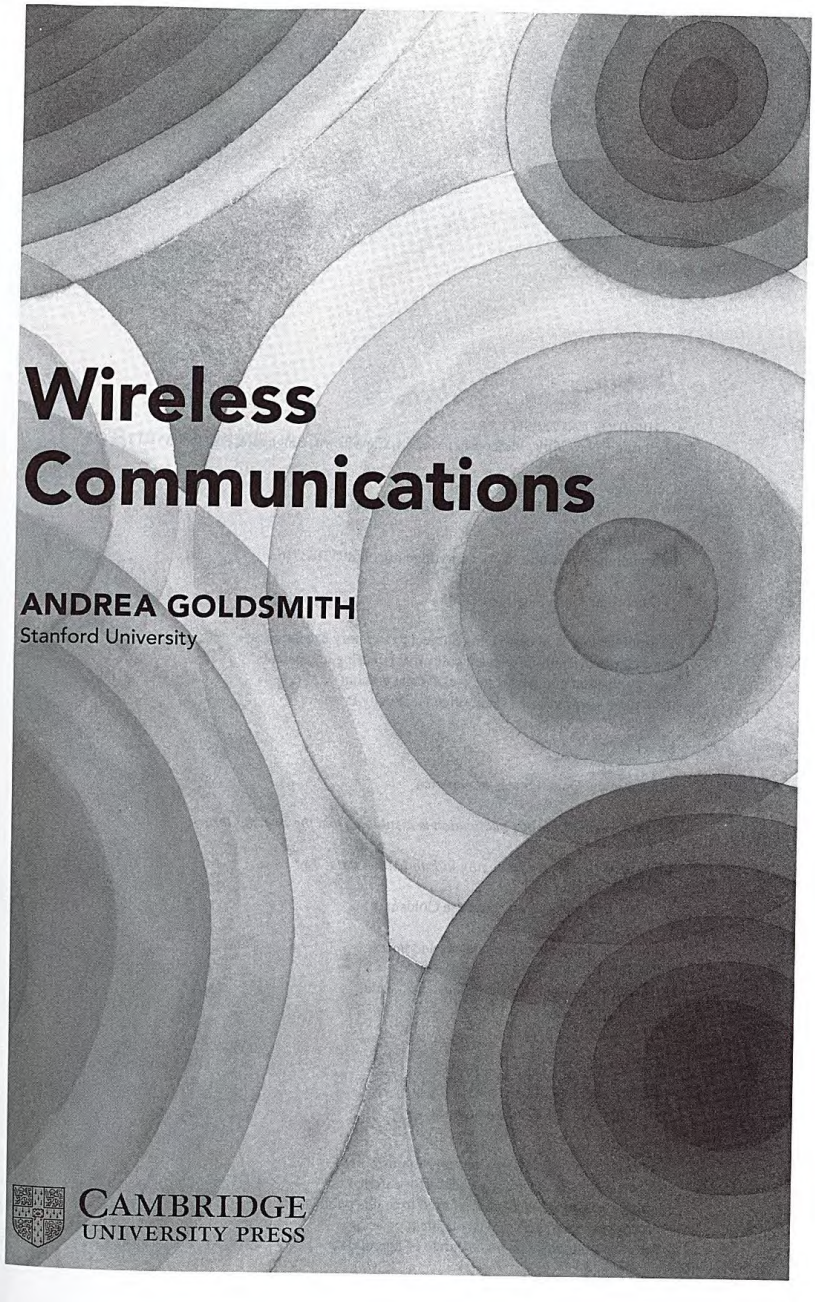


UMass Dartmouth

CAMBRIDGE UNIVERSITY PRESS
Cambridge, New York, Melbourne, Madrid, Cape Town, Singapore, São Paulo

Cambridge University Press
40 West 20th Street, New York, NY 10011-4211, USA

www.cambridge.org
Information on this title: www.cambridge.org/9780521837163

© Cambridge University Press 2005

This publication is in copyright. Subject to statutory exception
and to the provisions of relevant collective licensing agreements,
no reproduction of any part may take place without
the written permission of Cambridge University Press.

First published 2005

Printed in the United States of America

A catalog record for this publication is available from the British Library.

Library of Congress Cataloging in Publication data
Goldsmith, Andrea
Wireless communications / Andrea Goldsmith.
p. cm.
Includes bibliographical references and index.
ISBN-13: 978-0-521-83716-3
1. Wireless communication systems. I. Title.

TK5103.2.G65 2005
621.382 – dc22 2005047075

ISBN-13 978-0-521-83716-3 hardback
ISBN-10 0-521-83716-2 hardback

Cambridge University Press has no responsibility for
the persistence or accuracy of URLs for external or
third-party Internet Web sites referred to in this publication
and does not guarantee that any content on such
Web sites is, or will remain, accurate or appropriate.

from the Isle of Wight to a tugboat 18 miles away, and radio communications was born. Radio technology advanced rapidly to enable transmissions over larger distances with better quality, less power, and smaller, cheaper devices, thereby enabling public and private radio communications, television, and wireless networking.

Early radio systems transmitted analog signals. Today most radio systems transmit digital signals composed of binary bits, where the bits are obtained directly from a data signal or by digitizing an analog signal. A digital radio can transmit a continuous bit stream or it can group the bits into packets. The latter type of radio is called a *packet radio* and is often characterized by bursty transmissions: the radio is idle except when it transmits a packet, although it may transmit packets continuously. The first network based on packet radio, ALOHANET, was developed at the University of Hawaii in 1971. This network enabled computer sites at seven campuses spread out over four islands to communicate with a central computer on Oahu via radio transmission. The network architecture used a star topology with the central computer at its hub. Any two computers could establish a bi-directional communications link between them by going through the central hub. ALOHANET incorporated the first set of protocols for channel access and routing in packet radio systems, and many of the underlying principles in these protocols are still in use today. The U.S. military was extremely interested in this combination of packet data and broadcast radio. Throughout the 1970s and early 1980s the Defense Advanced Research Projects Agency (DARPA) invested significant resources to develop networks using packet radios for tactical communications in the battlefield. The nodes in these ad hoc wireless networks had the ability to self-configure (or reconfigure) into a network without the aid of any established infrastructure. DARPA's investment in ad hoc networks peaked in the mid 1980s, but the resulting systems fell far short of expectations in terms of speed and performance. These networks continue to be developed for military use. Packet radio networks also found commercial application in supporting wide area wireless data services. These services, first introduced in the early 1990s, enabled wireless data access (including email, file transfer, and Web browsing) at fairly low speeds, on the order of 20 kbps. No strong market for these wide area wireless data services ever really materialized, due mainly to their low data rates, high cost, and lack of "killer applications". These services mostly disappeared in the 1990s, supplanted by the wireless data capabilities of cellular telephones and wireless local area networks (WLANs).

The introduction of wired Ethernet technology in the 1970s steered many commercial companies away from radio-based networking. Ethernet's 10-Mbps data rate far exceeded anything available using radio, and companies did not mind running cables within and between their facilities to take advantage of these high rates. In 1985 the Federal Communications Commission (FCC) enabled the commercial development of wireless LANs by authorizing the public use of the Industrial, Scientific, and Medical (ISM) frequency bands for wireless LAN products. The ISM band was attractive to wireless LAN vendors because they did not need to obtain an FCC license to operate in this band. However, the wireless LAN systems were not allowed to interfere with the primary ISM band users, which forced them to use a low power profile and an inefficient signaling scheme. Moreover, the interference from primary users within this frequency band was quite high. As a result, these initial wireless LANs had very poor performance in terms of data rates and coverage. This poor performance – coupled with concerns about security, lack of standardization, and high cost

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 3 of 47

gradually evolved to support data services such as email, Internet access, and short messaging. Unfortunately, the great market potential for cellular phones led to a proliferation of second-generation cellular standards: three different standards in the United States alone, other standards in Europe and Japan, and all incompatible. The fact that different cities have different incompatible standards makes roaming throughout the United States and the world with only one cellular phone standard impossible. Moreover, some countries have initiated service for third-generation systems, for which there are also multiple incompatible standards. As a result of this proliferation of standards, many cellular phones today are multimode: they incorporate multiple digital standards to faciliate nationwide and worldwide roaming and possibly the first-generation analog standard as well, since only this standard provides universal coverage throughout the United States.

Satellite systems are typically characterized by the height of the satellite orbit: low-earth orbit (LEOs at roughly 2000 km altitude), medium-earth orbit (MEOs, 9000 km), or geosynchronous orbit (GEOs, 40,000 km). The geosynchronous orbits are seen as stationary from the earth, whereas satellites with other orbits have their coverage area change over time. The concept of using geosynchronous satellites for communications was first suggested by the science-fiction writer Arthur C. Clarke in 1945. However, the first deployed satellites – the Soviet Union's *Sputnik* in 1957 and the NASA/Bell Laboratories' *Echo-1* in 1960 – were not geosynchronous owing to the difficulty of lifting a satellite into such a high orbit. The first GEO satellite was launched by Hughes and NASA in 1963; GEOs then dominated both commercial and government satellite systems for several decades.

Geosynchronous satellites have large coverage areas, so fewer satellites (and dollars) are necessary to provide wide area or global coverage. However, it takes a great deal of power to reach the satellite, and the propagation delay is typically too large for delay-constrained applications like voice. These disadvantages caused a shift in the 1990s toward lower-orbit satellites [2; 3]. The goal was to provide voice and data service competitive with cellular systems. However, the satellite mobile terminals were much bigger, consumed much more power, and cost much more than contemporary cellular phones, which limited their appeal. The most compelling feature of these systems is their ubiquitous worldwide coverage, especially in remote areas or third-world countries with no landline or cellular system infrastructure. Unfortunately, such places do not typically have large demand or the resources to pay for satellite service either. As cellular systems became more widespread, they took away most revenue that LEO systems might have generated in populated areas. With no real market left, most LEO satellite systems went out of business.

A natural area for satellite systems is broadcast entertainment. Direct broadcast satellites operate in the 12-GHz frequency band. These systems offer hundreds of TV channels and are major competitors to cable. Satellite-delivered digital radio has also become popular. These systems, operating in both Europe and the United States, offer digital audio broadcasts at near-CD quality.

1.2 Wireless Vision

The vision of wireless communications supporting information exchange between people or devices is the communications frontier of the next few decades, and much of it already

exists in some form. This vision will allow multimedia communication from anywhere in the world using a small handheld device or laptop. Wireless networks will connect palmtop, laptop, and desktop computers anywhere within an office building or campus, as well as from the corner cafe. In the home these networks will enable a new class of intelligent electronic devices that can interact with each other and with the Internet in addition to providing connectivity between computers, phones, and security/monitoring systems. Such "smart" homes can also help the elderly and disabled with assisted living, patient monitoring, and emergency response. Wireless entertainment will permeate the home and any place that people congregate. Video teleconferencing will take place between buildings that are blocks or continents apart, and these conferences can include travelers as well – from the salesperson who missed his plane connection to the CEO off sailing in the Caribbean. Wireless video will enable remote classrooms, remote training facilities, and remote hospitals anywhere in the world. Wireless sensors have an enormous range of both commercial and military applications. Commercial applications include monitoring of fire hazards, toxic waste sites, stress and strain in buildings and bridges, carbon dioxide movement, and the spread of chemicals and gasses at a disaster site. These wireless sensors self-configure into a network to process and interpret sensor measurements and then convey this information to a centralized control location. Military applications include identification and tracking of enemy targets, detection of chemical and biological attacks, support of unmanned robotic vehicles, and counterterrorism. Finally, wireless networks enable distributed control systems with remote devices, sensors, and actuators linked together via wireless communication channels. Such systems in turn enable automated highways, mobile robots, and easily reconfigurable industrial automation.

The various applications described here are all components of the wireless vision. So then what, exactly, is wireless communications? There are many ways to segment this complex topic into different applications, systems, or coverage regions [4]. Wireless applications include voice, Internet access, Web browsing, paging and short messaging, subscriber information services, file transfer, video teleconferencing, entertainment, sensing, and distributed control. Systems include cellular telephone systems, wireless LANs, wide area wireless data systems, satellite systems, and ad hoc wireless networks. Coverage regions include in-building, campus, city, regional, and global. The question of how best to characterize wireless communications along these various segments has resulted in considerable fragmentation in the industry, as evidenced by the many different wireless products, standards, and services being offered or proposed. One reason for this fragmentation is that different wireless applications have different requirements. Voice systems have relatively low data-rate requirements (around 20 kbps) and can tolerate a fairly high probability of bit error (bit error rates, or BERs, of around 10^{-3}), but the total delay must be less than about 100 ms or else it becomes noticeable to the end user.[1] On the other hand, data systems typically require much higher data rates (1–100 Mbps) and very small BERs (a BER of 10^{-8} or less, and all bits received in error must be retransmitted) but do not have a fixed delay requirement. Real-time video systems have high data-rate requirements coupled with the same delay constraints as voice systems,

[1] Wired telephones have a delay constraint of ~30 ms. Cellular phones relax this constraint to ~100 ms, and voice over the Internet relaxes the constraint even further.

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 5 of 47

spectrum sharing are very complex, and the decision of which technique is best for a given system and operating environment is never straightforward.

Efficient cellular system designs are *interference limited* – that is, the interference dominates the noise floor, since otherwise more users could be added to the system. As a result, any technique to reduce interference in cellular systems leads directly to an increase in system capacity and performance. Some methods for interference reduction in use today or proposed for future systems include cell sectorization, directional and smart antennas, multiuser detection, and dynamic resource allocation. Details of these techniques will be given in Chapter 15.

The first-generation (1G) cellular systems in the United States, called the Advance Mobile Phone Service (AMPS), used FDMA with 30-kHz FM-modulated voice channels. The FCC initially allocated 40 MHz of spectrum to this system, which was increased to 50 MHz shortly after service introduction to support more users. This total bandwidth was divided into two 25-MHz bands, one for mobile-to-base station channels and the other for base station-to-mobile channels. The FCC divided these channels into two sets that were assigned to two different service providers in each city to encourage competition. A similar system, the Total Access Communication System (TACS), emerged in Europe. AMPS was deployed worldwide in the 1980s and remains the only cellular service in some areas, including rural parts of the United States.

Many of the first-generation cellular systems in Europe were incompatible, and the Europeans quickly converged on a uniform standard for second-generation (2G) digital systems called GSM.[2] The GSM standard uses a combination of TDMA and slow frequency hopping with frequency-shift keying for the voice modulation. In contrast, the standards activities in the United States surrounding the second generation of digital cellular provoked a raging debate on spectrum-sharing techniques, resulting in several incompatible standards [9; 10; 11]. In particular, there are two standards in the 900-MHz cellular frequency band: IS-136,[3] which uses a combination of TDMA and FDMA and phase-shift keyed modulation; and IS-95, which uses direct-sequence CDMA with phase-shift keyed modulation and coding [12; 13]. The spectrum for digital cellular in the 2-GHz PCS (personal communication system) frequency band was auctioned off, so service providers could use any standard for their purchased spectrum. The end result has been three different digital cellular standards for this frequency band: IS-136, IS-95, and the European GSM standard. The digital cellular standard in Japan is similar to IS-136 but in a different frequency band, and the GSM system in Europe is at a different frequency than the GSM systems in the United States. This proliferation of incompatible standards in the United States and internationally makes it impossible to roam between systems nationwide or globally without a multimode phone and/or multiple phones (and phone numbers).

All of the second-generation digital cellular standards have been enhanced to support high-rate packet data services [14]. GSM systems provide data rates of up to 140 kbps by aggregating all timeslots together for a single user. This enhancement is called GPRS. A

[2] The acronym GSM originally stood for Groupe Spéciale Mobile, the name of the European charter establishing the GSM standard. As GSM systems proliferated around the world, the underlying acronym meaning was changed to Global Systems for Mobile Communications.

[3] IS-136 is the evolution of the older IS-54 standard and subsumes it.

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 6 of 47

18-GHz band, had data rates on the order of 6 Mbps. However, performance of Altair was hampered by the high cost of components and the increased path loss at 18 GHz, and Altair was discontinued within a few years of its release.

The second-generation wireless LANs in the United States operate with 83.5 MHz of spectrum in the 2.4-GHz ISM band. A wireless LAN standard for this frequency band, the IEEE 802.11b standard, was developed to avoid some of the problems with the proprietary first-generation systems. The standard specifies direct-sequence spread spectrum with data rates of around 1.6 Mbps (raw data rates of 11 Mbps) and a range of approximately 100 m. The network architecture can be either star or peer-to-peer, although the peer-to-peer feature is rarely used. Many companies developed products based on the 802.11b standard, and after slow initial growth the popularity of 802.11b wireless LANs has expanded considerably. Many laptops come with integrated 802.11b wireless LAN cards. Companies and universities have installed 802.11b base stations throughout their locations, and many coffee houses, airports, and hotels offer wireless access, often for free, to increase their appeal.

Two additional standards in the 802.11 family were developed to provide higher data rates than 802.11b. The IEEE 802.11a wireless LAN standard operates with 300 MHz of spectrum in the 5-GHz U-NII band. The 802.11a standard is based on multicarrier modulation and provides 54-Mbps data rates at a range of about 30 m. Because 802.11a has much more bandwidth and consequently many more channels than 802.11b, it can support more users at higher data rates. There was some initial concern that 802.11a systems would be significantly more expensive than 802.11b systems, but in fact they quickly became quite competitive in price. The other standard, 802.11g, has the same design and data rates as 802.11a, but it operates in the the 2.4-GHz band with a range of about 50 m. Many wireless LAN cards and access points support all three standards to avoid incompatibilities.

