of RSC codes, known as turbo codes, was presented in [Ber 93] showing amazing BER performance at very low SNR. Until the advent of turbo codes, the concatenation of a Reed Solomon outer code and a convolutional inner code had widespread applications. The concatenation of component codes produces a stronger code and results in better BER performance. Although multiple component codes could be concatenated to improve performance, it is seen that a vast improvement in performance is achieved with two component codes but the performance improvement decreases as the number of component codes increases. Moreover, the complexity of the decoder increases exponentially with an increase in the number of component codes.

Turbo codes are conventionally a parallel concatenation of two identical RSC codes. Fig. 5.5 shows a rate 1/3 turbo code. An interleaver separates the two RSC encoders. The same input stream x feeds both encoders but the input to the bottom RSC encoder $\tilde{x}$ is a permuted version of that to the top RSC encoder due to the interleaver. The interleaver accepts an input block and scrambles the position of the bits before it feeds it to the bottom RSC encoder. Due to the presence of the interleaver, turbo codes are essentially block codes. The rate of the overall code is given by [Ber 93]

$$\frac{1}{R} = \frac{1}{R_1} + \frac{1}{R_2} - 1, \tag{5.2.1}$$

where R is the rate of the turbo code, R_1 and R_2 are the rates of the individual RSC encoders.

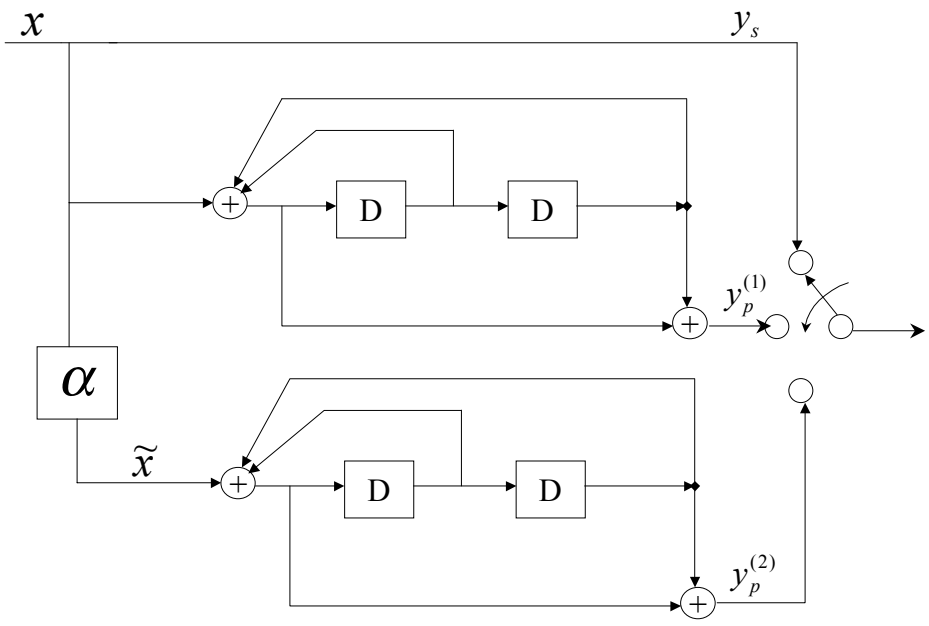


Fig. 5.5 A Rate 1/3 Turbo Code.

The systematic output y_s of the top RSC encoder is used while that of the bottom encoder is neglected. The other two outputs giving rise to an overall rate 1/3 code are the parity outputs $y_p^{(1)}$ and $y_p^{(2)}$ of the respective RSC encoders. The component RSC encoders in the figure have g = (7 5) and are the same encoders used in the previous section. It is not essential that both encoders be identical. However, identical encoders are usually preferred. Thus, the specification of the generator polynomial of an overall turbo code is the same as that for an individual component RSC encoder: $g = (g_1\ g_2)$ where g_1 is the feedback generator and g_2 is the feedforward generator unless specified otherwise.

The systematic nature of RSC codes makes it simple to have a parallel concatenation and still possess the systematic information at the output of the concatenated code. Although the systematic nature is useful, RSC codes have been chosen as component codes due more to their recursive nature than their systematic nature [Ben 98]. The recursive nature results in arbitrary inputs producing high weight code words with high probability and low weight code words with low probability.

Consider the NSC and RSC encoders in Fig. 5.2 and Fig. 5.3 respectively. Let an input stream of [1 0 0 0 0 0 0 0] (8 bits) be fed into each of these encoders. The outputs are as follows:

- NSC Encoder: Output $y^{(1)} = [1\ 1\ 1\ 0\ 0\ 0\ 0\ 0]$.
  Output $y^{(2)} = [1\ 0\ 1\ 0\ 0\ 0\ 0\ 0]$.

- RSC Encoder: Output $Y^{(1)} = [1\ 0\ 0\ 0\ 0\ 0\ 0\ 0]$.
  Output $Y^{(2)} = [1\ 1\ 1\ 0\ 1\ 1\ 0\ 1]$.

Hence, it is seen that with the same input frame of 1 followed by 7 zeros, the NSC encoder produces an output with Hamming weight 5 while the RSC encoder has an output with Hamming weight 7.

It is this recursive nature of RSC codes that is exploited by turbo codes. Owing to the feedback, the RSC encoders produce high weight outputs with high probability. Also, the combination of interleaving and RSC encoding ensures that the turbo code outputs generally have high Hamming weight. This is because of the fact that even if an input pattern is found that produces a low weight code word at the output of one of the RSC encoders, the probability that the interleaved version of the same input pattern will also produce a low weight code word at the output of the second RSC encoder is very low. Unfortunately, there are always such patterns that can generate low weight output code words at the output of both encoders, and hence, the minimum Hamming distance of the overall turbo code tends to be rather low. Fortunately, the multiplicity of these low weight code words is very small.

5.2.1 Performance of Turbo Codes

The code word error probability of a soft decision decoded (n,k) block code is given as

$$P_c \leq \sum_{d=d_{\min}}^{n} a_d \, Q\left(\sqrt{\frac{2rd_{\min}E_b}{N_o}}\right), \tag{5.2.2}$$

where a_d is the total number of output code words with weight d.

Similarly, the BER of a soft decision decoded (n,k) block code is given as

$$P_b \le \sum_{d=d_{\min}}^{n} C_d \, Q\left(\sqrt{\frac{2rdE_b}{N_o}}\right). \tag{5.2.3}$$

The parameter C_d for block codes is as follows

$$C_d = \frac{a_d \, \widetilde{w}_d}{k} \ , \tag{5.2.4}$$

where $\widetilde{w}_d$ is the average information weight corresponding to the weight d output code words.

At high SNR, the performance of block codes is dominated by the minimum (free) distance.

$$P_b \approx C_{d\min} \, Q\left(\sqrt{\frac{2rd_{\min}E_b}{N_o}}\right). \tag{5.2.5}$$

A performance improvement (i.e. reduction in bit error probability) can be achieved by the following two measures

- *Maximizing* d_{min} : A conventional code design approach
- *Minimizing* $C_{d\,min}$: Turbo code approach.

Maximizing minimum distance is a conventional code design approach. However, as seen in the previous section, turbo codes have low minimum distances. Irrespective of the minimum distance, the approach with turbo codes called spectral thinning is to minimize the multiplicity of low weight output code words. Since the number of low weight output code words is low, the weight spectrum is sparse, and hence, spectral thinning takes place.

The BER performance curve of a constraint length K = 5, rate 1/3 turbo code with input frame size k = 1024 over an AWGN channel is shown in Fig. 5.6. It is seen that the performance shown in the figure and the performance of any turbo code in general can be categorized into three regions.

- Low SNR region: The performance of the turbo code in this SNR region is very poor. There is just marginal reduction in BER with an increase in SNR.
- Transition region: This region is known as the waterfall or the "cliff" region. In this region, the bit error rates fall dramatically with a minute increase in SNR, thus making the curve seem like a waterfall. At low SNR, the number of low weight output code words plays a dominant role in the performance of the code.

The reason for the dramatic fall in bit error rates with slight increase in SNR is the low multiplicity of low weight output code words in a turbo code.

- High SNR region: This region is known as a "floor". At high SNR, the performance of a code is dominated by the minimum (free) distance of the code. Since, a turbo code has low minimum distance, the BER performance curve starts to flatten at high SNR giving rise to a BER floor at bit error rates of $10^{-5} - 10^{-6}$.

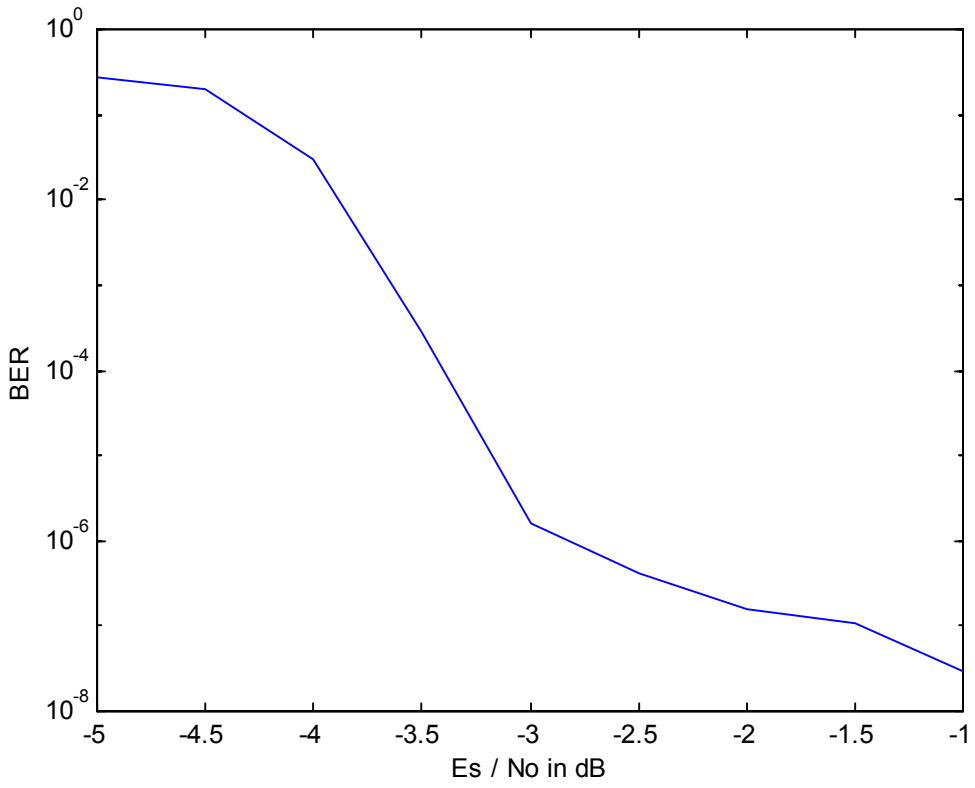


Fig. 5.6 Performance of a K = 5, rate 1/3 turbo code in an AWGN channel.

5.2.2 Turbo Code Interleaver Design

The interleaver in the turbo encoder performs three important functions:

- *Imparts randomness to the code*: It is well known that long random codes can achieve the best possible performance (Shannon's capacity limit). However, purely random codes cannot be decoded. Code structuring is required to facilitate decoding at the receiver. Therefore, a good balance must be struck between the random nature of the codes to achieve near capacity performance, and the structure of the code in order to decode. Block and convolutional codes strike a

reasonable balance between the two. Interleavers in turbo codes impart additional randomness to the code (scrambling the order of the input data) and possess enough structure to allow decoding at the receiver.

- *Decreases the number of low weight output code words*: As seen in the previous sections, even if there is an input pattern that causes a low weight code word at the output of one RSC encoder, the probability that the interleaved version of the input data will also cause a low weight code word at the output of the other RSC encoder is very low. This minimizes the number of low weight code words at the output of the overall turbo encoder.

- *Increases the performance of turbo code in the presence of burst errors*: Burst errors (a chain of successive bits in error) are detrimental to the performance of a code. Since the interleaver scrambles the input data bits, adjacent bits in the original input data will be placed a distance away from each other after interleaving. Thus, the number of burst errors is reduced and performance of a turbo code is improved.

The above issues motivate the design of efficient interleavers. Various interleaver designs have been considered and implemented with turbo codes. The most commonly used interleavers are as follows:

1. Block Interleaver: This is the simplest interleaver used with a turbo code. A block interleaver is one in which the input data is written column-wise and is read-out row-wise as shown in Fig. 5.7. This ensures that adjacent bits in the original input data are separated at least by the number of columns in the interleaver.

2. Pseudo-Random Interleaver: This interleaver pseudo-randomly permutes the input data. It picks a random number (i) within the size of the input data frame and maps the input bit to the i^{th} position in the interleaved frame as shown in Fig. 5.8.

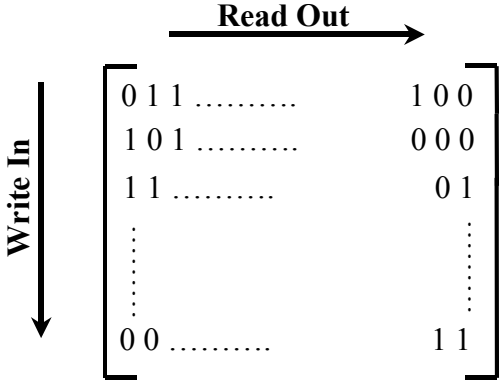


Fig. 5.7 A Block Interleaver.

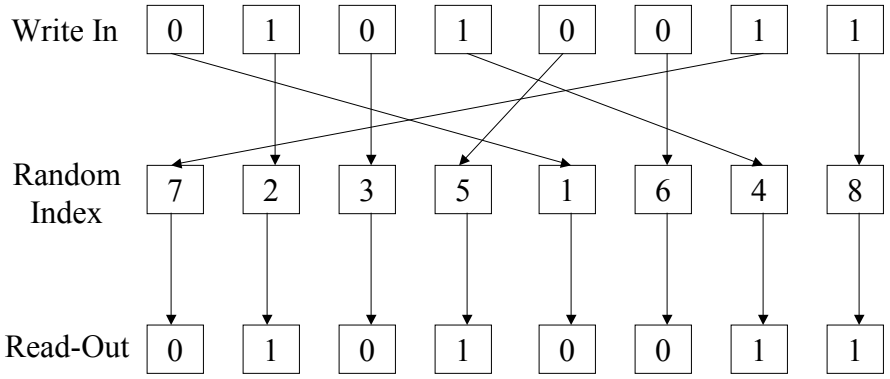


Fig. 5.8 A Pseudo-Random Interleaver.

