Copyright © 1998, by the author(s).
All rights reserved.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

RESOURCE ALLOCATION FOR MULTIMEDIA ON WIRELESS NETWORKS

by

Yuming Lu

Memorandum No. UCB/ERL M98/32

5 June 1998

RESOURCE ALLOCATION FOR MULTIMEDIA ON WIRELESS NETWORKS

Copyright © 1998

by

Yuming Lu

Memorandum No. UCB/ERL M98/32

5 June 1998

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Abstract

Resource Allocation for Multimedia on Wireless Networks

by

Yuming Lu

Doctor of Philosophy in Electrical Engineering and Computer Sciences

University of California at Berkeley

Professor Robert W. Brodersen, Chair

The personal communication industry has experienced phenomenal growth in the past few years. With the steady increase in demand for cellular telephones, wireless modems and other emerging wireless technologies, it has become evident that the next generation wireless system will integrate voice, video, image, and data. Transmitting multimedia data over a wireless channel presents a new set of challenges: data demands will sometimes exceed the system capacity, in which case the system must make the most efficient use of its limited resources.

This thesis addresses design and control issues for multiuser, multimedia indoor wireless communication systems. The first half of this thesis focuses on network resource allocations for supporting various quality of services imposed by multimedia traffic; the resources we consider in this work are link bandwidth and transmit power. We present our approach for unifying power control, variable forward error correction (VFEC), and scheduling for a downlink system by allocating the system resources. Our objective is to maximize the overall system satisfaction, which we call "system utility". This objective is achieved by applying a distributed algorithm which divides the overall optimization problem into a hierarchy of three levels (system, cell and user), with each performing independent and parallel optimizations. Following this theoretical framework, we then perform simulation-based evaluation of the system performance with a simple cell structure and uniformly distributed users. The overall performance is studied in detail as a function of user distribution, traffic statistics, FEC coding types. We also evaluated the effectiveness of power control comparing with variable forward error correction, the impact to the overall system performance due to the imposed fairness constraint, and finally, the performance gain due to hand-off.

Variable forward error correction is the generally the most performance critical part among three control variables. It is often implemented on custom IC or programmable logic devices. The second half of the thesis investigates the implementation of 63-bit family BCH codes with the error correction capability $t = 0, 1, 2, 3, 4, 5, 6, 7, 10, 11, 13, 15, 31$ errors (in a block of 63 bits). The core of the decoder adopts an iterative algorithm called Berlekamp's algorithm. By exploiting the redundancies between BCH codes of the same length, the architecture of a VFEC decoder is only slightly more complicated than a single FEC decoder. With pipelining, the decoding can be completed within one block time frame.

[illegible] 5/11/98

Professor Robert W. Brodersen, Chair Date

To mom and dad

with forever love and respect

Table of Contents

List of Figures

Acknowledgements

I am very indebted to Professor Brodersen for his vision and encouragement throughout the years, for indulging us with the research facilities and the staff support, and most of all, for the freedom he allowed me to pursue the research I want to do.

I would like to thank Richard Edell for our numerous and informative discussions which made great impact on my research, and of course, for his generous help on most of my computer problems. I would also like to thank my good friends Anita, Malik, Phyllis, and Tetiana, for making my Berkeley experience a very enjoyable one.

Finally, my deep gratitude goes to my husband George, who brightens my life with his spirit and love!

1 Introduction

1.1. Global wireless network for multimedia communications

Over the past few years, the personal communication industry has experienced phenomenal growth in providing new types of services and technologies. With the steady increase in demand for cellular telephones, paging services, wireless modems and other emerging wireless technologies, it has become evident that the next generation wireless system will integrate voice, video, image, and data. This system will not only provide the freedom for people to communicate at anytime from anywhere, it will also distinguish itself from the existing wireless systems by providing information that has been only available through wired networks.

Under the existing infrastructure for voice and data communications, it is likely that current wireless networks will be connected via wide-area network (WAN) for providing wide-area wireless services, as shown in Figure 1-1. The gateways are used for protocol conversion and adaptation of new network constraints. A couple of assumptions must be made for this proposed framework. First, the backbone network is assumed to have abundant bandwidth comparing to the wireless network. This assumption makes wireless network management possible when control information is exchanged between nodes, for functionality such as location tracking, handoff, packet forwarding, etc. In addition, this abundance in bandwidth also allows access capability to databases from a large number of users simultaneously. The second assumption is that basestations are installed in the service area, where each basestation transmits and receives from a set of portable units reside in its coverage area. These basestations are connected to the wired backbone network and are acted as gateways to the wireless networks.

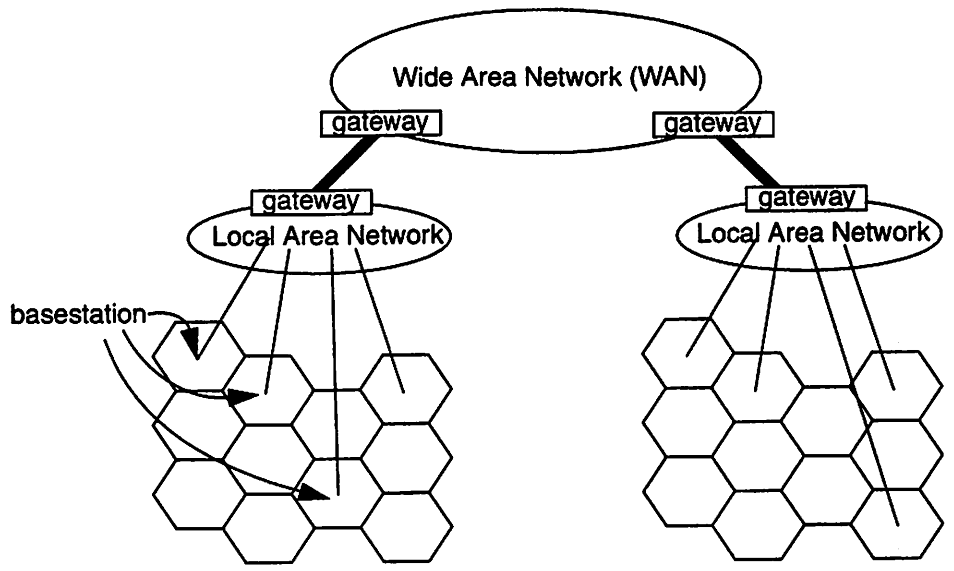


Figure 1-1. Global network for multimedia wireless communications.

1.2. Design issues

From the past research and development effort in this area, this next generation wireless system will have to address the following issues:

- Low power design for the portable unit

The transmit power and the computational power in a portable unit must be minimized in order to maintain a reasonable battery life. Extensive research has been done in this area with approaches ranging from high level algorithmic and architecture design to transistor level CMOS design [21][37][6]. The results have shown that the power consumption can be reduced significantly when low power design techniques are properly applied.

- Efficient use of limited network resources

The success of such global wireless services highly depends on gaining more in-depth understanding and making more efficient use of the underlying time-varying wireless channel. From a system's prospective, the wireless channels are often treated as the bottleneck as far as the channel capacities are concerned, for the available wireless bandwidth is significantly less than what has been provided by the traditional wired network. In addition, the channel quality is always subject to highly time-varying interference. To adapt to this environment, system design must focus on better utilization of the limited bandwidth. We will describe some key characteristics of wireless channels in the next section. The bandwidth allocation aspect of the channel will be discussed in the proceeding chapters of this thesis.

- Additional design constraints imposed by multimedia traffic

As the next generation wireless system will support multiple traffic types, the design has to address several user imposed system performance requirements. These requirements can be most easily understood if we consider the differences between this new system and conventional wireless systems.

The first difference is that conventional portable communication systems have been designed mostly for low-rate, single-data-type applications such as mobile voice and paging services; the new system, on the other hand, supports multimedia data consisting of real-time video, audio, text/graphics and control data. This difference yields several design requirements: (1) the bandwidth requirements are orders of magnitude greater. (2) Multimedia data consist of several data types with highly variable qualities of service (QoS) (e.g., typical speech data can tolerate a BER around 10^{-3}, whereas control data require a BER around 10^{-9}). (3) A multimedia system requires a scheduler because bandwidth demands vary greatly and will sometimes exceed channel capacity. This scheduler dynamically allocates bandwidth for each application, leading to a priority scheme for multimedia data.

The second difference is the relative burstiness of multimedia data as compared to voice data. For example, the channel that carries X-server display data is only active when there is a screen update. To fully utilize the available bandwidth, statistical sharing between multiple data streams is necessary.

The last difference is that the traditional voice/data communication systems respond to congestion by blocking or dropping calls. However, the more acceptable strategy is to provide a partial service during congested periods, and therefore gracefully degrade the QoS. This is possible because the use of variable rate compression algorithm for video and text/ graphics, and degradation can be achieved through the control of both bandwidth and error rate.

1.3. Quality of service and network resources

In contrast to traditional voice/data-only communications, multimedia applications are diverse and require a larger set of metrics to characterize their quality of service (QoS). These metrics will provide guidelines to wireless network for resource allocations. Let us first examine these in more details.

- Data rate

As we have discussed earlier, data rates for multimedia applications vary greatly from high bandwidth video to low bandwidth control information. For a system to accommodate various data rate and underlying behavior of the arrival process, a rate control algorithm is often employed to schedule multiple data sources for transmission.

- Error rate

The reliability of an application is usually measured by its received error rate. Many factors contribute to receiving errors. For example, buffer overflow result in packet loses, or excessive interference causes received bit errors. Many error control algorithms have been introduced during the past thirty years, ranging from bit level error correction coding (ECC), to packet level automatic repeat-request (ARQ) protocol. Recently, Han [22] proposed a scheme based on ARQ protocol which asymptotically achieves reliable transmission of multimedia/graphics over wireless channels. All these are very effective techniques with emphasis on different layers of the network.

- Delay

Roughly speaking, applications can be grouped into three categories according to their delay tolerances: end-to-end delay sensitive, jitter sensitive, and delay tolerant. The first group includes application such as interactive video-conferencing, in which case end-to-end delay is absolutely critical to visual quality. However, for applications such as movies, the relative delay between frames is more important than the end-to-end delay. And finally, when it comes to data transfer, delay may not be very important as long as data are received correctly (even though a timely transfer is still preferred).

These quality of service needs can be fulfilled via allocating the available resources. This thesis is mostly focused on managing network resources, and by network resources, we mean link bandwidth, transmit power level and buffer size. The allocation of the link bandwidth is controlled by a multiplexer, which schedules packets from various applications to a shared link. The amount of transmit power for each user is controlled by a power control algorithm. It is worth pointing out that a power control algorithm has to be designed from a system's perspective since increasing one user's transmit power increases overall interference level, which may severely degrades other users' receiving quality. From information theory's stand point, link bandwidth, transmit power and interference noise power ultimately define the upper bound on the channel capacity, which consequently determines the throughput and the maximum number of users a system can support.

The goal of this thesis is to design a framework which maps the resources to the various quality of service metrics. As we will see in later chapters, applications have flexibility in trading off bit rate, error rate, and delay. For example, we may increase the channel reliability by giving up some bandwidth for error control coding. Similarly, we can also achieved a higher data rate by increasing transmit power and adopting a higher constellation, which consequently increase the spectral efficiency. Compare to a conventional system which considers channel coding and source coding separately, we can also trade off between compression rate and channel coding rate. Many of these issues will be discussed in greater detail in the chapters of this thesis.

1.4. Network access technique

A communication system that allows multiple users to access the wireless network is called a *multiple-access communication system*. For a common shared bandwidth, there are many ways to divide it. Three approaches are widely used: (1) frequency-division-multiple-access (FDMA) divides the total bandwidth into small segments, and each user communicates within the assigned frequency band; (2) time-division-multiple-access (TDMA) is similar to FDMA except division is occurred in time domain rather than frequency domain; and (3) code-division-multiple-access (CDMA) is a spread spectrum technology, and it can be visualized as a combination of FDMA and TDMA. Since most of the research in this thesis is carried out under the framework of CDMA, this section therefore focuses on CDMA technology, specifically, the direct sequence CDMA (DS CDMA).

1.4.1. The DS CDMA technology

In a DS CDMA system, each user's bit stream is modulated with a pseudo-noise (PN) sequence, which is an output of the pseudo-random generator. A rectangular pulse in a PN sequence is called a ***chip***, and let us denote its duration as T_c. If the total available bandwidth is W, the rate of PN sequence is then W chips per second. In other words, $T_c = \frac{1}{W}$. Suppose information stream has bit rate R (with bit duration $T_b = \frac{1}{R}$), most practical systems choose the ratio $\frac{T_b}{T_c}$ to be an integer, we call this ratio the *spreading factor*, and it is denoted as N. As a result, the modulated signal transmitting over the channel expands to the entire available bandwidth. Because N is normally large, the PN modulated signals from other users therefore appear as white noise. An example of a PN modulated signal is shown in the next figure.

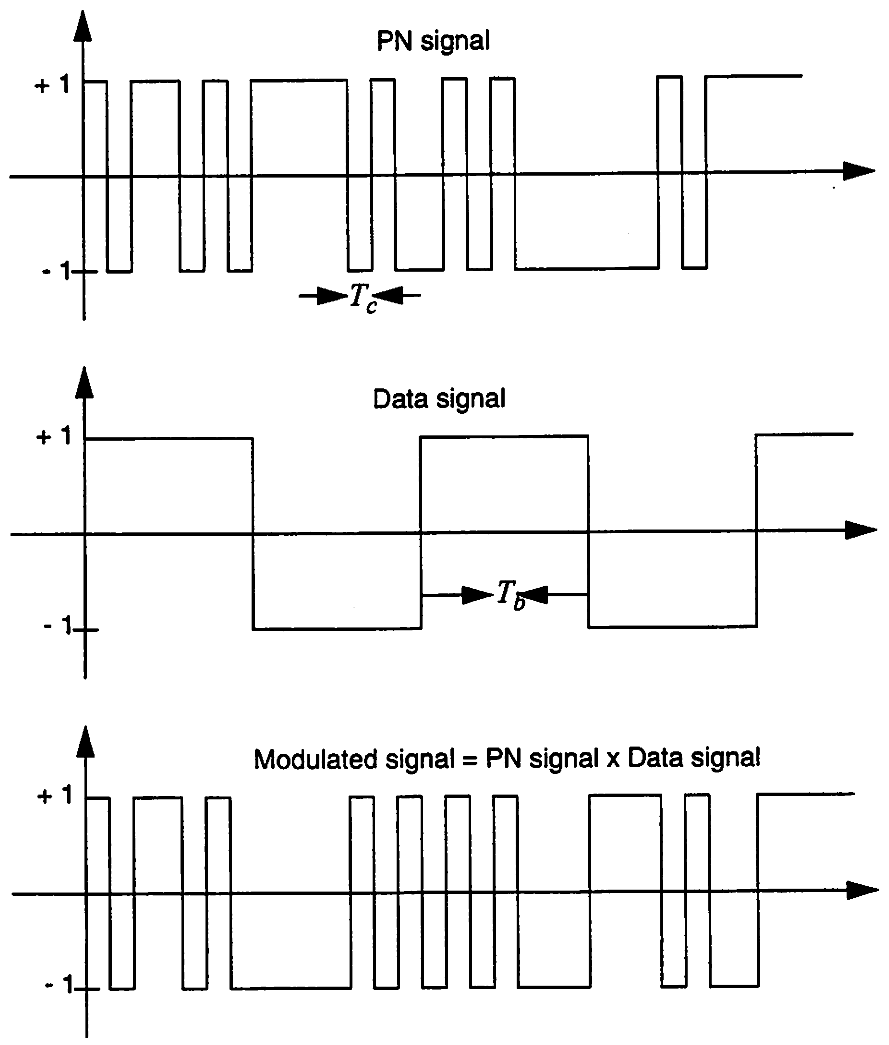


Figure 1-2. CDMA PN signal, data signal, modulated signal.

To reduce user-to-user interference, the user signals are multiplied with a set of orthonormal codes, called Walsh codes. By definition, a set of codes $W_i(t)$, $i \in \{1, 2, 3, \ldots\}$ is orthonormal over the interval $0 \le t \le T$ if:

$$\int_0^T W_i(t) W_j(t) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

Figure 1-3 shows two Walsh codes: W_1 and W_2.

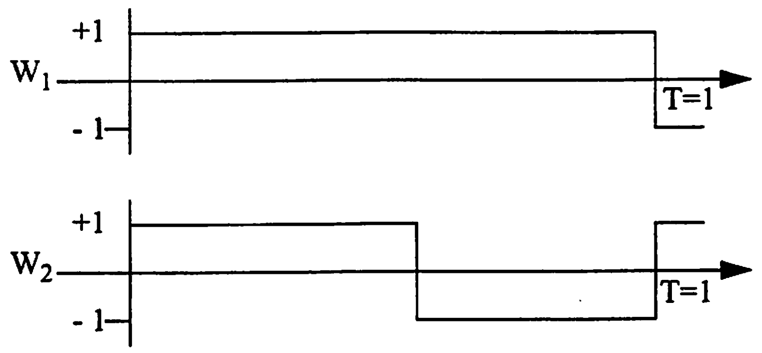


Figure 1-3. Basic Walsh Codes.

Putting everything together, it has been shown this combination of Walsh and PN codes will yield a processing gain of N, independent of how energy is distributed among signal components [50]. A system schematic indicating Walsh and PN codes is shown below:

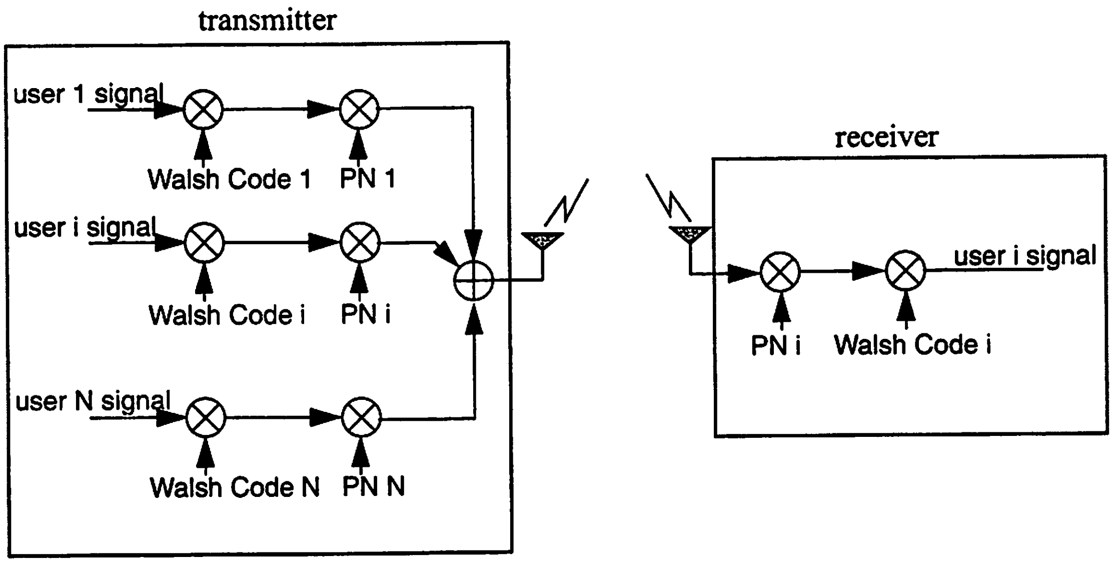


Figure 1-4. Transmitter and receiver architecture with Walsh and PN modulated codes.

At the receiver's end, say for user i, the aggregated signals are first multiplied with user i's PN code and then with user i's Walsh code. In a perfect world where the transmitter and the receiver are synchronized, and there is no noise and interference, then the desired signal can be perfectly detected since all user signals are orthogonally coded. However, it is known that a wireless environment can be full of background noise, and there are interferences from reflected signals and nearby transmitters; therefore, the received signal is imperfect with added interference and noise components.

1.4.2. Interference model for CDMA systems

Generally, the received signal for user i can be divided into four parts:

$$r_i(t) = s_i(t) + I_i^{\text{intra}}(t) + I_i^{\text{inter}}(t) + n_i(t)$$

where $s_i(t)$ is the desired signal component, $I_i^{\text{intra}}(t)$ and $I_i^{\text{inter}}(t)$ are the intra-cell and inter-cell interference, respectively, and $n_i(t)$ is the background noise. Two sources of interference are present in the system. The intra-cell interference, on one hand, is generated by the signals inside the same cell coverage. The inter-cell interference, on the other hand, is generated by the signals from nearby cells which arrived at user i's receiver. Both inter-cell and intra-cell interference appear in uplink and downlink channels.

During a downlink transmission, signals arrived at the receiver through a direct path and multiple reflected paths, as shown below:

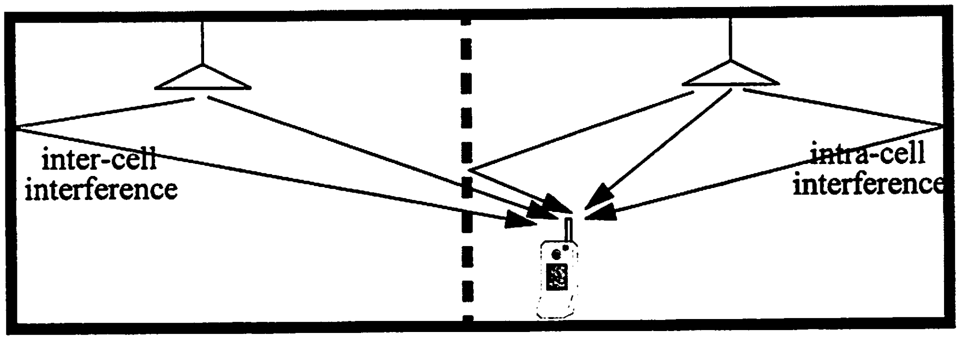


Figure 1-5. Interference sources for a downlink system.