In Europe, wireless LAN development revolves around the HIPERLAN (high-performance radio LAN) standards. The HIPERLAN/2 standard is similar to the IEEE 802.11a wireless LAN standard. In particular, it has a similar link layer design and also operates in a 5-GHz frequency band similar to the U-NII band. Hence it has the same maximum date rate of 54 Mbps and the same range of approximately 30 m as 802.11a. It differs from 802.11a in its access protocol and its built-in Quality-of-Service (QoS) support.

1.4.4 Wide Area Wireless Data Services

Wide area wireless data services provide wireless data to high-mobility users over a large coverage area. In these systems a given geographical region is serviced by base stations mounted on towers, rooftops, or mountains. The base stations can be connected to a backbone wired network or form a multihop ad hoc wireless network.

Initial wide area wireless data services had very low data rates, below 10 kbps, which gradually increased to 20 kbps. There were two main players providing this service: Motient and Bell South Mobile Data (formerly RAM Mobile Data). Metricom provided a similar service using architecture that consisted of a large network of small, inexpensive base stations with small coverage areas. The increased efficiency of the small coverage areas allowed for higher data rates in Metricom, 76 kbps, than in the other wide area wireless data systems. However, the high infrastructure cost for Metricom eventually forced it into bankruptcy, and the system was shut down. Some of the infrastructure was bought and is operating in a few areas as Ricochet.

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 7 of 47

The cellular digital packet data (CDPD) system is a wide area wireless data service overlayed on the analog cellular telephone network. CDPD shares the FDMA voice channels of the analog systems, since many of these channels are idle owing to the growth of digital cellular. The CDPD service provides packet data transmission at rates of 19.2 kbps and is available throughout the United States. However, since newer generations of cellular systems also provide data services and at higher data rates, CDPD is mostly being replaced by these newer services. Thus, wide area wireless data services have not been very successful, although emerging systems that offer broadband access may have more appeal.

1.4.5 Broadband Wireless Access

Broadband wireless access provides high-rate wireless communications between a fixed access point and multiple terminals. These systems were initially proposed to support interactive video service to the home, but the application emphasis then shifted to providing both high-speed data access (tens of Mbps) to the Internet and the World Wide Web as well as high-speed data networks for homes and businesses. In the United States, two frequency bands were set aside for these systems: part of the 28-GHz spectrum for local distribution systems (local multipoint distribution service, LMDS) and a band in the 2-GHz spectrum for metropolitan distribution service (multichannel multipoint distribution services, MMDS). LMDS represents a quick means for new service providers to enter the already stiff competition among wireless and wireline broadband service providers [5, Chap. 2.3]. MMDS is a television and telecommunication delivery system with transmission ranges of 30–50 km [5, Chap. 11.11]. MMDS has the capability of delivering more than a hundred digital video TV channels along with telephony and access to the Internet. MMDS will compete mainly with existing cable and satellite systems. Europe is developing a standard similar to MMDS called Hiperaccess.

WiMax is an emerging broadband wireless technology based on the IEEE 802.16 standard [16; 17]. The core 802.16 specification is a standard for broadband wireless access systems operating at radio frequencies between 2 GHz and 11 GHz for non–line-of-sight operation, and between 10 GHz and 66 GHz for line-of-sight operation. Data rates of around 40 Mbps will be available for fixed users and 15 Mbps for mobile users, with a range of several kilometers. Many manufacturers of laptops and PDAs (personal digital assistants) are planning to incorporate WiMax once it becomes available to satisfy demand for constant Internet access and email exchange from any location. WiMax will compete with wireless LANs, 3G cellular services, and possibly wireline services like cable and DSL (digital subscriber line). The ability of WiMax to challenge or supplant these systems will depend on its relative performance and cost, which remain to be seen.

1.4.6 Paging Systems

Paging systems broadcast a short paging message simultaneously from many tall base stations or satellites transmitting at very high power (hundreds of watts to kilowatts). Systems with terrestrial transmitters are typically localized to a particular geographic area, such as a city or metropolitan region, while geosynchronous satellite transmitters provide national or international coverage. In both types of systems, no location management or routing functions are needed because the paging message is broadcast over the entire coverage area. The high complexity and power of the paging transmitters allows low-complexity, low-power,

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 8 of 47

pocket paging receivers capable of long usage times from small and lightweight batteries. In addition, the high transmit power allows paging signals to easily penetrate building walls. Paging service also costs less than cellular service, both for the initial device and for the monthly usage charge, although this price advantage has declined considerably in recent years as cellular prices dropped. The low cost, small and lightweight handsets, long battery life, and ability of paging devices to work almost anywhere indoors or outdoors are the main reasons for their appeal.

Early radio paging systems were analog 1-bit messages signaling a user that someone was trying to reach him or her. These systems required callback over a landline telephone to obtain the phone number of the paging party. The system evolved to allow a short digital message, including a phone number and brief text, to be sent to the pagee as well. Radio paging systems were initially extremely successful, with a peak of 50 million subscribers in the United States alone. However, their popularity began to wane with the widespread penetration and competitive cost of cellular telephone systems. Eventually the competition from cellular phones forced paging systems to provide new capabilities. Some implemented "answer-back" capability (i.e., two-way communication). This required a major change in design of the pager because now it needed to transmit signals in addition to receiving them, and the transmission distance to a satellite or base station can be very large. Paging companies also teamed up with palmtop computer makers to incorporate paging functions into these devices [18]. Despite these developments, the market for paging devices has shrunk considerably, although there is still a niche market among doctors and other professionals who must be reachable anywhere.

1.4.7 Satellite Networks

Commercial satellite systems are another major component of the wireless communications infrastructure [2; 3]. Geosynchronous systems include Inmarsat and OmniTRACS. The former is geared mainly for analog voice transmission from remote locations. For example, it is commonly used by journalists to provide live reporting from war zones. The first-generation Inmarsat-A system was designed for large (1-m parabolic dish antenna) and rather expensive terminals. Newer generations of Inmarsats use digital techniques to enable smaller, less expensive terminals, about the size of a briefcase. Qualcomm's OmniTRACS provides two-way communications as well as location positioning. The system is used primarily for alphanumeric messaging and location tracking of trucking fleets. There are several major difficulties in providing voice and data services over geosynchronous satellites. It takes a great deal of power to reach these satellites, so handsets are typically large and bulky. In addition, there is a large round-trip propagation delay; this delay is quite noticeable in two-way voice communication. Geosynchronous satellites also have fairly low data rates of less than 10 kbps. For these reasons, lower-orbit LEO satellites were thought to be a better match for voice and data communications.

LEO systems require approximately 30–80 satellites to provide global coverage, and plans for deploying such constellations were widespread in the late 1990s. One of the most ambitious of these systems, the Iridium constellation, was launched at that time. However, the cost to build, launch, and maintain these satellites is much higher than costs for terrestrial base stations. Although these LEO systems can certainly complement terrestrial systems in low-population areas and are also appealing to travelers desiring just one handset and phone

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 9 of 47

number for global roaming, the growth and diminished cost of cellular prevented many ambitious plans for widespread LEO voice and data systems from materializing. Iridium was eventually forced into bankruptcy and disbanded, and most of the other systems were never launched. An exception to these failures was the Globalstar LEO system, which currently provides voice and data services over a wide coverage area at data rates under 10 kbps. Some of the Iridium satellites are still operational as well.

The most appealing use for a satellite system is the broadcast of video and audio over large geographic regions. Approximately 1 in 8 U.S. homes have direct broadcast satellite service, and satellite radio is also emerging as a popular service. Similar audio and video satellite broadcasting services are widespread in Europe. Satellites are best tailored for broadcasting, since they cover a wide area and are not compromised by an initial propagation delay. Moreover, the cost of the system can be amortized over many years and many users, making the service quite competitive in cost with that of terrestrial entertainment broadcasting systems.

1.4.8 Low-Cost, Low-Power Radios: Bluetooth and ZigBee

As radios decrease their cost and power consumption, it becomes feasible to embed them into more types of electronic devices, which can be used to create smart homes, sensor networks, and other compelling applications. Two radios have emerged to support this trend: Bluetooth and ZigBee.

Bluetooth[4] radios provide short-range connections between wireless devices along with rudimentary networking capabilities. The Bluetooth standard is based on a tiny microchip incorporating a radio transceiver that is built into digital devices. The transceiver takes the place of a connecting cable for devices such as cell phones, laptop and palmtop computers, portable printers and projectors, and network access points. Bluetooth is mainly for short-range communications – for example, from a laptop to a nearby printer or from a cell phone to a wireless headset. Its normal range of operation is 10 m (at 1-mW transmit power), and this range can be increased to 100 m by increasing the transmit power to 100 mW. The system operates in the unlicensed 2.4-GHz frequency band, so it can be used worldwide without any licensing issues. The Bluetooth standard provides one asynchronous data channel at 723.2 kbps. In this mode, also known as Asynchronous Connection-Less (ACL), there is a reverse channel with a data rate of 57.6 kbps. The specification also allows up to three synchronous channels each at a rate of 64 kbps. This mode, also known as Synchronous Connection Oriented (SCO), is mainly used for voice applications such as headsets but can also be used for data. These different modes result in an aggregate bit rate of approximately 1 Mbps. Routing of the asynchronous data is done via a packet switching protocol based on frequency hopping at 1600 hops per second. There is also a circuit switching protocol for the synchronous data.

Bluetooth uses frequency hopping for multiple access with a carrier spacing of 1 MHz. Typically, up to eighty different frequencies are used for a total bandwidth of 80 MHz. At any given time, the bandwidth available is 1 MHz, with a maximum of eight devices sharing the bandwidth. Different logical channels (different hopping sequences) can simultaneously

[4] The Bluetooth standard is named after Harald I Bluetooth, the king of Denmark between 940 and 985 A.D. who united Denmark and Norway. Bluetooth proposes to unite devices via radio connections, hence the inspiration for its name.

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 10 of 47

share the same 80-MHz bandwidth. Collisions will occur when devices in different piconets that are on different logical channels happen to use the same hop frequency at the same time. As the number of piconets in an area increases, the number of collisions increases and performance degrades.

The Bluetooth standard was developed jointly by 3 Com, Ericsson, Intel, IBM, Lucent, Microsoft, Motorola, Nokia, and Toshiba. The standard has now been adopted by over 1,300 manufacturers, and many consumer electronic products incorporate Bluetooth. These include wireless headsets for cell phones, wireless USB or RS232 connectors, wireless PCMCIA cards, and wireless settop boxes.

The ZigBee[5] radio specification is designed for lower cost and power consumption than Bluetooth [19]. Its specification is based on the IEEE 802.15.4 standard. The radio operates in the same ISM band as Bluetooth and is capable of connecting 255 devices per network. The specification supports data rates of up to 250 kbps at a range of up to 30 m. These data rates are slower than Bluetooth, but in exchange the radio consumes significantly less power with a larger transmission range. The goal of ZigBee is to provide radio operation for months or years without recharging, thereby targeting applications such as sensor networks and inventory tags.

1.4.9 Ultrawideband Radios

Ultrawideband (UWB) radios are extremely wideband radios with very high potential data rates [20; 21]. The concept of ultrawideband communications actually originated with Marconi's spark-gap transmitter, which occupied a very wide bandwidth. However, since only a single low-rate user could occupy the spectrum, wideband communications was abandoned in favor of more efficient communication techniques. Renewed interest in wideband communications was spurred by the FCC's decision in 2002 to allow operation of UWB devices underlayed beneath existing users in the 3.1–10.6-GHz range. The underlay in theory interferes with all systems in that frequency range, including critical safety and military systems, unlicensed systems such as 802.11 wireless LANs and Bluetooth, and cellular systems where operators paid billions of dollars for dedicated spectrum use. The FCC's ruling was quite controversial given the vested interest in interference-free spectrum of these users. To minimize the impact of UWB on primary band users, the FCC put in place severe transmit power restrictions. This requires UWB devices to be within close proximity of their intended receiver.

Ultrawideband radios come with unique advantages that have long been appreciated by the radar and communications communities. Their wideband nature provides precise ranging capabilities. Moreover, the available UWB bandwidth has the potential for extremely high data rates. Finally, power restrictions dictate that the devices be small and with low power consumption.

Initial UWB systems used ultrashort pulses with simple amplitude or position modulation. Multipath can significantly degrade performance of such systems, and proposals to mitigate the effects of multipath include equalization and multicarrier modulation. Precise

[5] ZigBee takes its name from the dance that honey bees use to communicate information about newly found food sources to other members of the colony.

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 11 of 47

and rapid synchronization is also a big challenge for these systems. Although many technical challenges remain, the appeal of UWB technology has sparked great interest both commercially and in the research community to address these issues.

1.5 The Wireless Spectrum

1.5.1 Methods for Spectrum Allocation

Most countries have government agencies responsible for allocating and controlling use of the radio spectrum. In the United States, spectrum is allocated by the Federal Communications Commission (FCC) for commercial use and by the Office of Spectral Management (OSM) for military use. Commercial spectral allocation is governed in Europe by the European Telecommunications Standards Institute (ETSI) and globally by the International Telecommunications Union (ITU). Governments decide how much spectrum to allocate between commercial and military use, and this decision is dynamic depending on need. Historically the FCC allocated spectral blocks for specific uses and assigned licenses to use these blocks to specific groups or companies. For example, in the 1980s the FCC allocated frequencies in the 800-MHz band for analog cellular phone service and provided spectral licenses to two operators in each geographical area based on a number of criteria. The FCC and regulatory bodies in other countries still allocate spectral blocks for specific purposes, but these blocks are now commonly assigned through spectral auctions to the highest bidder. Some argue that this market-based method is the fairest and most efficient way for governments to allocate the limited spectral resource and that it provides significant revenue to the government besides; others believe that this mechanism stifles innovation, limits competition, and hurts technology adoption. Specifically, the high cost of spectrum dictates that only large companies or conglomerates can purchase it. Moreover, the large investment required to obtain spectrum can delay the ability to invest in infrastructure for system rollout and results in high initial prices for the end user. The 3G spectral auctions in Europe, in which several companies ultimately defaulted, have provided ammunition to the opponents of spectral auctions.

In addition to spectral auctions, spectrum can be set aside in specific frequency bands that are free to use without a license according to a specific set of etiquette rules. The rules may correspond to a specific communications standard, power levels, and so forth. The purpose of these *unlicensed bands* is to encourage innovation and low-cost implementation. Many extremely successful wireless systems operate in unlicensed bands, including wireless LANs, Bluetooth, and cordless phones. A major difficulty of unlicensed bands is that they can be killed by their own success. If many unlicensed devices in the same band are used in close proximity then they interfere with each other, which can make the band unusable.

Underlay systems are another alternative for allocating spectrum. An underlay system operates as a secondary user in a frequency band with other primary users. Operation of secondary users is typically restricted so that primary users experience minimal interference. This is usually accomplished by restricting the power per hertz of the secondary users. UWB is an example of an underlay system, as are unlicensed systems in the ISM frequency bands. Such underlay systems can be extremely controversial given the complexity of characterizing how interference affects the primary users. Yet the trend toward spectrum allocation for

8

Coding for Wireless Channels

Coding allows bit errors introduced by transmission of a modulated signal through a wireless channel to be either detected or corrected by a decoder in the receiver. Coding can be considered as the embedding of signal constellation points in a higher-dimensional signaling space than is needed for communications. By going to a higher-dimensional space, the distance between points can be increased, which provides for better error correction and detection.

In this chapter we describe codes designed for additive white Gaussian noise channels and for fading channels. Codes designed for AWGN channels typically do not work well on fading channels because they cannot correct for long error bursts that occur in deep fading. Codes for fading channels are mainly based on using an AWGN channel code combined with interleaving, but the criterion for the code design changes to provide fading diversity. Other coding techniques to combat performance degradation due to fading include unequal error protection codes and joint source and channel coding.

We first provide an overview of code design in both fading and AWGN, along with basic design parameters such as minimum distance, coding gain, bandwidth expansion, and diversity order. Sections 8.2 and 8.3 provide a basic overview of block and convolutional code designs for AWGN channels. Although these designs are not directly applicable to fading channels, codes for fading channels and other codes used in wireless systems (e.g., spreading codes in CDMA) require background in these fundamental techniques. Concatenated codes and their evolution to turbo codes – as well as low-density parity-check (LDPC) codes – for AWGN channels are also described. These extremely powerful codes exhibit near-capacity performance with reasonable complexity levels. Coded modulation was invented in the late 1970s as a technique to obtain error correction through a joint design of the modulation and coding. We will discuss the basic design principles behind trellis and more general lattice coded modulation along with their performance in AWGN.