3. Spread interleaver: A spread interleaver ensures that all the adjacent bits in the input data sequence are separated by a factor S which is related to the frame size k as follows

$$S < \sqrt{\frac{k}{2}}. \tag{5.2.6}$$

The algorithm consists of randomly picking an index (a) and comparing it with S previous integers to see if a lies within the range $\pm S$ from the input data bit position. If it does, then another random index is picked and the process is repeated all over again. Otherwise, the random index is kept. A spread interleaver with k = 10 and S = 2 is shown in Fig. 5.9.

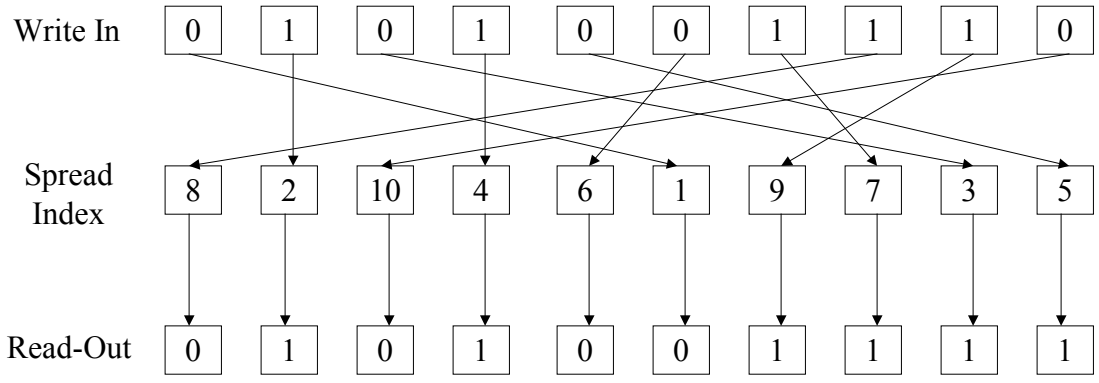


Fig. 5.9 A Spread Interleaver.

5.2.3 Trellis Termination in Turbo Codes

The problem of trellis termination in turbo codes does not have a trivial solution. In NSC codes, the trellis can be terminated by padding the input sequence with m zeros where m is the length of the shift register. This brings the NSC encoder back to state zero and the trellis is terminated. With turbo codes, the following two issues make trellis termination of both encoders much more difficult.

1. *The component RSC codes possess feedback and hence an input stream of m zeros does not necessarily terminate the trellis.*
2. *Owing to the interleaver, the tail bits that terminate one RSC encoder do not necessarily terminate the other. Moreover, these tail bits do not lie at the end of the input information sequence due to interleaving.*

The solution to the first issue is not a difficult one while that to the second is not as easy. Most turbo code applications require one of the two encoders to be terminated (usually the top one). The state in which the encoder finishes after encoding the entire input sequence is noted. Based on the memory contents of the positions that are tapped in the feedback loop, an input tail bit is found such that the feedback variable (the input to the shift register) is zero. This is repeated m times to bring the encoder back to state zero. In this manner, the top encoder can easily be terminated.

The encoder diagram in Fig. 5.10 is one of the many solutions proposed for the simultaneous trellis termination of both encoders [Val 99] and will be used in this thesis wherever applicable. The diagram shows each RSC encoder to possess a switch. The switch is in position A while the input data is being encoded and is in position B after the input data is encoded. These tail bits to each RSC encoder are input when the switch is in position B (via streams $t_i^{(1)}$ and $t_i^{(2)}$). The switches return to position A after the encoders are terminated. The tail bits used to terminate the bottom encoder have to be transmitted separately. The disadvantage of this scheme is that the encoders operate on different input data, which is in contrast to the assumption at the decoder.

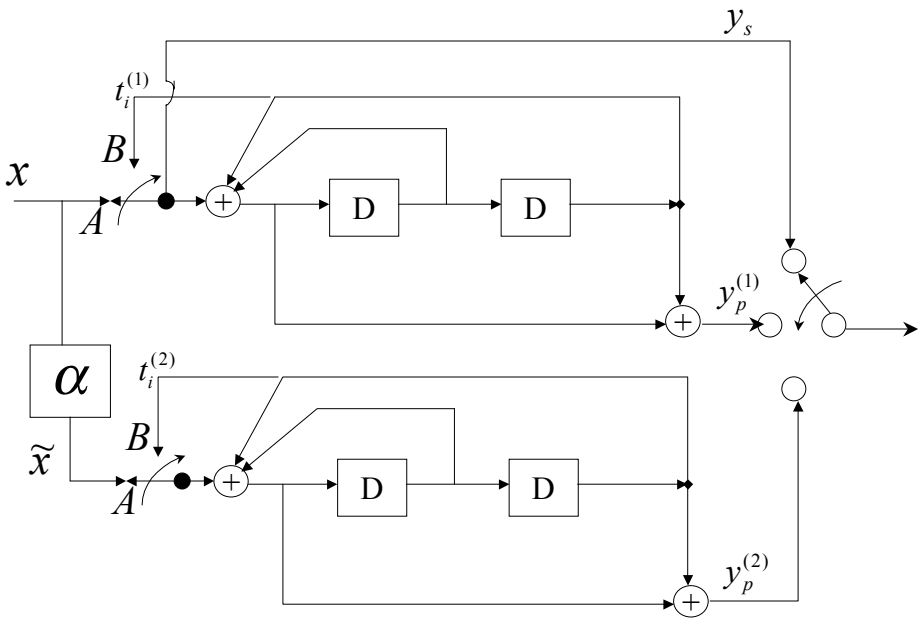


Fig. 5.10 Forced Trellis Termination in Turbo Rate 1/3 Encoder.

5.3 Iterative (Turbo) Decoding Algorithm

The essential features of turbo codes are parallel concatenation of codes, recursive systematic convolutional component codes, pseudo-random interleaving and an iterative decoding algorithm. However, it is the iterative decoding algorithm that distinguishes turbo codes from the previous classes of decoders. Since then, the iterative decoding algorithm has gained widespread usage in many other applications.

A maximum a posteriori (MAP) decoder evaluates the received sequence R and selects the estimate y' of the transmitted symbol that maximizes the probability $p(y'|R)$. An a posteriori algorithm was presented by Bahl *et al.* in 1974 [Bah 74] known as BCJR algorithm that bears the name of the authors. This algorithm was introduced to provide an alternative to the Viterbi algorithm for decoding convolutional codes. The Viterbi algorithm is a maximum-likelihood decoding algorithm that minimizes the sequence error probability but does not necessarily minimize the bit error probability. The symbol by symbol BCJR-MAP algorithm minimizes the probability of a symbol (or bit) error. The Viterbi algorithm considers the most likely state transition in a particular trellis section while the MAP algorithm considers the likelihood of all the bit transitions caused due to input 1 and input 0 separately and outputs a likelihood ratio.

The main building block of the turbo decoder is the soft-input soft-output (SISO) decoder, which accepts soft values from the channel and soft a priori values of input bits and produce soft a posteriori probability (APP) values of the bits at the output. This is shown in Fig. 5.11. Since, the turbo decoder outputs soft information about each information bit, it is beneficial to provide an algorithm that minimizes the probability of bit error. Also, the presence of the interleaver makes the trellis structure very complex and hence making it difficult to employ the maximum-likelihood scheme. Therefore, [Ber 93] used the BCJR-MAP algorithm to decode turbo codes and it is this algorithm that is the heart of the turbo decoding process.

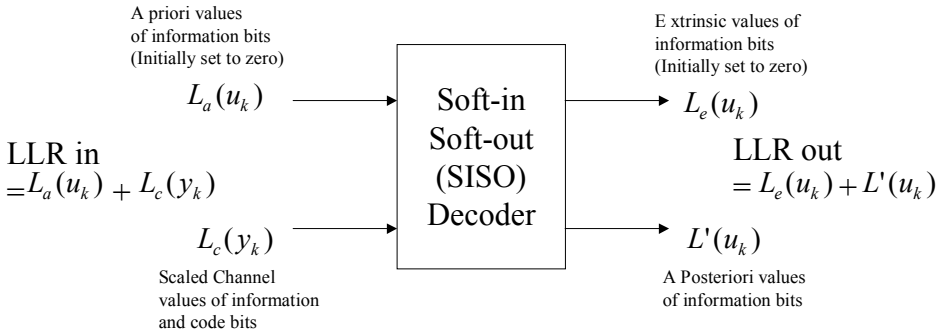


Fig. 5.11 A SISO Decoder for a Systematic Code.

Let u = (u_1 u_2 u_3…..u_k)[1] be the information bits produced by the source. The turbo encoder output is represented as x = (x_1 x_2 x_3 …..x_{nk}) where n is the number of outputs streams from the encoder. The encoded sequence is modulated and transmitted over the channel. The noisy received sequence is given by y = (y_1 y_2 ……. y_{nk}).

The soft-output of the SISO decoder is usually in the form of the log-likelihood ratio (LLR) of the probability of a data bit 1 being transmitted to that of a data bit 0 being transmitted given that y is received from the channel. The LLR output can be expressed as

$$L(u_k) = \log\left(\frac{P(u_k = +1|y)}{P(u_k = -1|y)}\right). \tag{5.3.1}$$

Using Bayes theorem, (5.3.1) is written as

[1] Change of notation is to be consistent with references

$$L(u_k) = \log\left[\frac{p(y|u_k=+1)P(u_k=+1)}{p(y|u_k=-1)P(u_k=-1)}\right]$$

$$= \log\left[\frac{p(y|u_k=+1)}{p(y|u_k=-1)}\right] + \log\left[\frac{P(u_k=+1)}{P(u_k=-1)}\right] \quad (5.3.2)$$

$$= L_c(y) + L_a(u_k), \quad (5.3.3)$$

where $L_c(y)$ is the LLR of the channel measurements of y under the conditions that $u_k = \pm 1$ may have been transmitted and $L_a(u_k)$ is the a priori information of the data bit u_k.

Consider a BPSK modulated input data frame transmitted over an AWGN or flat-fading channel with noise variance σ^2.

Therefore,

$$L_c(y_k) = \log_e\left[\frac{p(y_k|u_k=+1)}{p(y_k|u_k=-1)}\right]$$

$$= \log_e\left[\frac{\frac{1}{\sigma\sqrt{2\pi}}\exp\left[\frac{-1}{2}\left(\frac{y_k-a}{\sigma}\right)^2\right]}{\frac{1}{\sigma\sqrt{2\pi}}\exp\left[\frac{-1}{2}\left(\frac{y_k+a}{\sigma}\right)^2\right]}\right]$$

$$= \frac{-1}{2}\left(\frac{y_k-a}{\sigma}\right)^2 + \frac{1}{2}\left(\frac{y_k+a}{\sigma}\right)^2$$

$$= \frac{2}{\sigma^2} a\, y_k. \quad (5.3.4)$$

Since $\sigma^2 = 1/(2E_s/N_o)$, (5.3.4) becomes

$$L_c(y_k) = 4a\frac{E_s}{N_o} y_k, \quad (5.3.5)$$

where a is the fading amplitude which is = 1 for an AWGN channel.

L_c is thus called the channel reliability measure. For a systematic code, the output LLR in (5.3.3) can be broken up into data and parity sections and expressed as a sum of three factors.

$$L(u_k) = L_c(y_k) + L_a(u_k) + L^e(u_k)\,. \tag{5.3.6}$$

where $L^e(u_k)$ is the LLR information contributed by the parity associated with the input data bits and is called extrinsic LLR information.

The soft output LLR $L(u_k)$ of the SISO decoder is a real number that used to make two parameters associated with it: sign and magnitude. The sign of the real number is the hard decision (i.e. the transmitted bit is +1 if the sign is positive and –1 if the sign is negative). The magnitude of $L(u_k)$ is the reliability of the decision.

5.3.1 Principles of Turbo Decoding

Fig. 5.12 shows the schematic of a typical turbo decoder. Two MAP decoders called D1 and D2, (one for each RSC encoder called E1 or E2) each implementing the SISO algorithm discussed in the previous section are connected in series. Each decoder receives information encoded by its corresponding encoder after it is modulated, transmitted through a channel and scaled at the receiver by the factor L_c discussed in the previous section (i.e. D1 receives the scaled noisy systematic and parity information encoded by E1 whereas D2 receives the scaled noisy parity of E2 and the permuted version of the noisy systematic information at the input of decoder D1). Both decoders also receive a priori information, which is the deinterleaved extrinsic information produced by the other decoder during the previous iteration.

In (5.3.6), $L_a(u_k)$ is the a priori information at the input of by the decoder. Since the message bits are typically equiprobable, this a priori information is zero for conventional decoders when expressed as a LLR. However, owing to the presence of extrinsic information from the other decoder, the a priori value needs to be computed in iterative decoders. The extrinsic information produced by a decoder for each message bit u_k is derived from information that the other decoder does not receive. For example: D1 produces extrinsic information that forms the a priori information for D2 by using parity information of E1 (not available to D2) and vice-versa.

Initially, the a priori information for D1 is set to zero signifying that for the first iteration, the information bits 0 and 1 are equiprobable. A single iteration of the entire decoding process involving both decoders can be divided into two half-iterations i.e. D1 operates during the first half-iteration and D2 during the second. During the first half-iteration, D1 accepts the systematic and parity channel values, y_s and $y_p^{(1)}$ corresponding to E1 along with a priori information, $L_a^{(1)}$ (set to zero for the first half-iteration). It produces the LLR of the information bits at its output $L_o^{(1)}$ from which the a priori information and the

channel values of systematic bits are subtracted to form the extrinsic information $L_e^{(1)}$. This information is interleaved to form the a priori information for D2 in the second half-iteration.