For intra-cell, if the transmitter and the receiver are synchronized, the received signals would still remain orthogonal when arrived at the receiver, thus do not interfere with one another. However, the intra-cell interference is not completely eliminated because signals reflected from walls or objects would arrive with time delays, which would then destroy the orthogonality. As we have also indicated in Figure 1-5, the inter-cell interference exists for downlink transmission because signals transmitted from other basestations are non-orthogonal to the desired signal as they travelled completely different paths. Even though the reflected inter-cell signal is also a part of interference, the direct arrival paths are usually the strongest and are used to approximate the total inter-cell interference. Because signals are PN coded, and there is a large number of them, both intra-cell and inter-cell for downlink channel can be modeled as white Gaussian noise.

For uplink communications, there is no difference between the nature of inter-cell and intra-cell interference, because each user signal travels through a different path to reach the receiver. As they differ in travelling distances, there is no orthogonality between them. Since users reside in the same cells are closer to the receiver, they appear as a stronger source of interference (than users from nearby cells). Again, since they are all PN coded, and there is a large number of them, inter-cell and intra-cell interference for both uplink channel can be modeled as white Gaussian noise as well.

1.5. Power control

To combat excessive interference and improve the received signal to noise ratio, power control is an effective technique that has been applied to many wireless systems. Power control is a method which adjusts the transmit power level so that the received signal to noise ratio adapts to the interference and channel variation.

In general, power control has been applied to both uplink and downlink channels for different purposes. Previous research in this area has mostly been focused on uplink power control to solve the near/far problem [50]. This problem is caused by users having unequal distances to a common receiver basestation, so the received signals vary widely in signals power. Uplink power control can mitigate this effect as well as being used to minimize the transmit power level which critical due to the limited battery life at portables. The downlink communications, on the other hand, experiences no near/far effect since all signals propagate through the same path when arrived at the portable, but is used instead to combat interference and channel fading. Since the downlink is assumed to originate at a wired basestation, battery life is not a concern.

A power control algorithm can be either centralized or distributed, synchronous or asynchronous. For a centralized algorithm [19], information on every user's channel quality, power level and traffic demand is processed at a central unit, and the signal transmit power levels for all users are determined simultaneously. Centralized algorithms are usually synchronous, in which all transmit power levels are updated at the same time. Since all information is available at the time when decisions are made, centralized algorithms are often simpler to analyze, easier to implement and control. It is also non-iterative as the optimum can be computed in a single iteration. However, since wireless communication systems should be designed to be scalable to serve hundreds or thousands of active users at any given time, a centralized approach is usually not practical because of the computational complexity involved. This shortcoming makes a centralized algorithm unsuitable for large scale systems, but can be used as a performance upper-bound of what is achievable.

A distributed algorithm [20], on the other hand, uses the local power and interference information to make independent decisions. Algorithms in this case are often scalable, meaning adding new users to the system imposes no increase in algorithm complexity. This property makes distributed algorithms highly desirable for practical systems. However, the distributed algorithms are generally more difficult to analyze, since the algorithms ar usually iterative. For this type of algorithms, convergence becomes an issue, and even when an algorithm does converge, the rate of convergence must also be considered.

Early work has shown a significant increase in the system capacity after applying power control [34][58]. Foschini and Miljanic proposed a distributed algorithm that uses local measurements on power and interference in order to meet the required carrier to interference ratio (CIR) [15]. In [55], Yates proves the existence of a transmit power vector that satisfies the given interference constraint. In addition, Yates also proved the convergence of synchronous and asynchronous power update to a unique fixed point at which total transmitted power is minimized. This result will be applied in Chapter 3 to proved convergence of our power control algorithm. Both Hanly and Yates et. al. have independently come up with algorithms the combines basestation selection and power control [24][56]. They presented distributed algorithms that converge to an allocation which has the minimum interference.

Recently, power control has also been applied to multimedia systems for delivering multiple data types having a wide range of SNR requirements [57][29]. In [57], the transmitted power is adjusted at the packet level according to both data type and channel quality; in other words, different segments of a packet may be delivered with different transmit power; this is a form of fast power control. In [29], power control has been combined with FEC and scheduling to optimize the system utility, and will be presented in later chapters of this thesis.

1.6. Case Study: InfoPad

The Infopad is a multimedia wireless system developed at UC Berkeley [40]. This system, shown in Figure 1-6, is a wide-band multimedia communication network which is designed to serve a large number of users simultaneously through portable terminals. The communication channels for this system are asymmetric as the high speed, real-time video is only available for the downlink transmission. We assume the downlink transmission employs DS CDMA. Since measuring absolute phase of arrival signals is a difficult task, differential QPSK is adopted as the modulation scheme, in which he information is carried by the phase difference between two consecutive arrival symbols. Each user transmits at the rate of 1Msymbols per second (2Mbit per second). 64 distinct Walsh codes are employed, corresponding to a coding gain of 64.

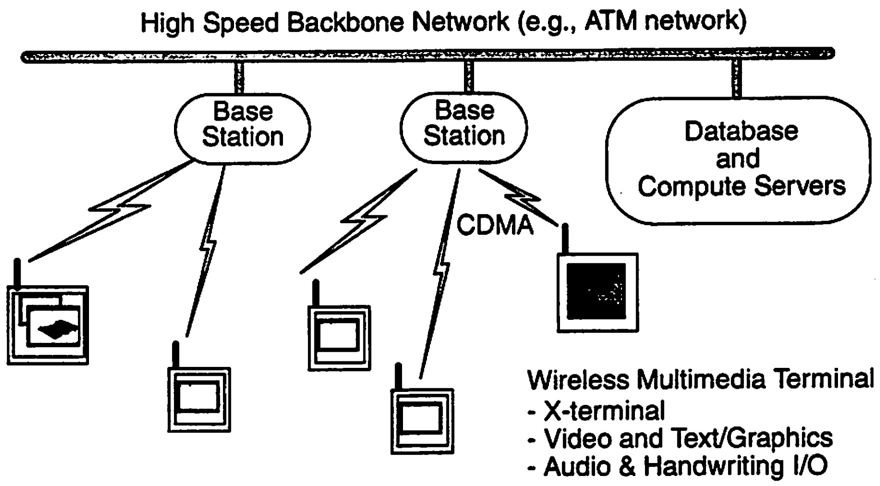

High Speed Backbone Network (e.g., ATM network)
Base Station
Base Station
Database and Compute Servers
CDMA
Wireless Multimedia Terminal
- X-terminal
- Video and Text/Graphics
- Audio & Handwriting I/O

Figure 1-6. Infopad system overview.

For low power, cost and usability reasons, these terminals have minimal general purpose computational power; instead, only computations that are absolutely necessary are carried out by the portable units. The computations that are not so time critical are pushed off to compute servers connected through the a high speed backbone network. The computational results, together with other types of multimedia data, are sent back to the terminals via backbone network, basestation, and wireless channel.

1.7. Thesis outline

As will be presented in the proceeding chapters, the main contribution of this thesis is to provide a systemic approach which efficiently allocates wireless network resources. The resulting allocation corresponds to a wide range of QoS imposed by multimedia users. Simulation of this system further provides a more in-depth understanding of the dependencies and relative effectiveness of different parts of the system.

The thesis outline is the following. We begin Chapter 2 with a detailed description of basestation architecture and resource constraints for a DS CDMA system. We then claim the system design is sub-optimum if the techniques of power control, forward error correction and scheduling are not considered simultaneously. As a result, we propose a framework for unifying power control, variable forward error correction, and scheduling for a downlink system. Our objective is to maximize the overall system satisfaction, which we call "system utility." This objective is achieved by applying a distributed algorithm which divides the overall optimization problem into a hierarchy of three levels (system, cell and user), with each performing independent and parallel optimizations.

Chapter 3 is focused simulation-based evaluation of the system performance with a simple cell structure and uniformly distributed users. The system is simulated using a specific utility function: the step function. This utility function coincides with conventional design objectives for many wireless or cellular systems, where a BER (or SNR) requirement is imposed. The objective of this chapter is to gain a more in-depth understanding of the system performance with concrete metrics. The system performance is studied as a function of user distribution, traffic statistics, FEC coding types. We also evaluated the effectiveness of power control comparing with variable forward error correction, the impact to the overall system performance due to the imposed fairness constraint, and finally, the performance gain due to hand-off.

In most wireless systems, power control and scheduling algorithms are implemented in software since they operate at the packet level thus are not very timing critical. The error correction, on the other hand, performs in real time at data rate, which can be as high as 2Mbit/sec such as in the case of the Infopad. In order to achieve this high performance requirement, FEC decoder design is often a critical part of the overall system design. The rest of the thesis is devoted to architecture design of a variable forward error correction decoder based on 63 bit family BCH codes. This decoder has the correction capability of $t = 0, 1, 2, 3, 4, 5, 6, 7, 10, 11, 13, 15, 31$ errors (in a block of 63 bits).

To understand the decoder architecture, the work is divided into two parts. In Chapter 4, we first provide readers with sufficient background on algebraic coding theory, since a cyclic code is used as the building block for variable forward error correction. In Chapter 5, an architecture is presented together with its control logic which is implemented using finite state machine. By exploiting the redundancies between BCH codes that have the same block length, we observe that a majority of the decoder hardware can be reused among codes. Therefore, the implementation of a VFEC decoder is only slightly more complicated than a single FEC decoder. With pipelining, we discover the decoding can be completed within one block period. However, the control logic will be simpler if more pipeline stages are introduced.

This thesis is concluded with Chapter 6, in which we will discuss the possibility of introducing delay as an additional system constraint. Other system control technique such as automatic repeat-request (ARQ) will also be discussed.

2 Utility Based System Control

2.1. Goals

This chapter addresses design and control issues for multiuser, multimedia indoor wireless communication systems. The motivation of this work is the InfoPad system, which is a multimedia wireless downlink system developed at UC Berkeley. However, the approach and the techniques described here can be extended to other wireless systems as well.

This system is designed to fully utilize the available resources while meeting the various QoS requirements of multimedia data. In order to achieve the optimal system performance, we consider three control "knobs" for fulfilling various bandwidth or BER requirements. These knobs are: variable forward error correction (VFEC), power control and scheduling. Among them, VFEC selects a particular FEC code; this FEC then introduces redundancy to combat transmission errors. Downlink power control varies the transmit power to adjust the received signal quality. Together VFEC and power control are used to support applications with widely varying QoS *and* to mitigate excessive interference. Finally, prioritizing (i.e. scheduling) allocates bandwidth among data types; this is especially needed when application demands exceed channel capacity.

The goal of our design is to unify VFEC, power control and scheduling into one system. We claim that the system design is incomplete if the methods of VFEC, power control and scheduling are not considered simultaneously. For example, let us consider a design that only controls power. Since data in our system are time varying, it is very possible that all active applications would sometimes consume less than the available bandwidth. In this case, the remaining bandwidth is wasted. Compare this to a system that, instead of wasting this bandwidth, uses it for FEC coding. Due to the coding gain, we are able to reduce the transmit power while maintaining the same QoS; and the consequence of lowering transmit power is the reduced interference to the rest of the system. This example illustrates the interaction between power control and FEC. Finally, the result from scheduling affects both the VFEC selection and the transmit power level in order to meet the specific QoS of that data type.

2.2. Downlink system architecture

It has been discussed in Chapter 1 that a CDMA system has a limited capacity due to resource constraints that arise from the nature of the multiple access scheme. These constraints are: (1) each user is allocated a fixed link bandwidth; the total bandwidth allocated to a user's applications with or without FEC encoding must be less than this link bandwidth. In the case of the Infopad, each user is given a 2Mit/sec link bandwidth; and (2) the total transmit power from a basestation (which serves all users in a cell) must be less than an allowable power level set by implementation constraint and FCC regulation. Figure 2-1 shows the downlink radio structure together with these practical constraints.

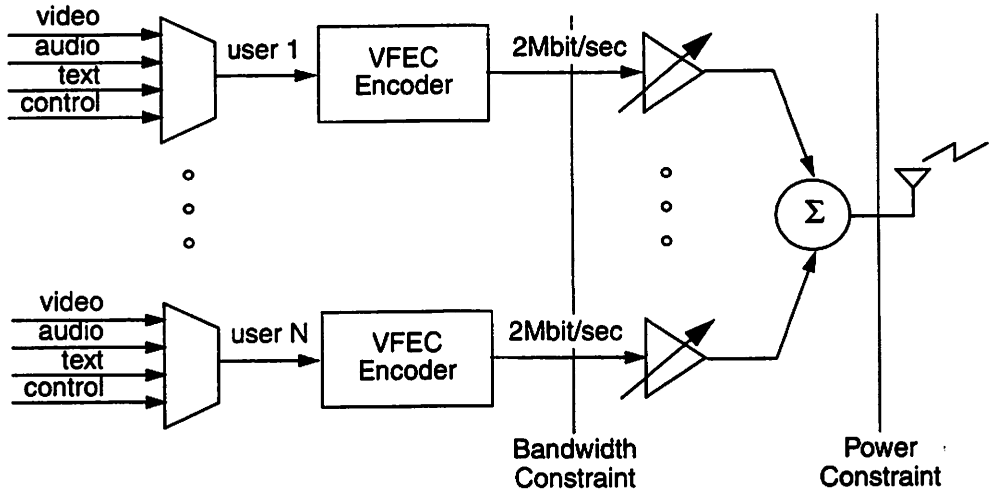


Figure 2-1. Basestation architecture and the system constraints.

For each user, the multiple data types are multiplexed and send to a VFEC encoder. The data are then encoded with the most suitable FEC code (according to individual data type's QoS specification) and passed to the power amplifier, where the data are amplified according to the power control algorithm. Finally, the amplified data are summed and transmitted to all users simultaneously. In the above Figure, we have only shown the modules that are related to this study, in other words, we have left out the part of the system such as PN generator and Walsh code.

2.3. Concept of capacity

2.3.1. Conventional measure of capacity

A large set of solutions are feasible by various combinations of VFEC, power control and scheduling, where each combination yields a set of resource allocation resulting in different bandwidth and BER received for each data type. It is therefore crucial to define a design objective, or the notion of capacity, that can be used as a guideline to control these techniques.

In the past, extensive work has been carried out to analyze the wireless system in terms of the Shannon capacity [17][18]. For a time-invariant channel such as the telephone network, the Shannon capacity is the mutual information between channel input and output maximized over all possible input distribution. The mutual information is defined as:

$$I(X;Y) = E_{x,y}\log[p(x,y)/p(x)p(y)]$$

where $p(x,y)$ is the joint distribution of the channel input and output, $p(x)$ and $p(y)$ denote the distribution for channel input and output, respectively. Shannon also proved that, for any data rate below capacity, there exists a block code at that rate with an error probability that goes to zero; however, the block code has no restriction on its code complexity or delay. For a time-varying channel, there is no analogous definition of mutual information, as the conditional input–output probabilities are time dependent. Therefore, the channel capacity is defined to be the maximum achievable rate with arbitrarily small probability of error (without restriction on the code complexity or delay). Even though this whole framework built around the Shannon capacity inherits a very rich mathematical structure that provides an upper bound on how well a system can perform, it is nevertheless impractical for designing practical systems as delay and implementation complexity are often important issues.

Until now, existing wireless and cellular systems are designed to maximize the average number of users at a given time with a given quality, which is called the Erlang capacity. This concept of Erlang capacity stems from the telephony industry, where the capacity is established around the availability of assigned slot for wired telephone traffic. The availability is defined as the probability a user does not receive service at any time because all slots are assigned to other calls, which would consequently evoke a busy signal. In wireless systems, the concept of the availability becomes more complicated as it depends on the channel quality, data rate, total bandwidth and many other factors. However, for the existing wireless systems, the concept of Erlang capacity has been widely applied, mostly because these systems are designed to support a single data type, such as voice or data service. In these cases, QoS for a user can be concretely defined as a function of received SNR, thus the number of active users is also well defined.

2.3.2. Utility

Complications arise after introducing multimedia data to our system. One difficulty is that these data types have a wide range of error tolerances, with BER ranging from 10^{-3} to 10^{-9}. In addition, with the invention of layered coding [31] [45], partial delivering is made possible for achieving graceful degradation. As a result, the notion of an active user is not well defined because the amount of network resources a user needs is highly dependent on the type of requested service. For example, suppose a user can choose receiving low resolution video together with text graphics *or* just high resolution video; or, the system may either support 6 users with high QoS *or* 10 users with medium QoS, how do we choose between these alternatives? A multi-user, multimedia system is ultimately designed to satisfy users; therefore, we argue that any design decisions should be based on user satisfaction. For instance, if users prefer receiving more data at cost of having a higher error rate, the system design should reflect that accordingly. Therefore, our objective is to maximize total user satisfaction, which we call "system utility".

In general, the concept of system utility is somewhat vague. However, if we assume that utility is additive, then system utility becomes the sum of user utilities, and a user's utility is the sum of that user's application utilities. For each application, the performance clearly depends on both quality and quantity of delivered data. In this study, we express application utility in terms of delivered BER and bandwidth.[1] An example on application utility function is shown in Figure 2-2.

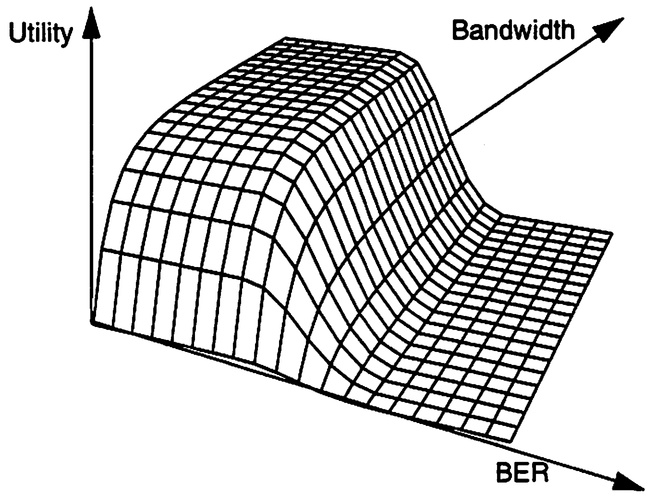


Figure 2-2. A typical application utility function.

At this point, let us discuss the qualitative properties of the application utility functions.[2] We proceed by first holding BER constant so that utility only depends on delivered application bandwidth.

For all applications, the application utility is a monotone non-decreasing function with respect to the bandwidth. We can categorize applications into many classes; nevertheless, we will discuss and contrast only two such classes. One class includes applications for which performance gradually improves as their allocated bandwidth increases; however, with a decreasing marginal utility (e.g., video and text/graphics). The utility functions for this class of applications are therefore concave everywhere, as shown in Figure 2-3 (a). Another class includes applications such as control information for which the received data are of no value to users if only partial information is delivered; however, once the neces-

1. End-to-end delay is not considered in this study.
2. The actual utility function for any particular application can be determined through either simulations or experiments with users.

sary amount of data is delivered, there is no extra benefit for receiving more data. Figure 2-3 (b) shows the utility function for this class of applications.

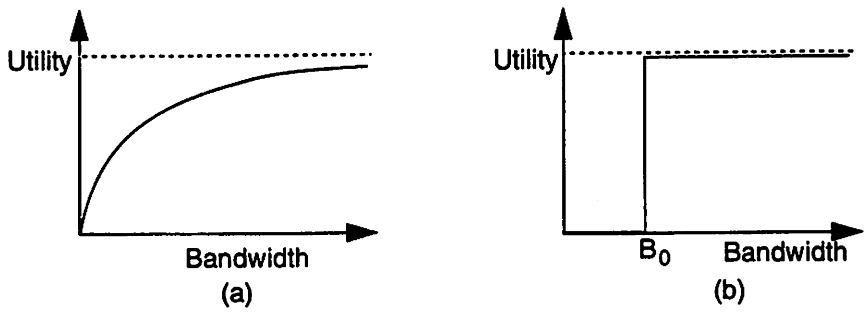


Figure 2-3. Two types of utility functions with respect to application bandwidth.

Let us now turn our attention to the other parameter of the utility function, the error rate. When the received BER is high, users are generally unsatisfied with application performance. As the error rate improves, their satisfactions rise as well. However, once the BER improves beyond a certain level, very little additional satisfaction is achieved. For instance, the reception quality of video is nearly identical between BERs of 10^{-5} and 10^{-8}. Figure 2-4 illustrates the utility function with respect to BER. As a final remark, the application utility as a function of bandwidth and BER shown in Figure 2-2 is obtained by multiplying utilities from Figure 2-3 (a) and Figure 2-4.

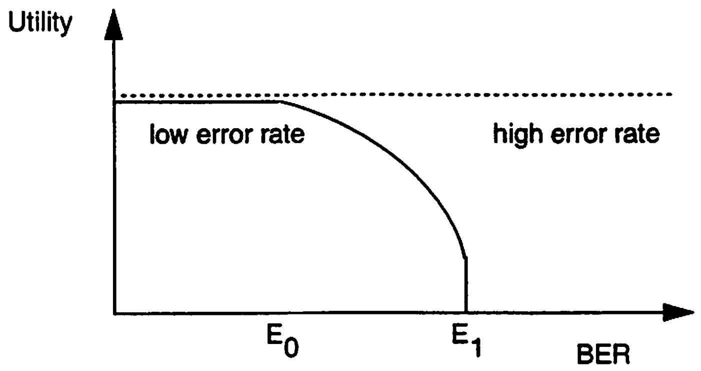


Figure 2-4. Utility function with respect to the error rate.