Code designs for fading channels are covered in Section 8.8. These designs combine block or convolutional codes designed for AWGN channels with interleaving and then modify the AWGN code design metric to incorporate maximum fading diversity. Diversity gains can also be obtained by combining coded modulation with symbol or bit interleaving, although bit interleaving generally provides much higher diversity gain. Thus, coding combined with interleaving provides diversity gain in the same manner as other forms of diversity, with the diversity order built into the code design. Unequal error protection is an alternative to

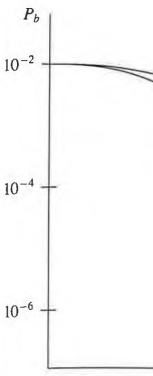


Figure 8.1: Coding ga

diversity in fading
encoded with strong
source code design
coding, which we a

Coding is a very
this topic. This cha
discussion of code
treated. This in-dep
wireless channels b

8.1 Overview o

The main reason to
bility of bit or block
that a bit is decoded
is the probability th
error probability is
blocks. The amoun
by its coding gain i

Coding gain in
power ratio can be
coding gain for P_b i
than the gain C_{g2} at
codes designed for h
extra redundancy of

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 13 of 47

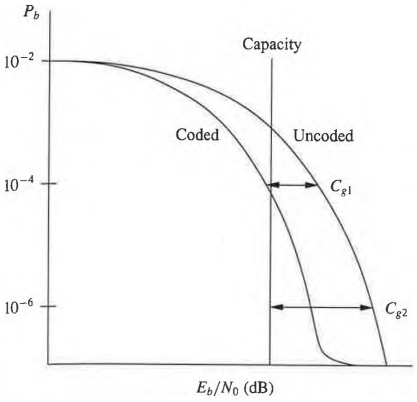


Figure 8.1: Coding gain in AWGN channels.

diversity in fading mitigation. In these codes bits are prioritized, and high-priority bits are encoded with stronger error protection against deep fades. Since bit priorities are part of the source code design, unequal error protection is a special case of joint source and channel coding, which we also describe.

Coding is a very broad and deep subject, with many excellent books devoted solely to this topic. This chapter assumes no background in coding, and thus it provides an in-depth discussion of code designs for AWGN channels before designs for wireless systems can be treated. This in-depth discussion can be omitted for a more cursory treatment of coding for wireless channels by focusing on Sections 8.1 and 8.8.

8.1 Overview of Code Design

The main reason to apply error correction coding in a wireless system is to reduce the probability of bit or block error. The bit error probability P_b for a coded system is the probability that a bit is decoded in error. The block error probability P_{bl}, also called the packet error rate, is the probability that one or more bits in a block of coded bits is decoded in error. Block error probability is useful for packet data systems where bits are encoded and transmitted in blocks. The amount of error reduction provided by a given code is typically characterized by its coding gain in AWGN and its diversity gain in fading.

Coding gain in AWGN is defined as the amount that the bit energy or signal-to-noise power ratio can be reduced under the coding technique for a given P_b or P_{bl}. We illustrate coding gain for P_b in Figure 8.1. We see in this figure that the gain C_{g1} at $P_b = 10^{-4}$ is less than the gain C_{g2} at $P_b = 10^{-6}$, and there is negligible coding gain at $P_b = 10^{-2}$. In fact, codes designed for high-SNR channels can have negative coding gain at low SNRs, since the extra redundancy of the code does not provide sufficient performance gain in P_b or P_{bl} at

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 14 of 47

low SNRs to compensate for spreading the bit energy over multiple coded bits. Thus, unexpected fluctuations in channel SNR can significantly degrade code performance. The coding gain in AWGN is generally a function of the minimum Euclidean distance of the code, which equals the minimum distance in signal space between codewords or error events. Thus, codes designed for AWGN channels maximize their Euclidean distance for good performance.

Error probability with or without coding tends to fall off with SNR as a waterfall shape at low to moderate SNRs. Whereas this waterfall shape holds at all SNRs for uncoded systems and many coded systems, some codes (such as turbo codes) exhibit error floors as SNR grows. The error floor, also shown in Figure 8.1, kicks in at a threshold SNR that depends on the code design. For SNRs above this threshold, the slope of the error probability curve decreases because minimum distance error events dominate code performance in this SNR regime.

Code performance is also commonly measured against channel capacity. The capacity curve is associated with the SNR (E_b/N_0) where Shannon capacity $B \log_2(1 + \text{SNR})$ equals the data rate of the system. At rates up to capacity, the capacity-achieving code has a probability of error that goes to zero, as indicated by the straight line in Figure 8.1. The capacity curve thus indicates the best possible performance that any practical code can achieve.

For many codes, the error correction capability of a code does not come for free. This performance enhancement is paid for by increased complexity and – for block codes, convolutional codes, turbo codes, and LDPC codes – by either a decrease in data rate or an increase in signal bandwidth. Consider a code with n coded bits for every k uncoded bits. This code effectively embeds a k-dimensional subspace into a larger n-dimensional space to provide larger distances between coded symbols. However, if the data rate through the channel is fixed at R_b, then the information rate for a code that uses n coded bits for every k uncoded bits is $(k/n)R_b$; that is, coding decreases the data rate by the fraction k/n. We can keep the information rate constant and introduce coding gain by decreasing the bit time by k/n. This typically results in an expanded bandwidth of the transmittted signal by n/k. Coded modulation uses a joint design of the code and modulation to obtain coding gain without this bandwidth expansion, as discussed in more detail in Section 8.7.

Codes designed for AWGN channels do not generally work well in fading owing to bursts of errors that cannot be corrected for. However, good performance in fading can be obtained by combining AWGN channel codes with interleaving and by designing the code to optimize its inherent diversity. The interleaver spreads out bursts of errors over time, so it provides a form of time diversity. This diversity is exploited by the inherent diversity in the code. In fact, codes designed in this manner exhibit performance similar to MRC diversity, with diversity order equal to the minimum Hamming distance of the code. Hamming distance is the number of coded symbols that differ between different codewords or error events. Thus, coding and interleaving designed for fading channels maximize their Hamming distance for good performance.

8.2 Linear Block Codes

Linear block codes are conceptually simple codes that are basically an extension of single-bit parity-check codes for error detection. A single-bit parity-check code is one of the most

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 15 of 47

common forms of detecting transmission errors. This code uses one extra bit in a block of n data bits to indicate whether the number of 1-bits in a block is odd or even. Thus, if a single error occurs, either the parity bit is corrupted or the number of detected 1-bits in the information bit sequence will be different from the number used to compute the parity bit; in either case the parity bit will not correspond to the number of detected 1-bits in the information bit sequence, so the single error is detected. Linear block codes extend this notion by using a larger number of parity bits to either detect more than one error or correct for one or more errors. Unfortunately, linear block codes – along with convolutional codes – trade their error detection or correction capability for either bandwidth expansion or a lower data rate, as we shall discuss in more detail. We will restrict our attention to binary codes, where both the original information and the corresponding code consist of bits taking a value of either 0 or 1.

8.2.1 Binary Linear Block Codes

A binary block code generates a block of n coded bits from k information bits. We call this an (n, k) binary block code. The coded bits are also called *codeword symbols.* The n codeword symbols can take on 2^n possible values corresponding to all possible combinations of the n binary bits. We select 2^k codewords from these 2^n possibilities to form the code, where each k bit information block is uniquely mapped to one of these 2^k codewords. The rate of the code is $R_c = k/n$ information bits per codeword symbol. If we assume that codeword symbols are transmitted across the channel at a rate of R_s symbols per second, then the information rate associated with an (n, k) block code is $R_b = R_c R_s = (k/n)R_s$ bits per second. Thus we see that block coding reduces the data rate compared to what we obtain with uncoded modulation by the code rate R_c.

A block code is called a *linear* code when the mapping of the k information bits to the n codeword symbols is a linear mapping. In order to describe this mapping and the corresponding encoding and decoding functions in more detail, we must first discuss properties of the vector space of binary n-tuples and its corresponding subspaces. The set of all binary n-tuples B_n is a vector space over the binary field, which consists of the two elements 0 and 1. All fields have two operations, addition and multiplication: for the binary field these operations correspond to binary addition (modulo 2 addition) and standard multiplication. A subset S of B_n is called a *subspace* if it satisfies the following conditions.

1. The all-zero vector is in S.
2. The set S is closed under addition; that is, if $S_i \in S$ and $S_j \in S$ then $S_i + S_j \in S$.

An (n, k) block code is linear if the 2^k length-n codewords of the code form a subspace of B_n. Thus, if $\mathbf{C}_i$ and $\mathbf{C}_j$ are two codewords in an (n, k) linear block code, then $\mathbf{C}_i + \mathbf{C}_j$ must form another codeword of the code.

EXAMPLE 8.1: The vector space B_3 consists of all binary tuples of length 3:

$$B_3 = \{[000], [001], [010], [011], [100], [101], [110], [111]\}.$$

Note that B_3 is a subspace of itself, since it contains the all-zero vector and is closed under addition. Determine which of the following subsets of B_3 form a subspace:

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 16 of 47

- $A_1 = \{[000], [001], [100], [101]\}$;
- $A_2 = \{[000], [100], [110], [111]\}$;
- $A_3 = \{[001], [100], [101]\}$.

Solution: It is easily verified that A_1 is a subspace, since it contains the all-zero vector and the sum of any two tuples in A_1 is also in A_1. A_2 is not a subspace because it is not closed under addition, as $110 + 111 = 001 \notin A_2$. A_3 is not a subspace because it is not closed under addition ($001 + 001 = 000 \notin A_3$) and it does not contain the all-zero vector.

Intuitively, the greater the distance between codewords in a given code, the less chance that errors introduced by the channel will cause a transmitted codeword to be decoded as a different codeword. We define the *Hamming distance* between two codewords $\mathbf{C}_i$ and $\mathbf{C}_j$, denoted as $d(\mathbf{C}_i, \mathbf{C}_j)$ or d_{ij}, as the number of elements in which they differ:

$$d_{ij} = \sum_{l=1}^{n} (\mathbf{C}_i(l) + \mathbf{C}_j(l)), \tag{8.1}$$

where $\mathbf{C}_m(l)$ denotes the lth bit in $\mathbf{C}_m$. For example, if $\mathbf{C}_i = [00101]$ and $\mathbf{C}_j = [10011]$ then $d_{ij} = 3$. We define the *weight* of a given codeword $\mathbf{C}_i$ as the number of 1-bits in the codeword, so $\mathbf{C}_i = [00101]$ has weight 2. The weight of a given codeword $\mathbf{C}_i$ is just its Hamming distance d_{0i} from the all-zero codeword $\mathbf{C}_0 = [00\ldots0]$ or, equivalently, the sum of its elements:

$$w(\mathbf{C}_i) = \sum_{l=1}^{n} \mathbf{C}_i(l). \tag{8.2}$$

Since $0 + 0 = 1 + 1 = 0$, the Hamming distance between $\mathbf{C}_i$ and $\mathbf{C}_j$ is equal to the weight of $\mathbf{C}_i + \mathbf{C}_j$. For example, with $\mathbf{C}_i = [00101]$ and $\mathbf{C}_j = [10011]$ as given before, $w(\mathbf{C}_i) = 2$, $w(\mathbf{C}_j) = 3$, and $d_{ij} = w(\mathbf{C}_i + \mathbf{C}_j) = w([10110]) = 3$. Since the Hamming distance between any two codewords equals the weight of their sum and since this sum is also a codeword, we can determine the minimum distance between all codewords in a code by just looking at the minimum distance between all nonzero codewords and the all-zero codeword $\mathbf{C}_0$. Thus, we define the minimum distance of a code as

$$d_{\min} = \min_{i, i \neq 0} d_{0i}. \tag{8.3}$$

We will see in Section 8.2.6 that the minimum distance of a linear block code is a critical parameter in determining its probability of error.

8.2.2 Generator Matrix

The generator matrix is a compact description of how codewords are generated from information bits in a linear block code. The design goal in linear block codes is to find generator matrices such that their corresponding codes are easy to encode and decode yet have powerful error correction/detection capabilities. Consider an (n, k) code with k information bits, denoted as

$$\mathbf{U}_i = [u_{i1}, \ldots, u_{ik}],$$

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 17 of 47

that are encoded into the codeword

$$\mathbf{C}_i = [c_{i1}, \ldots, c_{in}].$$

We represent the encoding operation as a set of n equations defined by

$$c_{ij} = u_{i1}g_{1j} + u_{i2}g_{2j} + \cdots + u_{ik}g_{kj}, \quad j = 1, \ldots, n, \tag{8.4}$$

where g_{ij} is binary (0 or 1) and where binary (standard) multiplication is used. We can write these n equations in matrix form as

$$\mathbf{C}_i = \mathbf{U}_i\mathbf{G}, \tag{8.5}$$

where the $k \times n$ *generator matrix* $\mathbf{G}$ for the code is defined as

$$\mathbf{G} = \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1n} \\ g_{21} & g_{22} & \cdots & g_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ g_{k1} & g_{k2} & \cdots & g_{kn} \end{bmatrix}. \tag{8.6}$$

If we denote the lth row of $\mathbf{G}$ as $\mathbf{g}_l = [g_{l1}, \ldots, g_{ln}]$, then we can write any codeword $\mathbf{C}_i$ as linear combinations of these row vectors as follows:

$$\mathbf{C}_i = u_{i1}\mathbf{g}_1 + u_{i2}\mathbf{g}_2 + \cdots + u_{ik}\mathbf{g}_k. \tag{8.7}$$

Since a linear (n, k) block code is a subspace of dimension k in the larger n-dimensional space, it follows that the k row vectors $\{\mathbf{g}_l\}_{l=1}^{k}$ of $\mathbf{G}$ must be linearly independent so that they span the k-dimensional subspace associated with the 2^k codewords. Hence, $\mathbf{G}$ has rank k. Since the set of basis vectors for this subspace is not unique, the generator matrix is also not unique.

A *systematic* linear block code is described by a generator matrix of the form

$$\mathbf{G} = [\mathbf{I}_k \mid \mathbf{P}] = \left[\begin{array}{cccc|cccc} 1 & 0 & \cdots & 0 & p_{11} & p_{12} & \cdots & p_{1(n-k)} \\ 0 & 1 & \cdots & 0 & p_{21} & p_{22} & \cdots & p_{2(n-k)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & p_{k1} & p_{k2} & \cdots & p_{k(n-k)} \end{array}\right], \tag{8.8}$$

where $\mathbf{I}_k$ is the $k \times k$ identity matrix and $\mathbf{P}$ is a $k \times (n-k)$ matrix that determines the redundant, or parity, bits to be used for error correction or detection. The codeword output from a systematic encoder is of the form

$$\mathbf{C}_i = \mathbf{U}_i\mathbf{G} = \mathbf{U}_i[\mathbf{I}_k \mid \mathbf{P}] = [u_{i1}, \ldots, u_{ik}, p_1, \ldots, p_{(n-k)}], \tag{8.9}$$

where the first k bits of the codeword are the original information bits and the last $(n-k)$ bits of the codeword are the parity bits obtained from the information bits as

$$p_j = u_{i1}p_{1j} + \cdots + u_{ik}p_{kj}, \quad j = 1, \ldots, n-k. \tag{8.10}$$

Note that any generator matrix for an (n, k) linear block code can be reduced by row operations and column permutations to a generator matrix in systematic form.

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 18 of 47

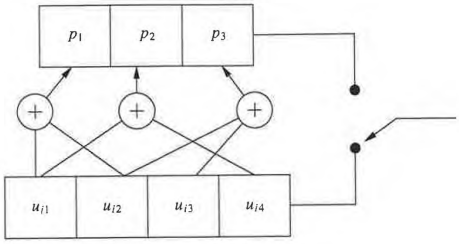


Figure 8.2: Implementation of (7, 4) binary code.

EXAMPLE 8.2: Systematic linear block codes are typically implemented with $n-k$ modulo-2 adders tied to the appropriate stages of a shift register. The resulting parity bits are appended to the end of the information bits to form the codeword. Find the corresponding implementation for generating a (7, 4) binary code with the generator matrix

$$\mathbf{G} = \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{array}\right]. \tag{8.11}$$

Solution: The matrix $\mathbf{G}$ is already in systematic form with

$$\mathbf{P} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}. \tag{8.12}$$

Let P_{lj} denote the ljth element of $\mathbf{P}$. By (8.10) we see that the first parity bit in the codeword is $p_1 = u_{i1}P_{11} + u_{i2}P_{21} + u_{i3}P_{31} + u_{i4}P_{41} = u_{i1} + u_{i2}$. Similarly, the second parity bit is $p_2 = u_{i1}P_{12} + u_{i2}P_{22} + u_{i3}P_{32} + u_{i4}P_{42} = u_{i1} + u_{i4}$ and the third parity bit is $p_3 = u_{i1}P_{13} + u_{i2}P_{23} + u_{i3}P_{33} + u_{i4}P_{43} = u_{i2} + u_{i3}$. The shift register implementation to generate these parity bits is shown in Figure 8.2. The codeword output is $[u_{i1}u_{i2}u_{i3}u_{i4}p_1p_2p_3]$, where the switch is in the down position to output the systematic bits u_{ij} $(j = 1, \ldots, 4)$ of the code or in the up position to output the parity bits p_j $(j = 1, 2, 3)$ of the code.