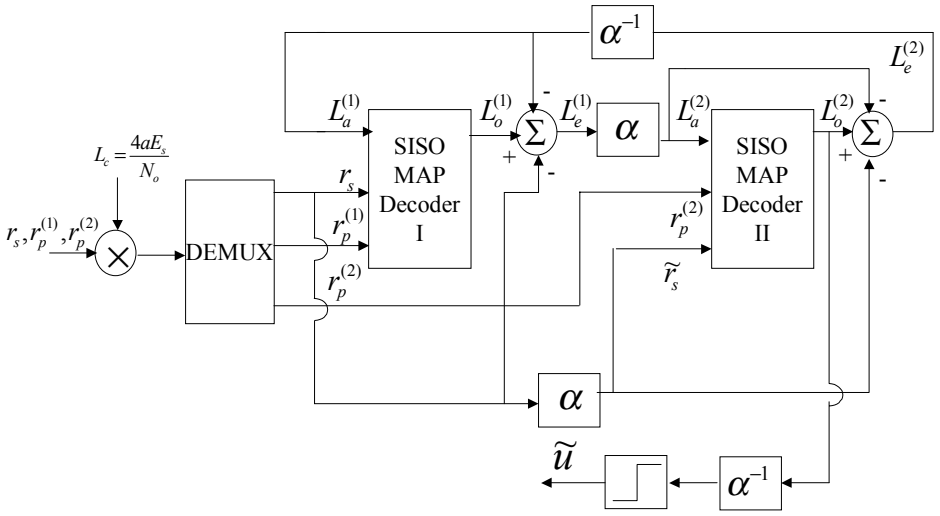


Fig. 5.12 Schematic of a Rate 1/2 Turbo Decoder.

Similarly, D2 accepts the received parity corresponding to E2 and a permuted version of the received systematic channel values. It also produces the LLR of information bits $L_o^{(2)}$ from which the systematic channel values and a priori information from D1 are subtracted to form the extrinsic information $L_e^{(2)}$. $L_e^{(2)}$ is then de-interleaved to form the a priori information for D1 during the next iteration.

In this manner, the two decoders assist each other in making decisions about an information bit by passing extrinsic information back and forth in an iterative fashion. The iterations continue until a preset maximum number of iterations are reached. The interleavers and de-interleavers arrange information in proper order for each decoder as shown in the figure. Only after the last iteration does the decoder D2 make a hard decision on a message bit, using the log-likelihood value of $L_o^{(2)}$.

Therefore, by producing soft-decision outputs and sharing information between constituent decoders iteratively, a vast performance improvement is achieved.

5.3.2 Practical MAP Decoding Strategies

MAP decoding is explained in detail with the algorithm and practical implementation techniques in Appendix A. This decoding algorithm computes the a posteriori probability of the information bits and is the algorithm that was originally proposed for use in constituent RSC decoders. However, it suffers from severe practical implementation problems [Rob 97]. It is very computationally complex, requiring $6 * 2^m$ multiplications and an equal number of additions per estimate. Also, it is very sensitive to round-off errors that result from representing numbers with finite precision. These two problems are alleviated by performing the MAP decoding in the log-domain. Two benefits of performing decoding in the log-domain are as follows.

- The multiplications become simpler addition operations in the log-domain
- The soft-output of the MAP decoders are in the form of log-likelihood ratios as discussed earlier. Now, instead of waiting until the end of each half-iteration to perform calculations in the log-domain as in the conventional MAP algorithm, the MAP algorithm in the log-domain facilitates the integrated computation of LLR during each iteration.

However, the addition of exponential values in the log-domain is not as easy. The Jacobian algorithm stated below illustrates how addition is performed in the log-domain.

$$\begin{aligned} \ln(e^x + e^y) &= \max(x, y) + \ln(1 + \exp\{-|y - x|\}) \\ &= \max(x, y) + f_c(|y - x|) . \end{aligned} \tag{5.3.7}$$

Equation (5.3.7) shows that addition of exponential values in the log-domain is equivalent to the addition of a maximization operation of the indices x and y and a correction function ($f_c(.)$) of the modulus of subtraction of the two indices. The Jacobi logarithm must be computed twice for each node of the trellis for each decoder (in what can be called a half-iteration) and hence, contributes significantly to the decoder complexity. Based on the manner in which the correction function is computed, a variety of classes of MAP algorithms have been instituted [Val 01]. They are briefly described below.

1. Log-MAP algorithm: In this algorithm, the Jacobi logarithm in (5.3.7) is evaluated exactly. The correction function is computed using log and exp C function calls or by having a large look-up table for values corresponding to various ranges of the subtraction of the two indices. This is the most complex of the algorithms to be described but owing to the complete evaluation of the Jacobi algorithm, it offers best performance.

2. Max-log-MAP algorithm: This algorithm only performs the maximization operation in (5.3.7). Hence, the name max-log-MAP. Since the correction function is not computed, the Jacobi logarithm is approximated as

$$\ln(e^x + e^y) \approx \max(x, y). \tag{5.3.8}$$

The max-log-MAP algorithm is the easiest to compute but has the worst performance of all the algorithms since it neglects the correction function. It performs about 0.4 dB worse than log-MAP.

3. Constant-log-MAP algorithm: This algorithm was presented in [Gro 98] and approximates the Jacobian logarithm using the following expression.

$$\ln(e^x + e^y) = \max(x, y) + \begin{cases} 0 & \text{if } |y - x| > T \\ C & \text{if } |y - x| \le T \end{cases}. \tag{5.3.9}$$

Optimal values of C and T are computed based on the parameters of the code. This algorithm is similar to using a 2-element look-up table with the log-MAP algorithm [Val 01]. Its complexity and performance lies between the log-MAP and the max-log-MAP. Although it performs very close (about 0.03 dB away) to log-MAP performance, it is very susceptible to errors if the variance of the noise required while scaling the received channel values with the channel reliability measure L_c in (5.3.4) is measured incorrectly.

4. Linear-log-MAP algorithm: This algorithm is proposed in [Val 01]. It renders a linear approximation to the Jacobian logarithm and is expressed as

$$\ln(e^x + e^y) = \max(x, y) + \begin{cases} 0 & if\ |y - x| > T \\ a|y - x| + b & if\ |y - x| \le T \end{cases}, \tag{5.3.10}$$

where the values of a, b and T are chosen to minimize the total squared error between the exact correction function and its linear approximation. It is seen that the values a = -.236, b = 0.592 and T = 2.508 minimize the function. This algorithm lies between the log-MAP and the constant-log-MAP in complexity and performance. The benefits of this algorithm are that it converges faster than constant-log-MAP and it is less susceptible to noise variance estimation errors as compared to the constant-log-MAP algorithm.

The four algorithms discussed above are the classes of MAP algorithms that provide ease in practical implementation of the constituent decoder blocks of Fig. 5.12 over the conventional MAP algorithm.

5.4 A Hybrid RCPT-ARQ System Model

Now that the various building blocks of the turbo encoder and decoder have been comprehensively studied and the encoding/decoding algorithms have been practically implemented, we can incorporate these into an end-to-end communication system. Puncturing can be done with turbo codes as well and provides a means to increase the transmission rate of the system by periodically deleting bits from the encoded sequence. as it was done with convolutional codes seen in the previous chapter. For example: a rate 1/3 turbo code can be punctured with the following puncturing pattern to form a rate 1/2 code.

$$P = \begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix}, \tag{5.4.1}$$

where the first row of the puncturing matrix is for the systematic bits from the turbo encoder, the second row is for encoder 1 parity and the third row is for encoder 2 parity.

As seen from the matrix, the systematic bits of the turbo-encoded sequence are not normally punctured to increase the code rate. Usually, only the parity bits are punctured and the systematic bits are transmitted since the performance of a turbo code deteriorates to a large extent when the systematic bits are also punctured. Only in the case where the system demands transmission at very high rates are the systematic bits punctured. In the same manner as seen in the previous chapter, a range of punctured code rates from highest (only information bits) to lowest (rate of the mother code) can be provided for a turbo-encoded system, as investigated in [Row 00].

The rate compatibility criterion is used when puncturing and hence only the bits that were punctured while transmitting at a higher rate are to be transmitted during successive attempts. Again, a rate-compatible punctured turbo (RCPT) code[2] blends well with the type-II hybrid ARQ system discussed in the previous chapters. It is seen that such a system has vastly improved BER and throughput performance over its convolutional code counterpart with the same decoder complexity.

5.4.1 Simulation Parameters of a Hybrid RCPT-ARQ System

The parameters used in the simulation of a hybrid RCPT-ARQ system in this thesis are tabulated in Table 5.1. The puncturing patterns used to generate the range of punctured code rates are given in Table 5.2.

[2] RCPT's are also called rate compatible punctured parallel concatenated convolutional code (RCP-PCCC)

Parameter	Value
Encoder	
Encoder type	PCCC (Turbo) encoder
Constituent encoder type	Two identical rate 1/2 RSC encoders
Generator Polynomial Feedback: Feedforward:	 35 (octal) 23 (octal)
Constraint length (K)	5
Parent code rate	1/3
Input frame size	1024
Interleaver size	1024
Interleaver type	Spread interleaver (S = 20)
Trellis termination type	Upper encoder only
Puncturing	
Puncturing Period	8
Punctured code rates	Ranges between 4/5 and 1/3 (Refer Table 5.2)
Puncturing Patterns	Refer Table 5.2 [Row 00]
Modulation	
Modulation type	BPSK with coherent detection
Channel	
Channel types	1. AWGN Channel 2. Fully Interleaved Rayleigh Flat Fading Channel
Decoder	
Decoder type	Max-log-MAP decoder
Number of Iterations	10
Automatic Repeat Request	
Hybrid ARQ type	Type II Hybrid ARQ
ARQ protocol	Selective repeat with error and delay free feedback channel

Table 5.1 Simulation Parameters of an hybrid RCPT-ARQ System.

	PCCC							
Overall Rate	4/5	8/11	2/3	4/7	1/2	4/9	2/5	1/3
Puncturing Patterns	377 020 020	377 024 020	377 024 024	377 224 224	377 264 264	377 265 274	377 275 374	377 377 377

Table 5.2 Puncturing patterns in octal for each code rate [Row 00].

5.4.2 Performance of hybrid RCPT-ARQ System

Fig. 5.13 and Fig. 5.14 show the BER performance of the turbo code with various punctured code rates over an AWGN and a fully interleaved Rayleigh flat-fading channel respectively.

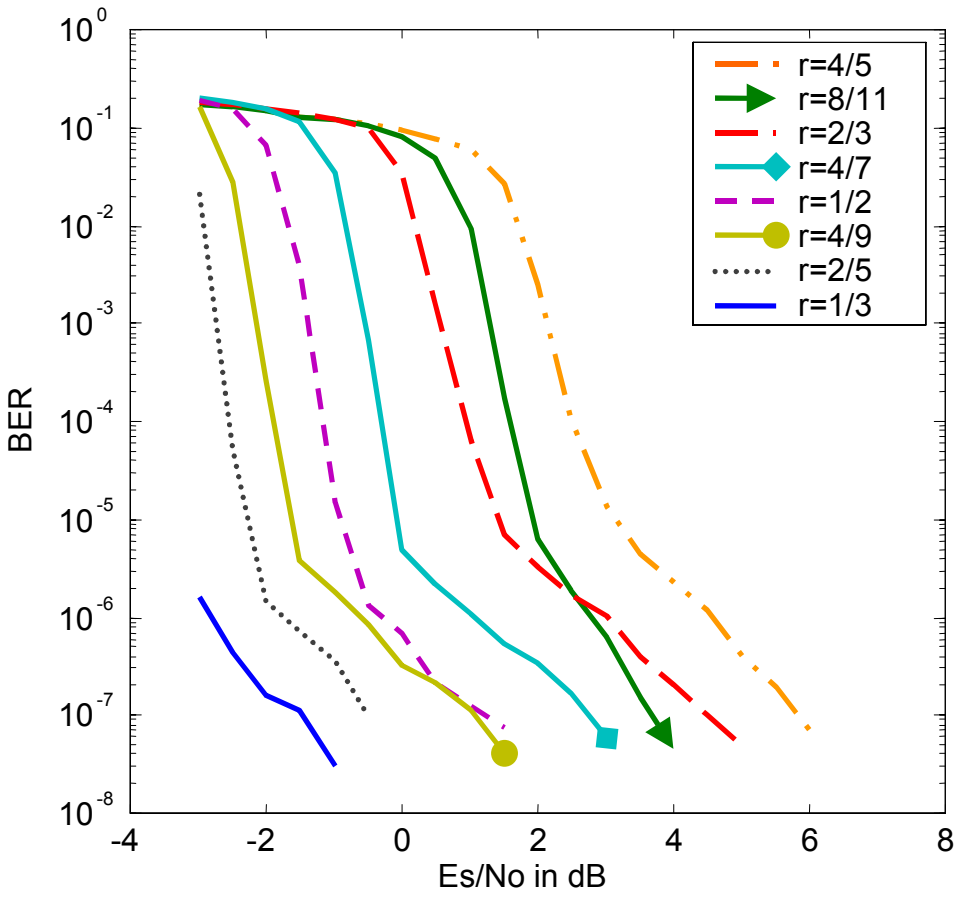


Fig. 5.13 BER performance in AWGN of a N=1024 bit PCCC with generator (35,23). From left to right, the curves are in order of ascending code rate.