2.4. Layered Approach to Utility Maximization

Now that we have discussed the qualitative behavior of the application utility functions, let us return to the problem of maximizing the system utility subject to constraints. Suppose a user has several applications, each with utility $u_i(B_i, E_i)$, where B_i is the application bandwidth, and E_i is the received BER (after FEC decoding). Recall the utilities are assumed to be additive, therefore, the user, cell, and system utilities can be expressed as:

$$\text{user util} = \sum_{i=1}^{L} (\text{appl. util})_i = \sum_{i=1}^{L} u_i(B_i, E_i) \tag{2.1}$$

$$\begin{aligned} \text{cell util} &= \sum_{j=1}^{M} (\text{user util})_j \\ &= \sum_{j=1}^{M} \sum_{i=1}^{L} (u_i(B_i, E_i))_j \end{aligned} \tag{2.2}$$

$$\begin{aligned} \text{system util} &= \sum_{k=1}^{N} (\text{cell util})_k \\ &= \sum_{k=1}^{N} \sum_{j=1}^{M} \sum_{i=1}^{L} (u_i(B_i, E_i))_{j,k} \end{aligned} \tag{2.3}$$

With this formulation of the utility functions, we are able to overcome the shortcomings from the previous design objectives. First, a user may prioritize various data types using the utility functions according to (2.1). Second, by introducing an appropriate weight factor, users in the system can be prioritized as well by (2.2).

Our objective is to maximize (2.3) subject to constraints, where parameters B_i and E_i are controlled by power control, VFEC, and scheduling. One approach to this global optimization is to apply a centralized algorithm which considers all users' utilities in the system simultaneously. The advantage of this approach is that it does not require any iterative steps for achieving the optimum; however, the computational complexity and communication requirements are impractical for any reasonable sized system.

Instead, we propose a distributed algorithm that divides the system optimization problem into three separate levels: user level, cell level, and system level. Figure 2-5 shows the optimization hierarchy, in which the optimizations are represented by nodes and communications are represented by edges.

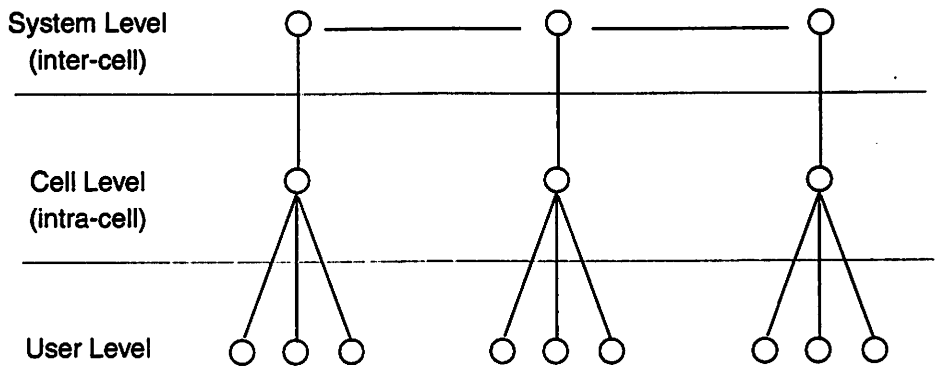


Figure 2-5. Three levels of optimization hierarchy.

The nodes within each level are independent and can be optimized in parallel. The optimization results, expressed in terms of the resource requirements, are passed up to the higher level. For example, the optimal user utility, expressed in terms of user's channel SNR, is directly proportional to the transmit power level; the transmit power level is then determined by the cell level optimization which performs intra-cell power allocation. In addition, at the system level, a cell communicates with its interfering neighbor cells to negotiate its cell power budget so as to maximize the entire neighborhood utility. This layering approach yields a distributed algorithm, and the details are explained in the following section using a bottom-up strategy: from user level to cell level, to system level.

For the rest of the chapter, we assume the feedback loop from portable to basestation gives the basestation a perfect estimate of the channel SNR. The basestation then calculates the propagation loss and the interference factors for each user, based on the user's channel SNR and the transmit power levels of the neighbor cells. In Section 2.4.3, we will discuss how channel estimation can be done in reality by having a synchronized power control algorithm.

2.4.1. User Level Optimization

At the user level, we optimize user utility. Two techniques are applied: first, the system performs scheduling by allocating bandwidth to each application. Second, the system optimizes error rate by applying VFEC; VFEC enables the system to trade-off the quality with quantity of delivered data for each application.

Recall that the user utility is the sum of the application utilities:

$$U_{user}(\bar{B}, \bar{E}) = \sum_{i=1}^{L} u_i(B_i, E_i) \tag{2.4}$$

Our objective is to maximize (2.4), over the variables $\bar{B} = (B_1, B_2, \ldots, B_L)$ and $\bar{E} = (E_1, E_2, \ldots, E_L)$, subject to the link bandwidth constraint:

$$\sum_{i=1}^{L} B_i \cdot \eta(SNR, E_i) \leq \text{Link Bandwidth} \tag{2.5}$$

where

$$\eta(SNR, E_i) = \frac{\text{link bandwidth}}{\text{data bandwidth}} \tag{2.6}$$

The maximal value of user utility is denoted as U^*_{user}.

To maximize the user utility, we do not directly choose the received BER E_i; instead, we select the level of VFEC which, together with the known link SNR, to determine the received BER. The choice of FEC code is reflected by the "bandwidth expansion function", $\eta(SNR, E_i)$, which corresponds to the FEC code that achieves E_i (after decoding) for a given channel SNR. $\eta(SNR, E_i)$ is the ratio of the link bandwidth to the data bandwidth of the FEC code, as in (2.6); thus $B_i \cdot \eta(SNR, E_i)$ is the actual channel bandwidth consumed by application *i*.

Maximizing (2.4) under the constraint (2.5) is an optimization problem over $2L$ variables ($\bar{B}$ and $\bar{E}$). The optimal $\bar{B}$ and $\bar{E}$ can be obtained by applying the Lagrange multipliers. Observe that the channel SNR is the only undetermined variable during the optimization; as a result, both the optimal $\bar{B}$ and $\bar{E}$, and thus the optimal user utility are functions of the channel SNR, as shown below:

$$U^*_{user}(SNR) = \underset{\bar{B},\bar{E}}{Max}\ U_{user}(\bar{B},\bar{E})$$

$$\text{such that } \sum_{i=1}^{L} B_i \cdot \eta(SNR, E_i) \leq \text{Link Bandwidth}$$

Let us now focus on the characters of $U^*_{user}(SNR)$. Since the user utility is a monotone increasing function with respect to application bandwidth, higher link bandwidth would always yield higher user utility. Therefore, the maximum user utility would occur when all available link bandwidth is consumed, i.e. at the boundary of the inequality (2.5). Thus this inequality can be reduced to equality, which is a convex set. Since our objective function, the user utility, is a concave continuous function, and the constraint set is a convex set, by the fundamental theorem of mathematical programming, $U^*_{user}(SNR)$ is a global optimum.

To summarize, two results are achieved through this user level utility optimization. First, for any given channel SNR, we can apply scheduling and FEC selection (i.e. choosing $\bar{B}$ and $\bar{E}$) such that the user utility is maximized. Second, the maximum user utility can be expressed as a function of SNR (presuming an optimal choice of $\bar{B}$ and $\bar{E}$). As will be shown in the next section, this maximal user utility function becomes the cornerstone of the cell level power allocation.

2.4.2. Cell Level Optimization

2.4.2.1. Derivation of user SNR

Before we proceed with the cell-level optimization, let us first derive the user channel SNR. This will help us understanding the intra-cell power allocation which will be presented later.

An indoor wireless communication environment is interference limited. In such an environment, users are subject to three sources of interference and noise: intercell interference, intracell interference and background noise. Let us assume we are interested in user *i* located in cell #0, the channel SNR for user *i*, is the ratio of the received signal power to the noise power:

$$SNR_i = \frac{\text{received signal power}}{\text{total interference and noise power}} \tag{2.7}$$

Let us first derive the received signal power. Let P_0 be the total available transmit power for cell #0, and ϕ_i be the fraction of P_0 allocated to user i. The transmit power level for user i, denoted as p_i, is $p_i = \phi_i P_0$. In most radio applications, the transmission over the free space is given by the approximation [25]:

$$P_r = P_t g_b g_r \left(\frac{h_b h_r}{d^2}\right)^2$$

where

P_r: received signal power;

P_t: transmitted signal power;

g_b: basestation antenna gain;

g_r: receiver antenna gain;

h_b: basestation height;

h_r: receiver height;

d: distant between basestation and receiver.

Therefore, the received signal power for user i can be written as:

$$P_r = g_i p_i \tag{2.8}$$

where g_i represents the overall antenna gain, height factors and path loss of user i.

Let us now derive the total interference and noise power which includes inter-cell, intra-cell and background noise. For illustration purposes, we assume a hexagonal cell topology as shown in Figure 2-6., where a center cell has six neighboring cells. Suppose cell $\#i$ is in the center, and let $\overline{P_i} = (P_i, P_1^i, \ldots, P_6^i)$ be the total power level at the basestation for each one of the seven cells, where $(P_1^i, \ldots, P_6^i)$ are neighboring cell power levels with ith cell being the center cell. Using this notation, for example, $\overline{P_0}$ represents the power budget for a cell neighborhood where #0 is in the center.

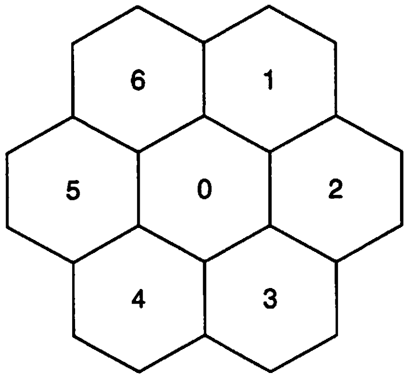


Figure 2-6. Cell topology.

As we have discussed earlier, the intra-cell and inter-cell interference are modeled as white Gaussian noise. Without loss of generality, our focus is on user i in cell #0 with six adjacent neighbor cells. The intra-cell interference results from reflected signals from the same cell, thus it only depends on the total power level of the residing cell, which is P_0. By assuming a large number of reflective paths, the intra-cell interference can be modeled as white Gaussian noise and is a fraction of cell power level [50]:

$$(\sigma^2_{\text{intra}})_i = \alpha_i \cdot P_0 \tag{2.9}$$

where α_i is called the intra-cell interference coefficient, and it equals to reflection coefficient divided by path loss and the spreading gain N.

The inter-cell interference, on the other hand, is generated by the signals from interfering neighbor cells[1], thus it depends on the total transmit power level from these cell basestations. Recall $(P_1^0, \ldots, P_6^0)$ represents the power levels of six neighbor transmitters when cell #0 being the center cell, the inter-cell interference can then be expressed as:

$$(\sigma^2_{\text{inter}})_i = \beta_{1,i} \cdot P_1^0 + \beta_{2,i} \cdot P_2^0 + \ldots + \beta_{6,i} \cdot P_6^0 \tag{2.10}$$

where $\beta_{n,i}$ is the inter-cell interference coefficient for user i generated by cell $\#n$. For simplicity, we only consider LOS interference, thus β equals to attenuation dividing the spreading gain. As a remark, these interference coefficients should always be verified and updated with measurements.[2]

Combining (2.9) and (2.10), we can use a vector notation to represent the overall interference noise power,

$$(\sigma^2_{\text{intra}})_i + (\sigma^2_{\text{inter}})_i = \overline{\gamma_i} \bullet \overline{P_0} \tag{2.11}$$

where $\overline{\gamma_j} = (\alpha_j, \beta_{1,j}, \ldots, \beta_{6,j})$, $\overline{P_0} = (P_0, P_1^0, \ldots, P_6^0)$, and $\bullet$ is the vector dot product. Consequently, substituting the expression of interference noise power (2.11) into (2.7), the channel SNR for user i in cell #0 is:

$$SNR_i = \frac{g_i \cdot p_i}{(\sigma^2_{\text{intra}})_i + (\sigma^2_{\text{inter}})_i + (\sigma^2_B)_j} = \frac{g_i \cdot \phi_i \cdot P_0}{\overline{\gamma_i} \bullet \overline{P_0} + (\sigma^2_B)_i} \tag{2.12}$$

1. Throughout this study, we only consider the first order intercell interference, i.e. interference coming from the adjacent cells.
2. The intracell and intercell interference coefficients for a user depend on the location of the user and the indoor environment, both of which are time varying. These interference coefficients can be estimated by correlating total noise power (which can be measured) with this and neighboring cell power levels.

where g_i is the product of antenna gain, path loss and the spreading gain of user *i*, $\overline{P_0}$ is the total transmit power for entire neighborhood of cells when cell #0 is in the center, and $(\sigma_B^2)_i$ background noise power.

2.4.2.2. Cell level optimization

During the downlink transmission, a CDMA radio transmits to all users within a cell simultaneously and is subject to a power budget which is determined through the higher layer optimization. The goal of performing cell level optimization, for cell #0 in our case, is to distribute the power budget P_0 to each user, i.e. finding ϕ_j so that the total cell utility is maximized.

Suppose cell #0 has M users, the total cell utility is therefore the sum of maximal user utilities:

$$\begin{aligned} U_{\text{cell }0}(\bar{\phi}, \overline{P_0}) &= \sum_{j=1}^{M} U_j^*(SNR_j) \\ &= \sum_{j=1}^{M} U_j^*\left(\frac{g_j \cdot \phi_j \cdot P_0}{\overline{\gamma_j} \bullet \overline{P_0} + (\sigma_B^2)_j}\right) \end{aligned} \tag{2.13}$$

Our objective is to maximize (2.13) subject to the power budget constraint:

$$\sum_{j=1}^{M} \phi_j \le 1 \tag{2.14}$$

Maximizing cell utility is an optimization problem of M variables. The optimal power allocated to each user, $(\phi_1, \phi_2, \ldots, \phi_M)$, can again be obtained by applying the Lagrange multipliers.

Notice that the cell power budget for the entire neighborhood, $\overline{P_0}$, remains unknown during the optimization; therefore, the optimal user power allocation, $(\phi_1, \phi_2, \ldots, \phi_M)$, thus the maximal cell utility, denoted as $U_{\text{cell }0}^*$, is a function of $\overline{P_0}$:

$$U^*_{\text{cell }0}(\overline{P_0}) = \underset{\phi_1, \ldots, \phi_M}{Max} \sum_{j=1}^{M} U^*_j(SNR_j) \tag{2.15}$$

$$\text{such that } \sum_{j=1}^{M} \phi_j = 1$$

The constraint function expressed in (2.14) can again be reduced to equality; since higher power budget yields higher cell utility, all the available power budget would be consumed by users. Again, the objective function is continuous and concave over the constraint set, the optimum $U^*_{\text{cell }0}(\overline{P_0})$ obtained by the Lagrange multiplier is a global maximum.

It is not difficult to prove that the two-step optimization process (i.e. the user level and then the cell level optimization) for maximizing the cell utility is equivalent to an one-step optimization at the cell level. This one-step optimization determines the B_i, the E_i for each application among all users, and ϕ_j for each user in the cell. However, the layered approach introduces parallelism thus reduces the intercell communication.

To summarize, we have achieved the optimal user power allocation for multiple users within a cell. The optimal cell utility depends on the cell power levels for the entire neighborhood. This result provides us with a platform to perform intercell power allocation, which is to determine the total cell power budget for each cell.

2.4.3. System Level Optimization

Our ultimate goal is to maximize the overall system utility which is achieved by setting the cell power budget for each cell. Since a large number of users and cells are in the system, we seek a distributed algorithm that is scalable in both computation and communication.

One difficulty for updating power levels for multiple cells independently and simultaneously is that they can introduce great amount of inaccuracy to channel estimate. For example, let us reconsider the cluster of sever cells as shown in Figure 2-7. If the power budget for two adjacent interfering cells, say cell #1 and cell #2, are changed independently and simultaneously, then the channel estimators in cell #1 would not react to the change made in cell #2 immediately, and thus produce a wrong estimate for some period of time. To overcome this problem, we developed a scheduling scheme for achieving our goal.

As intercell interference is localized to a finite region, changing the power level for a cell only affects its nearby cells; we call this interference region a cell's "neighborhood". This observation suggests that we are able to simultaneously change power levels for several cells (without concerning of their cross-interference), provided that their neighborhoods do not overlap. For this discussion, we assume only first order interference[1]. Figure 2-7 shows several non-overlapping neighborhoods (indicated by the "stars" superimposed on the cell topology).

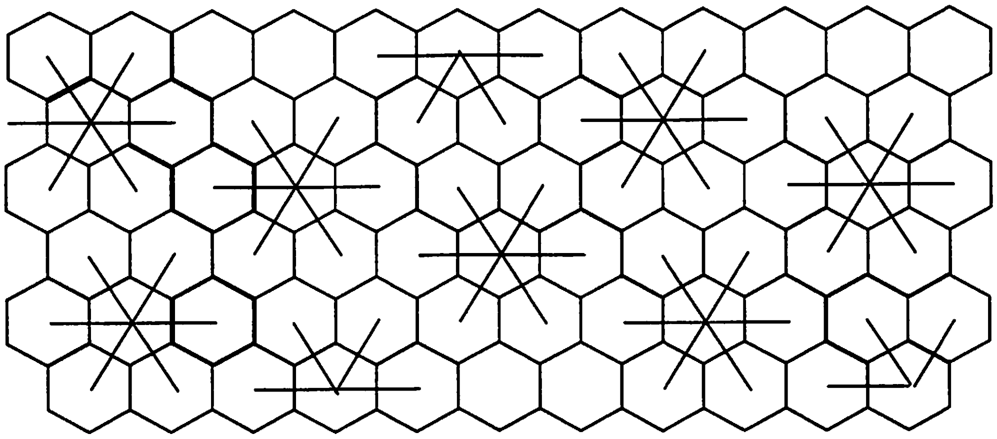


Figure 2-7. Non-overlapping interference neighborhoods.

Several key properties result if we restrict ourselves to changing power levels only for the center cells of these non-overlapping neighborhoods. First of all, the effects from changing a center cell's power level are limited to the neighborhood boundary; therefore, cells only need to communicate within the neighborhood. Secondly, a center cell faces a fixed interference environment; therefore, calculating the power budget is simplified. Finally, the remaining cells within the neighborhood have exactly one interfering cell changing its power. Therefore, estimating intracell and intercell interference coefficients

1. First order interference means that a basestation only interferes with its six immediate neighbor cells. This assumption is only for illustrative purposes, and is not necessary for our distributed algorithm. If the assumption does not hold, we can increase the neighborhood size, which will decrease the rate of convergence.

can be made with a single measurement, with the new interference noise power yielding an updated intercell interference coefficient.

So far, we have updated cell power budgets for only a fraction of cells. Our ultimate goal is to update power budgets for all cells in the system. This is achieved by iterating according to a periodic schedule[1], where the system adjusts a different subset of cells at each iteration; after one period, all cells in the system are updated at least once.

We will demonstrate this algorithm using the example of first order interference and hexagonal cell topology. Figure 2-8 shows the assignment of cells to iteration steps, and the iteration period equals to 7 in this case. Notice at every time step, the subset of cells which change power levels have non-overlapping neighborhoods.

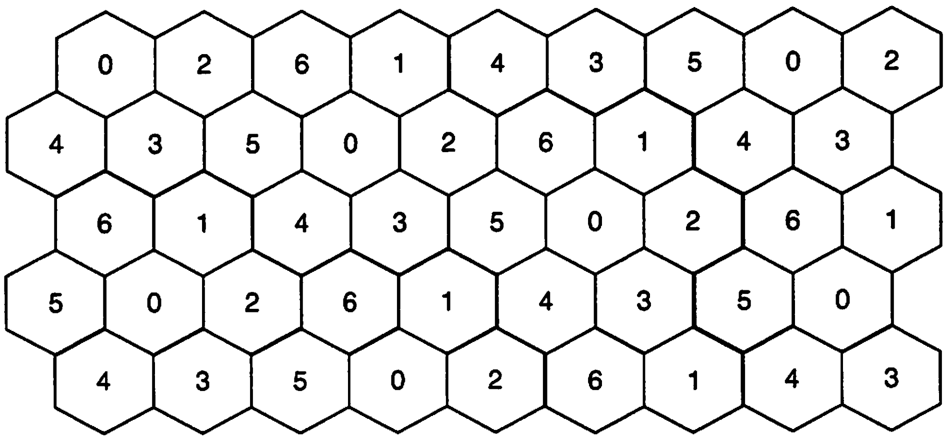


Figure 2-8. Iteration step assignments.