8.2.3 Parity-Check Matrix and Syndrome Testing

The parity-check matrix is used to decode linear block codes with generator matrix $\mathbf{G}$. The parity-check matrix $\mathbf{H}$ corresponding to a generator matrix $\mathbf{G} = [\mathbf{I}_k \mid \mathbf{P}]$ is defined as

$$\mathbf{H} = [\mathbf{P}^T \mid \mathbf{I}_{n-k}]. \tag{8.13}$$

It is easily verified that $\mathbf{G}\mathbf{H}^T = \mathbf{0}_{k,n-k}$, where $\mathbf{0}_{k,n-k}$ denotes an all-zero $k \times (n-k)$ matrix. Recall that a given codeword $\mathbf{C}_i$ in the code is obtained by multiplication of the information bit sequence $\mathbf{U}_i$ by the generator matrix $\mathbf{G}$: $\mathbf{C}_i = \mathbf{U}_i\mathbf{G}$. Thus,

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 19 of 47

$$\mathbf{C}_i\mathbf{H}^T = \mathbf{U}_i\mathbf{G}\mathbf{H}^T = \mathbf{0}_{n-k} \tag{8.14}$$

for any input sequence $\mathbf{U}_i$, where $\mathbf{0}_{n-k}$ denotes the all-zero row vector of length $n-k$. Thus, multiplication of any valid codeword with the parity-check matrix results in an all-zero vector. This property is used to determine whether the received vector is a valid codeword or has been corrupted, based on the notion of *syndrome testing,* which we now define.

Let $\mathbf{R}$ be the received codeword resulting from transmission of codeword $\mathbf{C}$. In the absence of channel errors, $\mathbf{R} = \mathbf{C}$. However, if the transmission is corrupted, then one or more of the codeword symbols in $\mathbf{R}$ will differ from those in $\mathbf{C}$. We therefore write the received codeword as

$$\mathbf{R} = \mathbf{C} + \mathbf{e}, \tag{8.15}$$

where $\mathbf{e} = [e_1, e_2, \ldots, e_n]$ is the *error pattern* indicating which codeword symbols were corrupted by the channel. We define the *syndrome* of $\mathbf{R}$ as

$$\mathbf{S} = \mathbf{R}\mathbf{H}^T. \tag{8.16}$$

If $\mathbf{R}$ is a valid codeword (i.e., $\mathbf{R} = \mathbf{C}_i$ for some i) then $\mathbf{S} = \mathbf{C}_i\mathbf{H}^T = \mathbf{0}_{n-k}$ by (8.14). Thus, the syndrome equals the all-zero vector if the transmitted codeword is not corrupted – or is corrupted in a manner such that the received codeword is a valid codeword in the code but is different from the transmitted codeword. If the received codeword $\mathbf{R}$ contains detectable errors, then $\mathbf{S} \neq \mathbf{0}_{n-k}$. If the received codeword contains correctable errors, the syndrome identifies the error pattern corrupting the transmitted codeword, and these errors can then be corrected. Note that the syndrome is a function only of the error pattern $\mathbf{e}$ and not the transmitted codeword $\mathbf{C}$, since

$$\mathbf{S} = \mathbf{R}\mathbf{H}^T = (\mathbf{C}+\mathbf{e})\mathbf{H}^T = \mathbf{C}\mathbf{H}^T + \mathbf{e}\mathbf{H}^T = \mathbf{0}_{n-k} + \mathbf{e}\mathbf{H}^T. \tag{8.17}$$

Because $\mathbf{S} = \mathbf{e}\mathbf{H}^T$ corresponds to $n-k$ equations in n unknowns, there are 2^k possible error patterns that can produce a given syndrome $\mathbf{S}$. However, since the probability of bit error is typically small and independent for each bit, the most likely error pattern is the one with minimal weight, corresponding to the least number of errors introduced in the channel. Thus, if an error pattern $\hat{\mathbf{e}}$ is the most likely error associated with a given syndrome $\mathbf{S}$, the transmitted codeword is typically decoded as

$$\hat{\mathbf{C}} = \mathbf{R} + \hat{\mathbf{e}} = \mathbf{C} + \mathbf{e} + \hat{\mathbf{e}}. \tag{8.18}$$

If the most likely error pattern does occur, then $\hat{\mathbf{e}} = \mathbf{e}$ and $\hat{\mathbf{C}} = \mathbf{C}$ – that is, the corrupted codeword is correctly decoded. The decoding process and associated error probability will be covered in Section 8.2.6.

Let $\mathbf{C}_w$ denote a codeword in a given (n,k) code with minimum weight (excluding the all-zero codeword). Then $\mathbf{C}_w\mathbf{H}^T = \mathbf{0}_{n-k}$ is just the sum of $d_{\min}$ columns of $\mathbf{H}^T$, since $d_{\min}$ equals the number of 1-bits (the weight) in the minimum weight codeword of the code. Since the rank of $\mathbf{H}^T$ is at most $n-k$, this implies that the minimum distance of an (n,k) block code is upper bounded by

$$d_{\min} \leq n - k + 1, \tag{8.19}$$

which is referred to as the *Singelton bound.*

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 20 of 47

8.2.4 Cyclic Codes

Cyclic codes are a subclass of linear block codes in which all codewords in a given code are cyclic shifts of one another. Specifically, if the codeword $\mathbf{C} = [c_0, c_1, \ldots, c_{n-1}]$ is a codeword in a given code, then a cyclic shift by 1, denoted as $\mathbf{C}^{(1)} = [c_{n-1}, c_0, \ldots, c_{n-2}]$, is also a codeword. More generally, any cyclic shift $C^{(i)} = [c_{n-i}, c_{n-i+1}, \ldots, c_{n-i-1}]$ is also a codeword. The cyclic nature of cyclic codes creates a nice structure that allows their encoding and decoding functions to be of much lower complexity than the matrix multiplications associated with encoding and decoding for general linear block codes. Thus, most linear block codes used in practice are cyclic codes.

Cyclic codes are generated via a *generator polynomial* instead of a generator matrix. The generator polynomial $g(X)$ for an (n, k) cyclic code has degree $n - k$ and is of the form

$$g(X) = g_0 + g_1 X + \cdots + g_{n-k} X^{n-k}, \tag{8.20}$$

where g_i is binary (0 or 1) and $g_0 = g_{n-k} = 1$. The k-bit information sequence $[u_0, \ldots, u_{k-1}]$ is also written in polynomial form as the *message polynomial*

$$u(X) = u_0 + u_1 X + \cdots + u_{k-1} X^{k-1}. \tag{8.21}$$

The codeword associated with a given k-bit information sequence is obtained from the polynomial coefficients of the generator polynomial multiplied by the message polynomial; thus, the codeword $C = [c_0, \ldots, c_{n-1}]$ is obtained from

$$c(X) = u(X)g(X) = c_0 + c_1 X + \cdots + c_{n-1} X^{n-1}. \tag{8.22}$$

A codeword described by a polynomial $c(X)$ is a valid codeword for a cyclic code with generator polynomial $g(X)$ if and only if $g(X)$ divides $c(X)$ with no remainder (no remainder polynomial terms) – that is, if and only if

$$\frac{c(X)}{g(X)} = q(X) \tag{8.23}$$

for a polynomial $q(X)$ of degree less than k.

EXAMPLE 8.3: Consider a (7, 4) cyclic code with generator polynomial $g(X) = 1 + X^2 + X^3$. Determine if the codewords described by polynomials $c_1(X) = 1 + X^2 + X^5 + X^6$ and $c_2(X) = 1 + X^2 + X^3 + X^5 + X^6$ are valid codewords for this generator polynomial.

Solution: Division of binary polynomials is similar to division of standard polynomials except that, under binary addition, subtraction is the same as addition. Dividing $c_1(X) = 1 + X^2 + X^5 + X^6$ by $g(X) = 1 + X^2 + X^3$, we have:

$$\begin{array}{r} X^3 + 1 \\ X^3 + X^2 + 1 \overline{)\, X^6 + X^5 + X^2 + 1} \\ \underline{X^6 + X^5 + X^3} \\ X^3 + X^2 + 1 \\ \underline{X^3 + X^2 + 1} \\ 0. \end{array}$$

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 21 of 47

Since $g(X)$ divides $c(X)$ with no remainder, it is a valid codeword. In fact, we have $c_1(X) = (1 + X^3)g(X) = u(X)g(X)$ and so the information bit sequence corresponding to $c_1(X)$ is $\mathbf{U} = [1001]$, corresponding to the coefficients of the message polynomial $u(X) = 1 + X^3$.

Dividing $c_2(X) = 1 + X^2 + X^3 + X^5 + X^6$ by $g(X) = 1 + X^2 + X^3$ yields

$$\begin{array}{r} X^3 + 1 \\ X^3 + X^2 + 1 \overline{)\, X^6 + X^5 + X^3 + X^2 + 1} \\ \underline{X^6 + X^5 + X^3} \\ X^2 + 1, \end{array}$$

where we note that there is a remainder of $X^2 + 1$ in the division. Thus, $c_2(X)$ is not a valid codeword for the code corresponding to this generator polynomial.

Recall that systematic linear block codes have the first k codeword symbols equal to the information bits and the remaining codeword symbols equal to the parity bits. A cyclic code can be put in systematic form by first multiplying the message polynomial $u(X)$ by X^{n-k}, yielding

$$X^{n-k}u(X) = u_0X^{n-k} + u_1X^{n-k+1} + \cdots + u_{k-1}X^{n-1}. \tag{8.24}$$

This shifts the message bits to the k rightmost digits of the codeword polynomial. If we next divide (8.24) by $g(X)$, we obtain

$$\frac{X^{n-k}u(X)}{g(X)} = q(X) + \frac{p(X)}{g(X)}, \tag{8.25}$$

where $q(X)$ is a polynomial of degree at most $k - 1$ and $p(X)$ is a remainder polynomial of degree at most $n - k - 1$. Multiplying (8.25) through by $g(X)$, we now have

$$X^{n-k}u(X) = q(X)g(X) + p(X). \tag{8.26}$$

Adding $p(X)$ to both sides yields

$$p(X) + X^{n-k}u(X) = q(X)g(X). \tag{8.27}$$

This implies that $p(X) + X^{n-k}u(X)$ is a valid codeword, since it is divisible by $g(X)$ with no remainder. The codeword is described by the n coefficients of the codeword polynomial $p(X) + X^{n-k}u(X)$. Note that we can express $p(X)$ (of degree $n - k - 1$) as

$$p(X) = p_0 + p_1X + \cdots + p_{n-k-1}X^{n-k-1}. \tag{8.28}$$

Combining (8.24) and (8.28), we get

$$\begin{aligned} &p(X) + X^{n-k}u(X) \\ &\quad = p_0 + p_1X + \cdots + p_{n-k-1}X^{n-k-1} + u_0X^{n-k} + u_1X^{n-k+1} + \cdots + u_{k-1}X^{n-1}. \end{aligned} \tag{8.29}$$

Thus, the codeword corresponding to this polynomial has the first k bits consisting of the message bits $[u_0, \ldots, u_k]$ and the last $n - k$ bits consisting of the parity bits $[p_0, \ldots, p_{n-k-1}]$, as is required for the systematic form.

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 22 of 47

Note that the systematic codeword polynomial is generated in three steps: first multiplying the message polynomial $u(X)$ by X^{n-k}; then dividing $X^{n-k}u(X)$ by $g(X)$ to obtain the remainder polynomial $p(X)$ (along with the quotient polynomial $q(X)$, which is not used); and finally adding $p(X)$ to $X^{n-k}u(X)$ to get (8.29). The polynomial multiplications are straightforward to implement, and the polynomial division is easily implemented with a feedback shift register [1, Chap. 8.1; 2, Chap. 6.7]. Thus, codeword generation for systematic cyclic codes has very low cost and low complexity.

Let us now consider how to characterize channel errors for cyclic codes. The codeword polynomial corresponding to a transmitted codeword is of the form

$$c(X) = u(X)g(X). \tag{8.30}$$

The received codeword can also be written in polynomial form as

$$r(X) = c(X) + e(X) = u(X)g(X) + e(X), \tag{8.31}$$

where $e(X)$ is the error polynomial with coefficients equal to 1 where errors occur. For example, if the transmitted codeword is $\mathbf{C} = [1011001]$ and the received codeword is $\mathbf{R} = [1111000]$, then $e(X) = X + X^{n-1}$. The *syndrome polynomial* $s(X)$ for the received codeword is defined as the remainder when $r(X)$ is divided by $g(X)$, so $s(X)$ has degree $n-k-1$. But by (8.31), the syndrome polynomial $s(X)$ is equivalent to the error polynomial $e(X)$ modulo $g(X)$. Moreover, we obtain the syndrome through a division circuit similar to the one used for generating the code. As stated previously, this division circuit is typically implemented using a feedback shift register, resulting in a low-cost implementation of low complexity.

8.2.5 Hard Decision Decoding (HDD)

The probability of error for linear block codes depends on whether the decoder uses soft decisions or hard decisions. In hard decision decoding (HDD), each coded bit is demodulated as a 0 or 1 – that is, the demodulator detects each coded bit (symbol) individually. For example, in BPSK, the received symbol is decoded as a 1 if it is closer to $\sqrt{E_b}$ and as a 0 if it is closer to $-\sqrt{E_b}$. This form of demodulation removes information that can be used by the channel decoder. In particular, for the BPSK example the distance of the received bit from $\sqrt{E_b}$ and $-\sqrt{E_b}$ can be used in the channel decoder to make better decisions about the transmitted codeword. In soft decision decoding, these distances are used in the decoding process. Soft decision decoding of linear block codes, which is more common in wireless systems than hard decision decoding, is treated in Section 8.2.7.

Hard decision decoding uses *minimum distance* decoding based on Hamming distance. In minimum distance decoding the n bits corresponding to a codeword are first demodulated to a 0 or 1, and the demodulator output is then passed to the decoder. The decoder compares this received codeword to the 2^k possible codewords that constitute the code and decides in favor of the codeword that is closest in Hamming distance to (differs in the least number of bits from) the received codeword. Mathematically, for a received codeword $\mathbf{R}$, the decoder uses the formula

$$\text{pick } \mathbf{C}_j \text{ s.t. } d(\mathbf{C}_j, \mathbf{R}) \leq d(\mathbf{C}_i, \mathbf{R}) \; \forall i \neq j. \tag{8.32}$$

If there is more than one codeword with the same minimum distance to $\mathbf{R}$, one of these is chosen at random by the decoder.

Figure 8.3: Maximum

Maximum likel
bility of having pro
$\mathbf{R}$, the maximum li

Since the most pro
number of errors n
(8.32) and the maxi
lihood codeword $\mathbf{C}$

Because maxim
metric, we can best
mum Hamming dis
figure, is $d_{\min}$. Each
denotes the largest
quences where one
indicates the Hamm

Minimum distan
rors in a data block
correction allows th
mum distance deco
distance t from the
correct up to t errors
ing to t or fewer er
will be decoded as th

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 23 of 47

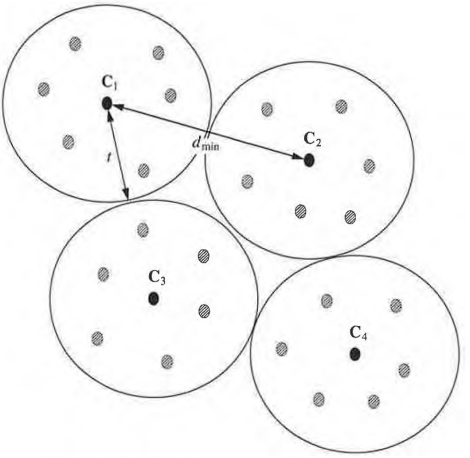


Figure 8.3: Maximum likelihood decoding in code space.

Maximum likelihood decoding picks the transmitted codeword that has the highest probability of having produced the received codeword. In other words, given the received codeword $\mathbf{R}$, the maximum likelihood decoder choses the codeword $\mathbf{C}_j$ as

$$\mathbf{C}_j = \arg\max_i p(\mathbf{R} \mid \mathbf{C}_i), \quad i = 0, \ldots, 2^k - 1. \tag{8.33}$$

Since the most probable error event in an AWGN channel is the event with the minimum number of errors needed to produce the received codeword, the minimum distance criterion (8.32) and the maximum likelihood criterion (8.33) are equivalent. Once the maximum likelihood codeword $\mathbf{C}_i$ is determined, it is decoded to the k bits that produce codeword $\mathbf{C}_i$.

Because maximum likelihood detection of codewords is based on a distance decoding metric, we can best illustrate this process in code space, as shown in Figure 8.3. The minimum Hamming distance between codewords, which are illustrated by the black dots in this figure, is $d_{\min}$. Each codeword is centered inside a sphere of radius $t = \lfloor .5d_{\min} \rfloor$, where $\lfloor x \rfloor$ denotes the largest integer less than or equal to x. The shaded dots represent received sequences where one or more bits differ from those of the transmitted codeword. The figure indicates the Hamming distance between $\mathbf{C}_1$ and $\mathbf{C}_2$.