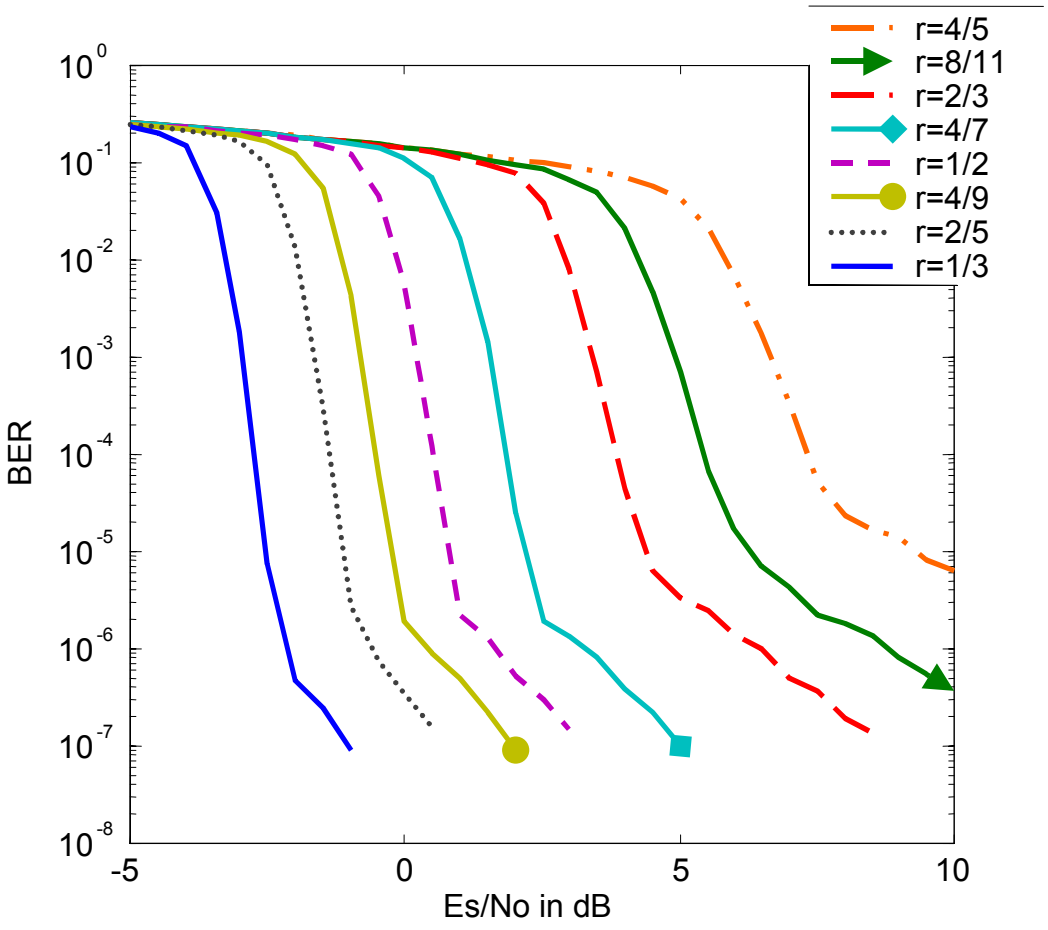


Fig. 5.14 BER performance in flat-fading Rayleigh channel of a N=1024 bit PCCC with generator (35,23). From left to right, the curves are in order of ascending code rate.

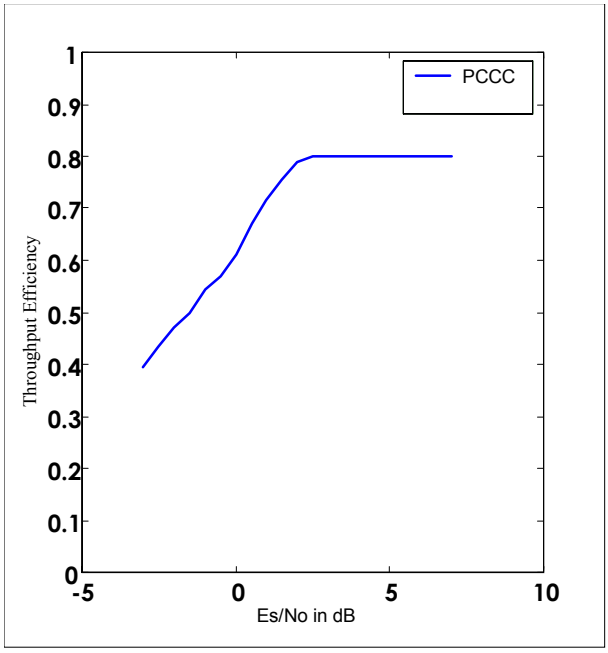


Fig. 5.15 Throughput of a hybrid RCP-PCCC-ARQ system in an AWGN channel.

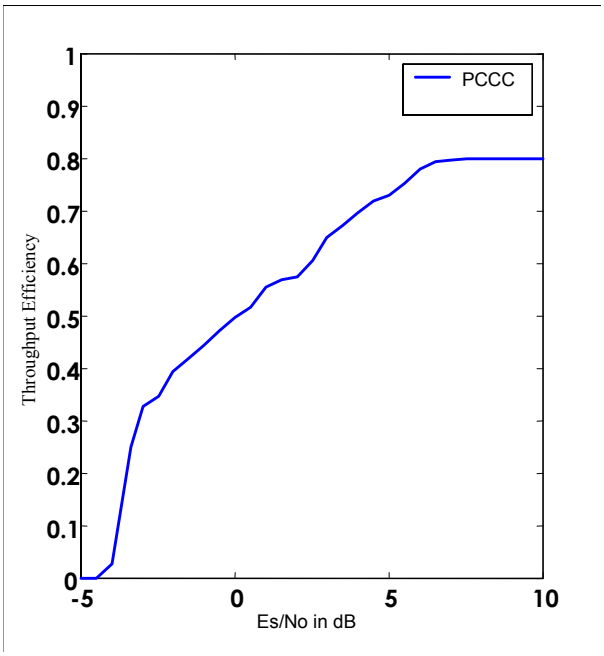


Fig. 5.16 Throughput of a hybrid RCP-PCCC-ARQ system in a Rayleigh flat fading channel.

As seen in the figures, the turbo-coded system performs at very high energy efficiencies while offering bit error rates down to $10^{-6} – 10^{-7}$. The waterfall region of these curves, as discussed earlier, provides vast improvement in bit error rates with very little increase in signal to noise ratio. This is because of the low multiplicity of low weight code words produced by the encoder. However, at high signal to noise ratios (where the minimum free distance dominates), the PCCC (turbo) system develops a bit error rate floor owing to its low minimum free distance. The nature of the performance curves of the various punctured rates are similar to those discussed in the RCPC chapter but only at lower signal to noise ratios (for the AWGN channel case). The nature of the curves in the fading case are similar to those in the AWGN case only at higher signal to noise ratios. As expected, the performance degrades with increase in code rate.

Fig. 5.15 shows the throughput of a hybrid RCPT-ARQ system over AWGN and fading channels. Again, the throughput increases with increase in the signal to noise ratio since at high SNR the high rate codes can be used. The improvement in BER performance is attributed to the turbo coding principle while that in the throughput is owing to the provision of a rate-compatible punctured turbo coded system with type-II hybrid ARQ.

5.5 Chapter Summary

Turbo codes have shown to perform at very high energy efficiencies as seen in the plots of simulation results. The steepness of the BER curve at low SNR is owing to the low multiplicity of low weight code words while the error floor at high SNR is owing to the low minimum distance of turbo codes. A hybrid RCP-ARQ system works very well with turbo codes as its FEC component as seen from the throughput performance plots. It is seen that high transmission rates with low error probabilities can be achieved with this kind of a system. However, the error floors in the BER performance occur at rather high error rates. This problem is addressed by considering serially concatenated encoder structures, which are the focus of the next chapter.

Chapter 6

A Serial Concatenated Approach

Forney introduced the concept of concatenated codes in 1966 [For 66]. A striking feature of these codes is that the bit error probability reduces exponentially with increasing frame size while the complexity of the decoder increases only algebraically. Just before the advent of turbo codes, there was an increased motivation in the coding community to explore the concatenation of convolutional and Reed-Solomon codes. An outer RS code encodes the frame of data first. The output of the RS code is fed to a convolutional inner encoder after symbol interleaving is performed. Owing to its performance with just a reasonable increase in decoding complexity, this kind of RS-convolutional serial concatenation still has widespread applications.

It was seen in chapter 5 that although PCCCs perform very well at low SNR (i.e. the "cliff region"), they give rise to a fairly high BER floor at high SNR. In order to alleviate this problem, serial concatenated convolutional codes (SCCCs) were proposed by Benedetto, et al [Ben 98] using the concepts introduced by Forney. The basic principles behind PCCCs such as code concatenation, recursive systematic convolutional component codes, efficient non-uniform interleaving, trellis-termination, and the iterative soft-input soft-output MAP decoding are borrowed by the SCCCs. However, the serially concatenated structure leads to some very interesting and inspiring results. It is seen that the SCCC outperforms the PCCC at high SNR by going down to bit error rates in the range 10^{-7} – 10^{-9} in the waterfall region and thereby alleviating the high bit error floor generated in the case of the PCCC.

This chapter begins with an introduction to serial concatenated recursive systematic convolutional encoders and the functions of the component blocks. A theoretical comparison of the performance of the SCCC system against that of PCCC system follows next. The next section of this chapter explains the SISO-MAP decoding algorithm for SCCCs with block and schematic diagrams. Finally, a rate compatible punctured SCCC system with hybrid ARQ (hybrid RCP-SCCC-ARQ) is developed and simulated over AWGN and fading channels. The work in this chapter constitutes the original contribution of this thesis and has been published in [Cha 01a].

6.1 Serial Concatenated Convolutional Codes (SCCC)

Figure 6.1 shows the block diagram of a typical serial concatenated convolutional encoder with two component codes.

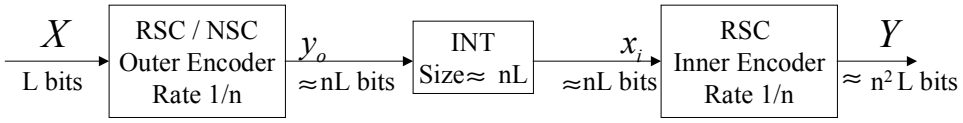


Fig. 6.1 A typical serial concatenated structure.

The outer convolutional encoder with rate 1/n transforms the input frame X with frame size of L bits to approximately[1] nL bits at its output. The interleaver scrambles the output of the outer encoder and feeds it to the inner encoder. The inner encoder maps the interleaved output of nL bits to approximately n^2L coded bits at the output of the SCCC encoder Y. This output Y is modulated and transmitted through a channel.

Ideally, both encoders should be RSC encoders. But the decoder can track the systematic bits of the outer encoder once the output of the outer encoder is passed through the interleaver. Also, there is no compelling reason for the outer encoder to be systematic. However, there are benefits of the recursive nature of convolutional codes that are exploited by the concatenated encoder as seen with PCCCs. These benefits coupled with the fact that the systematic bits at the input of the inner encoder are produced directly at the output of the concatenated encoder makes it imperative for the inner encoder to be an RSC encoder. Henceforth, it is assumed that both encoders are RSC encoders.

Similar to the PCCC case, the pseudo-random interleaver has two important functions.

- Introduces randomness to the encoder with enough structuring so as to be able to decode at the receiver.
- Transforms, with high probability, any outer encoder output pattern that causes a low weight codeword at the output of the inner encoder to a pattern that produces a high weight codeword at the inner encoder output.

RSC encoders generally produce high weight outputs. However, as was seen in the case of PCCCs, there exist input patterns that cause the first RSC encoder to produce a low weight output. The interleaver transforms such input patterns with high probability so that the output of the second encoder has a high weight and hence the output of the overall turbo code can have a moderate weight.

[1] It is an approximation since the tail bits are not considered

The probability of finding such input patterns that cause a low weight codeword at the output of the outer encoder in SCCCs is the same as that in PCCCs. However, with SCCCs, this low weight output does not get reproduced directly in the output of the concatenated code. Since the interleaver in the SCCC encoder works on the output of the outer encoder, it attempts to prevent such low weight output codewords to appear at the input of the inner encoder (i.e at the output of the concatenated encoder) with high probability. Therefore, the effective multiplicity of low weight codewords at the output of the SCCC encoder is reduced.

The overall rate of the SCCC encoder is the product of the individual RSC encoder rates. The outer and / or the inner code can be punctured if very high rates of transmission are desired. In [Oci 00], high rate punctured SCCCs were considered. As would be expected, the BER floors of the high rate SCCCs were substantially lower than those of similar PCCCs. However, a surprising result was that at very high rates (i.e. r=15/16), it is possible to find SCCCs with cliffs that occur at the same or even slightly lower SNR than comparable PCCCs. Unfortunately, the codes presented in [Oci 00] are not rate compatible, and thus not suitable for hybrid FEC-ARQ applications.

6.2 Theoretical Performance of SCCC in comparison to PCCC

The BER of a soft decision decoded (n,k) turbo block code as given in (5.2.3) is

$$P_b \leq \sum_{d=d_{\min}}^{n} C_d \, Q\left(\sqrt{\frac{2rdE_b}{N_o}}\right). \tag{6.2.1}$$

The parameter C_d is defined as follows

$$C_d = \frac{a_d \, \tilde{w}_d}{k}, \tag{6.2.2}$$

where $\tilde{w}_d$ is the average information weight corresponding to the weight d output code words and k = the interleaver size N.

It is seen that the probability of bit error of a PCCC scheme is proportional to N^{-1}. This is known as the interleaver gain. Larger frame size corresponds to larger interleaver size resulting in a steeper slope of the BER curve. Therefore, the high floor in the performance of the PCCC is lowered by increasing interleaver size.

The performance improvement with increasing interleaver size also holds good for the SCCC scheme. However, with the SCCC scheme, performance can be improved even by keeping the interleaver size constant. This is because the SCCC attempts to decrease the

exponent of N, thereby increasing the rate at which the bit error probability reduces with increase in SNR i.e. P_b of the SCCC scheme can decrease at a rate of N^{-2}, N^{-3}, and so on.

The factor that determines the value of the exponent of N in the PCCC case at high SNR is $(1-w_{dfree})$ where w_{dfree} is the minimum input weight producing the free distance codewords at the output of the PCCC encoder [Wu 00]. Since recursive systematic convolutional codes of rate 1/2 are used, the minimum input weight that causes the smallest output weight codeword is 2. Therefore, the exponent = -1 for PCCC.

Similarly, in the SCCC case, the factor that determines the exponent of N at high SNR is $(1-d_{fo})$ where d_{fo} is the free distance of the outer encoder. Since the input to the inner code is restricted to the permuted version of the output of the outer encoder, the ratio of the free distances of the outer to inner encoder is approximately = 1 if a pseudo-random interleaver is assumed. It is seen that the exponent is now based on the free distance of the outer encoder. As the free distance of the outer code increases, the exponent of the overall SCCC code decreases, resulting in a lower BER floor. It is relatively simple to design convolutional codes with free distance > 2 and hence the exponent of N can be smaller than –1.