Now we need to determine the cell power budgets. The procedure is the same for all cells; however, we will only focus on cell #0 at this time, with neighboring cells 1 through 6. When updating the power budget for cell #0, its power level is chosen so as to maximize its overall neighborhood utility. We know from the previous section that the maximal cell utility for each cell in the neighborhood of cell #0 depends on P_0. Therefore, the total

1. This schedule is fixed and determined by the system designer.

neighborhood utility, which is the sum of the maximal cell utilities, is also a function of P_0, as shown below:

$$U_{\text{Neighborhood}}(\overline{P_0}, \overline{P_1}, \ldots, \overline{P_6}) \tag{2.16}$$
$$= U^*_{\text{cell }0}(\overline{P_0}) + \sum_{i=1}^{6} U^*_{\text{cell }i}(\overline{P_i})$$

While P_0 is upper bounded by the implementation limit, the goal of adjusting P_0 is to maximize the total neighborhood utility, thus P_0 has to satisfy:

$$\frac{\partial}{\partial P_0}[U_{\text{Neighborhood}}(\overline{P_0}, \overline{P_1}, \ldots, \overline{P_6})] = 0$$

This is equivalent to:

$$\frac{\partial}{\partial P_0} U^*_{\text{cell }0}(\overline{P_0}) = -\frac{\partial}{\partial P_0}\left(\sum_{i=1}^{6} U^*_{\text{cell }i}(\overline{P_i})\right) \tag{2.17}$$

Notice the left side of (2.17) is cell #0's marginal utility as a function of P_0, and on the right is the total marginal utility of the neighbor cells. At the optimal point, with respect to P_0, the marginal utility of cell #0 offsets the marginal utility of the neighbor cells.

2.4.4. Computational complexity

The utility optimization algorithm is fully distributed, with its level of complexity independent of the number of cells in the system. The optimizations are performed independently and are in parallel at each level. At the highest system level, the iterative algorithm determines power level for a cell at each iteration (with given information on inter-cell interference), thus the algorithm only deals with one variable at each iteration. At the cell level, the complexity of cell utility optimization depends on the number of users in a cell. This step can be quite computational intensive for a cellular environment where hundreds of users maybe present during a congestion period; however, in a pico-cell situation where a cell usually has just a few users, the complexity is easily manageable given that computational power is not a concern at the basestation. Finally at the user level, the number of variables for user level optimization is twice the number of applications (since both bandwidth and error rate are determined for each application). For example, if a user has three distinct applications, then the control algorithm has six variables.

Since the wireless environment and user behaviors are time varying, this optimization algorithm is dynamic and updates itself frequently with the latest channel information and user demands. In a mobile cellular environment where a car may move as fast as 70 mile per hour, the updates have to be carried out very frequently (in the order of seconds), especially at the cell level for adapting the fast time-varying channel. As a result, implementing this algorithm under this time constraint can be very challenging, especially when the number of users in a cell can also be large. On the other hand, in an indoor pico-cell environment where channel quality varies slowly, the algorithm focuses more on changes in user behavior in terms of requested applications. Therefore, updates occur most frequently at the user level., which is a more manageable problem as the complexity of user level algorithm is only a function of the application count.

2.5. Study of Constraint Set and Feasible Region

The system utility optimization algorithm we described above is indeed an iterative, one-dimensional search algorithm, in which we optimize along each P_i iteratively until reaching the maximum. There are known numerical algorithms [8][37] that meet this purpose. However, our objective function, the system utility, is non-concave, thus the iterative algorithm provides no guarantee that it will converge to the global maximum (instead of a local maximum). The problem of solving the global optimization is a very difficult one indeed. In this case, either additional constraints are added as a part of the problem formulation for eliminating the local optima, or the utility function has to be restricted to certain classes of functions in order to guarantee that any optimum is a global optimum.

The fundamental theorem of mathematical programming states that a continuous concave objective function with a convex constraint set guarantees the uniqueness of a global optimum. Therefore, we will approach the same optimization problem from a different perspective by transforming the objective function into a concave function. The goal here is to study the property of the constraint set in this new domain.

Recall that the system utility is the sum of the cell utilities, and the cell utility is the sum of the user utilities. Therefore, the system utility can be expressed as the sum of all user utilities. The optimum user utility is a function of the channel SNR, which is denoted by $u_i(S_i)$ for user i.

$$\text{system utility} = \sum_i u_i(S_i) \tag{2.18}$$

We have made the assumption that the user utility is a non-decreasing function with respect to the channel SNR, indicating higher SNR would always bring a user more satisfaction. Furthermore, we assume the marginal utility (i.e. the first derivative of utility function) decreases as a function of SNR, indicating diminishing marginal satisfaction for every user. Therefore, the system utility is a non-decreasing and concave function with respect to individual user's SNR.

Let us now focus on the constraint set. To demonstrate the non-convexity of the constraint set, we consider a simple example consisting of a single cell with two users; these users have transmit power levels p_1 and p_2, respectively. Recall that the total cell transmit power is subject to a constraint, for convenience, we set this power constraint equals to 1, which can then be expressed as:

$$p_1 + p_2 \le 1 \text{ and } p_1 \ge 0, p_2 > 0. \tag{2.19}$$

Now we want to rewrite the above inequality in terms of users' SNR, so that the objective function (2.18) and the constraint function (2.19) are expressed in the same parameters. Using the channel model we have developed earlier, the SNR for each user can then be written as:

$$S_1 = \frac{p_1}{a_1(p_1 + p_2) + c_1}$$

$$S_2 = \frac{p_2}{a_2(p_1 + p_2) + c_2}$$

where a_i is the intracell interference coefficient, and c_i is the background thermal noise power received by each user. Both a_i and c_i are normalized by the path loss and antenna gain of user i. Notice there is no intercell interference term because we have a single cell system here.

Let us first focus on the power level for user 1, p_1. Solving the system of linear equations for p_1, we have:

$$p_1 = \frac{c_1 S_1 + (a_1 c_2 - a_2 c_1) S_1 S_2}{1 - a_1 S_1 - a_2 S_2}$$

Since $p_1 + p_2 \le 1$ and $0 \le p_1 \le 1$, we can rewrite the above equation as:

$$c_1 S_1 + (a_1 c_2 - a_2 c_1) S_1 S_2 \le 1 - a_1 S_1 - a_2 S_2$$

or

$$S_2 \le \frac{1 - (a_1 + c_1) S_1}{a_2 + (a_1 c_2 - a_2 c_1) S_1}. \tag{2.20}$$

Now the constraint set is expressed in user SNR S_1 and S_2. Plotting it, we have:

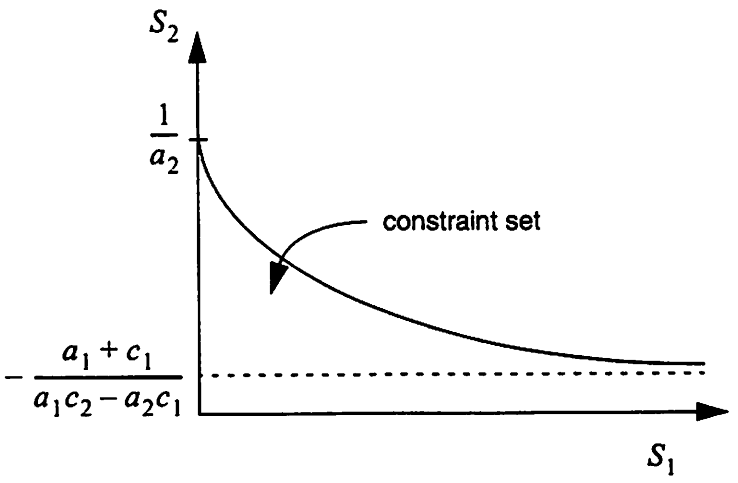


Figure 2-9. Power constraints expressed in SNR.

Clearly, the constraint set shown in Figure 2-9 is non-convex. The second constraint set derived from the inequality $0 \le p_2 \le 1$ is similar to (2.20) and is also non-convex. The overall constraint set is the intersection of these two, and the intersection of two non-convex sets is also non-convex. Therefore, we conclude the constraint set for the optimization problem is non-convex. All of these proved that a maximum obtained by optimization may be a local maximum.

There exist powerful numerical techniques, such as simulated annealing [46][33][25], for searching the global optimum. However, these tools have been mostly used for non-real time applications, such as routing in integrated circuit design. For a real-time application involving a large number of variables such as ours, the computational delay is likely to be intolerable.

Despite of the difficulties encountered in obtaining the global maximum, we should not feel despair. Observe that any local maximum is associated with a feasible region, such that the local optimum is the optimum within this feasible region. An example is illustrated in Figure 2-10, where the shaded region corresponds to the feasible region associated to the local maximum x^*. In general, if the feasible region covers the majority of the constraint set, such as in this example, then the corresponding local maximum is quite superior with respect to all achievable SNR. In other words, a local optimum may serve as a good alternative to the global optimum when it out-performs a large set of SNR. For our power control problem, we may combine the optimization algorithm with a fairness scheme to eliminate the extreme SNR differentials among users, e.g., some of the blank areas shown in Figure 2-10. In this case, a local optimum may indeed become the global optimum. The implementation and the impact of a fairness scheme will be discussed in Chapter 4.

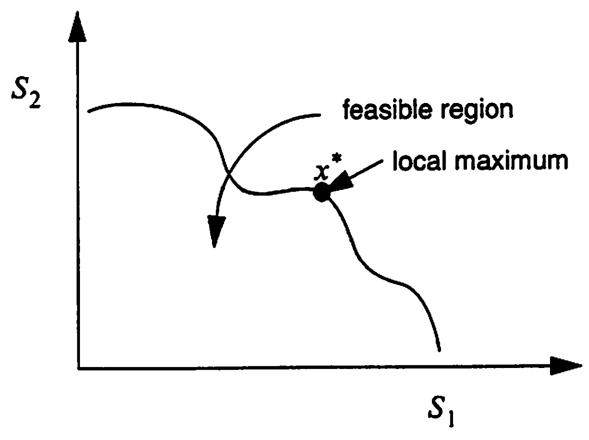


Figure 2-10. Feasible region subject to the constraint set.

2.6. A different approach to the problem: economics

2.6.1. Concept of pricing

So far we have only approached the overall system optimization using pure mathematical techniques. The problem of optimal resource allocation has been extensively studied in the economics community, where the commodities are distributed through the interactions of supply and demand. On the demand side, users trade-off various service qualities with their willingness to pay; while on the supply side, producers trade-off the provided service quality with the charge they are able to collect. For the rest of this section, we will explore the possibility of adopting pricing for allocating shared wireless networks resources, the transmit power.

It has been argued that network users in fact can tolerate any kind of service quality, and there is no required QoS as long as the price is right. This argument is supported by examples such as the internet telephone, where delay is unpredictable and data sometimes get lost; however, this kind of service still remains popular among people who do not want to pay the premium price for the toll-rate quality. In reality, pricing can often be used as a mean to reveal the true utility function for a user: when the asking price is higher than the marginal utility, a user would be discouraged from requesting higher QoS. On the other hand, if the price is lower than the marginal utility, this user would want to receive service at higher quality since he is willing to pay more money. To exploit this idea further, let us first look how we may use pricing to solve a day-to-day problem: the problem of rush hour commute. As we will see very soon, this problem is very analogous to the congestion in wireless network, which is mainly caused by user interference.

In the case of rush hour commute, the public or collective goods are the state highway systems with a finite number of lanes. When the vehicles per hour exceeds the highway capacity, the highway becomes congested. As more and more vehicles join the highway, the congestion becomes worse. Notice each extra vehicle delays other vehicles taking the same route. In other words, a commuter can only enjoy the goods at the cost of reducing others' enjoyment. This phenomenon is called the *externality*, where the pursuit of private gain may not promote the social welfare. When dealing with wireless network resource allocations, we face the same kind of externality, for increasing one user's transmit power level would always mean an increase in interference to others in the system. Thus a gain in one person's satisfaction does not always lead to an improvement in overall system utility.

One possible solution for solving this type of the problem is to set prices for the peak and the off-peak periods so that they approximate the marginal costs in these respective periods. The result is a form of marginal cost pricing. This policy is reasonable since marginal cost tends to be higher during the peak hours so consumers are discouraged from using it. Of course, the optimal price for a consumer is when the marginal consumer utility equals to the marginal cost to the society, which is exactly what we have proposed in (2.17). One such pricing scheme has been implemented in England where the cost of electricity depends on the time of the day [1]. In that case, the rates are higher during the times of day when electricity usage is high than the time of the day when the usage is relatively small.

2.6.2. Congestion pricing for wireless network

Extending this concept on marginal cost pricing to wireless communication, we will consider a congestion charge with the basic idea that the charge is higher when the network is congested, and there is no charge when the network is un-congested. The goal of imposing congestion charges is to regulate the network congestion level through the interactions between prices and user demand, thus maximizing the overall system utility. The shared resource is the transmit power, and the congestion is measured by the user SNR. The key point of this approach is to set the price which equals (or approximates) a user's marginal utility so that each user would pay for the overall damage he has caused to others. Let us now proceed and calculate the price.

Recall that a user's satisfaction is measured by channel SNR. The user utility, $u_i(S_i)$, for user i is the dollar value when data are received at SNR $= S_i$, where S_i has the unit in dB. Suppose that the user is charged a congestion price of π_i (per unit dB) for the SNR he received, then the user i will choose to receive at a SNR such that it solves the following problem:

$$\max_{S_i} u_i(S_i) - \pi_i \cdot S_i \tag{2.21}$$

Note the term $\pi_i \cdot S_i$ is the total price this user pays for S_i. The optimum SNR is solved by:

$$\frac{\partial}{\partial S_i} u_i(S_i) = \pi_i \tag{2.22}$$

This is equivalent to saying the price is set to be the user's marginal utility. This result is reasonable since marginal utility is the additional satisfaction (measured in dollar) a user get for receiving the extra ΔS_i. If this additional satisfaction is less than the price he has to pay, then this user would decide not to receive the extra ΔS_i. In other words, the equilibrium occurs at the point when price equals to the marginal utility.

Let us now obtain the marginal utility, thus the congestion price, by considering all the interfering users in the system simultaneously. The cell power constraint is included in this process. For convenience, we let total allowable power for each cell equal to 1 unit. Let us consider the central controller that chooses transmit power level p_i for every user i subject to this constraint for maximizing the total system utility. Applying the Lagrange multipliers, the problem we need to solve is:

$$\max_{p_i}\left\{\sum_{j\in J}u_j(S_j)-\lambda_1\cdot\left(\sum_{j_1\in J_1}p_{j_1}\right)-\lambda_2\cdot\left(\sum_{j_2\in J_2}p_{j_2}\right)-\lambda_3\cdot\left(\sum_{j_3\in J_3}p_{j_3}\right)-\ \ldots\right\}$$

where

J is the index for all users,

J_i is the index for all users in cell i,

λ_n is the Lagrange multiplier for cell n as each cell is subject to a total power constraint.

The optimum value of p_i is obtained by solving the equations

$$\frac{\partial}{\partial p_i}u_i(S_i) = \frac{\partial}{\partial p_i}\left(-\sum_{j\neq i}u_j(S_j)+\lambda_1\sum_{j_1\in J_1}p_{j_1}+\lambda_2\sum_{j_2\in J_2}p_{j_2}+\lambda_3\sum_{j_3\in J_3}p_{j_3}+\ \ldots\right)$$

Therefore,

$$\frac{\partial}{\partial p_i}u_i(S_i) = -\frac{\partial}{\partial p_i}\sum_{j\neq i}u_j(S_j)+\lambda_n \text{ for some } n \tag{2.23}$$

Let us first discuss the value for the Lagrange multipliers, λ_n. Intuitively, we are maximizing the sum of non-decreasing functions. During the optimization, each one of these functions will keep increasing (at various rates) until one cell's power consumption reaches the power limit. Therefore, there is always one cell that would consume all the available power budget. In fact, this is the cell that has the highest marginal utility, i.e. $\sum_{j\in J}\frac{\partial}{\partial p_j}\left(\sum_{j_i\in J_i}u_{j_i}(S_{j_i})\right)$ being the largest for all J_i. So we know the maximum always occurs at the boundary condition, thus $\lambda_i\in\{0,1\}$: 0 when the cell i consumes less than the total allowable power limit, and 1 when cell i consumes all the allowable power limit.

Let us go back to the optimization problem stated in (2.23). Expanding $\frac{\partial}{\partial p_i}u_i(S_i)$ by applying the chain rule, we have:

$$\frac{\partial}{\partial p_i}u_i(S_i) = \frac{\partial}{\partial S_i}u_i(S_i)\cdot\frac{\partial S_i}{\partial p_i} = -\sum_{j\neq i}\frac{\partial}{\partial S_j}u_j(S_j)\cdot\frac{\partial S_j}{\partial p_i}+\lambda$$

or

$$\frac{\partial}{\partial S_i}u_i(S_i) = \frac{\lambda-\sum_{j\neq i}\frac{\partial}{\partial S_j}u_j(S_j)\cdot\frac{\partial S_j}{\partial p_i}}{\frac{\partial S_i}{\partial p_i}} \qquad \text{where } \lambda\in\{0,1\} \qquad (2.24)$$

To obtain an explicit expression for $\frac{\partial}{\partial S_i}u_i(S_i)$, we now need to derive the partial derivatives $\frac{\partial S_i}{\partial p_i}$ and $\frac{\partial S_j}{\partial p_i}$.

First we rewrite the user SNR in terms of transmit power level of other users and interference coefficients:

$$S_i = \frac{p_i}{\sum_k a_{ik}p_k + c_i}$$

where a_{ik} denotes the fraction of user k's transmit power that interferes with user i, and c_i is the background thermal noise received by user i. Both a_{ik} and c_i are normalized by path-loss, the antenna gain and the spreading factor of user i. Note a_{ij} represents the inter-user interference coefficient when $i\neq j$, and the self interference when $i = j$. With this notation, the SNR for user j, S_j is: $S_j = \frac{p_j}{\sum_k a_{jk} p_k + c_j}$. Now taking the partial derivatives of S_i and S_j with respect to p_i, we have:

$$\frac{\partial S_i}{\partial p_i} = \frac{\sum_{k \neq i} a_{ik} p_k + c_i}{\left[\sum_k a_{ik} p_k + c_i\right]^2}$$

$$\frac{\partial S_j}{\partial p_i} = -\frac{a_{ji} p_j}{\left[\sum_k a_{jk} p_k + c_j\right]^2}.$$

Plug them into (2.24), the marginal utilities then becomes:

$$\frac{\partial}{\partial S_i} u_i(S_i) = \frac{\lambda + \sum_{j \neq i} \frac{\partial}{\partial S_j} u_j(S_j) \cdot \frac{a_{ji} p_j}{\left[\sum_k a_{jk} p_k + c_j\right]^2}}{\frac{\sum_{k \neq i} a_{ik} p_k + c_i}{\left[\sum_k a_{ik} p_k + c_i\right]^2}}$$

Recall from (2.22) that $\frac{\partial}{\partial S_j} u_j(S_j) = \pi_j$, i.e. the congestion charge for user j, and $\frac{p_j}{\sum_k b_k p_k + c_j} = S_j$; we therefore can rewrite the above equation as:

$$\frac{\partial}{\partial S_i} u_i(S_i) = \frac{\lambda + \sum_{j \neq i} a_{ji} \frac{\pi_j S_j}{\sum_k a_{jk} p_k + c_j}}{\frac{\sum_{k \neq i} a_{ik} p_k + c_i}{\left[\sum_k a_{ik} p_k + c_i\right]^2}} \tag{2.25}$$

This is the congestion prices for user i expressed in interference power and SNR; note this price is independent of user i. This implies that the congestion level seen by user i only depends on the current activities of other users in the system.

Let us simplify the above equation by making the assumption that the system has a large number of users, thus the self-interference is negligible compared to the sum of all inter-user interference power, i.e.

$$a_{ii}p_i + c_i \ll \sum_k a_{ik}p_k + c_i$$

Therefore, the denominator of the above equation can be approximated as $\frac{1}{\sum_k a_{ik}p_k + c_i}$, which is the inverse of the total interference and noise power received by user i. Let us denote this by N_i, thus $N_j = \sum_k a_{jk}p_k + c_j$. With this notation, the marginal utility, or the congestion charge for user i, becomes:

$$\pi_i = \frac{\partial}{\partial S_i}u_i(S_i) = \lambda N_i + \sum_{j \neq i} a_{ji}\frac{N_i}{N_j} \cdot \pi_j S_j \qquad \text{where } \lambda_i \in \{0, 1\} \qquad (2.26)$$

The product term $\pi_j S_j$ in (2.26) is congestion price (in dollars) paid by user j for SNR equals to S_j (in dB). We can see that the unit price for user i is independent of his own SNR; instead, the unit price is linear with respect to the SNR of other users, which can be understood as an indicator of the congestion level of the wireless network.

Finally, we want to make the remark that most users may find it confusing to deal with a complicated pricing scheme such as congestion pricing, mostly because it depends on the system load and changes over time. One resolution for this is to have the service provider purchase network resources according to the congestion market, and then resell them to the consumers at a fixed price or a simple peak/non-peak price that are higher than the average congestion prices. This pricing model has been considered for wired networks such as the internet, where serious congestion occurs from time to time [11].

2.7. Summary and Conclusions

This chapter investigated design and control issues faced by multimedia indoor CDMA systems. We have proposed a distributed algorithm that fully utilizes the system resources such as bandwidth and transmit power. Our algorithm integrates three techniques for multimedia downlink transmission; these techniques are: power control, VFEC and scheduling. As a multimedia system is designed to satisfy users, our objective is to maximize the overall user satisfaction, which we call the "system utility".

We divided this system optimization problem into a hierarchy of three levels: user level, cell level and system level. This partitioning allows us to localize the problem, so that we can perform independent and local optimizations for each user and then each cell. Because users and cells in a CDMA system are subject to bandwidth and power constraints, we achieve the optimization at each level by applying the Lagrange multiplier. The optimization results are then presented to the next level in the hierarchy. At the topmost level, the system level, we allocated cell power to one set of cells while keeping their neighboring cell power levels constant. The cell power level is determined based on its contribution to the overall utility for its entire neighborhood of seven cells. This layering approach yields a distributed algorithm, which is essential given any practical wireless system has to be scalable.