Minimum distance decoding can be used either to detect or to correct errors. Detected errors in a data block cause either the data to be dropped or a retransmission of the data. Error correction allows the corruption in the data to be reversed. For error correction the minimum distance decoding process ensures that a received codeword lying within a Hamming distance t from the transmitted codeword will be decoded correctly. Thus, the decoder can correct up to t errors, as can be seen from Figure 8.3: since received codewords corresponding to t or fewer errors will lie within the sphere centered around the correct codeword, it will be decoded as that codeword using minimum distance decoding. We see from Figure 8.3

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 24 of 47

that the decoder can detect all error patterns of $d_{\min} - 1$ errors. In fact, a decoder for an (n, k) code can detect $2^n - 2^k$ possible error patterns. The reason is that there are $2^k - 1$ nondetectable errors, corresponding to the case where a corrupted codeword is exactly equal to a codeword in the set of possible codewords (of size 2^k) that is not equal to the transmitted codeword. Since there are $2^n - 1$ total possible error patterns, this yields $2^n - 2^k$ detectable error patterns. Note that this is not hard decision decoding because we are not correcting errors, just detecting them.

EXAMPLE 8.4: A $(5, 2)$ code has codewords $\mathbf{C}_0 = [00000]$, $\mathbf{C}_1 = [01011]$, $\mathbf{C}_2 = [10101]$, and $\mathbf{C}_3 = [11110]$. Suppose the all-zero codeword $\mathbf{C}_0$ is transmitted. Find the set of error patterns corresponding to nondetectable errors for this codeword transmission.

Solution: The nondetectable error patterns correspond to the three nonzero codewords. That is, $\mathbf{e}_1 = [01011]$, $\mathbf{e}_2 = [10101]$, and $\mathbf{e}_3 = [11110]$ are nondetectable error patterns, since adding any of these to $\mathbf{C}_0$ results in a valid codeword.

8.2.6 Probability of Error for HDD in AWGN

The probability of codeword error, P_e, is defined as the probability that a transmitted codeword is decoded in error. Under hard decision decoding a received codeword *may* be decoded in error if it contains more than t errors (it will not be decoded in error if there is not an alternative codeword closer to the received codeword than the transmitted codeword). The error probability is thus bounded above by the probability that more than t errors occur. Since the bit errors in a codeword occur independently on an AWGN channel, this probability is given by

$$P_e \leq \sum_{j=t+1}^{n} \binom{n}{j} p^j (1-p)^{n-j}, \tag{8.34}$$

where p is the probability of error associated with transmission of the bits in the codeword. Thus, p corresponds to the error probability associated with uncoded modulation for the given energy per codeword symbol, as treated in Chapter 6 for AWGN channels. For example, if the codeword symbols are sent via coherent BPSK modulation then $p = Q\left(\sqrt{2E_c/N_0}\right)$, where E_c is the energy per codeword symbol and $N_0/2$ is the noise power spectral density. Since there are k/n information bits per codeword symbol, the relationship between the energy per bit and the energy per symbol is $E_c = kE_b/n$. Thus, powerful block codes with a large number of parity bits (k/n small) reduce the channel energy per symbol and therefore increase the error probability in demodulating the codeword symbols. However, the error correction capability of these codes typically more than compensates for this reduction, especially at high SNRs. At low SNRs this may not happen, in which case the code exhibits *negative coding gain* – it has a higher error probability than uncoded modulation. The bound (8.34) holds with equality when the decoder corrects exactly t or fewer errors in a codeword and cannot correct for more than t errors in a codeword. A code with this property is called a *perfect* code.

At high SNRs, the most likely way to make a codeword error is to mistake a codeword for one of its nearest neighbors. Nearest neighbor errors yield a pair of upper and lower bounds

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 25 of 47

on error probability. The lower bound is the probability of mistaking a codeword for a given nearest neighbor at distance $d_{\min}$:

$$P_e \geq \sum_{j=t+1}^{d_{\min}} \binom{d_{\min}}{j} p^j (1-p)^{d_{\min}-j}. \tag{8.35}$$

The upper bound, a union bound, assumes that all of the other $2^k - 1$ codewords are at distance $d_{\min}$ from the transmitted codeword. Thus, the union bound is just $2^k - 1$ times (8.35), the probability of mistaking a given codeword for a nearest neighbor at distance $d_{\min}$:

$$P_e \leq (2^k - 1) \sum_{j=t+1}^{d_{\min}} \binom{d_{\min}}{j} p^j (1-p)^{d_{\min}-j}. \tag{8.36}$$

When the number of codewords is large or the SNR is low, both of these bounds are quite loose.

A tighter upper bound can be obtained by applying the Chernoff bound, $P(X \geq x) \leq e^{-x^2/2}$ for X a zero-mean, unit-variance, Gaussian random variable, to compute codeword error probability. Using this bound, it can be shown [3, Chap. 5.2] that the probability of decoding the all-zero codeword as the jth codeword with weight w_j is upper bounded by

$$P(w_j) \leq [4p(1-p)]^{w_j/2}. \tag{8.37}$$

Since the probability of decoding error is upper bounded by the probability of mistaking the all-zero codeword for any of the other codewords, we obtain the upper bound

$$P_e \leq \sum_{j=1}^{2^k-1} [4p(1-p)]^{w_j/2}. \tag{8.38}$$

This bound requires the weight distribution $\{w_j\}_{j=1}^{2^k-1}$ for all codewords (other than the all-zero codeword corresponding to $j = 0$) in the code. A simpler, slightly looser upper bound is obtained from (8.38) by using $d_{\min}$ instead of the individual codeword weights. This simplification yields the bound

$$P_e \leq (2^k - 1)[4p(1-p)]^{d_{\min}/2}. \tag{8.39}$$

Note that the probability of codeword error P_e depends on p, which is a function of the Euclidean distance between modulation points associated with the transmitted codeword symbols. In fact, the best codes for AWGN channels should not be based on Hamming distance: they should be based on maximizing the Euclidean distance between the codewords after modulation. However, this requires that the channel code be designed jointly with the modulation. This is the basic concept of coded modulation, which will be discussed in Section 8.7. However, Hamming distance is a better measure of code performance in fading when codes are combined with interleaving, as discussed in Section 8.8.

The probability of bit error after decoding the received codeword depends in general on the specific code and decoder and in particular on how bits are mapped to codewords, as in the bit mapping procedure associated with nonbinary modulation. This bit error probability is often approximated as

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 26 of 47

$$P_b \approx \frac{1}{n} \sum_{j=t+1}^{n} j \binom{n}{j} p^j (1-p)^{n-j} \tag{8.40}$$

[2, Chap. 6.5]; for $t = 1$, this can be simplified [2] to $P_b \approx p - p(1-p)^{n-1}$.

EXAMPLE 8.5: Consider a (24, 12) linear block code with a minimum distance $d_{\min} = 8$ (an extended Golay code, discussed in Section 8.2.8, is one such code). Find P_e based on the loose bound (8.39), assuming the codeword symbols are transmitted over the channel using BPSK modulation with $E_b/N_0 = 10$ dB. Also find P_b for this code using the approximation $P_b = P_e/k$ and compare with the bit error probability for uncoded modulation.

Solution: For $E_b/N_0 = 10$ dB we have $E_c/N_0 = \frac{12}{24}10 = 5$. Thus, $p = Q(\sqrt{10}) = 7.82 \cdot 10^{-4}$. Using this value in (8.39) with $k = 12$ and $d_{\min} = 8$ yields $P_e \leq 3.92 \cdot 10^{-7}$. Using the P_b approximation we get $P_b \approx (1/k)P_e = 3.27 \cdot 10^{-8}$. For uncoded modulation we have $P_b = Q(\sqrt{2E_b/N_0}) = Q(\sqrt{20}) = 3.87 \cdot 10^{-6}$. Thus we obtain over two orders of magnitude performance gain with this code. Note that the loose bound can be orders of magnitude away from the true error probability, so this calculation may significantly underestimate the performance gain of the code.

8.2.7 Probability of Error for SDD in AWGN

The HDD described in the previous section discards information that can reduce probability of codeword error. For example, in BPSK, the transmitted signal constellation is $\pm\sqrt{E_b}$ and the received symbol after matched filtering is decoded as a 1 if it is closer to $\sqrt{E_b}$ and as a 0 if it is closer to $-\sqrt{E_b}$. Thus, the distance of the received symbol from $\sqrt{E_b}$ and $-\sqrt{E_b}$ is not used in decoding, yet this information can be used to make better decisions about the transmitted codeword. When these distances are used in the channel decoder it is called *soft decision decoding* (SDD), since the demodulator does not make a hard decision about whether a 0 or 1 bit was transmitted but rather makes a soft decision corresponding to the distance between the received symbol and the symbol corresponding to a 0-bit or a 1-bit transmission. We now describe the basic premise of SDD for BPSK modulation; these ideas are easily extended to higher-level modulations.

Consider a codeword transmitted over a channel using BPSK. As in the case of HDD, the energy per codeword symbol is $E_c = (k/n)E_b$. If the jth codeword symbol is a 1 it will be received as $r_j = \sqrt{E_c} + n_j$ and if it is a 0 it will be received as $r_j = -\sqrt{E_c} + n_j$, where n_j is the AWGN sample of mean zero and variance $N_0/2$ associated with the receiver. In SDD, given a received codeword $\mathbf{R} = [r_1, \ldots, r_n]$, the decoder forms a *correlation metric* $C(\mathbf{R}, \mathbf{C}_i)$ for each codeword $\mathbf{C}_i$ $(i = 0, \ldots, 2^k - 1)$ in the code and then the decoder chooses the codeword $\mathbf{C}_i$ with the highest correlation metric. The correlation metric is defined as

$$C(\mathbf{R}, \mathbf{C}_i) = \sum_{j=1}^{n} (2c_{ij} - 1) r_j, \tag{8.41}$$

where c_{ij} denotes the jth coded bit in the codeword $\mathbf{C}_i$. If $c_{ij} = 1$ then $2c_{ij} - 1 = 1$ and if $c_{ij} = 0$ then $2c_{ij} - 1 = -1$. Thus the received codeword symbol is weighted by the polarity

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 27 of 47

associated with the corresponding symbol in the codeword for which the correlation metric is being computed. Hence $C(\mathbf{R}, \mathbf{C}_i)$ is large when most of the received symbols have a large magnitude and the same polarity as the corresponding symbols in $\mathbf{C}_i$, is smaller when most of the received symbols have a small magnitude and the same polarity as the corresponding symbols in $\mathbf{C}_i$, and is typically negative when most of the received symbols have a different polarity than the corresponding symbols in $\mathbf{C}_i$. In particular, at very high SNRs, if $\mathbf{C}_i$ is transmitted then $C(\mathbf{R}, \mathbf{C}_i) \approx n\sqrt{E_c}$ while $C(\mathbf{R}, \mathbf{C}_j) < n\sqrt{E_c}$ for $j \neq i$.

For an AWGN channel, the probability of codeword error is the same for any codeword of a linear code. Error analysis is typically easiest when assuming transmission of the all-zero codeword. Let us therefore assume that the all-zero codeword $\mathbf{C}_0$ is transmitted and the corresponding received codeword is $\mathbf{R}$. To correctly decode $\mathbf{R}$, we must have that $C(\mathbf{R}, \mathbf{C}_0) > C(\mathbf{R}, \mathbf{C}_i)$, $i = 1, \ldots, 2^k - 1$. Let w_i denote the Hamming weight of the ith codeword $\mathbf{C}_i$, which equals the number of 1-bits in $\mathbf{C}_i$. Then, conditioned on the transmitted codeword $\mathbf{C}_i$, it follows that $C(\mathbf{R}, \mathbf{C}_i)$ is Gauss distributed with mean $\sqrt{E_c}n(1 - 2w_i/n)$ and variance $nN_0/2$. Note that the correlation metrics are not independent, since they are all functions of $\mathbf{R}$. The probability $P_e(\mathbf{C}_i) = p(C(\mathbf{R}, \mathbf{C}_0) < C(\mathbf{R}, \mathbf{C}_i)$ can be shown to equal the probability that a Gauss-distributed random variable with variance $2w_iN_0$ is less than $-2w_i\sqrt{E_c}$; that is,

$$P_e(\mathbf{C}_i) = Q\left(\frac{2w_i\sqrt{E_c}}{\sqrt{2w_iN_0}}\right) = Q\left(\sqrt{2w_i\gamma_bR_c}\right). \tag{8.42}$$

Then, by the union bound, the probability of error is upper bounded by the sum of pairwise error probabilities relative to each $\mathbf{C}_i$:

$$P_e \leq \sum_{i=1}^{2^k-1} P_e(\mathbf{C}_i) = \sum_{i=1}^{2^k-1} Q\left(\sqrt{2w_i\gamma_bR_c}\right). \tag{8.43}$$

Computing (8.43) requires the weight distribution w_i $(i = 1, \ldots, 2^k - 1)$ of the code. This bound can be simplified by noting that $w_i \geq d_{\min}$, so

$$P_e \leq (2^k - 1)Q\left(\sqrt{2\gamma_bR_cd_{\min}}\right). \tag{8.44}$$

The Chernoff bound on the Q-function is $Q\left(\sqrt{2x}\right) < e^{-x}$. Applying this bound to (8.43) yields

$$P_e \leq (2^k - 1)e^{-\gamma_bR_cd_{\min}} < 2^ke^{-\gamma_bR_cd_{\min}} = e^{-\gamma_bR_cd_{\min}+k\ln 2}. \tag{8.45}$$

Comparing this bound with that of uncoded BPSK modulation,

$$P_b = Q\left(\sqrt{2\gamma_b}\right) < e^{-\gamma_b}, \tag{8.46}$$

we get a dB coding gain of approximately

$$G_c = 10\log_{10}[(\gamma_bR_cd_{\min} - k\ln 2)/\gamma_b] = 10\log_{10}[R_cd_{\min} - (k\ln 2)/\gamma_b]. \tag{8.47}$$

Note that the coding gain depends on the code rate, the number of information bits per codeword, the minimum distance of the code, and the channel SNR. In particular, the coding gain

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 28 of 47

decreases as γ_b decreases, and it becomes negative at sufficiently low SNRs. In general the performance of SDD is about 2–3 dB better than HDD [1, Chap. 8.1].

EXAMPLE 8.6: Find the approximate coding gain of SDD over uncoded modulation for the (24, 12) code with $d_{\min} = 8$ considered in Example 8.5, with $\gamma_b = 10$ dB.

Solution: Setting $\gamma_b = 10$, $R_c = 12/24$, $d_{\min} = 8$, and $k = 12$ in (8.47) yields $G_c = 5$ dB. This significant coding gain is a direct result of the large minimum distance of the code.

8.2.8 Common Linear Block Codes

We now describe some common linear block codes. More details can be found in [1; 2; 3; 4]. The most common type of block code is a Hamming code, which is parameterized by an integer $m \geq 2$. For an (n, k) Hamming code, $n = 2^m - 1$ and $k = 2^m - m - 1$, so $n - k = m$ redundant bits are introduced by the code. The minimum distance of all Hamming codes is $d_{\min} = 3$, so $t = 1$ error in the $n = 2^m - 1$ codeword symbols can be corrected. Although Hamming codes are not very powerful, they are perfect codes and thus have probability of error given exactly by the right side of (8.34).

Golay and extended Golay codes are another class of channel codes with good performance. The Golay code is a linear (23, 12) code with $d_{\min} = 7$ and $t = 3$. The extended Golay code is obtained by adding a single parity bit to the Golay code, resulting in a (24, 12) block code with $d_{\min} = 8$ and $t = 3$. The extra parity bit does not change the error correction capability (since t remains the same), but it greatly simplifies implementation because the information bit rate is exactly half the coded bit rate. Thus, both uncoded and coded bit streams can be generated by the same clock, using every other clock sample to generate the uncoded bits. These codes have higher $d_{\min}$ and thus better error correction capabilities than Hamming codes, but at a cost of more complex decoding and a lower code rate $R_c = k/n$. The lower code rate implies that the code either has a lower data rate or requires additional bandwidth.

Another powerful class of block codes is the Bose–Chadhuri–Hocquenghem (BCH) codes. These are cyclic codes, and at high rates they typically outperform all other block codes with the same n and k at moderate to high SNRs. This code class provides a large selection of blocklengths, code rates, and error correction capabilities. In particular, the most common BCH codes have $n = 2^m - 1$ for any integer $m \geq 3$.

The P_b for a number of BCH codes under hard decision decoding and coherent BPSK modulation is shown in Figure 8.4. The plot is based on the approximation (8.40), where for coherent BPSK we have

$$p = Q\left(\sqrt{\frac{2E_c}{N_0}}\right) = Q\left(\sqrt{2R_c\gamma_b}\right). \tag{8.48}$$

In the figure, the BCH (127, 36) code actually has a negative coding gain at low SNRs. This is not uncommon for powerful channel codes owing to their reduced energy per symbol, as discussed in Section 8.2.6.

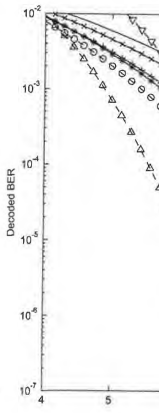


Figure 8.4: P_b for differ

8.2.9 Nonbinary B

A nonbinary block c
symbols mapped into
codeword are chosen
can take any value in
symbol.