Since the BER of SCCCs reduces at a faster rate and can reach the range 10^{-7}- 10^{-9} in the waterfall region with interleaver size kept constant, the error floor seen in the case of PCCCs at a rather high BER range is alleviated. However, the tradeoff is about a 1.5 fold increase in the decoder complexity and weaker BER performance at low SNR as compared to PCCCs.

6.3 SCCC SISO-MAP Decoder

The iterative SISO-MAP algorithm and the practical implementations of the MAP algorithm are explained in section 5.3 of Chapter 5 in conjunction with Appendix A. These underlying concepts are exactly the same for decoding of PCCC and the SCCC. However, improvisations need to be made in the decoder to incorporate the serial structure of the concatenated code. A block diagram of the SCCC SISO-MAP decoding algorithm is shown in Fig. 6.2 while a schematic of the detailed decoder diagram is given in Fig. 6.3.

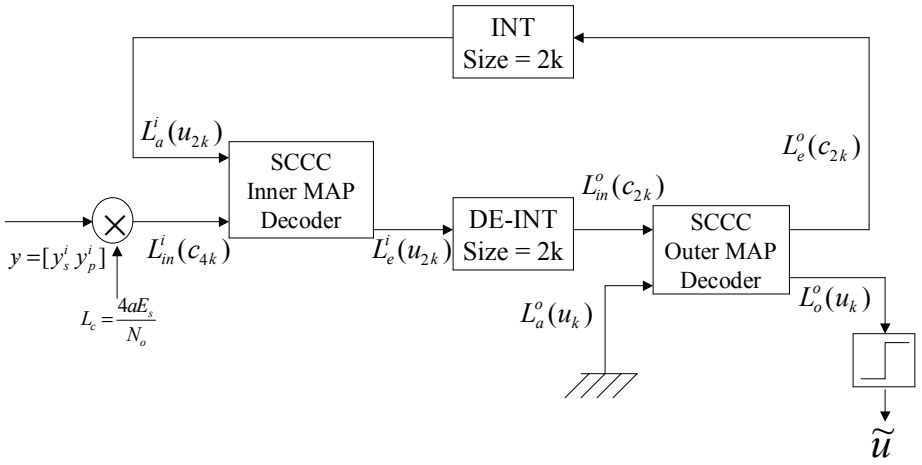


Fig. 6.2 A SCCC Decoder Block Diagram.

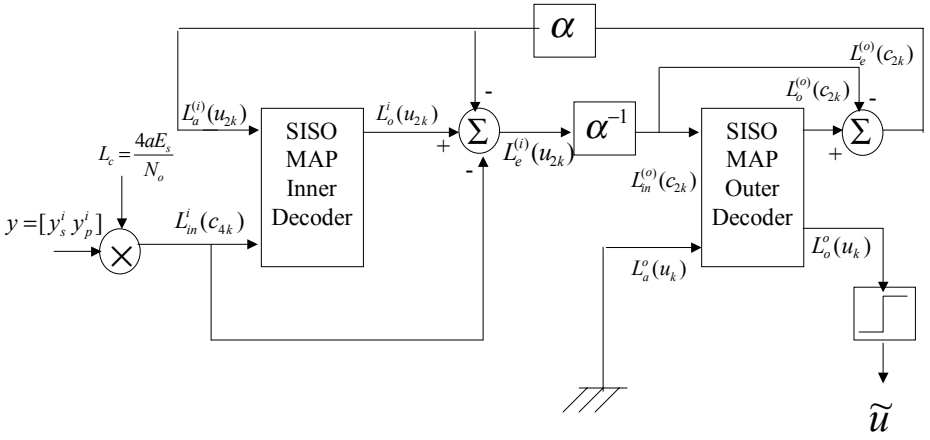


Fig. 6.3 A SCCC Decoder Schematic.

Consider the SCCC encoder to be composed of two RSC encoders of rate 1/2. Let $u=[u_1\,u_2\,u_3\,....u_k]$ be the information bits input to a constituent RSC encoder and $c=[c_1\,c_2\,c_3\,.....c_n]$ be the code bits at the output of the encoder. Let $L_a^b(x_c)$ be the LLR generated in the SCCC decoder where a, b, x and c can take on the following values.
a: (i for inner, o for outer) denoting the inner or outer decoder,
b: (a for a priori, in for input, and e for extrinsic and o for output) denoting kind of LLR information,

x: (u for information, c for code) distinguishing between information and codeword bits
c: value denoting the number of bits (information or code bits, depending on x).

The input to the decoder is a noisy version of the multiplexed systematic and parity bits of the inner encoder represented as $y = [y_s^i \; y_p^i]$. The input y to the decoder is scaled by the channel reliability factor

$$L_c = \frac{4aE_s}{N_o}, \tag{6.3.1}$$

where a is the fading amplitude and is =1 for AWGN channel.

The order of operation of the decoders is the reverse of that of the encoders. Hence, the scaled input is fed into the inner MAP decoder first. The operation of the SCCC decoder is iterative and every iteration is composed of two half-iterations (one for each constituent MAP decoder).

During the first half-iteration, the inner decoder accepts the input frame (size $\approx 4k$) and computes the extrinsic information $L_e^i(u_{2k})$ of the information bits only. The a priori information to the inner decoder is set to zero for the first iteration. The extrinsic information generated by the inner decoder is deinterleaved and forms the input $L_{in}^o(c_{2k})$ to the outer decoder. Recall that extrinsic information is generated by a decoder with the help of information that is not available to the other decoder (inner encoder parity is not available to the outer decoder).

The a priori information to the outer decoder is always set to zero and is not used. The outer decoder not only produces extrinsic information of information bits $L_e^o(u_k)$ but also computes LLR of code word bits $L_e^o(c_{2k})$. The computation of extrinsic information of codeword bits is the only major difference between the functioning of the SCCC outer decoder as compared to the SCCC inner decoder or any PCCC decoder.

The extrinsic information of the information and codeword bits is interleaved and fed back to the inner decoder forming the a priori information for the next iteration. The decoders keep sharing extrinsic information of information and codewords bits in the fashion described above for a number of pre-set iterations. After the last iteration is performed, the outer decoder computes the complete output LLR $L_o^o(u_k)$ of the information bits. A hard-decision is performed on these bits and an estimate of the original information sequence $\tilde{u}$ is produced.

6.4 A Hybrid RCP-SCCC-ARQ System

The SCCC encoder and decoder can be incorporated in an end-to-end communication system to evaluate the BER performance. The rate of the overall SCCC system with two rate 1/2 composite RSC encoders is 1/4. Low rate SCCCs such as rate 1/4 SCCCs can perform with very high energy efficiencies. This is because of the already existing performance benefits of a SCCC system discussed earlier coupled with the low rate of operation. High-rate SCCCs can be constructed using the same component RSC encoders with rate 1/2 by puncturing the output of each RSC encoder. The effective rate of the inner and outer RSC encoder increases and thus the overall rate of the system increases. Usually, the output of the outer encoder is punctured if only moderate rates are desired. However, if the requirement is for high rate transmission, both the inner and the outer encoder outputs must be punctured.

Similar to the PCCC case, a bunch of rate compatible punctured rates are considered for use with SCCCs. Simulations are run for each of these different rates. The simulation parameters are tabulated in Table 6.1. Table 6.2 shows the different rates and the puncturing patterns for each rate. These puncturing patterns are rate compatible.

An effort is made to keep the parameters of the hybrid RCP-SCCC-ARQ system consistent with the parameters used in the simulation of the hybrid RCP-PCCC-ARQ system discussed in the previous chapter to conduct a performance comparison between the two systems. However, it has been difficult to match up all the punctured rates of the two systems owing to the fact that not all of the rates considered for the PCCC case can be generated by products of the two RSC code rates, especially when the system must be rate-compatible. Nevertheless, most of the rates of the two systems are close enough to be compared.

The BER performance of an RCP-SCCC system simulated over an AWGN channel is shown in Fig. 6.4 for each of the punctured code rates given in Table 6.2. Fig. 6.5 shows the simulation of the same system over a fully interleaved Rayleigh flat fading channel.

The vast improvement in performance of the SCCC based system over its PCCC counterpart is obvious from the BER curves of Fig. 6.4 and 6.5 as compared to those seen in Chapter 5. The performance of the PCCC based system is superior in the waterfall region since the cliff appears well before it does for the SCCC based system. However, the SCCC system outperforms the PCCC system at high SNR where the PCCC develops a rather high error floor while the SCCC mitigates the appearance of such a high floor owing to fast reduction in BER with slight increase in SNR. It is also seen from the BER curves that performance degrades with increasing code rate. In the case of r=2/3 performance is particularly bad because all of the inner parity bits are punctured and thus there is effectively no inner code.

Parameter	Value
Encoder	
Encoder type	SCCC encoder
Constituent encoder type	Two identical rate 1/2 RSC encoders with outer code punctured to r = 2/3
Generator Polynomial Feedback: Feedforward:	 35 (octal) 23 (octal)
Constraint Length	5
Parent code rate	1/3
Input frame size	1024
Interleaver size	1544
Interleaver type	Spread interleaver (S = 20)
Trellis termination type	Both encoders
Puncturing	
Puncturing Period	8
Punctured code rates	Ranges between 2/3 and 1/3 (Refer Table 6.2)
Puncturing Patterns	Refer Table 6.2
Modulation	
Modulation type	BPSK
Channel	
Channel types	1. AWGN Channel 2. Fully Interleaved Rayleigh Flat Fading Channel
Decoder	
Decoder type	Max-log-MAP decoder
Number of Iterations	10
Automatic Repeat Request	
Hybrid ARQ type	Type II Hybrid ARQ
ARQ protocol	Selective repeat with assumption of error-free noise-free feedback

Table 6.1 Simulation Parameters of a hybrid RCP-SCCC-ARQ System.

	SCCC									
Encoder	Outer	Inner								
Rates	2/3	1	8/9	4/5	8/11	2/3	8/13	4/7	8/15	1/2
Puncturing Patterns	377 252	377 0	377 200	377 202	377 242	377 252	377 352	377 356	377 357	377 377

Table 6.2 Puncturing patterns in octal for each code rate.

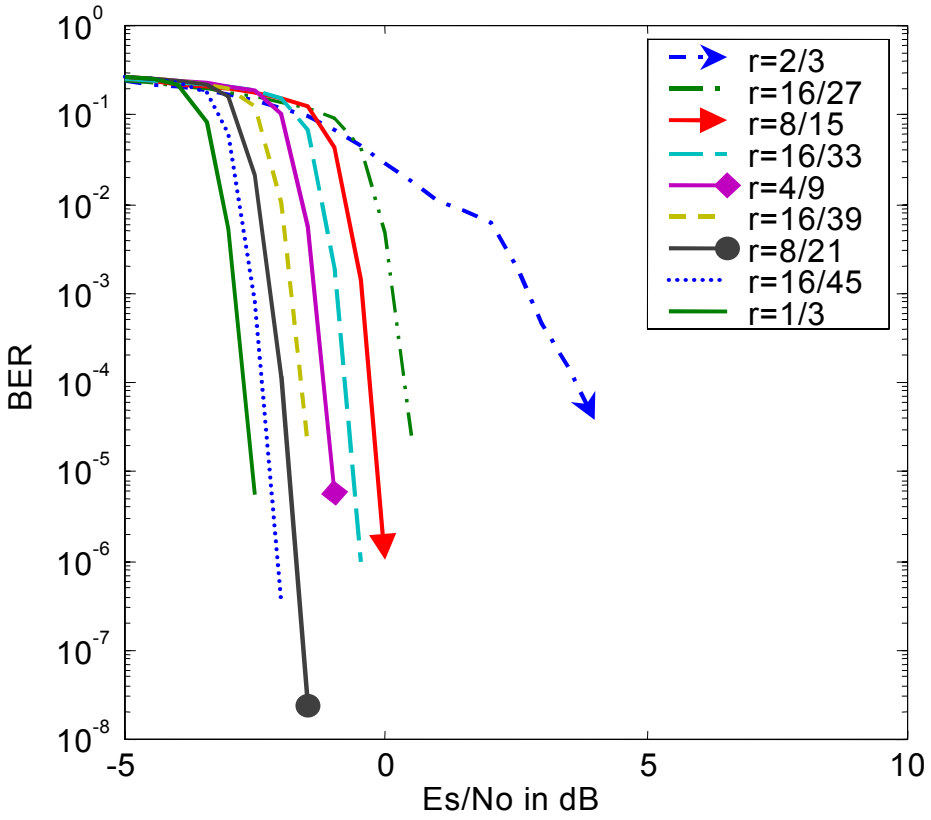


Fig. 6.4 Performance of a RCP-SCCC system in an AWGN channel (From left to right, the curves are in order of ascending code rate).

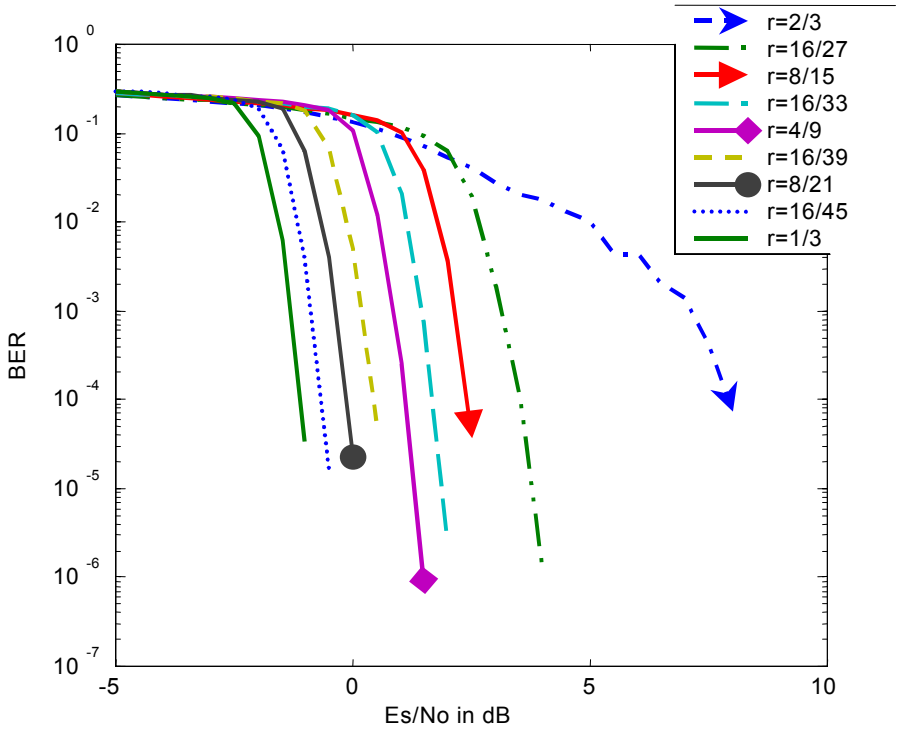


Fig. 6.5 Performance of a RCP-SCCC system in a flat-fading Rayleigh channel (From left to right, the curves are in order of ascending code rate).