We then studied the constraint set for the optimization problem and found out it is non-convex. As a consequence, the optimization algorithm may sometimes converge to a local optimum. For any local optimum, we can derive the feasible region indicating the domain of users' SNR (within the constraint set) that this local optimum is the optimum. If a feasible region covers the majority of the constraint set, then this local optimum is quite optimum.

Finally in the last section of this chapter, we presented a new approach to the network resource allocation problem using concepts from economics, namely, the concept of congestion pricing. With this approach, the interference introduced by each added user is treated as additional congestion to the system from this user. By setting the congestion prices according to the system load (measured in users' SNR), we can discourage users from heavy usage when the congestion level is high. The correct congestion prices reflect the optimum resource allocation. The derivation of the congestion price is included in the chapter. The result showed that the price is user independent. When dealing with a large system, we can simplify the result and the congestion price becomes linear with respect to other users' SNR.

3 Simulation Based Performance Evaluation

3.1. Goal

In the previous chapter, we have presented the "utility"-based approach to the overall control problem. The main advantage of this framework is that it offers large degrees of flexibility to the system control, because the policy issues of allocating resources are separated from the design process. In other words, one can always choose a utility function that reflects a specific design objective. For example, if we let the application utility function be a step function (with respect to both bandwidth and error rate), then this would coincide with conventional design objectives for many wireless or cellular systems, where a BER (or equivalently SNR) requirement is imposed. However, this advantage of flexibility makes it difficult to quantify and compare system performance with other approaches. Therefore, in this chapter, we will simulate the system using a specific utility function, the step function. This restriction will allow us to evaluate and quantify system performance with numerical results. In addition, these results will uncover some interesting dependencies which differentiate the multimedia optimization problem we are investigating from a single media application.

The goal of this chapter is to set up a system level simulation and use it to explore the system behavior, including the interactions from various parts of the control algorithm in detail. In order to do so, we will study the performance dependencies on user distribution and traffic models. We will also study the individual control variables and compare their relative effectiveness. In addition, the technique of hand-off is applied to highly congested "hot-spots" for studying the overall improvement. The range for the optimum packet sizes will also be studied. Finally in this chapter, we will study performance of various families of binary BCH codes (used for variable error correction purpose) while consider their implementation complexities. As a remark, the underlying system architecture for simulation is assumed the same as the downlink system presented in Chapter 2.

Before discussing the actual simulation, we will first present how some of the previous work on power control can be applied directly to this framework by proving convergence of the power control algorithm. The purpose for doing this is to prove that by adopting this special class of utility functions for our multimedia system, the power control algorithm converges.

3.2. Convergence of power control algorithm

Yates [55] has proved the convergence of a general power control algorithm for the uplink channel. In his work, he assumed users' basestation assignments are fixed, and each user has a pre-determined BER (or SNR) requirement. The goal of the power control algorithm is to find a set of minimum transmit power levels which satisfies these requirements. For both synchronous or asynchronous power update. Yates showed that for any initial power vector $\boldsymbol{p}$, the power control algorithm converges to a unique fixed point if a feasible solution exists. The criteria for convergence are based on the observation on interference function for each user j, $I_j(\boldsymbol{p})$, has the following properties:

- Positivity: $I(\boldsymbol{p}) > 0$
- Monotonicity: If $\boldsymbol{p} \geq \boldsymbol{p}'$, then $I(\boldsymbol{p}) \geq I(\boldsymbol{p}')$
- Scalability: for all $\alpha > 1$, $\alpha I(\boldsymbol{p}) > I(\alpha \boldsymbol{p})$.

The first condition indicates that SNR for any user is always positive; this is derived from the assumption that background noise is nonzero, and interference noise power are all non-negative. The second condition indicates the monotonicity of the SNR, namely, a user's SNR is always a monotone increasing function with respect to its transmit power. The last condition implies that if transmit power for all users are scaled up by the same fraction α, then every user would end up having a higher SNR, which again is due to non-zero background noise.

Once these three conditions are met, it has been shown that:

(1) If the power control algorithm has a fixed point [1], then the fixed point is unique, and
(2) If there exist a feasible solution, then for any initial power vector ***p***, the power control algorithm converges to a unique fixed point, ***p****.

This is a very general result for it provides convergence of power control algorithm without assuming a specific expression for the interference function ***I(p)***, where ***I(p)*** is highly dependent on the underlying wireless technology, whether it is CDMA, FDMA, slotted-aloha, etc. However, this result proves the convergence of power control algorithm only when a feasible solution exists. In the case when there is no solution satisfying all users due to excessive interference or traffic demand, call admission problem need to be solved in finding a subset of users that can obtain satisfactory connection [2] [7]; or in our case, provide partial service that only a fraction of all requested data are transmitted, and apply heavy error correction to achieve the desired BER.

We are interested in this convergence property because it can be extended to a downlink system like the Infopad where multiple traffic types are present. However for a downlink CDMA system, we must first realize its underline interference model is completely different from the uplink as the downlink channel is free of the near-far effect. This is because signals transmitted from a basestation to all users are orthogonally coded and sent to users simultaneously. As these signals propagate through the same path, they remain orthogonal when reaching the receiver. In other words, signals generated from the same basestation do not interfere with the main propagation path. The reflected waveforms and the waveforms from nearby cells are still treated as interference since orthogonality no longer holds.

Using the interference model developed in Chapter 1, it can be verified that the interference function,

$$SNR_j = \frac{g_j \cdot \phi_j \cdot P_0}{\overline{\gamma_j} \bullet \overline{P_0} + (\sigma_B^2)_j}$$

1. the *fixed point* is a point x such that $f(x) = x$; that is, a point x that does not move when applying function f.

satisfies the three properties of $I_j(p)$.

Even though our proposed downlink system serves multiple data types that have a wide range of BER requirements, however, the multiple BER requirements are achieved by applying VFEC, which is completely decoupled from the power control aspect of the problem. Therefore, this multimedia constraint impose no additional constraint on our power control algorithm, thus the power control algorithm converges in our proposed framework.

3.3. Algorithm discussion

For simulation purposes, we assume a linear cell topology as shown in Figure 3-1. This layout is chosen because it is the simplest topology which contains all aspects of a multi-cell environment. Since the system performance is a function of user population, location and traffic, the simplicity of this layout makes the dependencies and cell interactions more transparent for analysis. We believe the general results and trends from this case carry over to more realistic topologies, because the effect of inter-call and intra-cell interference are always present disregarding the number of neighbor cells.

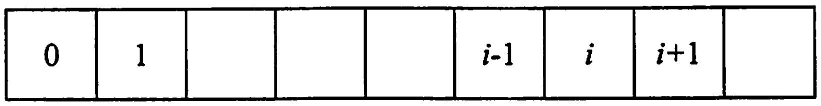


Figure 3-1. Cell topology.

Let $\overline{P_i} = (P_{i-1}, P_i, P_{i+1})$ be the total power budget for each one of the three adjacent cells. Using the same model and the notation developed in Chapter 1, the channel SNR for any user, say user j in cell#1 is:

$$SNR_j = \frac{g_j \cdot \phi_j \cdot P_1}{\alpha_j \cdot P_1 + \beta_j^1 \cdot P_0 + \beta_j^2 \cdot P_2 + (\sigma_B^2)_j} = \frac{g_j \cdot \phi_j \cdot P_1}{\overline{\gamma_j} \bullet \overline{P_0} + (\sigma_B^2)_j}$$

For the rest of the chapter, we assume a basestation transmitter is installed on the ceiling at the center of every cell. The basestation assignment is made based on the distance between user and receiver, so that all users in cell #*n* would receive signals from the basestation #*n*. Later in this chapter we will study some handoff schemes, in which case a user may communicate with a less congested neighboring cell.

3.3.1. User level

At the user level, the techniques of scheduling and VFEC are applied. Figure 3-2 illustrates the architecture for this sub-system. As previously mentioned, the system sometimes can only provide partial service due to excessive interference, in which case the scheduler would only send the high priority data to the VFEC encoder and then to the transmitter. For our purposes, we choose the rate-controlled static priority (RCSP) [59] scheduling scheme, where all data types are classified according to a set of priority levels. The relative priority levels are determined by many factors, including the relative importance of the data, their delay tolerance, and the amount of network resources needed for delivering them at the required QoS. These priority levels are fixed (or static) for the entire connection. In the case of Infopad, the priorities are set as: control, audio, text/graphic and video.

The scheduling decisions are simple: higher priority data are always processed ahead of lower priority data, and within the same priority class, packet are served in an FIFO fashion. In other words, the lower priority data which arrived the same time as higher priority data can only get ahead when there is insufficient network resource to transmit the higher priority data. This scheduling algorithm is simple and well understood and its detailed characteristics are not important when we later evaluate power control and VFEC. As a final remark, a perfect channel estimation is assumed for each user.

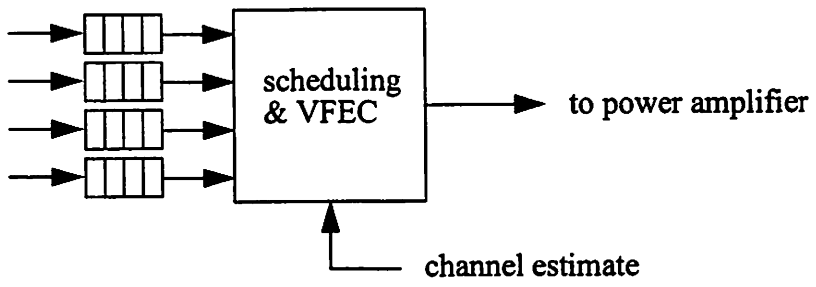


Figure 3-2. Architecture for user level resource allocation.

3.3.2. Cell level

At the cell level, the intra-cell power allocation is performed in such a way that users reside in the same cell compete for transmit power subject to the total cell power budget.

"Fairness" is imposed at this level because some users need more transmit power to compensate for a poor channel and/or heavy traffic demand. In the case when there is *not* enough power to meet everyone's demand, it is difficult to justify why one user should receive more network resource than another. In Chapter 3, we presented a utility driven optimization framework for allocating resources, in which case no fairness was considered. In this chapter, we will implement a "fairness" policy (defined below). We will compare the results against the algorithm that maximizes the number of users later in Section 3.5.2.

The "fair" power allocation consists of two stages: fairness-based power allocation followed by demand-based allocation, as shown in Figure 3-3. During the fairness-based power allocation, the total cell transmit power is equally divided among all users. The purpose of this step is to guarantee each user with a fair share of the available resource, independent of the channel quality or traffic demand. If the allocated resource is greater than a user's demand, the excess resource is then collected and then redistributed among users (within the same cell) whose demands are greater than the allocated power. We call this redistribution the demand-based power allocation.

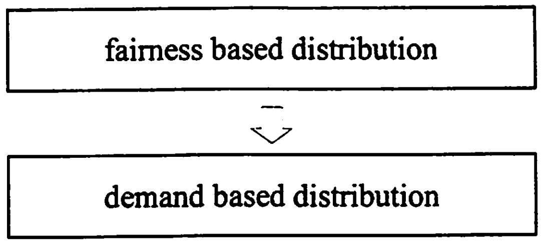


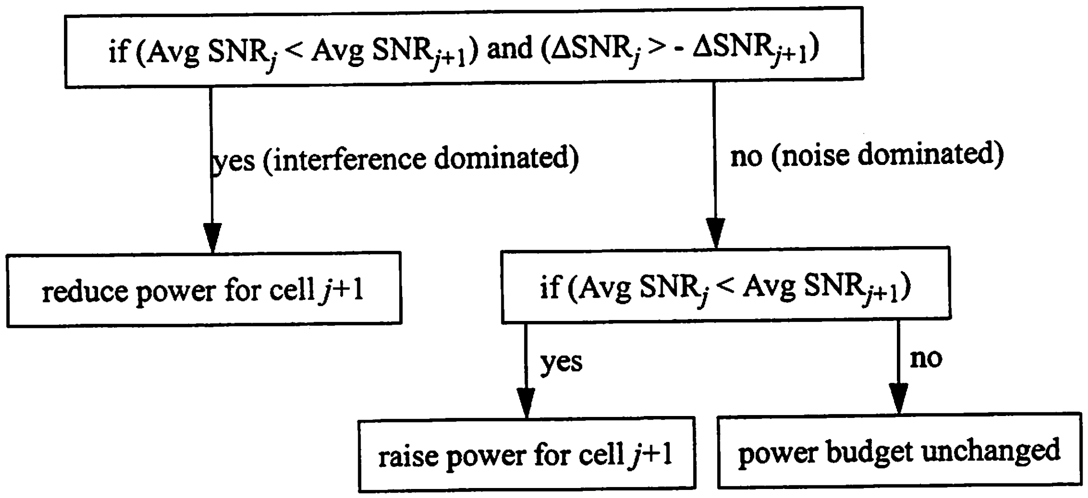
Figure 3-3. Two stage user level power control.

During each intra-cell iteration, more than two sets of state information are exchanged between the user level and the cell level allocations. The first exchange occurred during the fairness-based allocation, where the allocated power is sent to all users followed by their feedbacks. The second exchange occurred during the demand-based allocation, where the excess power is redistributed. In case there is still excess power and needing users after this second exchange, same redistribution is performed again. According to our simulation, the number of total exchanges is usually less (but sometimes equals) to three.

3.3.3. System level

Finally at the system level, the individual cell power budget is determined based on inter-cell interference and the total cell demand. Since our system is interference limited, fairness issues again come up at the system level. However, the nature of fairness at this level is somewhat different from what has been discussed at the user level. This is because cells do not share any network resources; instead, cells interfere with one another. As a result, allowing one cell too much power would generate excessive interference to its neighbors, thus degrade the QoS for the overall system.

Taking this into consideration, we propose a SNR-based inter-cell power control algorithm. The principle for this algorithm is that no cell should get more transmit power unless the increase in its average SNR is greater than the total decrease of average SNR from its neighbor cells due to this increase of power. This concept has been described by the utility model in Chapter 3, except we are now taking the cell utility function as the identical function with respect to the channel SNR. The detail of this algorithm is described by the flow chart below.



if (Avg SNR$_j$ < Avg SNR$_{j+1}$) and (ΔSNR$_j$ > - ΔSNR$_{j+1}$)
yes (interference dominated)
no (noise dominated)
reduce power for cell j+1
if (Avg SNR$_j$ < Avg SNR$_{j+1}$)
yes
no
raise power for cell j+1
power budget unchanged

Figure 3-4. SNR-based inter-cell power allocation.

By now we have completely described our distributed algorithm. This algorithm is implemented in the simulation. Next we will describe our simulation setup and parameters, followed by numerical results.

3.4. Simulation setup and parameters

3.4.1. Channel model

Our broadband CDMA multiple access scheme is designed based on the IS-95 standard [43]. The data for each user is spread by a PN code and then added to a Walsh code. As mentioned in Chapter 1, the reason for this is that when signals arrived at the receiver, the ones from other users would appear as white noise due to spreading. Therefore, we model the interference from nearby cells as white noise that depends on the spreading factor and attenuation. The attenuation factor is set to be 4. The background noise is also modeled as white Gaussian noise. Since our system is designed for indoor users, we assume people are mostly stationary or moving around slow, thus fading is not included in our model.

3.4.2. Simulation parameters

Recall that we assume a linear cell topology. Four cells are simulated; they are labelled as 0, 1, 2, 3, and each has the population 4, 6, 5, 5 users, respectively. The users are assumed to be uniformly distributed in each cell. Each cell is 5x5 meter^2 in dimension, and the ceiling is 4 meters high. Since it is difficult to compare users when they have different traffic profiles, we assume all users request four types of data. Each data type has a fixed BER requirement, so data are only transmitted when the estimated received BER (after FEC decoding) equals or exceeds this predetermined level. In addition, these are indoor users, and we assume change in user locations are negligible comparing to the high data rate and the frequencies which inter-cell and intra-cell algorithms are performed.

Experimentally, we found it is sufficient to simulate the system for 50,000 iterations, with each iteration corresponding to 1 msec. Both user level and cell level algorithms are executed at each iteration, whereas the system level algorithm is carried out once every three iterations. Later in Section 3.5.3, we will present the system performance using various traffic models, including Poisson arrival, constant-bit-rate, two-state continuous-time

Markov chain. However, for the most part of this study, the packets for each data type are modeled by Poisson arrivals, unless it is specified otherwise. Table 3-1 shows the parameters for the Poisson arrivals of all data types along with their relative priorities.

priority	BER requirement	packet size (bits)	arrival rate (msec)	data rate (kbit/sec)
1 (control)	10^{-9}	100	8	12.5
2 (text/graphics)	10^{-4}	200	1	200
3 (audio)	10^{-5}	100	2	50
4 (video)	$5x10^{-4}$	100	0.125	800

Table 3-1. Traffic profile.

3.5. Results

The system performance is evaluated using average aggregate throughput of delivered data for each user. This average aggregate throughput is calculated with the data that are received at the required BER. The maximum throughput at receiver for any user, is around 1100 kbit/sec, according to Table 3-1.

3.5.1. Dependencies on user distribution

First let us explore the dependencies of throughput on user distribution. Ten independent and random sets of user locations are simulated while the population in each cell remains the same. Figure 3-5 shows the average aggregate throughput for all users in the system. Users are ranked according to their throughput, from highest to lowest.

Clearly, the performance depends very much on the user distribution: the set of location which yields the best performance is shifted to the right by nearly five users comparing the location yields the worst performance. However, despite of the variations from one location to another, all throughput distributions are nevertheless skewed with more users residing on the left side, indicating majority of users have very little data loss.

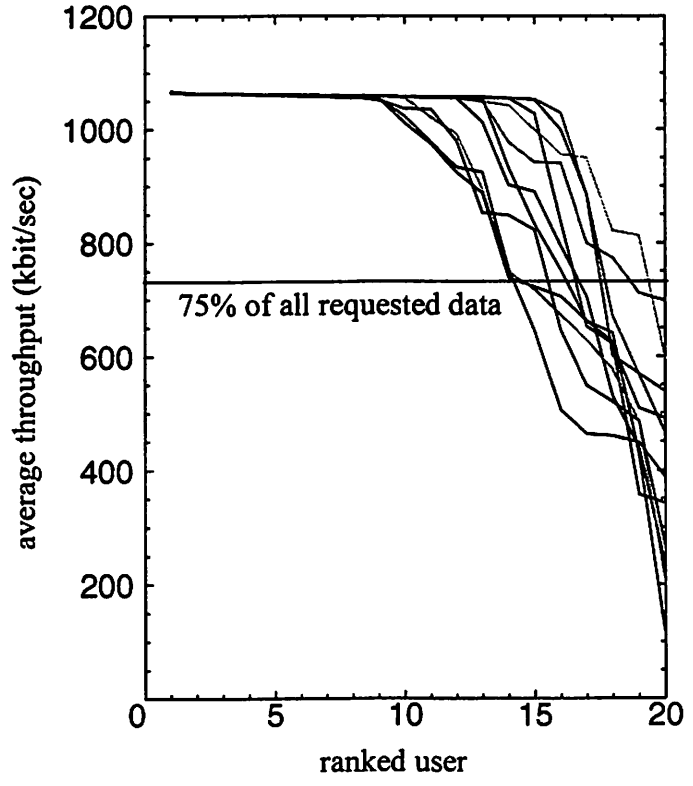


Figure 3-5. User throughput for ten independent set of user locations

3.5.2. Fairness constraint

"Fairness" is imposed during the intra-cell power allocation, because users in the same cell compete for transmit power subject to the total power budget constraint. As we have discussed earlier, the "fair" power allocation consists of two stages: fairness-based power allocation and demand-based power allocation.

In this section, we evaluate the impact of this "fairness" policy on the overall system performance by comparing its result against the algorithm that maximizes the number of "satisfied users"; a satisfied user is defined as a user who receives all requested data. Note this is *not* equivalent to maximizing throughput as in a voice-only system. When a system has multiple data types and thus multiple QoS requirements, maximizing throughput may allow lower priority data ahead of higher priority data, if lower priority data require less resources to transmit. Since some high priority data, such as control, may carry crucial information for decoding lower priority data, we want to maintain a strict priority among data types.

The system incorporating all three control techniques is simulated for both “fair” and “competitive” cases. The results are shown in Figure 3-6. The “competitive” algorithm slightly out-performs the “fair” algorithm for up to 14 users, but then becomes notably inferior to the “fair” system at the tail of the distribution. This is happening because the objective for the “competitive” algorithm is to maximize the number of active users; therefore, it tends to favor users who have better channels and thus allocates all resources to these users. From this result, we believe fairness benefits the overall system, especially to users who have poor channels. Fairness is implemented as a part of the power control unless it is specified otherwise. As a final remark, the users distribution used in this simulation is the one that yields the median throughput. This distribution will be used for the remainder of the results.