The most commo
range of applications
Solomon codes have
tates the error correct
$t = .5\lfloor N - K \rfloor$ code
tween codewords is c
differ. Reed Solomon
largest possible minim
coder input and outpu
requirements of a give

Because nonbinary
ing to 2^k bits, they are
with 2^k-ary modulatio
possible constellation
probability of mistaki
the transmitted point)
binary code is upper b

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 29 of 47

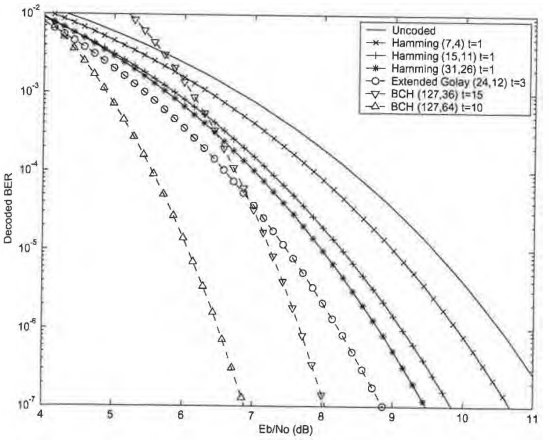


Figure 8.4: P_b for different BCH codes.

8.2.9 Nonbinary Block Codes: The Reed Solomon Code

A nonbinary block code has similar properties as the binary code: it has K information symbols mapped into codewords of length N. However, the N codeword symbols of each codeword are chosen from a nonbinary alphabet of size $q > 2$. Thus the codeword symbols can take any value in $\{0, 1, \ldots, q-1\}$. Usually $q = 2^k$, so that k bits can be mapped into one symbol.

The most common nonbinary block code is the Reed Solomon (RS) code, used in a range of applications from magnetic recording to cellular digital packet data (CDPD). Reed Solomon codes have $N = q - 1 = 2^k - 1$ and $K = 1, 2, \ldots, N-1$. The value of K dictates the error correction capability of the code. Specifically, an RS code can correct up to $t = .5\lfloor N - K \rfloor$ codeword symbol errors. In nonbinary codes the minimum distance between codewords is defined as the number of codeword symbols in which the codewords differ. Reed Solomon codes achieve a minimum distance of $d_{\min} = N - K + 1$, which is the largest possible minimum distance between codewords for any linear code with the same encoder input and output blocklengths. Reed Solomon codes are often *shortened* to meet the requirements of a given system [4, Chap. 5.10].

Because nonbinary codes – and RS codes in particular – generate symbols corresponding to 2^k bits, they are sometimes used with M-ary modulation where $M = 2^k$. In particular, with 2^k-ary modulation each codeword symbol is transmitted over the channel as one of 2^k possible constellation points. If the error probability associated with the modulation (the probability of mistaking the received constellation point for a constellation point other than the transmitted point) is P_M, then the probability of codeword error associated with the nonbinary code is upper bounded by

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 30 of 47

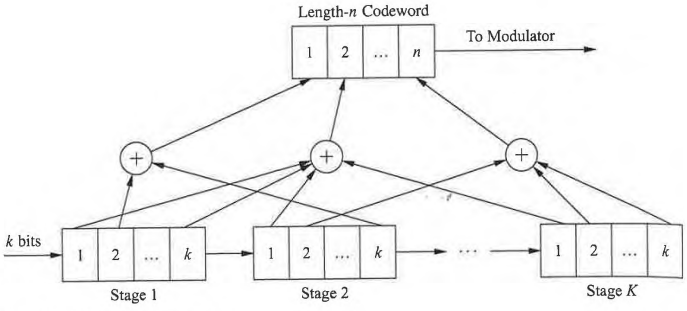


Figure 8.5: Convolutional encoder.

$$P_e \leq \sum_{j=t+1}^{N} \binom{N}{j} P_M^j (1 - P_M)^{N-j}, \tag{8.49}$$

which is similar to the form of (8.34) for the binary code. We can then approximate the probability of information symbol error as

$$P_s \approx \frac{1}{N} \sum_{j=t+1}^{N} j \binom{N}{j} P_M^j (1 - P_M)^{N-j}. \tag{8.50}$$

8.3 Convolutional Codes

A convolutional code generates coded symbols by passing the information bits through a linear finite-state shift register, as shown in Figure 8.5. The shift register consists of K stages with k bits per stage. There are n binary addition operators with inputs taken from all K stages: these operators produce a codeword of length n for each k-bit input sequence. Specifically, the binary input data is shifted into each stage of the shift register k bits at a time, and each of these shifts produces a coded sequence of length n. The rate of the code is $R_c = k/n$. The maximum span of output symbols that can be influenced by a given input bit in a convolutional code is called the *constraint length* of the code. It is clear from Figure 8.5 that a length-n codeword depends on kK input bits – in contrast to a block code, which only depends on k input bits. Thus, the constraint length of the encoder is kK bits or, equivalently, K k-bit bytes. Convolutional codes are said to have memory since the current codeword depends on more input bits (kK) than the number input to the encoder to generate it (k). Note that in general a convolutional encoder may not have the same number of bits per stage.

8.3.1 Code Characterization: Trellis Diagrams

When a length-n codeword is generated by the convolutional encoder of Figure 8.5, this codeword depends both on the k bits input to the first stage of the shift register as well as the *state* of the encoder, defined as the contents in the other $K - 1$ stages of the shift register. In order

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 31 of 47

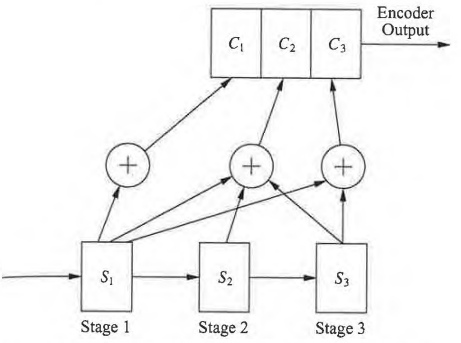


Figure 8.6: Convolutional encoder example ($n = 3$, $k = 1$, $k = 3$).

to characterize this convolutional code, we must characterize how the codeword generation depends both on the k input bits and the encoder state, which has $2^{k(K-1)}$ possible values. There are multiple ways to characterize convolutional codes, including a tree diagram, state diagram, and trellis diagram [1, Chap. 8.2]. The tree diagram represents the encoder in the form of a tree, where each branch represents a different encoder state and the corresponding encoder output. A state diagram is a graph showing the different states of the encoder and the possible state transitions and corresponding encoder outputs. A trellis diagram uses the fact that the tree representation repeats itself once the number of stages in the tree exceeds the constraint length of the code. The trellis diagram simplifies the tree representation by merging nodes in the tree corresponding to the same encoder state. In this section we focus on the trellis representation of a convolutional code, since this is the most common characterization. The details of trellis diagram representation are best described by an example.

Consider the convolutional encoder shown in Figure 8.6 with $n = 3$, $k = 1$, and $K = 3$. In this encoder, one bit at a time is shifted into Stage 1 of the three-stage shift register. At a given time t we denote the bit in Stage i of the shift register as S_i. The three stages of the shift register are used to generate a codeword of length 3, $C_1C_2C_3$; from the figure we see that $C_1 = S_1$, $C_2 = S_1 + S_2 + S_3$, and $C_3 = S_1 + S_3$. A bit sequence $\mathbf{U}$ shifted into the encoder generates a sequence of coded symbols, which we denote by $\mathbf{C}$. Note that the coded symbols corresponding to C_1 are just the original information bits. As with block codes, when one of the coded symbols in a convolutional code corresponds to the original information bits we say that the code is *systematic*. We define the encoder state as $S = S_2S_3$ (i.e., the contents of the last two stages of the encoder), and there are $2^2 = 4$ possible values for this encoder state. To characterize the encoder, we must show for each input bit and each possible encoder state what the encoder output will be, and we must also show how the new input bit changes the encoder state for the next input bit.

The trellis diagram for this code is shown in Figure 8.7. The solid lines in Figure 8.7 indicate the encoder state transition when a 0-bit is input to Stage 1 of the encoder, and the dashed lines indicate the state transition corresponding to a 1-bit input. For example, starting

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 32 of 47

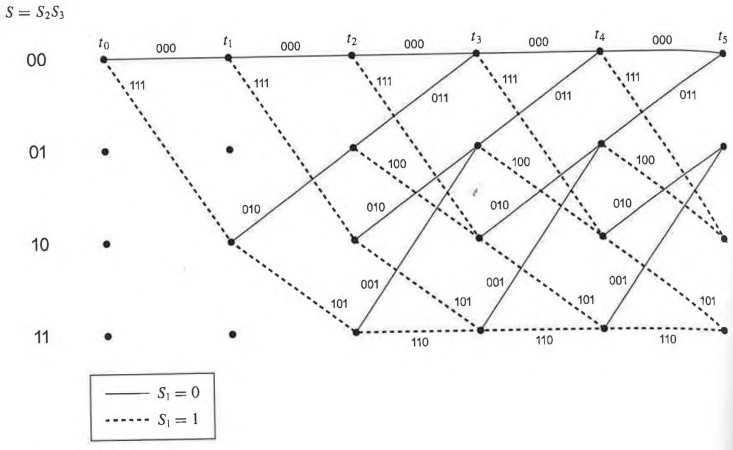
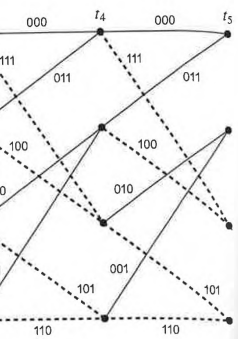


Figure 8.7: Trellis diagram.

at state $S = 00$, if a 0-bit is input to Stage 1 then, when the shift register transitions, the new state will remain as $S = 00$ (since the 0 in Stage 1 transitions to Stage 2, and the 0 in Stage 2 transitions to Stage 3, resulting in the new state $S = S_2S_3 = 00$). On the other hand, if a 1-bit is input to Stage 1 then, when the shift register transitions, the new state will become $S = 10$ (since the 1 in Stage 1 transitions to Stage 2, and the 0 in Stage 2 transitions to Stage 3, resulting in the new state $S = S_2S_3 = 10$). The encoder output corresponding to a particular encoder state S and input S_1 is written next to the transition lines in the figure. This output is the encoder output that results from the encoder addition operations on the bits S_1, S_2, and S_3 in each stage of the encoder. For example, if $S = 00$ and $S_1 = 1$ then the encoder output $C_1C_2C_3$ has $C_1 = S_1 = 1$, $C_2 = S_1 + S_2 + S_3 = 1$, and $C_3 = S_1 + S_3 = 1$. This output 111 is drawn next to the dashed line transitioning from state $S = 00$ to state $S = 10$ in Figure 8.7. Note that the encoder output for $S_1 = 0$ and $S = 00$ is always the all-zero codeword regardless of the addition operations that form the codeword $C_1C_2C_3$, since summing together any number of 0s always yields 0. The portion of the trellis between time t_i and t_{i+1} is called the ith *branch* of the trellis. Figure 8.7 indicates that the initial state at time t_0 is the all-zero state. The trellis achieves *steady state,* defined as the point where all states can be entered from either of two preceding states, at time t_3. After this steady state is reached, the trellis repeats itself in each time interval. Note also that, in steady state, each state transitions to one of two possible new states. In general, trellis structures starting from the all-zero state at time t_0 achieve steady state at time t_K.

For general values of k and K, the trellis diagram will have 2^{K-1} states, where each state has 2^k paths entering each node and 2^k paths leaving each node. Thus, the number of paths through the trellis grows exponentially with k, K, and the length of the trellis path.

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 33 of 47



register transitions, the new
Stage 2, and the 0 in Stage 2
. On the other hand, if a 1-bit
ew state will become $S = 10$
2 transitions to Stage 3, re-
corresponding to a particular
nes in the figure. This output
rations on the bits S_1, S_2, and
$= 1$ then the encoder output
$+ S_3 = 1$. This output 111 is
to state $S = 10$ in Figure 8.7.
he all-zero codeword regard-
since summing together any
een time t_i and t_{i+1} is called
state at time t_0 is the all-zero
here all states can be entered
dy state is reached, the trellis
state, each state transitions to
tarting from the all-zero state

2^{K-1} states, where each state
le. Thus, the number of paths
th of the trellis path.

EXAMPLE 8.7: Consider the convolutional code represented by the trellis in Figure 8.7. For an initial state $S = S_2S_3 = 01$, find the state sequence S and the encoder output C for input bit sequence $\mathbf{U} = 011$.

Solution: The first occurrence of $S = 01$ in the trellis is at time t_2. We see at t_2 that if the information bit $S_1 = 0$ then we follow the solid line in the trellis from $S = 01$ at t_2 to $S = 00$ at t_3, and the output corresponding to this path through the trellis is $C = 011$. Now at t_3, starting at $S = 00$, for the information bit $S_1 = 1$ we follow the dashed line in the trellis to $S = 10$ at t_4, and the output corresponding to this path through the trellis is $C = 111$. Finally, at t_4, starting at $S = 10$, for the information bit $S_1 = 1$ we follow the dashed line in the trellis to $S = 11$ at t_5, and the output corresponding to this path through the trellis is $C = 101$.

8.3.2 Maximum Likelihood Decoding

The convolutional code generated by the finite state shift register is basically a finite-state machine. Thus, unlike an (n, k) block code – where maximum likelihood detection entails finding the length-n codeword that is closest to the received length-n codeword – maximum likelihood detection of a convolutional code entails finding the most likely sequence of coded symbols $\mathbf{C}$ given the received sequence of coded symbols, which we denote by $\mathbf{R}$. In particular, for a received sequence $\mathbf{R}$, the decoder decides that coded symbol sequence $\mathbf{C}^*$ was transmitted if

$$p(\mathbf{R} \mid \mathbf{C}^*) \geq p(\mathbf{R} \mid \mathbf{C}) \quad \forall \mathbf{C}. \tag{8.51}$$

Since each possible sequence $\mathbf{C}$ corresponds to one path through the trellis diagram of the code, maximum likelihood decoding corresponds to finding the maximum likelihood path through the trellis diagram. For an AWGN channel, noise affects each coded symbol independently. Thus, for a convolutional code of rate $1/n$, we can express the likelihood (8.51) for a path of length L through the trellis as

$$p(\mathbf{R} \mid \mathbf{C}) = \prod_{i=0}^{L-1} p(R_i \mid C_i) = \prod_{i=0}^{L-1} \prod_{j=1}^{n} p(R_{ij} \mid C_{ij}), \tag{8.52}$$

where C_i is the portion of the code sequence $\mathbf{C}$ corresponding to the ith branch of the trellis, R_i is the portion of the received code sequence $\mathbf{R}$ corresponding to the ith branch of the trellis, C_{ij} is the jth coded symbol corresponding to C_i, and R_{ij} is the jth received coded symbol corresponding to R_i. The log likelihood function is defined as the log of $p(\mathbf{R} \mid \mathbf{C})$, given as

$$\log p(\mathbf{R} \mid \mathbf{C}) = \sum_{i=0}^{L-1} \log p(R_i \mid C_i) = \sum_{i=0}^{L-1} \sum_{j=1}^{n} \log p(R_{ij} \mid C_{ij}). \tag{8.53}$$

The expression

$$B_i = \sum_{j=1}^{n} \log p(R_{ij} \mid C_{ij}) \tag{8.54}$$

is called the *branch metric* because it indicates the component of (8.53) associated with the ith branch of the trellis. The sequence or path that maximizes the likelihood function also

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 34 of 47

maximizes the log likelihood function, since the log is monotonically increasing. However, it is computationally more convenient for the decoder to use the log likelihood function because it involves a summation rather than a product. The log likelihood function associated with a given path through the trellis is also called the *path metric,* which by (8.53) is equal to the sum of branch metrics along each branch of the path. The path through the trellis with the maximum path metric corresponds to the maximum likelihood path.

The decoder can use either hard or soft decisions for the expressions $\log p(R_{ij} \mid C_{ij})$ in the log likelihood metric. For hard decision decoding, the R_{ij} is decoded as a 1 or a 0. The probability of hard decision decoding error depends on the modulation and is denoted as p. If $\mathbf{R}$ and $\mathbf{C}$ are N symbols long and differ in d places (i.e., their Hamming distance is d), then

$$p(\mathbf{R} \mid \mathbf{C}) = p^d(1-p)^{N-d}$$

and

$$\log p(\mathbf{R} \mid \mathbf{C}) = -d \log \frac{1-p}{p} + N \log(1-p). \tag{8.55}$$

Since $p < .5$, (8.55) is maximized when d is minimized. So the coded sequence $\mathbf{C}$ with minimum Hamming distance to the received sequence $\mathbf{R}$ corresponds to the maximum likelihood sequence.

In soft decision decoding, the value of the received coded symbols (R_{ij}) are used directly in the decoder, rather than quantizing them to 1 or 0. For example, if the C_{ij} are sent via BPSK over an AWGN channel with a 1 mapped to $\sqrt{E_c}$ and a 0 mapped to $-\sqrt{E_c}$, then

$$R_{ij} = \sqrt{E_c}(2C_{ij} - 1) + n_{ij}, \tag{8.56}$$

where $E_c = kE_b/n$ is the energy per coded symbol and n_{ij} denotes Gaussian noise of mean zero and variance $\sigma^2 = .5N_0$. Thus,

$$p(R_{ij} \mid C_{ij}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{(R_{ij} - \sqrt{E_c}(2C_{ij}-1))^2}{2\sigma^2}\right]. \tag{8.57}$$

Maximizing this likelihood function is equivalent to choosing the C_{ij} that is closest in Euclidean distance to R_{ij}. In determining which sequence $\mathbf{C}$ maximizes the log likelihood function (8.53), any terms that are common to two different sequences $\mathbf{C}_1$ and $\mathbf{C}_2$ can be neglected, since they contribute the same amount to the summation. Similarly, we can scale all terms in (8.53) without changing the maximizing sequence. Thus, by neglecting scaling factors and terms in (8.57) that are common to any C_{ij}, we can replace $\sum_{j=1}^{n} \log p(R_{ij} \mid C_{ij})$ in (8.53) with the *equivalent branch metric*

$$\mu_i = \sum_{j=1}^{n} R_{ij}(2C_{ij} - 1) \tag{8.58}$$

and obtain the same maximum likelihood output.