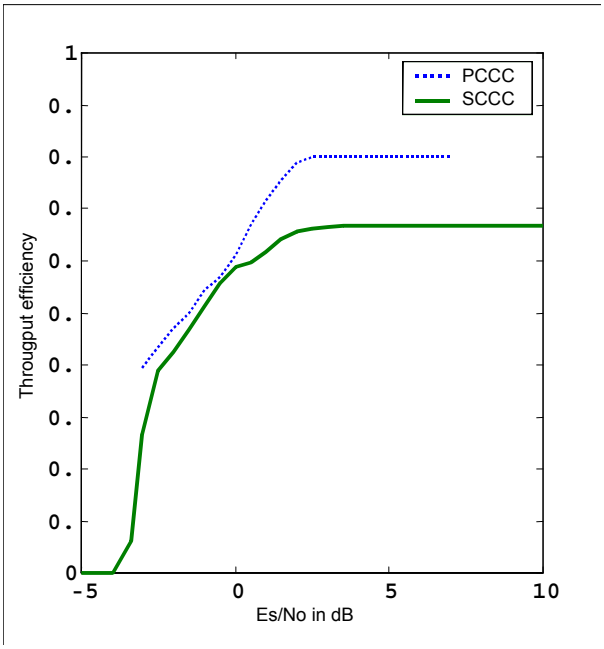


Fig. 6.6 Throughput of PCCC-system vs. SCCC-system in an AWGN channel.

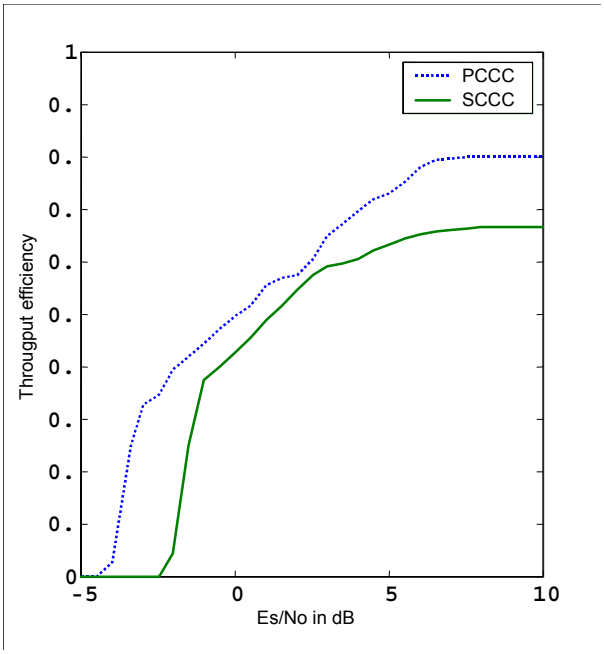


Fig. 6.7 Throughput of a PCCC-system vs. SCCC-system in a Rayleigh fading channel.

The performance of a RCP-SCCC system in a hybrid ARQ environment can be evaluated by means of the throughput of the system vs. signal to noise ratio curves. Throughput curves for both PCCC and SCCC are shown for AWGN in Fig. 6.6 and fading in Fig. 6.7. Note that in each case, the throughput of the PCCC code is superior to that of the SCCC code. This is primarily due to the fact that the cliff region for PCCC occurs at a lower SNR than for SCCC. As implied by Equation (4.6.2) in Chapter 4 which is

$$P[r = r_i] = \left(1 - FER\left(r_i, \frac{\varepsilon_s}{N_o}\right)\right)\left(\prod_{r_j > r_i} FER\left(r_j, \frac{\varepsilon_s}{N_o}\right)\right), \tag{6.4.1}$$

the throughput performance is dominated by the location of the cliff. The error floor does not significantly impact throughput, and thus the hybrid FEC-ARQ scheme is unable to exploit the lower error floors promised by SCCCs.

6.5 Chapter Summary

Although SCCCs offer lower BER floors than PCCCs, this comes at the cost of a "cliff" that occurs at a higher SNR. Like PCCCs, SCCCs can be used as part of a type-II hybrid FEC-ARQ scheme. However, because the throughput efficiency depends primarily on the location of the cliff, rather than the height of the BER floor, the throughput efficiency when using SCCCs is somewhat less than with PCCCs.

It is important to note that in this study, the puncturing patterns or code polynomials used by the SCCC code are not optimized. With better puncturing, the anticipation is that the gap between the throughput curves of the PCCC and SCCC based schemes can be closed. Finally, it should be noted that there is a tight relationship between PCCC and SCCC codes, and that in fact the PCCC code can be described as a particular type of punctured SCCC code [Wu 00a]. Thus, promising results can be obtained by using hybrid codes that combine the benefits of each type of code. The relationship between PCCC and SCCC codes is the topic of the next chapter.

Chapter 7

Equivalence of PCCC and SCCC

Purely random codes are the best codes. However, these cannot be used in a practical communication system. It has been seen from the past few chapters that a good tradeoff between randomness and structure i.e. stretching the random-like properties of the code to the extent that it still possesses enough structure to be able to decode (called pseudo-random) is instrumental in the performance improvement of the code. It is the cardinal goal of efficient designs to preserve this underlying structure but increase the disguised randomness to the extent possible.

Convolutional codes discussed as a part of the RCPC system in chapter 4 possess randomness in the manner in which these codes mathematically transform the input information to streams of parity information based on the number of outputs. However, these parity outputs possess too much structure owing to the mathematical transformation. Turbo codes (PCCCs) discussed as a component of the RCPT system in chapter 5 introduce further randomness to the system by means of the interleaver.

With SCCCs, the interleaver functions in exactly the same manner as it does in PCCCs but operates on the output of the outer encoder rather than the input directly. As has been seen in chapter 6, once the output of the outer encoder has been interleaved, the input information can no longer be traced from the output of the system. This double-encoded parity (parity of outer encoder transformed by inner encoder) increases the minimum distance of the code and thus, SCCC has a performance improvement over the PCCC.

Indeed, the performance improvement of the SCCCs can be attributed to this increase in randomness apart from all its other potential benefits. Therefore, there must be a method of constructing a PCCC from its equivalent SCCC by increasing the structure provided to the interleaver [Wu 00a]. This is the highlight of this chapter and will be explained briefly in the next couple of sections. This work is an extension of the previous work presented in [Wu 00a] and can be utilized to provide an increased number of transmission rates to a system as will be seen in this chapter.

7.1 PCCC and SCCC Encoder Outputs: A Comparison

Consider the block diagrams of the PCCC and SCCC encoders given in Fig. 7.1 and Fig. 7.2.

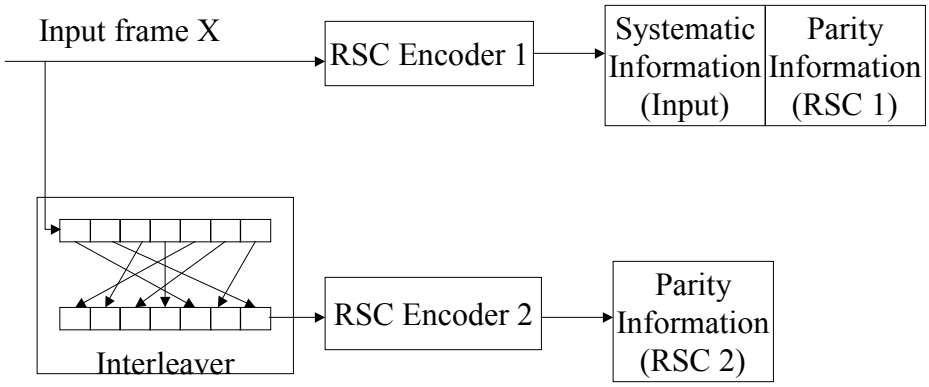


Fig. 7.1 Types of Information At Output of PCCC Encoder.

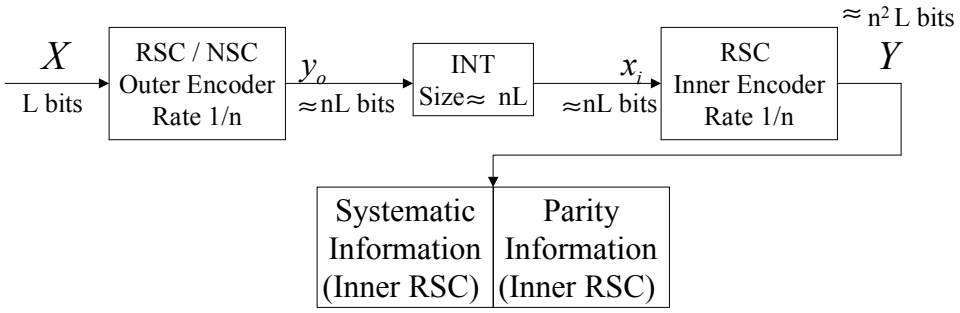


Fig. 7.2 Types of Information at Output of SCCC Encoder.

The output information from the PCCC encoder is comprised of the systematic information from RSC 1 (i.e. input frame X), parity information of RSC 1 for the input frame X, parity generated by RSC 2 for the interleaved input information ($\widetilde{X}$). The output from the SCCC encoder is comprised of systematic information from inner RSC consisting of the interleaved version ($\widetilde{X}$) of the input frame X and the interleaved version of the parity information from outer RSC for input frame X.

7.2 PCCC Encoding with an SCCC Encoder: Efficiently Structured Interleaver Design

Interleaver Restriction: If the interleaver in Fig. 7.2 is bounded by a restriction that the systematic bits from the output of the outer encoder are mapped to the first half of the interleaved frame and the parity bits are mapped to the second half of the interleaved frame, the output information from the SCCC encoder given in Fig. 7.2 will take the form

given in Fig. 7.3. Since the systematic and parity bits are multiplexed to form the output of the RSC encoder, it can be said that the interleaver must map the odd ordered bits[1] (systematic bits) in the outer encoder output frame to the first half and the even ordered bits (parity bits) to the second half of the interleaved frame.

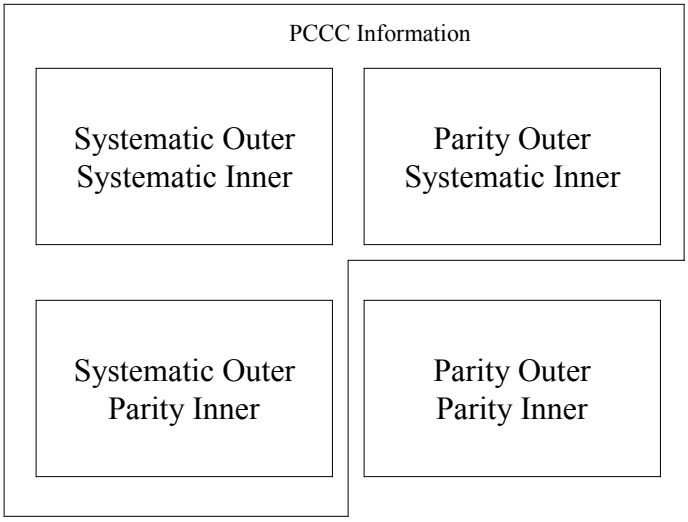


Fig. 7.3 Output Information from SCCC Encoder after Structured Interleaving.

The output information from the SCCC encoder can now be categorized as shown in Fig. 7.3. The systematic-outer systematic-inner subset of the information is the input provided to the SCCC encoder but in a permuted order. Thus, the input to the encoder can be traced from its output as a result of this structuring. Considering the output information of the PCCC encoder in Fig. 7.1, the systematic-outer systematic-inner information is equivalent to the systematic information from RSC 1. In fact, if the PCCC encoder transmits the systematic output from RSC 2 and not from RSC 1, the systematic information of the PCCC encoder and the systematic-outer systematic-inner information of the SCCC encoder will be exactly same (if the same interleaving pattern is used).

The parity-outer systematic-inner information is the interleaved version of the parity information of outer encoder, which is equivalent to the parity information from RSC 1 in Fig. 7.1. The systematic-outer parity-inner information is the parity of inner encoder for the interleaved version of the input frame, which is equivalent to the third output from the PCCC encoder i.e. parity information from RSC 2. Thus, the output of the SCCC encoder can be broken down into sections that map one-to-one with those in the PCCC output information as indicated in Fig. 7.3.

[1] Assuming indexing starts from one

Parity-outer parity-inner information is the additional information provided by the SCCC encoder. This information can be punctured before transmission to make the output of the SCCC encoder effectively that of the PCCC encoder. The SCCC decoder decodes the information as any other input frame and produces an estimate of the original input. Thus, PCCC is essentially a punctured case of the SCCC if the interleaver restriction is applied [Wu 00a] and the PCCC encoding and decoding can be performed using an SCCC codec (encoder-decoder).

7.3 Interleaver Implementation Issues

Consider the operation of the spread interleaver discussed in chapter 5. The interleaver maps adjacent bits at its input to positions with index (i) such that the distance between the two positions is S. S is upper-bounded by

$$S < \sqrt{\frac{k}{2}}, \tag{7.2.1}$$

where k is the input frame size.

S must be less than the bound but as close to it as possible for best performance. This is known as the spread interleaver since it spreads adjacent bits at least S positions apart.

Since the output of the SCCC encoder has multiplexed the systematic and parity information of the inner encoder, implementation of the interleaver restriction stated in section 7.2 actually is equivalent to spreading adjacent bits at the input of the interleaver given by (7.2.1). The spread interleaver algorithm given in chapter 5 now reduces to the following:

Step 1: Select the value of S from (7.2.1).