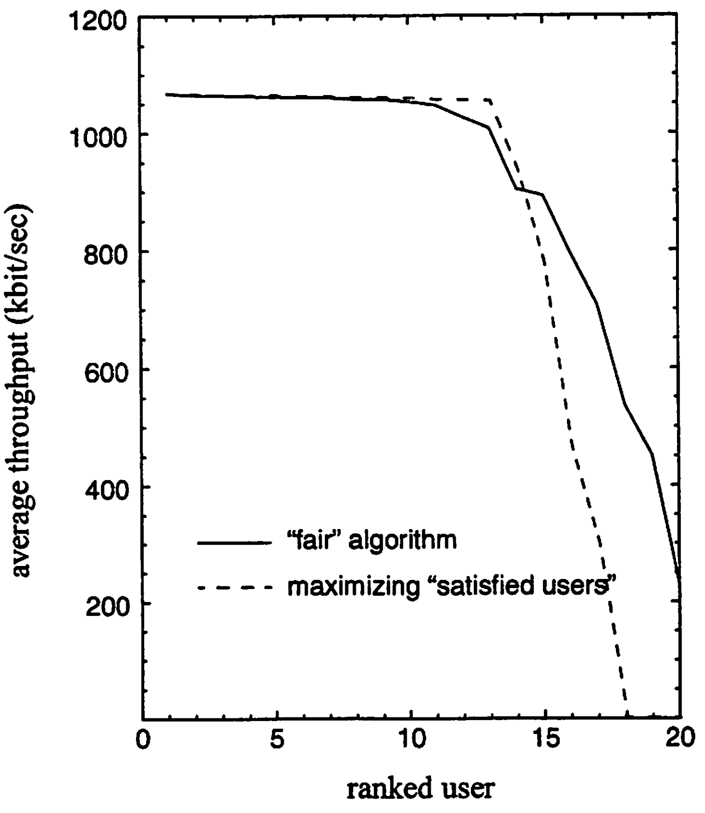


Figure 3-6. Performance comparison for systems with and without fairness constraint.

3.5.3. Dependencies on traffic profile

This system is designed to support traffic sources having diverse statistics. In order to design a robust system that supports data types for a wide range of arrival statistics, we test the system through several traffic models. In this study, the traffic sources are: (1) Poisson arrival (same as the one used in the previous section), (2) two-state continuous time Markov chain (CTMC) for modeling highly bursty traffic, (3) constant bit rate, and (4) a combination of two-state CTMC and constant bit rate. These traffic sources are chosen to be diverse, spanning from very bursty to constant rate transmission. The only characteristics they have in common are the mean arrival rate and the packet size for each data type. The parameters for these four models are listed in Table 3-2.

	type 1 (control)	**type 2 (text/graphics)**	**type 3 (audio)**	**type 4 (video)**
packet size (bit)	100	200	100	100
avg data rate (kbit/sec)	12.5	200	50	800
BER requirement	10^{-9}	10^{-4}	10^{-5}	5×10^{-4}
Poisson Arrival	refer Table 3-1			
2-state continuous time Markov chain				
p(on→off)	0.04	0.1	0.02	0.2
p(off→on)	0.01	0.2	0.01	0.8
constant-bit-rate/CTMC				
	2-state CTMC	2-state CTMC	constant bit rate	constant bit rate

Table 3-2. Parameters for various traffic models.

We choose the set of users from Figure 3-5 which yields the median throughput; this set of users will again be used for the rest of the chapter unless specified otherwise. Again, we assume a perfect channel estimation. Figure 3-7 presents user throughput for these four scenarios, where no major performance variation was present. We therefore conclude that mean data rate and packet size are good parameters for characterizing the overall received data.

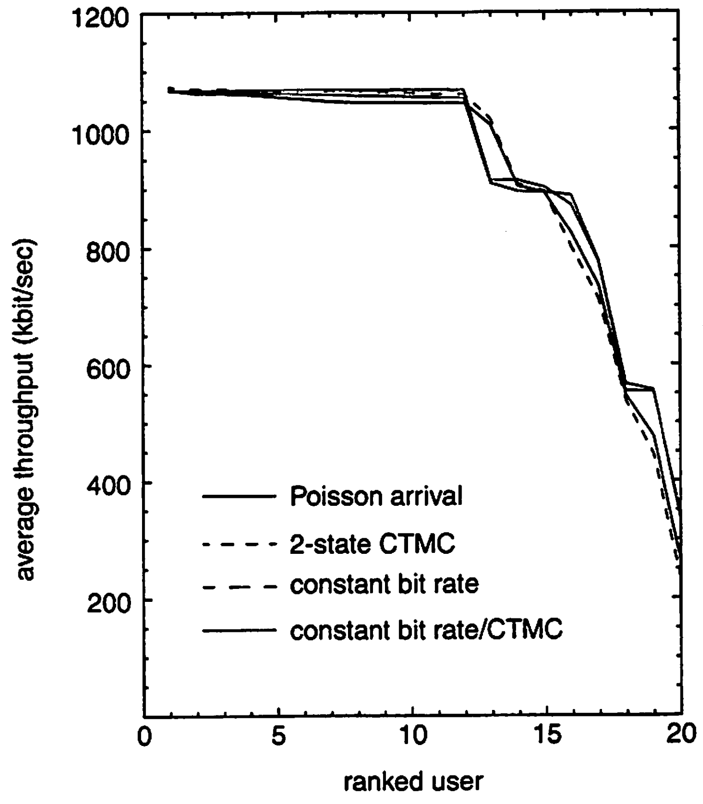


Figure 3-7. System performance with four traffic models.

To study the system dynamics in detail, we then look into the trace data at receiver for each data type. This is done by computing the moving average on arrival data throughput. The moving average window size is set to be 500 msec, and the consecutive windows are skewed by 100 msec. These parameters are chosen because we felt they are sufficient for revealing the dynamic details of the system, and at the same time, offer a good approximation for the packet level activity under various traffic model.

Since it is impractical to present the trace data of all data types for all system users, only three users (with four data types each) are chosen for each traffic model to demonstrate the behavioral characteristics. These users are the ones who achieve the highest, the median, and the lowest throughput. The moving average analysis performed on four data types for the highest throughput user, the median throughput user, and the lowest throughput user are shown in Figure 3-8, Figure 3-9, and Figure 3-10 respectively. Despite the little difference we saw in terms of the overall user throughput under various traffic, the trace data behaved vastly different as we moved from one traffic model to another, with a near constant throughput in the constant-bit-rate case and a highly varying throughput when the traffic is modeled by two-state CTMC.

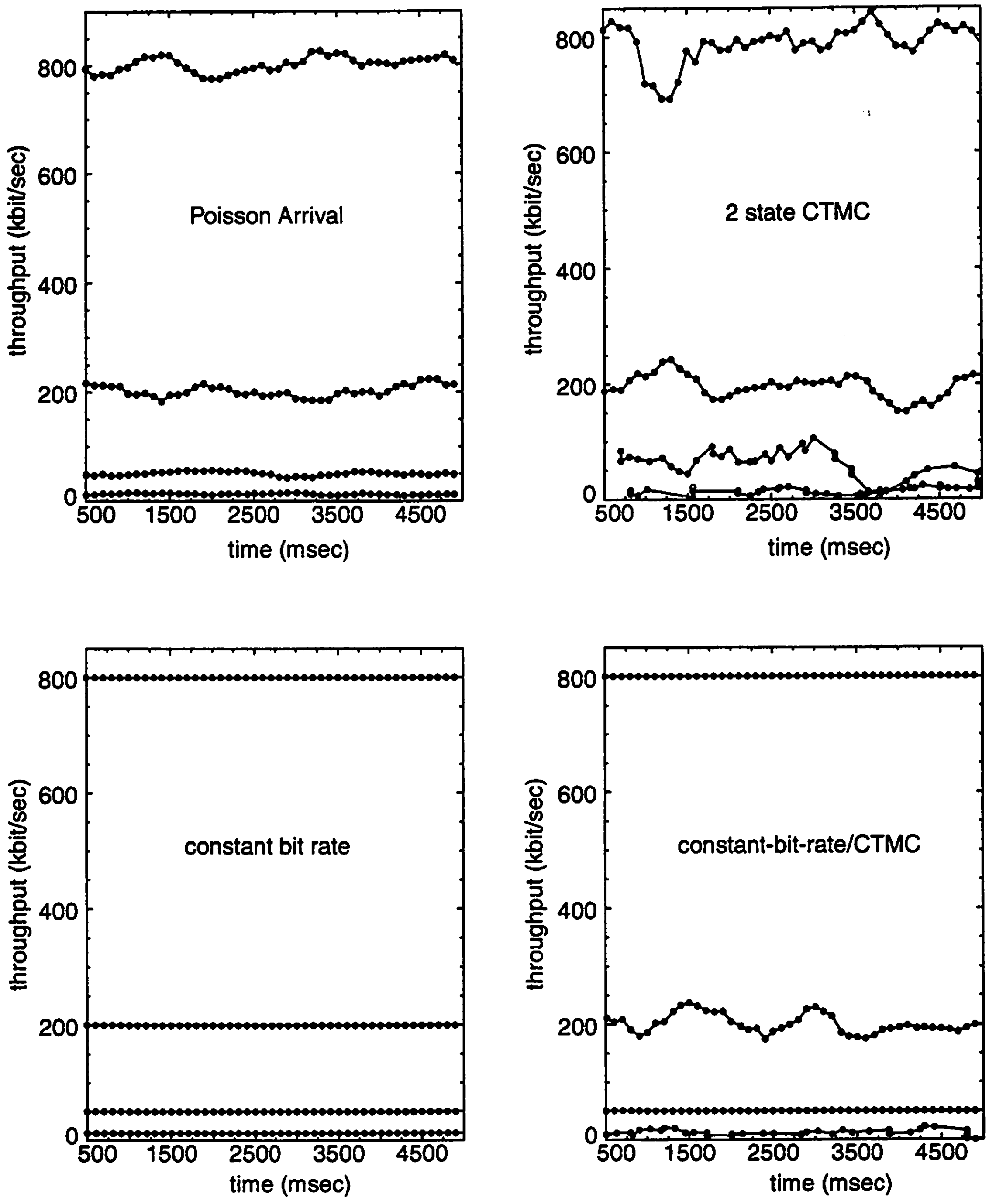


Figure 3-8. Moving average analysis for the highest ranked user under four traffic models.

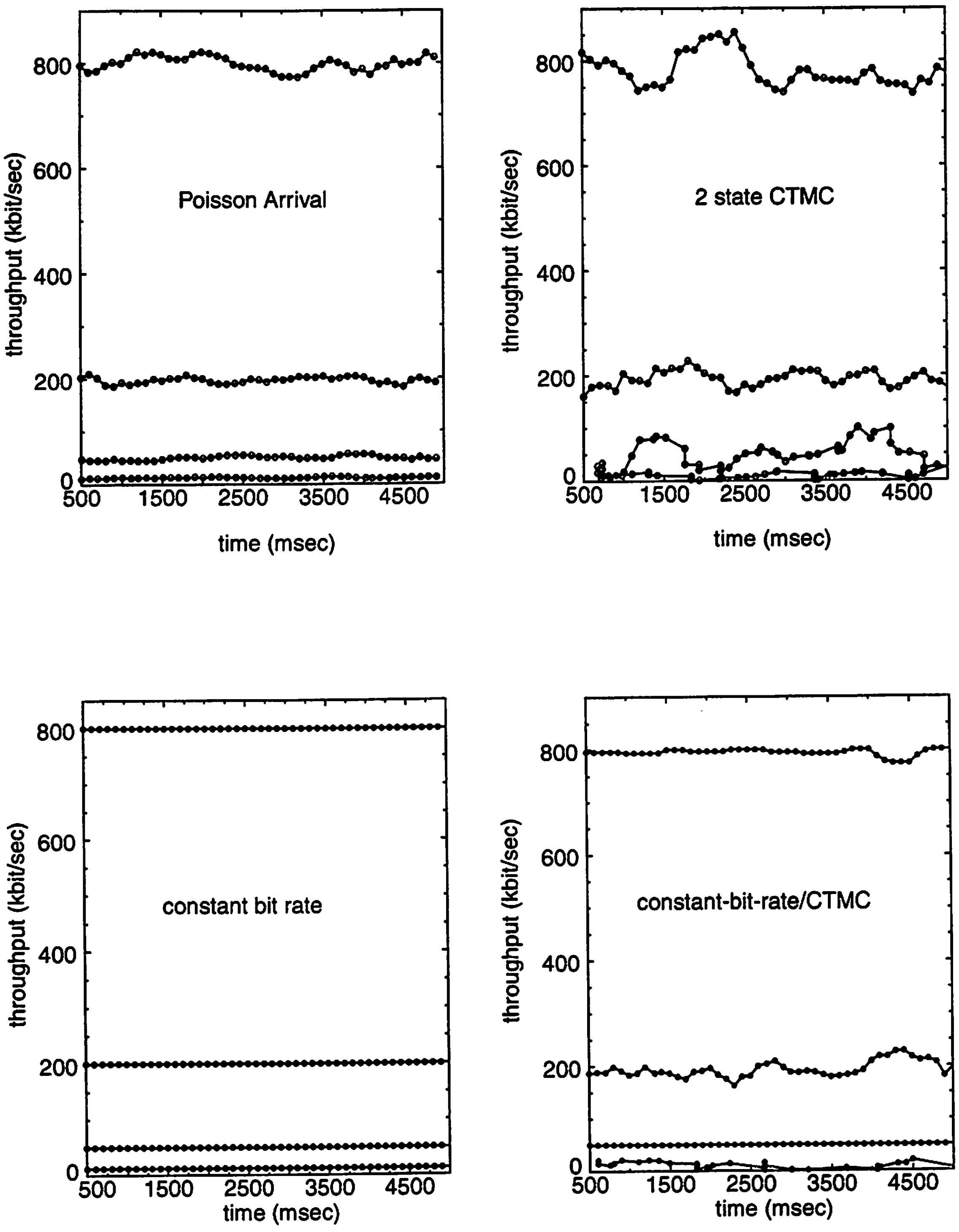


Figure 3-9. Moving average analysis for the median ranked user under four traffic models

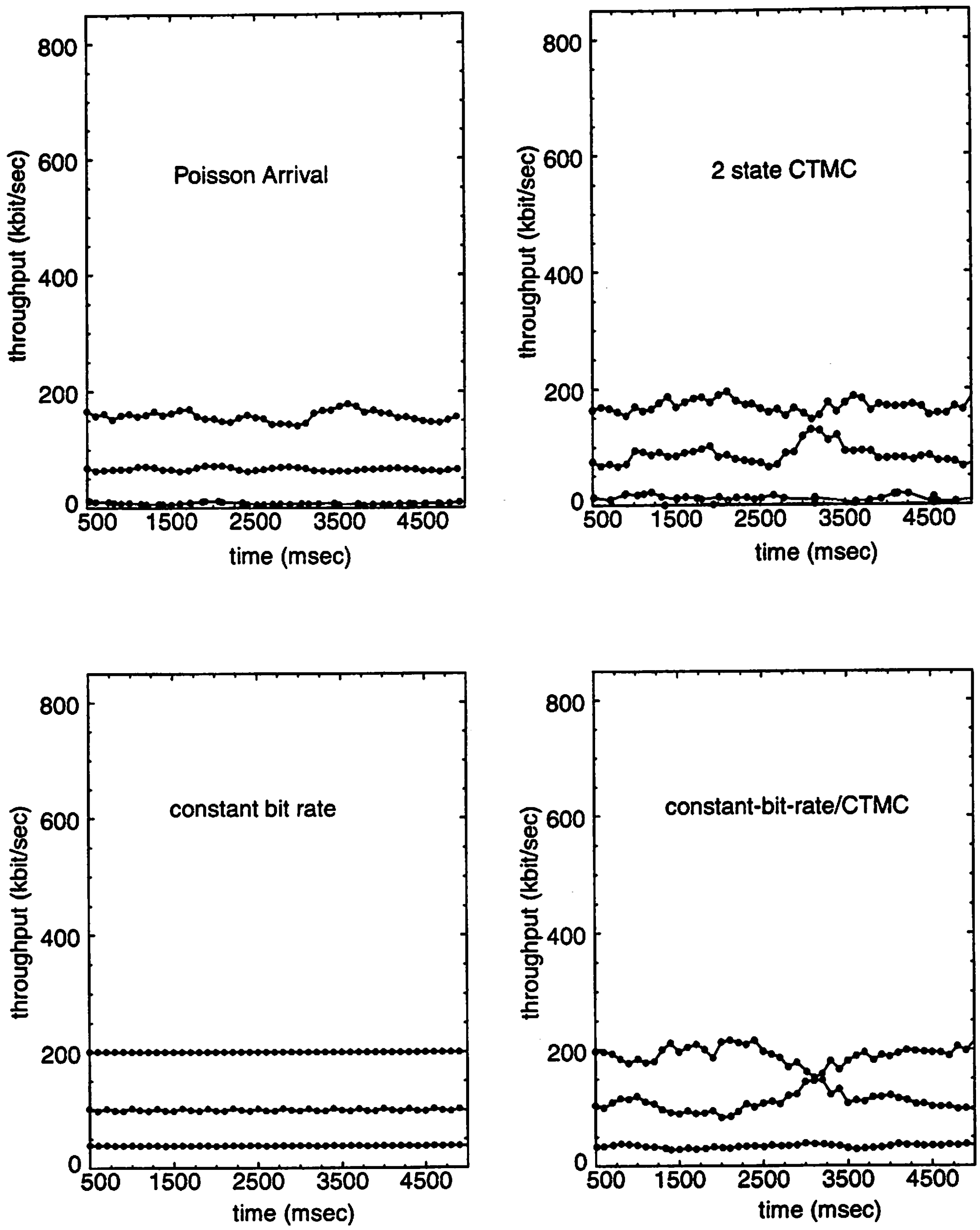


Figure 3-10. Moving average analysis for the lowest ranked user under four traffic models

3.5.4. Power control or VFEC?

The distributed algorithm we have described involves a number of control variables and thus has very complicated co-dependencies. To determine the necessity of this complexity we investigate if a simplified algorithm can offer comparable performance. For this reason, only two out of three techniques are applied: power control combined with scheduling, and VFEC combined with scheduling. In the case when power control is omitted from the control algorithm, we let each basestation transmit at its maximum allowable power budget while this budget is equally divided among cell users. Scheduling is always included since it is necessary to prioritize the data types.

From the traffic model, we know the average total traffic demand for each user is greater than half of the available link bandwidth. As a consequence, we expect power control to play a significant role in the process of optimum resource allocation. The simulation results are compared against the original algorithm which consists all three techniques. The results however, as shown in Figure 3-11 indicate power control/scheduling, with or without the fairness constraint, is inferior to VFEC/scheduling. There are two possible reasons to explain why power control adopted by a multimedia system is not as critical as it is in a single-medium case. First, the multimedia traffic we studied are highly diverse in bandwidth and error rate requirements. The power control algorithm, however, adjusts transmit power at the packet level, with each packet usually consists of multiple segments of different data types. Consequently, a packet is transmitted at the power level which fulfills the most stringent BER requirement among all data types within a packet. As a result, not only power is wasted on more error tolerant data, this allocation also introduces unnecessary interference to the rest of the system. For a single-medium system, there is no such problem because the data have only one BER requirement, so there is always the transmit power level that is just right for meeting this requirement. Second, the FEC coding rate is normally fixed in a single-medium system (because the data rate and the channel bandwidth are determined in advance); this constraint leaves power control to be the only parameter available for adjusting the system performance, which consequently putting it at a dominate position.

VFEC is different from the single forward error correction coding as it adepts to the time-varying traffic at packet level by encoding each data segment with the most suitable FEC; as a result, it is able to take full advantage of the total available bandwidth. Unless we adopt a fast power control algorithm (at the cost of increasing overall algorithm complexity and it may not converge) which updates transmit power at the packet level, power control does not enjoy as much flexibility as VFEC, and thus is inferior. In conclusion, the results here indicate the importance of providing direct error correction.

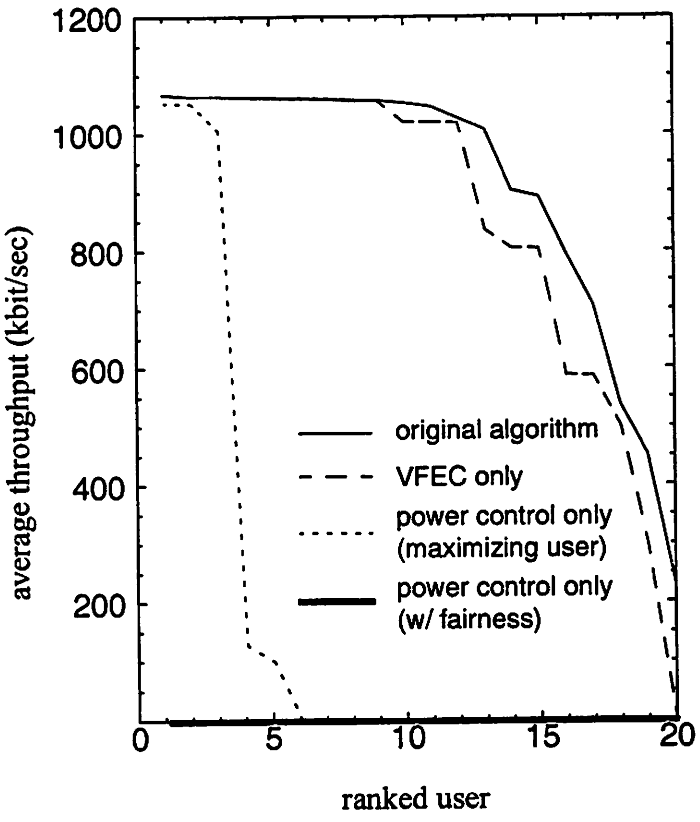


Figure 3-11. Performance comparison when applying only two out of three techniques.

3.5.5. VFEC statistics

Now we have seen the necessity of applying VFEC for our proposed multimedia system. Let us now investigate the statistics of the code rate among the FEC codes that have been applied as a part of the overall optimization algorithm. The percentage of the code usage is shown in Figure 3-12 for two independent sets of user distribution. It is clear that

(1) most of the data transmitted over the wireless link are FEC coded; and (2) both extremely low rate and high rate codes are rarely used. The most frequently used codes are codes having medium rate, with rate around 1/2.