We now illustrate the path metric computation under both hard and soft decisions for the convolutional code of Figure 8.6 with the trellis diagram in Figure 8.7. For simplicity, we will consider only two possible paths through the trellis and compute their corresponding likelihoods for a given received sequence $\mathbf{R}$. Assume we start at time t_0 in the all-zero state.

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 35 of 47

The first path we consider is the all-zero path, corresponding to the all-zero input sequence. The second path we consider starts in state $S = 00$ at time t_0 and transitions to state $S = 10$ at time t_1, then to state $S = 01$ at time t_2, and finally to state $S = 00$ at time t_3, at which point this path merges with the all-zero path. Since the paths and therefore their branch metrics at times $t < t_0$ and $t \geq t_3$ are the same, the maximum likelihood path corresponds to the path whose sum of branch metrics over the branches in which the two paths differ is smaller. From Figure 8.7 we see that the all-zero path through the trellis generates the coded sequence $\mathbf{C}_0 = 000000000$ over the first three branches in the trellis. The second path generates the coded sequence $\mathbf{C}_1 = 111010011$ over the first three branches in the trellis.

Let us first consider hard decision decoding with error probability p. Suppose the received sequence over these three branches is $\mathbf{R} = 100110111$. Note that the Hamming distance between $\mathbf{R}$ and $\mathbf{C}_0$ is 6 while the Hamming distance between $\mathbf{R}$ and $\mathbf{C}_1$ is 4. As discussed previously, the most likely path therefore corresponds to $\mathbf{C}_1$, since it has minimum Hamming distance to $\mathbf{R}$. The path metric for the all-zero path is

$$M_0 = \sum_{i=0}^{2} \sum_{j=1}^{3} \log P(R_{ij} \mid C_{ij}) = 6 \log p + 3 \log(1 - p), \tag{8.59}$$

while the path metric for the other path is

$$M_1 = \sum_{i=0}^{2} \sum_{j=1}^{3} \log P(R_{ij} \mid C_{ij}) = 4 \log p + 5 \log(1 - p). \tag{8.60}$$

Assuming $p \ll 1$, which is generally the case, this yields $M_0 \approx 6 \log p$ and $M_1 \approx 4 \log p$. Since $\log p < 1$, this confirms that the second path has a larger path metric than the first.

Let us now consider soft decision decoding over time t_0 to t_3. Suppose the received sequence (before demodulation) over these three branches, for $E_c = 1$, is $\mathbf{R} = (.8, -.35, -.15, 1.35, 1.22, -.62, .87, 1.08, .91)$. The path metric for the all-zero path is

$$M_0 = \sum_{i=0}^{2} \mu_i = \sum_{i=0}^{2} \sum_{j=1}^{3} R_{ij}(2C_{ij} - 1) = \sum_{i=0}^{2} \sum_{j=1}^{3} -R_{ij} = -5.11,$$

and the path metric for the second path is

$$M_1 = \sum_{i=0}^{2} \sum_{j=1}^{3} R_{ij}(2C_{ij} - 1) = 1.91.$$

Thus, the second path has a higher path metric than the first. In order to determine if the second path is the maximum likelihood path, we must compare its path metric to that of all other paths through the trellis.

The difficulty with maximum likelihood decoding is that the complexity of computing the log likelihood function (8.53) grows exponentially with the memory of the code, and this computation must be done for every possible path through the trellis. The Viterbi algorithm, discussed in the next section, reduces the complexity of maximum likelihood decoding by taking advantage of the structure of the path metric computation.

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 36 of 47

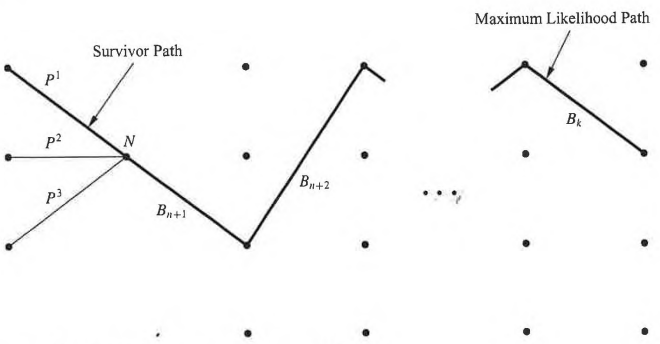


Figure 8.8: Partial path metrics on maximum likelihood path.

8.3.3 The Viterbi Algorithm

The Viterbi algorithm, introduced by Viterbi in 1967 [5], reduces the complexity of maximum likelihood decoding by systematically removing paths from consideration that cannot achieve the highest path metric. The basic premise is to look at the partial path metrics associated with all paths *entering* a given node (node N) in the trellis. Since the possible paths through the trellis *leaving* node N are the same for each *entering* path, the complete trellis path with the highest path metric that goes through node N must coincide with the path that has the highest partial path metric up to node N. This is illustrated in Figure 8.8, where path 1, path 2, and path 3 enter node N (at trellis depth n) with partial path metrics $P^l = \sum_{k=0}^{n-1} B_k^l$ $(l = 1, 2, 3)$ up to this node. Assume P^1 is the largest of these partial path metrics. The complete path with the highest metric has branch metrics $\{B_k\}$ after node N. The maximum likelihood path starting from node N (i.e., the path starting from node N with the largest path metric) has partial path metric $\sum_{k=n}^{\infty} B_k$. The complete path metric for path l $(l = 1, 2, 3)$ up to node N and the maximum likelihood path after node N is $P^l + \sum_{k=n}^{\infty} B_k$ $(l = 1, 2, 3)$, and thus the path with the maximum partial path metric P^l up to node N (path 1 in this example) must correspond to the path with the largest path metric that goes through node N.

The Viterbi algorithm takes advantage of this structure by discarding all paths entering a given node except the path with the largest partial path metric up to that node. The path that is not discarded is called the *survivor path*. Thus, for the example of Figure 8.8, path 1 is the survivor at node N and paths 2 and 3 are discarded from further consideration. Hence, at every stage in the trellis there are 2^{K-1} surviving paths, one for each possible encoder state. A branch for a given stage of the trellis cannot be decoded until all surviving paths at a subsequent trellis stage overlap with that branch; see Figure 8.9, which shows the surviving paths at time t_{k+3}. We see in the figure that all of these surviving paths can be traced back to a *common stem* from time t_k to t_{k+1}. At this point the decoder can output the codeword symbol C_i associated with this branch of the trellis. Note that there is not a fixed decoding delay associated with how far back in the trellis a common stem occurs for a given set of surviving paths,

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 37 of 47

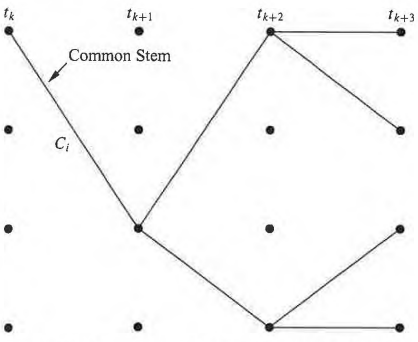


Figure 8.9: Common stem for all survivor paths in the trellis.

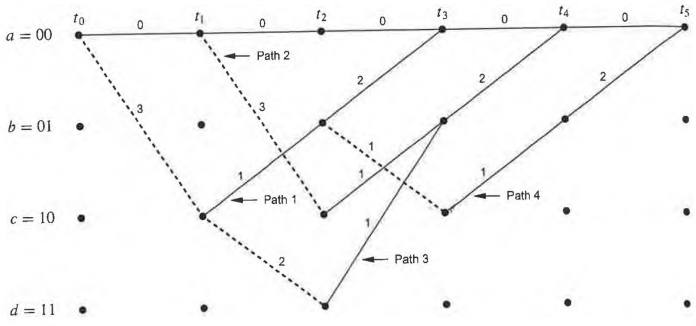
since this delay depends on k, K, and the specific code properties. To avoid a random decoding delay, the Viterbi algorithm is typically modified so that, at a given stage in the trellis, the most likely branch n stages back is decided upon based on the partial path metrics up to that point. Although this modification does not yield exact maximum likelihood decoding, for n sufficiently large (typically $n \geq 5K$) it is a good approximation.

The Viterbi algorithm must keep track of $2^{k(K-1)}$ surviving paths and their corresponding metrics. At each stage, in order to determine the surviving path, 2^k metrics must be computed for each node corresponding to the 2^k paths entering each node. Thus, the number of computations in decoding and the memory requirements for the algorithm increase exponentially with k and K. This implies that practical implementations of convolutional codes are restricted to relatively small values of k and K.

8.3.4 Distance Properties

As with block codes, the error correction capability of convolutional codes depends on the distance between codeword sequences. Since convolutional codes are linear, the minimum distance between all codeword sequences can be found by determining the minimum distance from any sequence or, equivalently, determining any trellis path to the all-zero sequence/trellis path. Clearly the trellis path with minimum distance to the all-zero path will diverge and remerge with the all-zero path, so that the two paths coincide except over some number of trellis branches. To find this minimum distance path we must consider all paths that diverge from the all-zero state and then remerge with this state. As an example, in Figure 8.10 we draw all paths in Figure 8.7 between times t_0 and t_5 that diverge and remerge with the all-zero state. Note that path 2 is identical to path 1 – just shifted in time – and thus is not considered as a separate path. Note also that we could look over a longer time interval, but any paths that diverge and remerge over this longer interval would traverse the same branches (shifted in time) as one of these paths plus some additional branches and would therefore have larger path metrics. In particular, we see that path 4 traverses the trellis branches 00-10-01-10-01-00, whereas path 1 traverses the branches 00-10-01-00. Since

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 38 of 47



Path 1 and 2: 00–10–01–00
Path 3: 00–10–11–01–00
Path 4: 00–10–01–10–01–00

Figure 8.10: Path distances to the all-zero path.

path 4 traverses the same branches as path 1 on its first, second, and last transition – and since it has additional transitions – its path metric will be no smaller than the meteric of path 1. Thus we need not consider a longer time interval to find the minimum distance path. For each path in Figure 8.10 we label the Hamming distance of the codeword on each branch to the all-zero codeword in the corresponding branch of the all-zero path. By summing up the Hamming distances on all branches of each path, we see that path 1 has a Hamming distance of 6 and that paths 3 and 4 have Hamming distances of 8. Recalling that dashed lines indicate 1-bit inputs while solid lines indicate 0-bit inputs, we see that path 1 corresponds to an input bit sequence from t_0 to t_5 of 10000, path 3 corresponds to an input bit sequence of 11000, and path 4 corresponds to an input bit sequence of 10100. Thus, path 1 results in one bit error relative to the all-zero squence, and paths 3 and 4 result in two bit errors.

We define the *minimum free distance* d_f of a convolutional code, also called simply the free distance, to be the minimum Hamming distance of all paths through the trellis to the all-zero path, which for this example is 6. The error correction capability of the code is obtained in the same manner as for block codes, with $d_{\min}$ replaced by d_f, so that the code can correct t channels errors with $t = \lfloor .5d_f \rfloor$.

8.3.5 State Diagrams and Transfer Functions

The transfer function of a convolutional code is used to characterize paths that diverge and remerge from the all-zero path, and it is also used to obtain probability of error bounds. The transfer function is obtained from the code's state diagram representing possible transitions from the all-zero state to the all-zero state. The state diagram for the code illustrated in Figure 8.7 is shown in Figure 8.11, with the all-zero state $a = 00$ split into a second node e to facilitate representing paths that begin and end in this state. Transitions between states due to a 0 input bit are represented by solid lines, while transitions due to a 1 input bit are represented

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 39 of 47

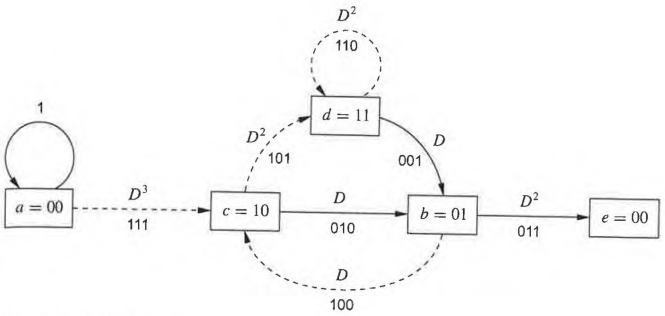


Figure 8.11: State diagram.

by dashed lines. The branches of the state diagram are labeled as either $D^0 = 1$, D^1, D^2, or D^3, where the exponent of D corresponds to the Hamming distance between the codeword (which is shown for each branch transition) and the all-zero codeword in the all-zero path. The self-loop in node a can be ignored because it does not contribute to the distance properties of the code.

The state diagram can be represented by state equations for each state. For the example of Figure 8.11, we obtain state equations corresponding to the four states

$$X_c = D^3 X_a + D X_b, \quad X_b = D X_c + D X_d, \quad X_d = D^2 X_c + D^2 X_d, \quad X_e = D^2 X_b, \tag{8.61}$$

where $X_a, \ldots, X_e$ are dummy variables characterizing the partial paths. The transfer function of the code, describing the paths from state a to state e, is defined as $T(D) = X_e / X_a$. By solving the state equations for the code, which can be done using standard techniques such as Mason's formula, we obtain a transfer function of the form

$$T(D) = \sum_{d=d_f}^{\infty} a_d D^d, \tag{8.62}$$

where a_d is the number of paths with Hamming distance d from the all-zero path. As stated before, the minimum Hamming distance to the all-zero path is d_f, and the transfer function $T(D)$ indicates that there are a_{d_f} paths with this minimum distance. For the example of Figure 8.11, we can solve the state equations given in (8.61) to get the transfer function

$$T(D) = \frac{D^6}{1 - 2D^2} = D^6 + 2D^8 + 4D^{10} + \cdots. \tag{8.63}$$

We see from the transfer function that there is one path with minimum distance $d_f = 6$ and two paths with Hamming distance 8, which is consistent with Figure 8.10. The transfer function is a convenient shorthand for enumerating the number and corresponding Hamming distance of all paths in a particular code that diverge and later remerge with the all-zero path.

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 40 of 47

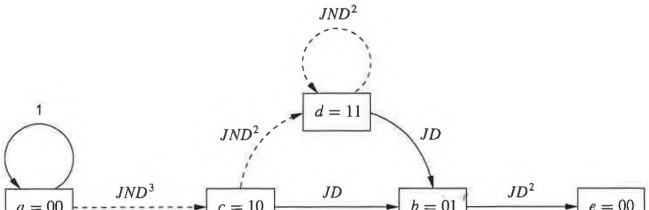


Figure 8.12: Extended state diagram.

Although the transfer function is sufficient to capture the number and Hamming distance of paths in the trellis to the all-zero path, we need a more detailed characterization to compute the bit error probability of the convolutional code. We therefore introduce two additional parameters into the transfer function, N and J, for this additional characterization. The factor N is introduced on all branch transitions associated with a 1 input bit (dashed lines in Figure 8.11). The factor J is introduced to every branch in the state diagram such that the exponent of J in the transfer function equals the number of branches in any given path from node a to node e. The extended state diagram corresponding to the trellis of Figure 8.7 is shown in Figure 8.12.

The extended state diagram can also be represented by state equations. For the example of Figure 8.12, these are given by:

$$\begin{aligned} X_c &= JND^3X_a + JNDX_b, \qquad X_b = JDX_c + JDX_d, \\ X_d &= JND^2X_c + JND^2X_d, \qquad X_e = JD^2X_b. \end{aligned} \tag{8.64}$$

Similarly to the previous transfer function definition, the transfer function associated with this extended state is defined as $T(D, N, J) = X_e/X_a$, which for this example yields

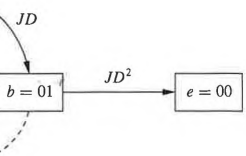
$$\begin{aligned} T(D, N, J) &= \frac{J^3ND^6}{1 - JND^2(1 + J)} \\ &= J^3ND^6 + J^4N^2D^8 + J^5N^2D^8 + J^5N^3D^{10} + \cdots . \end{aligned} \tag{8.65}$$

The factor J is most important when we are interested in transmitting finite-length sequences; for infinite-length sequences we typically set $J = 1$ to obtain the transfer function for the extended state:

$$T(D, N) = T(D, N, J = 1). \tag{8.66}$$

The transfer function for the extended state tells us more information about the diverging and remerging paths than just their Hamming distance; namely, the minimum distance path with Hamming distance 6 is of length 3 and results in a single bit error (exponent of N is unity), one path of Hamming distance 8 is of length 4 and results in two bit errors, and

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 41 of 47



ure the number and Hamming distance
more detailed characterization to com-
. We therefore introduce two additional
is additional characterization. The fac-
ted with a 1 input bit (dashed lines in
anch in the state diagram such that the
ber of branches in any given path from
sponding to the trellis of Figure 8.7 is

ed by state equations. For the example

$$\begin{aligned} {}_b &= JDX_c + JDX_d, \\ X_e &= JD^2X_b. \end{aligned} \tag{8.64}$$

, the transfer function associated with
, which for this example yields

$$N^2D^8 + J^5N^3D^{10} + \cdots . \tag{8.65}$$

in transmitting finite-length sequences;
to obtain the transfer function for the

$$J = 1). \tag{8.66}$$

us more information about the diverg-
stance; namely, the minimum distance
lts in a single bit error (exponent of N
gth 4 and results in two bit errors, and

the other path of Hamming distance 8 is of length 5 and results in two bit errors, consistent with Figure 8.10. The extended transfer function is a convenient shorthand for representing the Hamming distance, length, and number of bit errors that correspond to each diverging and remerging path of a code from the all-zero path. In the next section we show that this representation is useful in characterizing the probability of error for convolutional codes.