Step 2: Generate a frame of random indexes (i) with $i < \frac{k}{2}$ (first half) and map each systematic bit to the appropriate position in the interleaved frame.

Step 3: Generate a frame of random indexes (i) with $i > \frac{k}{2}$ (second half) and map each parity bit to the appropriate position in the interleaved frame.

Step 4: After all bits are mapped, check only the last S number of bits of the first half of interleaved frame and the first S number of bits of the second half of the interleaved frame to see if adjacent bits in the original input frame are present and if they do then check to see if they satisfy the S criterion in (7.2.1).

Step 5: If any two bit positions do not satisfy the condition, then rearrange any one of them by picking a random number $i \le \frac{k}{2} - S$ if a systematic bit and $i \ge \frac{k}{2} + S$ if a parity bit and exchange positions with the bit at this random index position.

Step 6: Repeat steps 4 and 5 (if required) for all 2*S number of bits.

This reduces the number of operations in the original spread interleaver per iteration and for large frame sizes the values map correctly within very few iterations. The rearranging procedure in steps 4 , 5 and 6 can be used if the values don't map or the process from step 1 to 3 can be repeated. Either way, there is a high probability for large frame sizes that the S positions on either side of the half way mark of the interleaved frame do not contain adjacent bits of the original frame.

Moreover, the quality of the interleaved frame (average of all the distances in the interleaved frame between adjacent bits of the original frame) generated with this interleaver restriction is exactly the same if not better than the conventional spread interleaver generated frame. This is revealed by the result in Fig. 7.4, which is a comparison plot of the SCCC with and without incorporating the interleaver restriction.

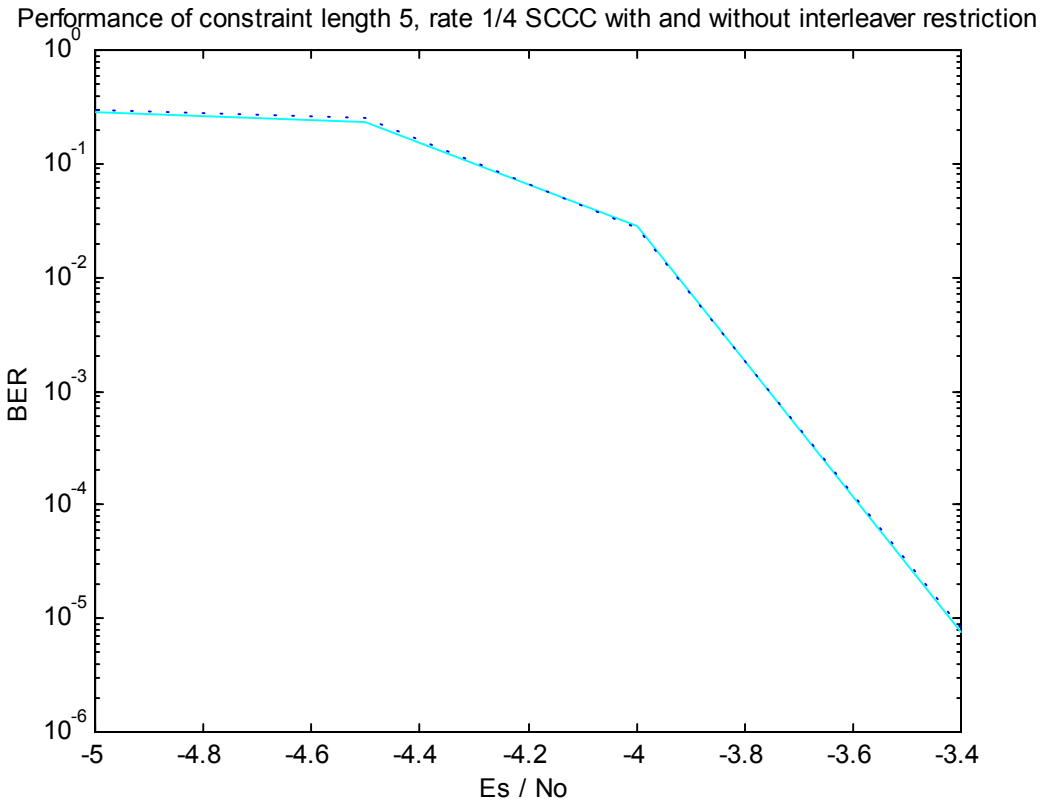


Fig. 7.4 Performance of SCCC with and without interleaver restriction.

7.4 Chapter Summary

It is obvious from Fig. 7.3 that the double-parity punctured SCCC encoder performance would match the equivalent PCCC performance. This reiterates the fact in [Wu 00a] that the PCCC is a punctured case of the SCCC and that PCCC encoding can be performed by an SCCC encoder with the decoding process remaining the same as before. Furthermore, various punctured rates can be considered using the RCP system with this SCCC encoder-decoder. By understanding the relationship between SCCC and PCCC, it is possible to design hybrid codes that benefit from the advantages of each. Furthermore, if applied to ARQ systems, it is possible to have a wider range of periodic code rates.

Chapter 8

Conclusions and Future Work

8.1 Thesis Summary

The focus of this thesis has been to improve the performance of a modern communication system by employing various forward error control techniques coupled with making the system adapt to the prevailing channel conditions. Such an adaptive system is imperative for efficient and speedy end-to-end communication especially when the channel is a hostile wireless channel.

The thesis can be categorized into three distinct sections. The first section served as a background to the thesis. The first chapter in the section began with an overview of wireless personal communications. Portions of this chapter were published by the author in IEEE Potentials [Cha 01]. The Shannon channel capacity theorem published in 1948 [Sha 48] that stemmed the need for error control coding was explained. Various forward error correcting (FEC) channel encoding and decoding techniques introduced since 1948 were then discussed. Once the need for channel coding was understood, the next step was to comprehend the nature of the channel and the various effects introduced by the channel in the stream of data passed through it. It was seen in chapter 2 that wired channels are less hostile than their wireless counterpart owing to the presence of fading in the latter. Signal degradation in the wireless channel is caused by a variety of factors like path loss due to the separation between the transmitter and the receiver and shadowing due to obstructions (called large-scale fading) and rapid fluctuations in signal strength due to motion over short distances (called small-scale fading). The manifestations of each type of fading and their effects were explained in detail. This chapter concluded by reiterating the importance of channel coding with the simulation of a BPSK system (without channel coding) in an AWGN and Rayleigh (small-scale) fading channel. It was seen that the performance of the BPSK system in an AWGN channel was 13 dB better than that in a Rayleigh fading channel at a BER of 10^{-3} (a benchmark for voice communications). It was also seen that the performance in both these channels could be improved with efficient channel coding. The third chapter introduced the concept of automatic repeat request over a feedback channel in an attempt to relay the channel state information from the receiver back to the transmitter. The advantage of ARQ of providing high system reliability can be coupled with that of FEC codes of achieving high throughput even in poor channel conditions to form a hybrid FEC-ARQ system. Multiple hybrid FEC-ARQ strategies were introduced based on the manner in which retransmission requests are

treated. Finally, this chapter explained the channel model and the hybrid FEC-ARQ model employed in the thesis.

The second section of the thesis was devoted to the development of various FEC codes used in this thesis. The three chapters in this section were very similar in format. The first part of each chapter introduced a particular type of FEC code. Later, each FEC encoder and the corresponding decoder were implemented in software (Matlab / C) and the implementation issues were discussed. Each FEC code's performance when incorporated in a hybrid FEC-ARQ system was simulated over an AWGN and fading channel and the results were plotted. More specifically, the fourth chapter of the thesis focused on non-systematic convolutional (NSC) codes while the fifth and the sixth chapters focused on the more advanced FEC techniques of turbo (parallel concatenated convolutional) codes and the serial concatenated convolutional codes respectively. The concept of using SCCC in a hybrid FEC-ARQ system was introduced for the first time in this thesis and forms a major contribution of the thesis. A paper titled 'Hybrid ARQ using Serial Concatenated Convolutional Codes over Fading Channels' based on these concepts was published in the IEEE Vehicular Technology Conference in Greece in May 2001 [Cha 01a].

Finally, the third section of the thesis discussed the development of hybrid PCCC / SCCC codes. The equivalence of the PCCC and the SCCC codes was shown and it was seen that an SCCC encoder could be modified to produce PCCC output information. The same decoder could then be used to decode the information encoded by the SCCC or the PCCC. Such hybrid codes benefit from the advantages of each type of component code. Furthermore, by appropriate puncturing it should be possible to create a code that lies somewhere between the PCCC and SCCC. The modification required is simple but a very efficient one because the SCCC code performs exactly the same even in the presence of the modification.

8.2 Conclusions

The following conclusions can be drawn from this research:

I. Rate Compatibility and Puncturing:
   1. Rate compatibility criterion reduces the amount of overhead in transmission and utilizes the channel resources efficiently.
   2. Puncturing and rate-compatibility provide an efficient means of implementing a variable rate error control system.

II. Hybrid FEC-ARQ Systems:
   1. ARQ systems obtain high system reliability by avoiding the delivery of erroneous packets to the user.
   2. FEC systems provide high throughput provided that the code rate matches the prevailing channel conditions.

3. A hybrid FEC-ARQ system combines forward error correction with ARQ thereby reaping benefits from both schemes.
4. Type-II hybrid FEC-ARQ is an efficient scheme to make the system adapt to varying channel conditions, thereby ensuring that the rate of the FEC code matches the channel.
5. Type-II hybrid RCPC-ARQ systems provide vast improvement in throughput performance over conventional NSC coded systems because of the presence of multiple rates of transmission.
6. Turbo codes have been shown to perform very close to channel capacity. Thus, their BER performance is far better than their NSC code counterpart.
7. As expected, a hybrid RCPT-ARQ system also provides large improvement in throughput performance over conventional turbo coded systems.
8. Like PCCCs, SCCCs can be used as part of a type II hybrid FEC-ARQ scheme.
9. SCCCs offer lower BER floors than PCCCs.
10. However, this comes at the cost of the "cliff" occurring at higher SNR.
11. Results show that since the cliff region is the predominant contributor towards throughput efficiency rather than the height of the BER floor, SCCCs have lower throughput efficiency than PCCCs.

III. Hybrid PCCC / SCCC:

1. It is possible for the SCCC to perform PCCC encoding by proper interleaving and puncturing.
2. Hybrid codes can be generated which can perform both types of concatenated convolutional encoding with a single encoder.
3. Hybrid codes require no modification in the SCCC decoding algorithm.
4. However, an interesting result is that the SCCC performance with and without the interleaver restriction remains exactly the same.

8.3 Future Work

1. Puncturing patterns for SCCC can be optimized. Optimized puncturing patterns can reduce the gap between the throughput efficiencies of PCCC and SCCC.
2. The performance of a practical hybrid ARQ system with noise in the feedback channel can be evaluated.
3. The concept of using an SCCC encoder to generate a PCCC code can be used within a hybrid FEC-ARQ system.
4. Code designs that lie between PCCC and SCCC can be achieved. These codes can reap benefits of the PCCC and the SCCC.

Appendix A

Maximum A Posteriori (MAP) Algorithm

The Viterbi algorithm is a maximum-likelihood decoding algorithm that minimizes the probability of sequence or word error rather than minimizing the symbol or bit error probability. The BCJR algorithm named after its founders is a symbol-by-symbol maximum a posteriori (MAP) algorithm, which minimizes the bit error probability. The Viterbi algorithm picks the maximum-likelihood path going into each state of the trellis section in consideration while the MAP algorithm computes the log-likelihood ratios of all state transitions caused due to an input 1 and all state transitions caused due an input 0 given a particular trellis section.

In other words, the MAP algorithm is the solution to the more general problem of estimating the a posteriori probabilities (APP) of the states and transitions of a Markov source observed through a noisy memoryless channel. The source of information is a discrete-time finite-state Markov process. The total number of states of this Markov process is M, which are indexed by integer m. S_t is the state of the source at time t and X_t is the corresponding output of the source. The state transitions of the Markov source are governed by the transition probabilities given by [Bah 74]

$$p_t(m|m') = P_r\{S_t = m | S_{t-1} = m'\}, \tag{A.1}$$

and the output by the probabilities

$$q_t(X|m', m) = P_r\{X_t = X | S_{t-1} = m'; S_t = m\}, \tag{A.2}$$

where X belongs to some finite discrete alphabet.

The output of the source X ($X_1^\tau = X_1, X_2 \ldots\ldots X_\tau$) is input to the channel whose output is Y ($Y_1^\tau = Y_1, Y_2 \ldots\ldots Y_\tau$). The objective of the decoder is to examine Y_1^τ and estimate the APP of the states and transitions of the Markov source, i.e. the conditional probabilities

$$P_r\{S_t = m | Y_1^\tau\} = \frac{P_r\{S_t = m; Y_1^\tau\}}{P_r\{Y_1^\tau\}} \tag{A.3}$$

and

$$P_r\left\{S_{t-1}=m';S_t=m|Y_1^\tau\right\}=\frac{P_r\left\{S_{t-1}=m';S_t=m;Y_1^\tau\right\}}{P_r\left\{Y_1^\tau\right\}}. \quad \text{(A.4)}$$

Consider the following two joint probabilities to compute the APP in (A.3) and (A.4)

$$\lambda_t(m)=P_r\left\{S_t=m;Y_1^\tau\right\} \quad \text{(A.5)}$$

$$\sigma_t(m',m)=P_r\left\{S_{t-1}=m';S_t=m;Y_1^\tau\right\}. \quad \text{(A.6)}$$

(A.5) and (A.6) can be divided by $P_r\left\{Y_1^\tau\right\}$, which is a constant for a given Y_1^τ, to obtain the APP in (A.3) and (A.4).