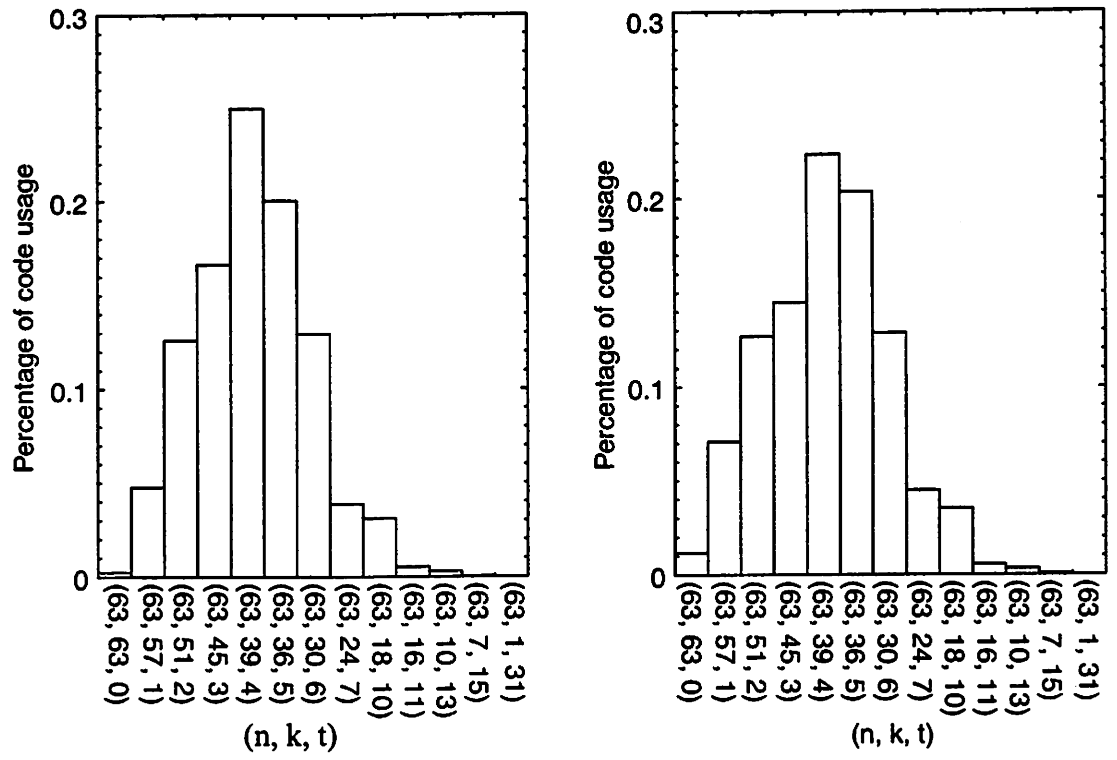


Figure 3-12. Statistics on FEC code usage.

3.5.6. Power control with fixed FEC

The question then arises as to the importance of including the complexity of variable rate error correction, since existing cellular standards, such as IS-95, implement power control with a fixed-rate convolutional code for both uplink and downlink communications. We will apply this to our multimedia system using block codes.

To conduct fair comparisons with the algorithm that incorporates VFEC, five 63-bit BCH codes are chosen from the same family of codes used for VFEC. These codes range from double-error-correction code to eleven-error-correction code; the parameters are listed in Table 3-3. Recall that the total link bandwidth for each user is 2000 kbit/sec, the net data bandwidth after applying FEC is then $\left(\frac{2000}{\text{FEC expansion factor}}\right)$kbit/sec.

code type (n,k,t)	bandwidth expansion factor	data bandwidth (kbit/sec)	index used in the Figure 3-13
(63, 51, 2)	1.24	1610	①
(63, 39, 4)	1.62	1238	②
(63, 36, 5)	1.75	1143	③
(63, 30, 6)	2.1	952	④
(63, 16, 11)	3.9	507	⑤

Table 3-3. FEC parameter.

The simulation results are shown in Figure 3-13, from which we observe a significant overall improvement in throughput as we increase the error correction capability from 2-error bits (shown by ①) to 4-error bits (shown by ②), and finally to 5-error bits (shown by ③) where the throughput reaches the maximum. As we move further in error correction capability, the system begins to shift from interference limited to bandwidth limited. This is clearly illustrated by ⑤ when the extremely powerful 11-bit error correction code is applied. In this case, users are immune to interference and noise so that the throughput for all users is nearly constant. However, the cost of this high reliability is the low transmission rate, which is near 500 kbit/sec for each user. The original algorithm that has power control/VFEC/scheduling is also shown in the figure for comparison purposes; this is indicated by "original". The algorithm that has just VFEC, is indicated "VFEC only". Clearly, when power control is combined properly with an FEC code, there can be a significant improvement over the power-control only system.

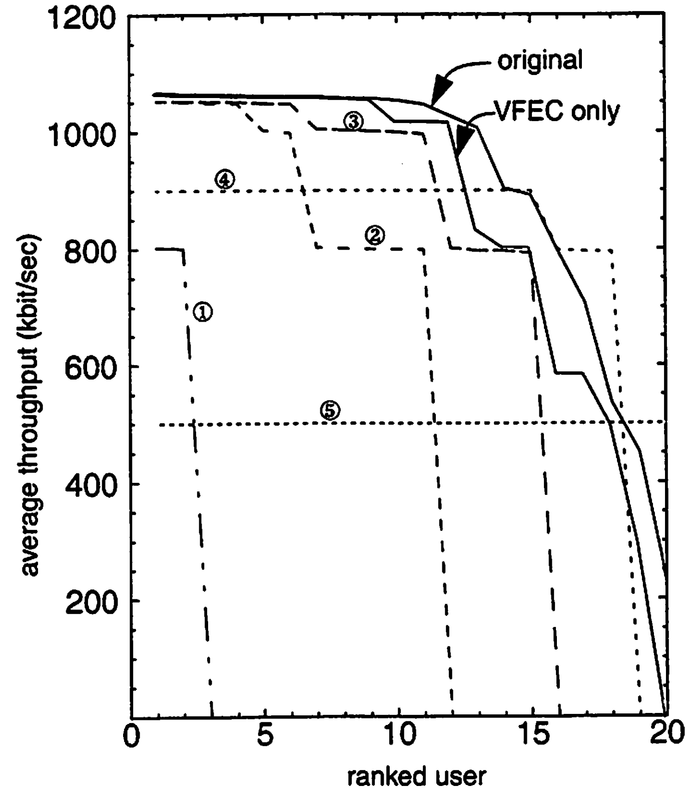


Figure 3-13. Power control combined with fixed FEC (see Table 3-3 for description of fixed error correction used in each curve) with comparisons to full algorithm and VFEC with out power control.

3.5.7. Handoff strategies

The system has now been investigated as whole where users were not distinguished from one another. In other words, we have not looked closely at the parameters that affect individual user's performance, since aggregate throughput was used as the metric. In this section, we explore how a given user's throughput can be influenced by location, cell population and interference. Moreover, we want to know if a technique such as handoff is effective for alleviating users from excessive interference.

Let us go back to the original system where power control, VFEC and scheduling are all applied; the overall system performance is indicated by "original" in Figure 3-14. The users in the shaded region are clearly the ones with the poorest performance. It is not surprising that all these users reside in the most crowded cell, i.e. cell #1. However, the throughput for users in the second most crowded cell, cell #2 (which is also the cell experiencing the most interference) are marked by the circles. As we can see, cell #1 users suffer a significant quality degradation even it has just one more user than the next crowded cell.

One strategy to relieve the hot spot is to initiate handoff. The handoff here is slightly different from the conventional one, where it is only initiated when a user is at the boundary of two adjacent cells. In our situation, handoff is used as a method for alleviating radio network congestion. This goal is achieved by letting a user in the most congested cell, cell #1 in our case, to communicate directly with its least crowded neighbor cell, cell #0. The user is chosen to be the one who is closest to cell #0, who is also the one furthest from cell #1.

The result of this handoff is compared with the case where the same user from cell #1 is removed and then randomly placed into cell #0. Physically relocating a user would definitely yield better performance than merely reassigning basestation. However, the result shown in Figure 3-14 indicates very small difference. Notice both techniques yield a significant performance gain (that can be as large as 350kbit/sec for some users) over the "original" algorithm where a user only communicates with the closest basestation.

Just as a reference, we also show the situation where the same user from cell #1 is removed from the system, which imply cutting off this user's service in a real system.

Obviously having one less user would relieve the system load, however, the gain is fairly small comparing to what can be achieved with other strategy such as handoff.

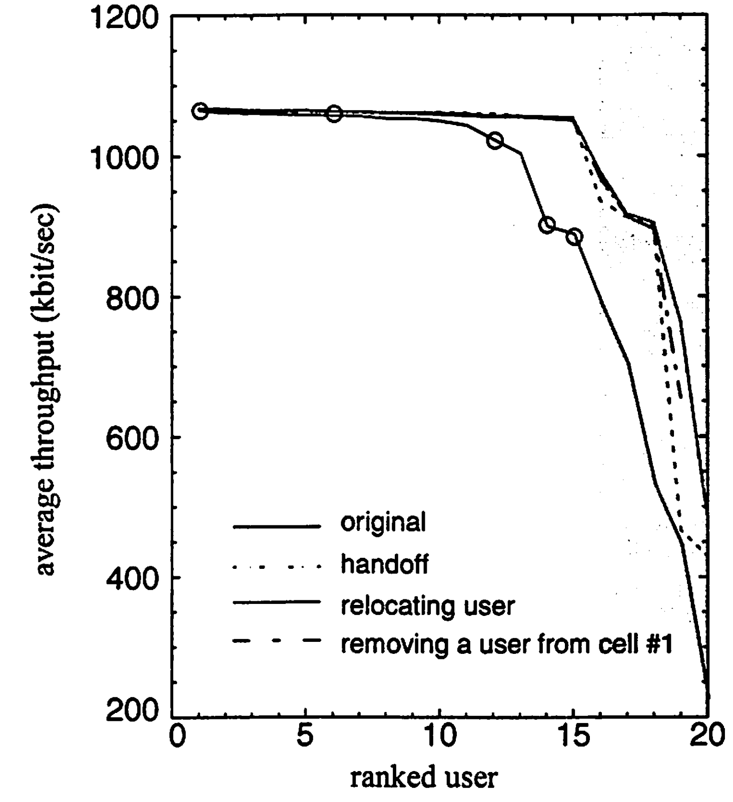


Figure 3-14. Effectiveness of two handoff schemes.

3.5.8. Study of packet size

So far we have only focused on the algorithmic aspect of the system design, such as comparing the technique of power control verses VFEC and studying the effectiveness of two handoff schemes. Those results will provide useful guidelines when we design the overall system architecture. However, at the time of actual implementation, we need to know more detailed system specification, such as the packet size for each data type, or the ideal family of BCH codes used for VFEC.

One difficulty arises when studying the optimal packet size is that the optimum packet size for one set of users may not be suitable for another, since users have a wide range of channel qualities. For those who have good channels, they probably would prefer longer packets as to reduce the percentage bandwidth spent on the overhead. On the other hand, people with poor channels may want shorter packets since shorter packets are less likely to be corrupted by errors. Therefore, instead of searching for the optimum packet size, we can only come up with a range in which packet sizes are optimal in one way or another.

For a given packet size, we divide users into three groups, where each group consists of users who achieved the highest, the median, and the lowest throughput, respectively. Our goal is to find the most suitable packet size for each one of these three groups. In this study, the header is assumed to be 16 bit long. Since majority of traffic is made of type #4 data which has the data rate 800kbit/sec, we therefore only vary its packet size (while keeping the data rate constant) and use it as an approximation for the entire wireless traffic. The packet sizes are set to be 50 bits, 100 bits, 200 bits, 400 bits and 800 bits.

User throughput of each group is plotted as a function of the packet size, which is shown in Figure 3-15. The peaks of each curve correspond to the optimal packet sizes for each one of these three user groups. To conclude, the optimal packet sizes for our system are in the range of 50 to 400 bits.

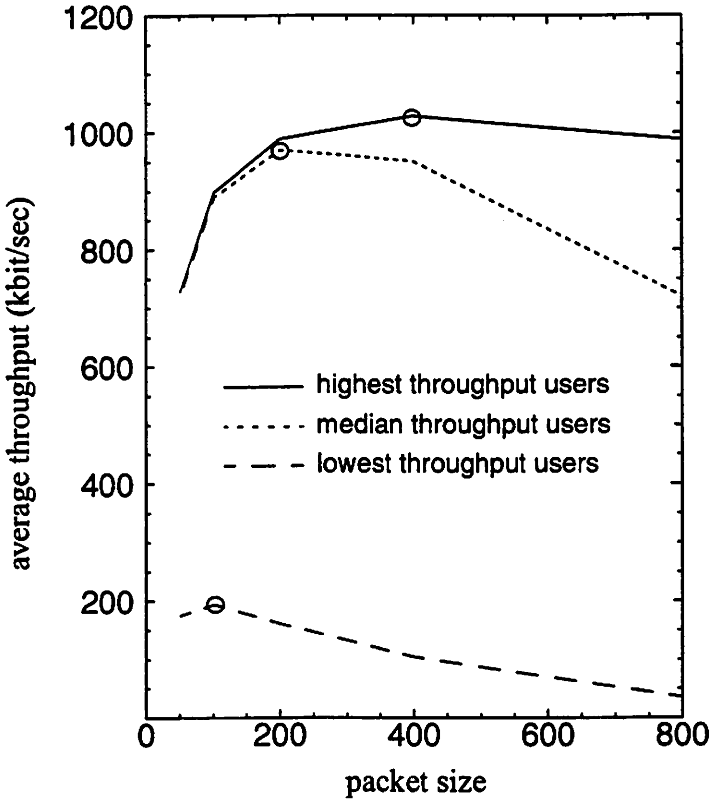


Figure 3-15. User throughput as a function of packet size.

3.5.9. Study of FEC family

The decision of using a single family of BCH code (i.e. codes that share the same block length) for VFEC purposes comes purely from implementation practicality. Comparing to convolutional code or punctured code, block codes offer a much wider range of error correction capabilities at reasonable implementation cost. For codes having the same block length, many building blocks such as syndrome calculation and polynomial manipulation can be shared because they all operate on the same finite field. In this section, we approach the BCH code from the system level by deciding the most suitable family of BCH codes for VFEC purpose.

From a purely performance standpoint, longer codes have better code efficiencies than shorter ones for the same percentage of bandwidth redundancy. In addition, longer codes offer greater selections for obtaining the right code to adapt to the time-varying channel and traffic. However, as the block length increases, more decoders have to be implemented; furthermore, as a code corrects more bit errors, the implementation complexity increases as well. Just as an example, one step in the decoding procedure (proposed by W. Peterson) involves computing the inverse of a $t \times t$ matrix, where t is the number of correctable errors; the complexity for this step is in the order of t^3.

From these considerations, we study four families of BCH codes with block lengths equal to 15, 31, 63, 127 bits; these are all primitive codes. The code redundancy and error correction capability for each are listed in Table 3-4. The algorithms that use power control, VFEC and scheduling is simulated for each one of these four families of codes. The results are shown in Figure 3-16.

block length	**# of information bits**	**# of correctable bit errors**
15	15, 11, 7 5, 1	0, 1, 2, 3, 7
31	31, 26, 21, 15, 11, 6, 1	0, 1, 2, 3, 5, 7, 15
63	63, 57, 51, 45, 39, 36, 30, 24, 18, 16, 10, 7, 1	0, 1, 2, 3, 4, 5, 6, 7, 10, 11, 13, 15, 31
127	127, 120, 113, 106, 99, 92, 85, 78, 71, 64, 57, 50, 43, 36, 29, 22, 15, 8, 1	0, 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 13, 14, 15, 21, 23, 27, 31, 63

Table 3-4. Error correction capabilities and redundancies for four families of BCH code.

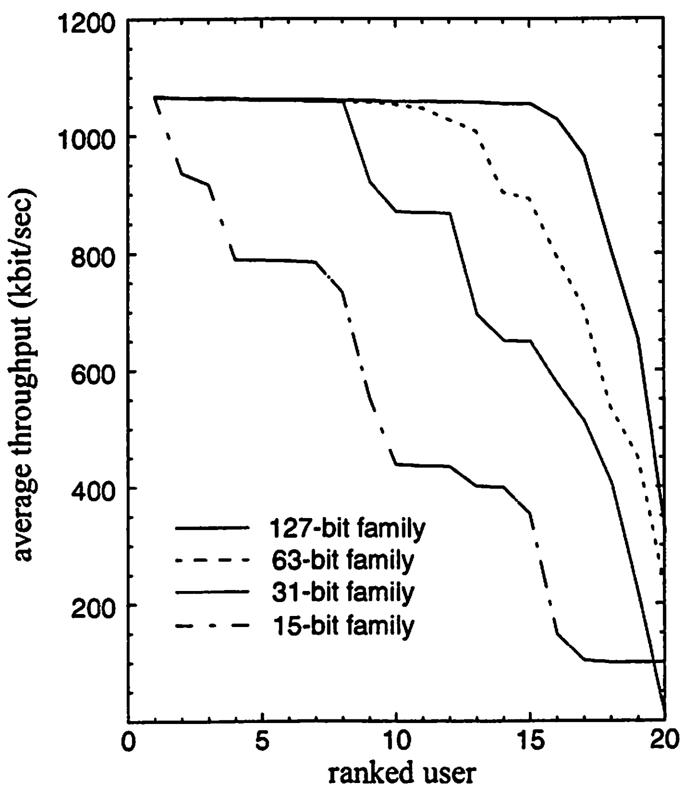


Figure 3-16. System performance with respect to four families of BCH codes.

It is expected that there will be a performance gap between the 127-bit family and the 15-bit family, since in the later case only three non-trivial error correction codes are available as oppose to seventeen in the 127-bit family. However, the net performance gain only decreases by a factor of two as we move up to larger codes, implying extreme fine control of bandwidth redundancy may not be necessary, especially when the practical implementation constraints are considered.

3.6. Summary

This chapter is devoted to simulation based evaluation of the system performance with a simple cell structure and uniformly distributed users. In the simulations, we focus on a special case where each data type has a BER requirement: data are only transmitted when the expected received BER meet this requirement. This constraint is adopted by many existing wireless systems, expressed in terms of received SNR. Notice this is equivalent to having a step utility function with respect to BER.

The results showed the performance for an individual user is highly dependent on the physical location, which agrees with our intuition. However, the system performance showed very little difference (in terms of the aggregate throughput) as we vary the traffic model for each data type from a highly bursty Markov chain to constant-bit-rate traffic, as long as these traffic sources have the same average data rate and packet size.

Later we studied the effectiveness of VFEC verses power control. The technique of VFEC showed to be significantly superior than power control. This is true even when the average traffic demand is heavy, i.e. greater than half of the available link bandwidth. Then we investigated the statistics of the code rate among the FEC codes that have been applied as a part of the overall optimization algorithm. The result indicated that most of the data transmitted over the wireless link are FEC coded; however, both extremely low rate and high rate codes are rarely used. The most frequently used codes are medium rate codes with the rate around 1/2.

When studying the optimal packet sizes, since there is no single packet size that is optimum for every user, we have derived a range of packet sizes that optimum. The ideal packet size for a user depends mostly on her location.

In the study of hand-off, our results showed handoff improves the system significantly over the situation when users communicate with the nearest basestation. Finally, we examined four families of BCH codes (block length = 15, 31, 63, 127) to study the trade-off between implementation complexity and variable error correction capability finding that the shorter codes provide sufficient variability to be effective.

4 Background on Forward Error Correcting Codes

We have demonstrated through simulation results that a multimedia wireless system performs best by employing power control, VFEC and scheduling. In general, power control and scheduling algorithms are implemented in software since they operate at the packet level thus are not very timing critical. The error correction, on the other hand, performs in real time at the transmission data rate, which can be as high as 2Mbit/sec such as the case of the Infopad. To achieve this high performance requirement, FEC decoding is often implemented using custom IC or programmable logic device.

The rest of this thesis is devoted to a design architecture that is suitable for VFEC decoder. This is done in two parts. In this chapter, we will provide readers with sufficient background on coding theory with focus on BCH code, which is a class of block codes. Both encoding and decoding algorithms for general BCH code will be discussed. Most of the materials discussed in this chapter can be found in standard coding text books; those who are familiar with coding theory are encouraged to skip this chapter. In the proceeding chapter, an architecture which can be mapped efficiently to a VLSI implementation is proposed for a variable forward error correction decoder. The design of this architecture requires a thorough understanding of the iterative decoding algorithm for BCH codes described in this chapter.

4.1. Forward Error Correction

Forward error-control coding is a technique for providing reliable digital data transmission over a noisy communication channel. The errors introduced by the noisy channel are controlled through the insertion of redundancies in the information messages. The concept was first presented by Shannon [41], in which Shannon showed that for a system with transmission rate R (in bits per second) that is less than the channel capacity C (in bits per second), it is possible through the use of error-control codes to reduce the output bit error probability to as small as desired. Shannon theorem did not tell us how to find such codes to achieve the promised arbitrary small probability of error, but it proved that they exist. Throughout 1950 and 1960, much effort was devoted to finding construction of the codes that would achieve the promised error probability performance. In general, two types of codes were found, namely, block codes and tree codes. Most modern digital communication systems use block codes, convolutional codes (which is a subset of tree codes) or concatenated codes.