8.3.6 Error Probability for Convolutional Codes

Since convolutional codes are linear codes, the probability of error can be obtained by first assuming that the all-zero sequence is transmitted and then determining the probability that the decoder decides in favor of a different sequence. We will consider error probability for both hard decision and soft decision decoding; soft decisions are much more common in wireless systems owing to their superior performance.

First consider soft decision decoding. We are interested in the probability that the all-zero sequence is sent but a different sequence is decoded. If the coded symbols output from the convolutional encoder are sent over an AWGN channel using coherent BPSK modulation with energy $E_c = R_cE_b$, then it can be shown [1] that, if the all-zero sequence is transmitted, the probability of mistaking this sequence with a sequence Hamming distance d away is

$$P_2(d) = Q\left(\sqrt{\frac{2E_c}{N_0}d}\right) = Q\left(\sqrt{2\gamma_b R_c d}\right). \tag{8.67}$$

We call this probability the *pairwise error probability,* since it is the error probability associated with a pairwise comparison of two paths that differ in d bits. The transfer function enumerates all paths that diverge and remerge with the all zero path, so by the union bound we can upper bound the probability of mistaking the all-zero path for another path through the trellis as

$$P_e \leq \sum_{d=d_f}^{\infty} a_d Q\left(\sqrt{2\gamma_b R_c d}\right), \tag{8.68}$$

where a_d denotes the number of paths of distance d from the all-zero path. This bound can be expressed in terms of the transfer function itself if we use the Chernoff upper bound for the Q-function, which yields

$$Q\left(\sqrt{2\gamma_b R_c d}\right) \leq e^{-\gamma_b R_c d}.$$

Using this in (8.68) we obtain the upper bound

$$P_e < T(D)|_{D=e^{-\gamma_b R_c}}. \tag{8.69}$$

This upper bound tells us the probability of mistaking one sequence for another, but it does not yield the more fundamental probability of bit error. We know that the exponent in the factor N of $T(D, N)$ indicates the number of information bit errors associated with selecting an incorrect path through the trellis. Specifically, we can express $T(D, N)$ as

$$T(D, N) = \sum_{d=d_f}^{\infty} a_d D^d N^{f(d)}, \tag{8.70}$$

where $f(d)$ denotes the number of bit errors associated with a path of distance d from the all-zero path. Then we can upper bound the bit error probability for $k = 1$ as

$$P_b \leq \sum_{d=d_f}^{\infty} a_d f(d) Q\left(\sqrt{2\gamma_b R_c d}\right) \tag{8.71}$$

[1, Chap. 8.2], which differs from (8.68) only in the weighting factor $f(d)$ corresponding to the number of bit errors in each incorrect path. If the Q-function is upper bounded using the Chernoff bound as before, we get the upper bound

$$P_b < \left.\frac{dT(D,N)}{dN}\right|_{N=1, D=e^{-\gamma_b R_c}}. \tag{8.72}$$

If $k > 1$ then we divide (8.71) or (8.72) by k to obtain P_b.

All of these bounds assume coherent BPSK transmission (or coherent QPSK, which is equivalent to two independent BPSK transmissions). For other modulations, the pairwise error probability $P_2(d)$ must be recomputed based on the probability of error associated with the given modulation.

Let us now consider hard decision decoding. The probability of selecting an incorrect path at distance d from the all-zero path, for d odd, is given by

$$P_2(d) = \sum_{k=.5(d+1)}^{d} \binom{d}{k} p^k (1-p)^{(d-k)}, \tag{8.73}$$

where p is the probability of error on the channel. This follows because the incorrect path will be selected only if the decoded path is closer to the incorrect path than to the all-zero path – that is, the decoder makes at least $.5(d+1)$ errors. If d is even then the incorrect path is selected when the decoder makes more than $.5d$ errors, and the decoder makes a choice at random if the number of errors is exactly $.5d$. We can bound the pairwise error probability as

$$P_2(d) < [4p(1-p)]^{d/2}. \tag{8.74}$$

Following the same approach as in soft decision decoding, we then obtain the error probability bound as

$$P_e < \sum_{d=d_f}^{\infty} a_d [4p(1-p)]^{d/2} < T(D)|_{D=\sqrt{4p(1-p)}}, \tag{8.75}$$

and

$$P_b < \sum_{d=d_f}^{\infty} a_d f(d) P_2(d) = \left.\frac{dT(D,N)}{dN}\right|_{N=1, D=\sqrt{4p(1-p)}}. \tag{8.76}$$

8.4 Concatenated Codes

A concatenated code uses two levels of coding: an inner code and an outer code, as shown in Figure 8.13. The inner code is typically designed to remove most of the errors introduced by the channel, and the outer code is typically a less powerful code that further reduces error

Figure 8.13: Concatenate

probability when the
most errors are correc
tive at correcting burst
fades. In addition, at
rors that occur in burst
often concatenated wi
correcting properties.
codes separated by an
coding techniques is d

Concatenated code
than a single code wit
nated codes is usually
decoded, and then the
in fact both codes are
mum likelihood decod
complex. It was disco
codes can be obtained
codes, described in the

8.5 Turbo Codes

Turbo codes, introduc
shima ([6]; see also [7
the Shannon capacity
codes on graphs with
some of their characte
graphs were introduce
ing techniques were th
field. The landmark 19
revisit the work of Ga

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 43 of 47

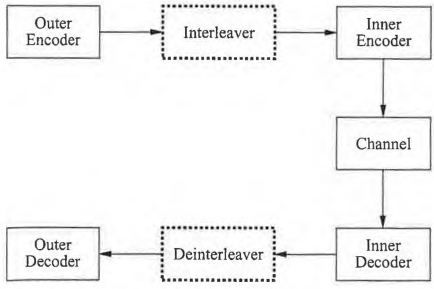


Figure 8.13: Concatenated coding.

probability when the received coded bits have a relatively low probability of error (since most errors are corrected by the inner code). Concatenated codes can be particularly effective at correcting bursts of errors, which are common in wireless channels as a result of deep fades. In addition, at low SNRs Viterbi decoding of a convolutional code tends to have errors that occur in bursts. To compensate for these error bursts, an inner convolutional code is often concatenated with an outer Reed Solomon code, since RS codes have good burst error correcting properties. In addition, concatenated codes frequently have the inner and outer codes separated by an interleaver to break up bursts of errors. Interleaver design for different coding techniques is described in Section 8.8.

Concatenated codes typically achieve very low error probability with less complexity than a single code with the same error probability performance. The decoding of concatenated codes is usually done in two stages, as indicated in the figure: first the inner code is decoded, and then the outer code is decoded separately. This is a suboptimal technique, since in fact both codes are working in tandem to reduce error probability. However, the maximum likelihood decoder for a concatenated code, which performs joint decoding, is highly complex. It was discovered in the mid-1990s that a near-optimal decoder for concatenated codes can be obtained based on iterative decoding. This is the basic premise behind turbo codes, described in the next section.

8.5 Turbo Codes

Turbo codes, introduced in 1993 in a landmark paper by Berrou, Glavieux, and Thitimajshima ([6]; see also [7]), are powerful codes that can come within a fraction of a decibel of the Shannon capacity limit on AWGN channels. Turbo codes and the more general family of codes on graphs with iterative decoding algorithms [8; 9] have been studied extensively, yet some of their characteristics are still not well understood. The main ideas behind codes on graphs were introduced by Gallager in the early sixties [10]; at the time, however, these coding techniques were thought impractical and were generally not pursued by researchers in the field. The landmark 1993 paper on turbo codes [6] provided more than enough motivation to revisit the work of Gallager and others on iterative, graph-based decoding techniques.

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 44 of 47

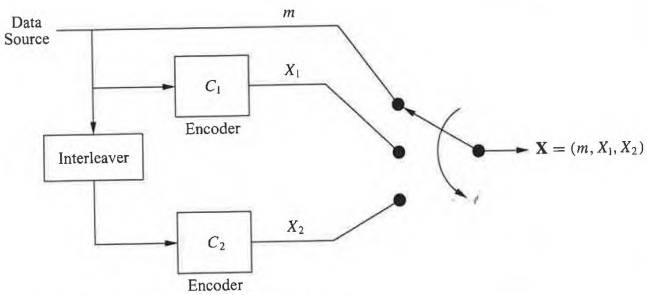


Figure 8.14: Parallel concatenated (turbo) encoder.

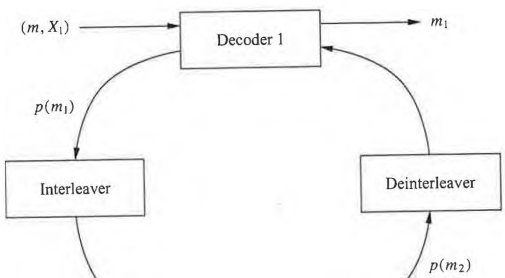


Figure 8.15: Turbo decoder.

As first described by Berrou et al., turbo codes consist of two key components: parallel concatenated encoding and iterative, "turbo" decoding [6; 11]. A typical parallel concatenated encoder is shown in Figure 8.14. It consists of two parallel convolutional encoders separated by an interleaver, with the input to the channel being the data bits m along with the parity bits X_1 and X_2 output from each of the encoders in response to input m. Since the m information bits are transmitted as part of the codeword, we consider this a systematic turbo code. The key to parallel concatenated encoding lies in the recursive nature of the encoders and the impact of the interleaver on the information stream. Interleavers also play a significant role in the reduction of error floors [11], which are commonly exhibited in turbo codes.

Iterative or "turbo" decoding exploits the component-code substructure of the turbo encoder by associating a component decoder with each of the component encoders. More specifically, each decoder performs soft input–soft output decoding, as shown in Figure 8.15 for the example encoder of Figure 8.14. In this figure decoder 1 generates a soft decision in

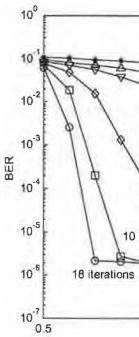


Figure 8.16: Turbo

the form of a prob
ceived codeword
a posteriori (MA
a reliability indic
formation is passe
its received code
mation is input t
and the original r
coder 2, which re
an iterative manne
measures. Ideally
hard decisions m
well-defined, in p
does not converge
proposed for dete
neural net–based

The simulated
in Figure 8.16 for
length $K = 5$ seps
after approximate
turbo codes. Firs
E_b/N_0 of less tha
.5 dB of the Shan
performance of tu
is similar to the c
decoder allows th
the turbo code ex
problematic for sy

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 45 of 47

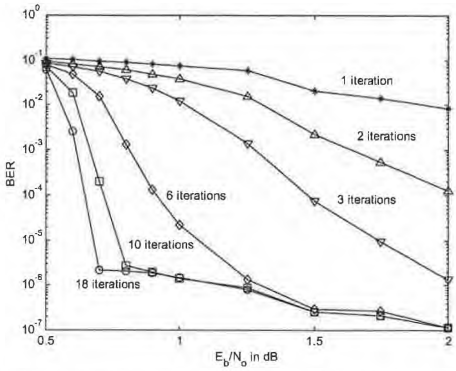


Figure 8.16: Turbo code performance (rate 1/2, $K = 5$ component codes with interleaver depth 2^{16}).

the form of a probability measure $p(m_1)$ on the transmitted information bits based on the received codeword (m, X_1). The probability measure is generated based on either a maximum a posteriori (MAP) probability or a soft output Viterbi algorithm (SOVA), which attaches a reliability indicator to the VA hard decision outputs [4, Chap. 12.5]. This probability information is passed to decoder 2, which generates its own probability measure $p(m_2)$ from its received codeword (m, X_2) and the probability measure $p(m_1)$. This reliability information is input to decoder 1, which revises its measure $p(m_1)$ based on this information and the original received codeword. Decoder 1 sends the new reliability information to decoder 2, which revises its measure using this new information. Turbo decoding proceeds in an iterative manner, with the two component decoders alternately updating their probability measures. Ideally the decoders will eventually agree on probability measures that reduce to hard decisions $m = m_1 = m_2$. However, the stopping condition for turbo decoding is not well-defined, in part because there are many cases in which the turbo decoding algorithm does not converge: the decoders cannot agree on the value of m. Several methods have been proposed for detecting convergence (if it occurs), including bit estimate variance [7] and neural net–based techniques [12].

The simulated performance of turbo codes over multiple iterations of the decoder is shown in Figure 8.16 for a code composed of two convolutional codes of rate $1/2$ with constraint length $K = 5$ separated by an interleaver of depth $d = 2^{16} = 65536$. The decoder converges after approximately eighteen iterations. This curve indicates several important aspects of turbo codes. First, note their exceptional performance: bit error probability of 10^{-6} at an E_b/N_0 of less than 1 dB. In fact, the original turbo code proposed in [6] performed within .5 dB of the Shannon capacity limit at $P_b = 10^{-5}$. The intuitive explanation for the amazing performance of turbo codes is that the code complexity introduced by the encoding structure is similar to the codes that achieve Shannon capacity. The iterative procedure of the turbo decoder allows these codes to be decoded without excessive complexity. However, note that the turbo code exhibits an error floor: in Figure 8.16 this floor occurs at 10^{-6}. This floor is problematic for systems that require extremely low bit error rates. Several mechanisms have

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 46 of 47

been investigated to lower the error floor, including bit interleaving and increasing the constraint length of the component codes.

An alternative to parallel concatenated coding is serial concatenated coding [13]. In this coding technique, one component code serves as an outer code, and then the output of this first encoder is interleaved and passed to a second encoder. The output of the second encoder comprises the coded bits. Iterative decoding between the inner and outer codes is used for decoding. There has been much work comparing serial and parallel concatenated code performance (see e.g. [13; 14; 15]). Whereas both codes perform very well under similar delay and complexity conditions, in some cases serial concatenated coding performs better at low bit error rates and also can exhibit a lower error floor.

8.6 Low-Density Parity-Check Codes

Low-density parity-check (LDPC) codes were originally invented by Gallager [10]. However, these codes were largely ignored until the introduction of turbo codes, which rekindled some of the same ideas. Subsequent to the landmark paper [6] on turbo codes in 1993, LDPC codes were announced by Mackay and Neal [16] and Wiberg [17]. Shortly thereafter it was recognized that these new code designs were actually reinventions of Gallager's original ideas, and subsequently much work has been devoted to finding the capacity limits, encoder and decoder designs, and practical implementation of LDPC codes for different channels.

Low-density parity-check codes are linear block codes with a particular structure for the parity check matrix $\mathbf{H}$, which was defined in Section 8.2.3. Specifically, a (d_v, d_c) regular binary LDPC has a parity-check matrix $\mathbf{H}$ with d_v 1s in each column and d_c 1s in each row, where d_v and d_c are chosen as part of the codeword design and are small relative to the codeword length. Since the fraction of nonzero entries in $\mathbf{H}$ is small, the parity-check matrix for the code has a low density – hence the name low-density parity-check codes.

Provided that the codeword length is long, LDPC codes achieve performance close to the Shannon limit and in some cases surpass the performance of parallel or serially concatenated codes [18]. The fundamental practical difference between turbo codes and LDPC codes is that turbo codes tend to have low encoding complexity (linear in blocklength) but high decoding complexity (due to their iterative nature and message passing). In contrast, LDPC codes tend to have relatively high encoding complexity but low decoding complexity. In particular, like turbo codes, LDPC decoding uses iterative techniques that are related to Pearl's belief propagation, which is commonly used by the artificial intelligence community (see [19]). However, the belief propagation corresponding to LDPC decoding may be simpler than for turbo decoding [19; 20]. In addition, this belief propagation decoding is parallelizable and can be closely approximated with decoders of very low complexity [21], although this may also be possible for turbo decoding. Finally, the decoding algorithm for LDPC codes can determine when a correct codeword has been detected, which is not necessarily the case for turbo codes. The trade-offs between turbo and LDPC codes is an active research area, and many open questions remain regarding their relative performance.

Additional work in the area of LDPC codes includes finding capacity limits for these codes [21], determining effective code designs [22] and efficient encoding and decoding algorithms [21; 23], and expanding the code designs to include nonregular [18] and nonbinary LDPC codes [24] as well as coded modulation [25].

Collision Communications, Inc. EX2005
Case No.: IPR2025-00303
Page 47 of 47