The following probability functions are defined to compute $\lambda_t(m)$ and $\sigma_t(m',m)$.

$$\alpha_t(m)=P_r\left\{S_t=m;Y_1^\tau\right\} \quad \text{(A.7)}$$

$$\beta_t(m)=P_r\left\{Y_{t+1}^\tau|S_t=m\right\} \quad \text{(A.8)}$$

$$\gamma_t(m',m)=P_r\left\{S_t=m;Y_t|S_{t-1}=m'\right\}. \quad \text{(A.9)}$$

Now

$$\begin{aligned}\lambda_t(m)&=P_r\left\{S_t=m;Y_1^\tau\right\}\\&=P_r\left\{S_t=m';Y_1^\tau;Y_{t+1}^\tau\right\}\end{aligned}$$

Using Bayes Rule,

$$=P_r\left\{S_t=m;Y_1^\tau\right\}P_r\left\{Y_{t+1}^\tau|S_t=m;Y_1^\tau\right\}$$

From (A.7) and from the Markov property that if S_t is known, events after time t do not depend on Y_1^τ,

$$=\alpha_t(m)P_r\left\{Y_{t+1}^\tau|S_t=m;Y_1^\tau\right\}$$

From (A.8),

$$=\alpha_t(m)\beta_t(m) \quad \text{(A.10)}$$

Similarly,

$$\begin{aligned}\sigma_t(m',m)&=P_r\left\{S_{t-1}=m';S_t=m;Y_1^\tau\right\}\\&=P_r\left\{S_{t-1}=m';S_t=m;Y_1^\tau;Y_{t+1}^\tau\right\}\\&=P_r\left\{S_{t-1}=m';S_t=m;Y_1^{t-1};Y_t;Y_{t+1}^\tau\right\}\\&=P_r\left\{S_{t-1}=m';Y_1^{t-1}\right\}P_r\left\{S_t=m;Y_t;Y_{t+1}^\tau|S_{t-1}=m';Y_1^{t-1}\right\}\\&=P_r\left\{S_{t-1}=m';Y_1^{t-1}\right\}P_r\left\{S_t=m;Y_t;Y_{t+1}^\tau|S_{t-1}=m'\right\}\\&=P_r\left\{S_{t-1}=m';Y_1^{t-1}\right\}P_r\left\{S_t=m;Y_t|S_{t-1}=m'\right\}P_r\left\{Y_{t+1}^\tau|S_t=m;Y_t\right\}\end{aligned}$$

$$= P_r\{S_{t-1} = m'; Y_1^{t-1}\} P_r\{S_t = m; Y_t | S_{t-1} = m'\} P_r\{Y_{t+1}^{\tau} | S_t = m\}$$
$$= \alpha_{t-1}(m')\gamma_t(m', m)\beta_t(m). \quad \text{(A.11)}$$

Now,

$$\alpha_t(m) = \sum_{m'=0}^{M-1} P_r\{S_{t-1} = m'; S_t = m; Y_1^t\}$$
$$= \sum_{m'} P_r\{S_{t-1} = m'; Y_1^{t-1}\} P_r\{S_t = m; Y_t | S_{t-1} = m'\}$$
$$= \sum_{m'} \alpha_{t-1}(m')\gamma_t(m', m). \quad \text{(A.12)}$$

Therefore, $\alpha_t(m)$ can be recursively calculated while moving forward in the trellis. The boundary conditions at t=0 are

$$\alpha_0(0) = 1 \text{ and } \alpha_0(m) = 0, \qquad \text{for } m \neq 0. \quad \text{(A.13)}$$

(A.13) is the result of the fact that the trellis always begins in state 0.
Similarly,

$$\beta_t(m) = \sum_{m'} \beta_{t+1}(m').\gamma_{t+1}(m, m'). \quad \text{(A.14)}$$

and the boundary conditions are

$$\beta_\tau(0) = 1 \text{ and } \beta_\tau(m) = 0, \qquad \text{for } m \neq 0 \quad \text{(A.15)}$$

Therefore, $\beta_t(m)$ can be recursively calculated while tracing back from the end of the trellis. (A.15) is the result of the fact that the trellis always ends in state 0 assuming that the trellis is terminated with tail bits.
Now

$$\gamma_t(m', m) = \sum_{m'} p_t(m|m')q_t(X|m', m)R(Y_t|X), \quad \text{(A.16)}$$

where R(.|.) is the transition probability of the discrete memoryless channel.

The symbol-by-symbol MAP decoder decides $X_k = +1$ if $P(X_k = +1|Y) > P(X_k = -1|Y)$ and it decides $X_k = -1$ otherwise. More precisely, the decision $\widetilde{X}_k$ is given by

$$\widetilde{X}_k = sign[L(X_k)],$$

where $L(X_k)$ is the log APP ratio defined as

$$L(X_k) \cong \log\left(\frac{P(X_k = +1|y)}{P(X_k = -1|y)}\right).$$

Incorporating the code's trellis seen from (A.4), (A.6) and (A.11), this can be written as

$$L(X_k) = \log\left(\frac{\sum_{S^+} \alpha_{k-1}(m')\gamma_k(m', m)\beta_k(m) / p(Y)}{\sum_{S^-} \alpha_{k-1}(m')\gamma_k(m', m)\beta_k(m) / p(Y)}\right), \quad \text{(A.17)}$$

where S^+ is the set of state pairs (s', s) corresponding to all state transitions caused by data input =1 and S^- is the set of state pairs (s', s) corresponding to all state transitions caused by data input = 0.

The factor $p(y)$ in (A.17) is used for normalization purposes and hence can be neglected for the purpose of analysis.

The right hand side of (A.17) can thus be expressed as

$$\log\left(\sum_{S^+}\alpha_{k-1}(m')\gamma_k(m',m)\beta_k(m)\right)-\log\left(\sum_{S^-}\alpha_{k-1}(m')\gamma_k(m',m)\beta_k(m)\right) \qquad \text{(A.18)}$$

The Jacobian algorithm given below illustrates the addition operation in the log-domain.

$$\log(e^x+e^y)=\max(x,y)+\log(1+\exp\{-|y-x|\}) \qquad \text{(A.19)}$$

(A.19) is defined as a $\max^*$ function where

$$\max^*(x,y)=\max(x,y)+\log(1+\exp\{-|y-x|\}) \qquad \text{(A.20)}$$

Using shorthand notation for $\alpha_{k-1}(m')$, $\beta_k(m)$ and $\gamma_k(m',m)$, and (A.20), (A.18) can be expressed as

$$\max_{S^+}^*(\log(\alpha\gamma\beta))-\max_{S^-}^*(\log(\alpha\gamma\beta))$$

Therefore,

$$L(X_k)=\max_{S^+}^*(\log\alpha+\log\gamma+\log\beta)-\max_{S^-}^*(\log\alpha+\log\gamma+\log\beta) \qquad \text{(A.21)}$$

Also, (A.12) now becomes,

$$\log\alpha_t=\max_{m'}^*(\log\alpha_{t-1}+\log\gamma_t) \qquad \text{(A.22)}$$

Similarly, (A.14) now becomes,

$$\log\beta_t=\max_{m'}^*(\log\beta_{t+1}+\log\gamma_{t+1}) \qquad \text{(A.23)}$$

Equations (A.21), (A.22) and (A.23) are used in the log-MAP decoding of convolutional codes. Since each of the equations are used recursively and the $\max^*$ function is used in each of these equations, the computation time of the $\max^*$ function is significant. Thus, there is a need to simplify the $\max^*$ function to reduce computational complexity and

time. Various practical MAP-decoding strategies such as the max-log-MAP, constant-log-MAP and the linear-log-MAP are discussed in Chapter 5 in section 5.3.2.

Bibliography

[Bah 74] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. 20, pp. 284-287, Mar. 1974.

[Ben 98] S. Benedetto, D. Divsalar, D. Montorsi, and F. Pollara, "Serial concatenation of interleaved codes: Performance analysis, design and iterative decoding," *IEEE Trans. Inform. Theory*, vol. 44, pp. 909-926, May 1998.

[Ber 93] C. Berrou, A. Glavieux, and P. Thitimasjshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes(1)," in *Proc., IEEE Int. Conf. on Commun.,* (Geneva, Switzerland), May 1993, pp. 1064-1070.

[Bos 60] R.C. Bose and D. K. Ray-Chaudhuri, "On a class of error correcting binary group codes," *Information and Control*, vol. 8, pp. 300-304, 1960.

[Cha 85] D. Chase, "Code combining – A maximum-likelihood decoding approach for combining an arbitrary number of noisy packets," *IEEE Transactions on Communications*, vol. COM-33, pp. 385-393, May 1985.

[Cha 01] N. Chandran and M.C. Valenti, "Three generations of wireless cellular systems," *IEEE Potentials*, vol. 20, no. 1, pp. 32-35, Feb/March 2001.

[Cha 01a] N. Chandran and M.C. Valenti, "Hybrid ARQ using serial concatenated convolutional codes over fading channels," in *Proc. IEEE Vehicular Tech. Conf. (VTC),* (Rhodes, Greece), May 2001.

[Cos 83] S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and Applications*, Englewood Cliffs, NJ: Prentice Hall Inc., 1983.

[Eli 55] P. Elias, "Coding for noisy channels," *IRE Conv. Record*, vol. 4, pp. 37-47, 1955.

[For 66] G. D. Forney, *Concatenated Codes*. Cambridge, MA: MIT Press, 1966.

[Fre 64] R. J. Benice and A. H. Frey, "An analysis of retransmission systems," *IEEE Transactions on Communications Tech.*, vol. COM-12, pp. 135-145, Dec. 1964.

[Goo 97] D.J. Goodman, *Wireless Personal Communication Systems.* Addison Wesley, 1997.

[Gro 98] W. J. Gross and P. G. Gulak, "Simplified MAP algorithm suitable for implementation of turbo decoders," *IEEE Electronic Letters*, vol. 34, pp. 1577-1578, Aug. 1998.

[Hag 88] J. Hagenauer, "Rate Compatible punctured convolutional codes (RCPC-codes) and their application," *IEEE Trans. Commun.*, vol. 36, pp. 389-400, Apr. 1988.

[Ham 50] R.W. Hamming, "Error detecting and correcting codes," *Bell Sys. Tech. J.*, vol. 29, pp. 147-160, 1950.

[Man 74] D. M. Mandelbaum, "An adaptive-feedback coding scheme using incremental redundancy," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 388-389, May 1974.

[Mil 98] B. Miller, "Satellites free the mobile phone", *IEEE Spectrum*, vol. 35, no. 3, Mar. 1998, pp. 26-35.

[Mul 54] D. E. Muller, "Application of boolean algebra to switching circuit design," *IEEE Trans. on Computers*, vol. 3, pp. 6-12, Sept. 1954.

[Oci 00] O.F. Acikel and W.E. Ryan, "Punctured high rate SCCCs for BPSK/QPSK channels," in *Proc., IEEE Int. Conf. on Commun.*, (New Orleans, LA), June 2000.

[Oli 99] M.W. Oliphant, "The mobile phone meets the Internet", *IEEE Spectrum*, vol. 6, no. 8, pp. 20-28, Aug. 1999.

[Pah 95] K. Pahlavan and A.H. Levesque, *Wireless Information Networks.* Wiley, 1995.

[Rap 96] T.S. Rappaport, *Wireless Communications: Principles and Practice.* Upper Saddle River, NJ: Prentice Hall PTR, 1996.

[Rob 97] P. Robertson, P. Hoeher, and E. Villebrun, "Optimal and sub-optimal maximum a posteriori algorithms suitable for turbo decoding," *European Trans. on Telecommun*., vol. 8, no. 2, pp. 119-125, Mar./Apr. 1997.

[Row 00] D. N. Rowitch and L. B. Milstein, "On the performance of hybrid FEC/ARQ systems using rate compatible punctured turbo (RCPT) codes," *IEEE Trans. Commun.*, vol. 48, pp. 948-959, June 2000.

[Sha 48] C.E. Shannon, “A mathematical theory of communication,” *Bell Sys. Tech. J.*, vol. 27, pp. 379-423 and 623-656, 1948.

[Sin 77] P. Sindhu, “Retransmission error control with memory,” *IEEE Transactions on Communications*, vol. COM-25, pp. 473-479, May 1977.

[Skl 97] B. Sklar, “Rayleigh fading channels in mobile digital communication systems .I. Characterization,” *IEEE Communications Magazine*, vol. 35, no. 9, pp. 148-155, Sept. 1997.

[Val 99] M. C. Valenti, “Iterative detection and decoding for wireless communications, PhD. Dissertation,” Virginia Tech. Blacksburg, VA, July 1999.

[Val 01] M. C. Valenti, “An efficient software radio implementation of the UMTS turbo codec,” in *Proc., IEEE PIMRC*, (San Diego, CA), Oct. 2001, to appear.

[Vit 67] A. J. Viterbi, “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm,” *IEEE Trans. Inform. Theory*, vol. 13, pp. 260-269, Apr. 1967.

[Wic 95] S. Wicker, *Error Control Systems for Digital Communications and Storage.* Englewood Cliffs, NJ: Prentice Hall Inc., 1995.

[Woz 60] J. M. Wozencraft and M. Horstein, “Digitalised communication over two-way channels,” *The Fourth London Symposium on Information Theory*, London, England, Aug. 29 – Sept. 3, 1960.

[Wu 00] Y. Wu, “Implementation of parallel and serial concatenated codes, PhD. Dissertation,” Virginia Tech. Blacksburg, VA, Apr 2000.

[Wu 00a] Y. Wu and M.C. Valenti, “An ARQ technique using related parallel and serial concatenated convolutional codes," in *Proc. Int. Conf. on Commun.* (ICC) 2000, (New Orleans, LA), June 2000.

[Zen 99] M. Zeng, A. Annamalai, and V.K. Bhargava, “Recent advances in cellular wireless communications”, *IEEE Communications Magazine,* vol. 37, no. 9, pp. 128-138, Sept. 1999.

Vita

Naveen Chandran was born in Chennai, India on June 22, 1978. He received the Bachelors degree (B.E.) in Electronics and Telecommunication Engineering with first class from the University of Bombay, India, in 1999. During his undergraduate degree, he was an intern at Nelco Ltd., Bombay, for a year. In the capacity of a trainee research engineer, he assisted the company in the development of a large-scale supervisory control and data acquisition (SCADA) system for automating the power plant of a renowned oil refinery located in northern India. He joined the department of Computer Science and Electrical Engineering (CSEE) at West Virginia University in August 1999 for a Masters of Science (M.S) program in Electrical Engineering. His research interests are in wireless data communications, signal processing, adaptive error control systems, and next generation cellular and satellite communications. He can be reached through email at naveen_chandran@ieee.org. A detailed curriculum vita is available upon request.