The goal of this chapter is to provide readers with sufficient background on coding theory, as the proceeding chapter will be completely focused on the design architecture of a variable forward error correction decoder. We will start this chapter by briefly discussing convolutional codes. Since the implementation of the variable forward error correction decoder is based on block codes, the emphasis of this chapter is on the linear block codes, specifically, the cyclic codes.

4.2. Convolutional Code

The convolutional encoder functions as a Markov-type finite state machine. A simple encoder is shown below:

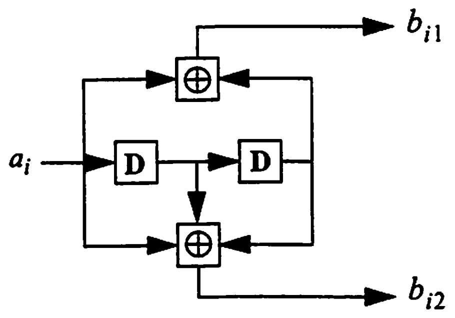


Figure 4-1. Rate-1/2, constraint length 2, convolutional encoder.

Some salient features of the above encoder include: (1) for each input bit, two output bits are created; therefore, it is called a rate-1/2 coder (2) each input bit is stored for two clock cycles; therefore, the constraint length K is said to be two. It is intuitively obvious that a larger K yields better performance because it stores more elements from the past into the registers as redundancy for the current element.

For each possible state and each possible input bit, we need to know which output bits result so maximum-likelihood decoding can be accomplished. Such transitions are easily represented by a trellis diagram and are shown in Figure 4-2 for $K = 2$.

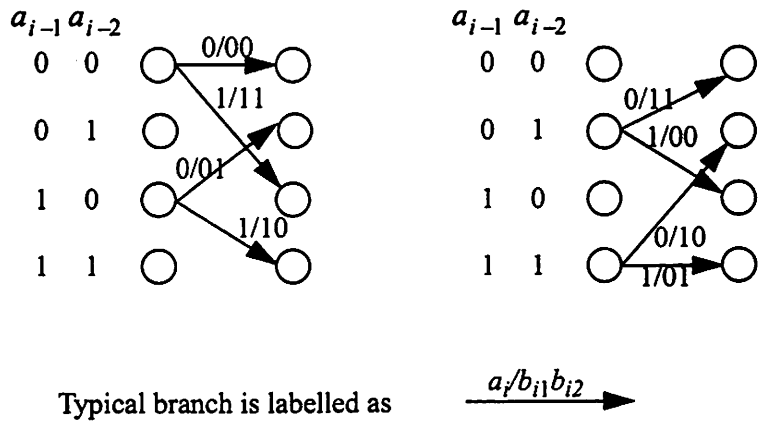


Figure 4-2. Trellis representation of a rate-1/2, K = 2 convolutional encoder.

For the $K = 4$ case, the trellis involves a rather large 16 states which is similar to Figure 4-2. Output bits b_i are trivially obtained via $b = aG$ where G is a generator polynomial. The G matrices are shown below for the two cases considered: $K = 2$ and $K = 4$, both of which are rate 1/2 code. They were chosen because they are optimal for rate-1/2 in the sense of maximal free distance. For $K = 2$, the free distance is 5 and for $K = 4$, the free distance is 7 [27].

$$G = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}^T, K = 2$$

$$G = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \end{bmatrix}^T, K = 4$$

A very popular way to decode convolutional codes is by maximum-likelihood sequence estimation (MLSE) using the Viterbi algorithm (VA). The VA (dynamic programming) is the optimum symbol-by-symbol maximum-likelihood decision method for a bandlimited multipath channel with intersymbol interference. The overall system diagram is shown below in Figure 4-3.

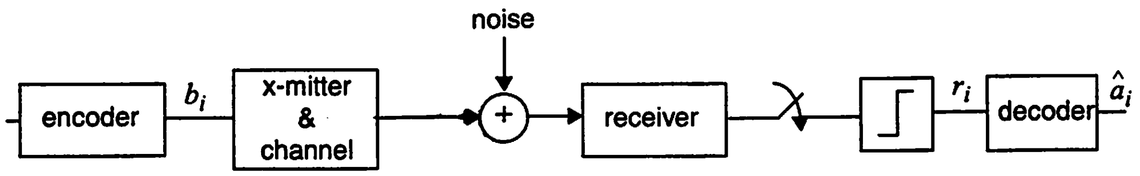


Figure 4-3. Block diagram of communication system showing convolutional encoder and decoder.

For clarity and simplicity, we demonstrate in detail the rate-1/2, $K = 2$ case. The objective of our VA is to recreate the trellis of Figure 4-2. Note that each node has two paths entering and exiting. Figure 4-4 shows the two entering paths for the "00" node at some time n.

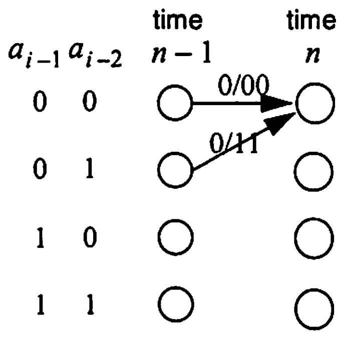


Figure 4-4. Two input paths into node "00" at time n.

At time $n-1$, the accumulated metric up to that time and the corresponding path back for that metric is stored in each of the four nodes. We then form partial path metric for the two branches coming into node "00", given by:

$$\rho_{i,0} = |r_{i1}| + |r_{i2}|$$

$$\rho_{i,1} = |r_{i1} \oplus 1| + |r_{i2} \oplus 1|$$

where r_i are the hard decisions out of the slicer in Figure 4-3. Then, at time n, we consider all incoming paths to a node. For each path, we form the sum of the accumulated metric up to time $n-1$ and the partial path metric. The path with the smallest sum is retained while the others are discarded - the essence of the VA is to keep the amount of information reasonable and still provide maximum-likelihood solutions. After repeating this for M steps, we look at all four nodes and pick the smallest accumulated metric, back-trace M steps and output resulting bits. The final consideration in implementing the VA is how long to wait before making a final decision. We consider the best path coming into each node and choose a truncation depth M, meaning we will look back M time steps to obtain the transmitted bit sent at that time. Ideally for maximum likelihood decoding, we would only start decoding when all the bits are received, this implies M is infinite. But to avoid high latency in practice, M can be set to a few integer multiples of the number of states.

Even for the rate-1/2, $K = 4$ case, which functions analogous to the one described, the process starts to become unwieldy. This is one disadvantage of a full-blown MLSE on a general channel - its complexity is just too large. Methods of mitigating this effect include truncating the overall channel impulse response [13] or a more recent idea of combining equalization and the MLSE [12].

Convolutional codes are widely used because of their ease of implementation and their abilities to utilize soft-decision information. However, convolution codes have difficulty achieving very low output bit error rates even at relatively low input bit error rates. Therefore, for high performance systems requiring low output error probability, block codes are generally preferred.

The rest of this chapter is organized as the following. Section 4.3 gives an general introduction to block codes. Section 4.4 provides concepts on abstract algebra which are necessary for understanding cyclic codes. Section 4.5.2 discusses the encoding rules for general BCH codes. The decoding of BCH codes using matrix and iterative approaches is discussed in Section 4.5.5; these two approaches are also known as Peterson algorithm and Berlekamp algorithm, respectively.

4.3. Block Codes

For any block code, the information stream is grouped into blocks which are k symbols in length. The encoder takes the block of k symbols and maps them into a block of n symbols based on an encoding rule, where $n \geq k$. The block of n symbols is called a codeword, and n is the block length of the code. Some well known block codes are single-error-correcting Hamming code and BCH codes.

A block code is normally characterized by its length n, the number of information symbols k, and its minimum distance d. The ratio k/n is called the *code rate* and is used to indicate the bandwidth expansion or redundancy of the code. An (n, k) code is said to be *systematic* if first k symbols are information symbols and the last n-k symbols are the parity check symbols. Systematic codes are generally implemented in practice because it is easy for decoder to separate information symbols from parity symbols.

For a block code, the minimum distance d determines the random error detection and/or correction capability of a code. It is defined as the smallest of all Hamming distances between any two codewords. A code is guaranteed to correct any pattern of t errors and e erasures if $2t + e < t$. Therefore, when a code is used only for error correction, i.e. $e = 0$, this inequality implies a code can correct all patterns of t errors when $t \leq \left\lfloor \frac{d-1}{2} \right\rfloor$. The error correction capability of a code can also be visualized through sphere packing where each sphere has radius $\left\lfloor \frac{d-1}{2} \right\rfloor$. The codewords are the centers of the spheres, so the space occupied by each sphere characterizes the number of tolerable errors.

The bit error rate (BER) is a good indicator of the channel quality when a channel has uniform independent random errors. Let p_e denote the probability of receiving a bit with error, then mean bit error rate (BER) is p_e. For a binary (n, k) t-error-correcting code, the decoder fails when there are t+1 or more bit errors in the received block of n bits. If we assume a decoder introduces no extra errors in the case of un-correctable errors (by leaving the bits as they are received), the BER at the decoder output can be calculated as:

$$\mathrm{BER} = \frac{1}{n}\sum_{i=t+1}^{n} i\binom{n}{i}p_e^{\,i}(1-p_e)^{n-i} \quad (4.1)$$

Figure below illustrate the error correction capabilities of some typical linear block codes.

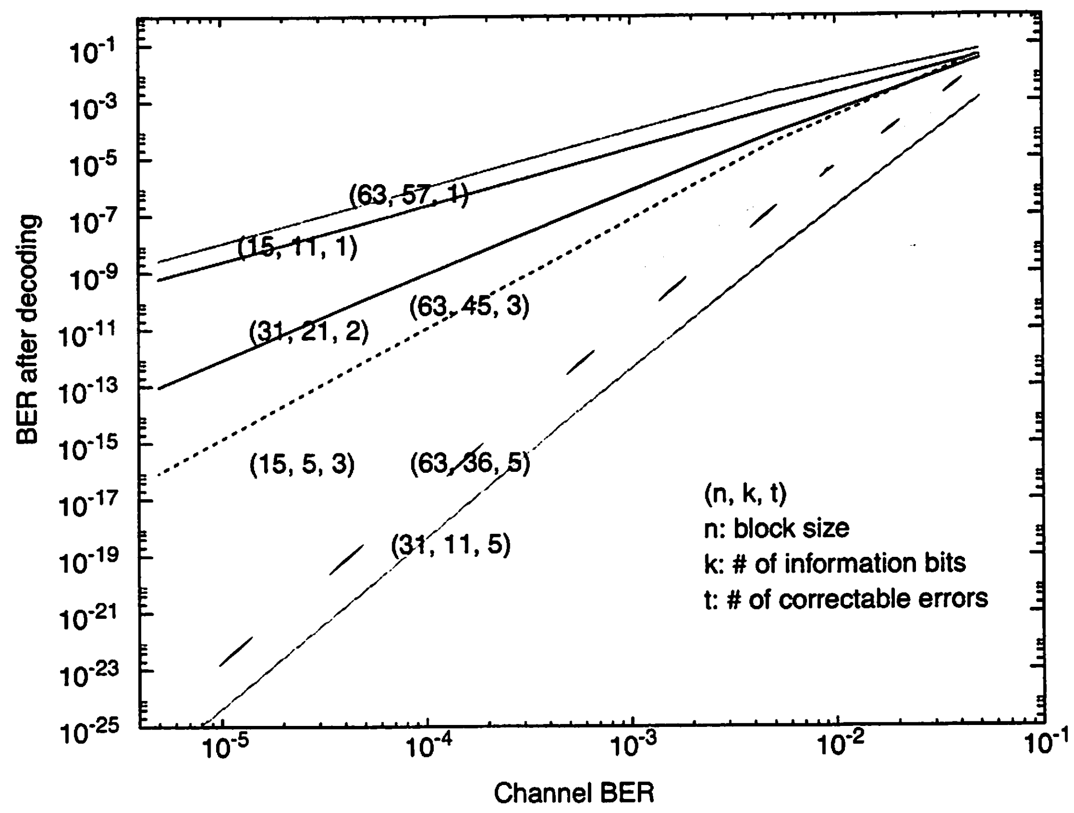


Figure 4-5. Error correcting capabilities of some BCH codes.

The rest of the chapter will review various approaches to the coding theory. We will start with matrix description of block codes and then move on to the cyclic codes and their decoding algorithms. The emphasis will be on the decoding of BCH codes, in which case two algorithms will be presented: one by Peterson and one by Berlekamp.

4.3.1. Matrix description of a block code

4.3.1.1. Hamming code

The Hamming code is the simplest non-trivial linear block code which corrects all single bit errors. To illustrate the concept, suppose we have a binary code that has code length n=6, and k=3. A code word is represented by $[c_0, c_1, c_2, \dots, c_5]$, where $[c_0, c_1, c_2]$ are the information bits. Note this code is systematic. Let the encoding rule be:

$$
\begin{aligned}
c_3 &= c_0 + c_1 \\
c_4 &= c_1 + c_2 \\
c_5 &= c_0 + c_2
\end{aligned}
\tag{4.2}
$$

For example, if the information bits are $\vec{v} = [1, 0, 1]$, the above encoding rule would yield the codeword $\vec{c} = [1, 0, 1, c_3, c_4, c_5] = [1, 0, 1, 1, 1, 0]$.

This encoding rule can be described more concisely via a generator matrix G, where G is a $k \times n$ matrix, and the codeword is the product of information vector and the generator matrix, i.e. $\vec{c} = \vec{v} \cdot G$. For the above example, the generator matrix of a (6, 3) code is a 3×6 matrix, where $G = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$. G is of the form $[I|P]$, where I is a $k \times k$ identity matrix and P is a $k \times (n-k)$ matrix which computes the parity check bits by the above encoding rule.

Notice the encoding rule expressed a system of linear equations in (4.2) can also be written as:

$$
\begin{aligned}
c_0 + c_1 + c_3 &= 0 \\
c_1 + c_2 + c_4 &= 0 \\
c_0 + c_2 + c_5 &= 0
\end{aligned}
$$

This in fact describes the null space of G, and it can be characterized by a $k \times n$ matrix called parity check matrix, H. In this example, $H = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$, with the first row corresponds to $c_0 + c_1 + c_3 = 0$, the second row corresponds to $c_1 + c_2 + c_4 = 0$, and the third row corresponds to $c_0 + c_2 + c_5 = 0$. Therefore, $\vec{c}$ is a codeword if and only if:

$$H \cdot \vec{c}^T = 0. \tag{4.3}$$

To summarize, generator matrix G and parity check matrix H are orthogonal to each other, i.e. $G \cdot H^T = 0$; they have the properties that rank(G)=k and rank(H)=n-k. Furthermore, a linear block code be characterized either by its generator matrix or parity check matrix.

For the rest of this section, we will describe the decoding procedure using the parity check matrix H. The following notations will be used in our discussion.

$\vec{c}$ = transmitted codeword

$\vec{e}$ = error vector, where $e_i = \begin{cases} 0 & \text{if } i\text{th bit is received correctly} \\ 1 & \text{otherwise} \end{cases}$

$\vec{r}$ = received vector = $\vec{c} + \vec{e}$

To decode, the received vector is first multiplied with the parity check matrix H. Since $H \cdot \vec{c}^T = 0$, the product (of parity check matrix and the received vector) can be reduced according to $H \cdot \vec{r}^T = H \cdot (\vec{c} + \vec{e})^T = H \cdot \vec{c}^T + H \cdot \vec{e}^T = H \cdot \vec{e}^T$, where the result only depends on the error position(s). This decoding algorithm is therefore called the syndrome decoding; the syndrome of a received vector $\vec{r}$ is defined as $Syn(\vec{r}) = H \cdot \vec{r}^T$.

In the previous example, the information bits $\vec{v} = [1, 0, 1]$ yields a codeword $\vec{c} = [1, 0, 1, 1, 1, 0]$. Suppose the 4th bit is received with an error so the received vector $\vec{r} = [1, 0, 1, 0, 1, 0]$. Applying the syndrome decoding, the product is $H \cdot \vec{r}^T = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$, which is the 4th column of H. Because $H \cdot \vec{r}^T = H \cdot \vec{e}^T$, we would observe the same syndrome pattern if the 4th component of the error vector is non-zero, i.e. $\vec{e} = [0, 0, 0, 1, 0, 0]$. Therefore, we know the 4th received bit is corrupted.

However, if there are two bit errors, say at 3rd and 4th position, then the received vector is $\vec{r} = [1, 1, 1, 1, 1, 1]$. The product $H \cdot \vec{r}^T = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$ matches no column of H, which indicate decoding failure and multiple errors have occurred during the transmission.

4.3.1.2. A double error-correcting block code

The (6, 3) Hamming code we have just described can correct a single bit error out of 6 transmitted bits. Some more elaborate codes such as BCH(15, 7) code can correct up to 2 bit errors from any position within a block of 15 bits. Note this double error-correcting code is different from concatenating two Hamming codes; for Hamming codes to correct two errors, each error has to reside precisely within each block of 6 bits. The difficulty in correcting multiple consecutive errors is the reason why it took so long to develop a multiple-error-correction algorithm after the discovery of a single error correction algorithm.

The generator matrix for BCH(15, 7) code is a 7×15 matrix, thus the parity check matrix has dimension 8×15. To decode BCH(15, 7) codes, we apply the syndrome decoding algorithm. Let us start with a parity check matrix of 8 rows with the following form:

$$H = \begin{bmatrix} 1 & 2 & \cdot & \cdot & 15 \\ f(1) & f(2) & \cdot & \cdot & f(15) \end{bmatrix}$$

where the numbers $1, 2, \ldots, 15$ are 4 bits in binary representation, and f is a function which has yet to be determined. If $\vec{r}$ is the received vector with 2 errors occurring in the ith and

jth columns, the corresponding error vector $\vec{e}$ would then have two non-zero components, $\vec{e} = \begin{bmatrix}0 & \dots & 1 & 0 & \dots & 1 & \dots & 0\end{bmatrix}$, with 1's at ith and jth positions. The syndrome of $\vec{r}$ is

$$Syn(\vec{r}) = H \cdot \vec{r} = H \cdot (\vec{c} + \vec{e}) = H \cdot \vec{e}$$

$$= \begin{bmatrix} i+j \\ f(i)+f(j) \end{bmatrix} = \begin{bmatrix} y_1 \\ y_3 \end{bmatrix}$$

which corresponds to two equations with two unknowns.

The goal here is to determine the function f so there are enough information to solve for i and j. We know f can not be the identity function, since this will yield $y_1 = y_3 = i+j$, which will not give a unique solution for i and j. Next we try $f(i) = i^2$. Since the space of possible codes is defined over a binary field, which implies $i^2 + j^2 = i^2 + 2ij + j^2 = (i+j)^2$. Again, we have the same problem as we had with the identity function. Let us try $f(i) = (i)^3$, the syndrome of $\vec{r}$ gives

$$Syn(\vec{r}) = \begin{bmatrix} y_1 \\ y_3 \end{bmatrix} = \begin{bmatrix} i+j \\ i^3+j^3 \end{bmatrix}$$

and $y_3 = i^3 + j^3 = (i+j)(i^2+ij+j^2)$ so that $y_3/y_1 = i^2+ij+j^2 = y_1^2+ij$. Noticing that $i+j = y_1$, and $ij = -y_1^2 + y_3/y_1 = y_1^2 + y_3/y_1$, we see that i and j are roots of the equation $(x+i)(x+j) = x^2 + (i+j)x + ij = x^2 + y_1 x + (y_3/y_1 + y_1^2)$. We know the coefficients of this quadratic equation, so we can solve for its roots. Once again, the solution for i and j will correspond to the bit error positions.

The size of generator matrix and parity check matrix is always proportional to the block length n. This makes the overall matrix approach very inefficient for implementation when long codes are used, because both encoding and decoding algorithms require matrix multiplication. A subclass of block codes called cyclic codes are commonly used in practice because encoding and decoding procedures are described in polynomials, and these can be implemented efficiently using linear shift registers. The cyclic codes are widely used with applications ranging from error detection (CRC) to deep space communications. Before we can describing cyclic codes, we will review some basic concepts of finite field algebra [4].

4.4. A quick introduction to abstract algebra

To define a field, we adopt a rather informal definition by Berlekamp [3]:

> A *field* is a set of elements, including 0 and 1. Any pair may be added or multiplied (denoted by + and *, respectively) and the result is a unique element in the same field. The addition and multiplication are associative and commutative, and multiplication distributes over addition in the usual way: x*(y+z) = x*y + x*z. Every field element a has a unique inverse, $-a$, that is also a field element such that $a + (-a) = 0$. Every non-zero field element a has a unique reciprocal field element $1/a$, such that $a * (1/a) = 1$. For every field element a, $0 + a =$ a, $1 * a = 1$, and $0 * a = 0$.

The simplest finite field is the field of two elements, 0 and 1. This field is often called the binary field which is useful when we later study the BCH codes. A slightly more complicated field is the field of three elements. We may denote the field elements as 0, 1, and -1. It can be verified easily that these three elements form a field using the above definition. All these finite fields are unique up to their presentation, meaning there always exist a one-to-one and onto mapping of one finite field to another finite field with the same number of elements.

It has been shown that for any prime number p, there is a finite field of p elements. In eighteenth century, the French mathematician Evariste Galois discovered the existence of unique finite fields aside from those of p elements, namely, he discovered the field of p^m elements for any integer m. These fields of p^m elements are called Galois fields and is commonly denoted by GF(p^m). A Galois field of order p^m is said to have *characteristic p*, given $\sum_{i=1}^{p} 1 = 0$. For error correction purposes, we will concentrate mostly on the field of characteristic two, in which case sign can be ignored because $1 + 1 = 0$. Other fields are less practical when implemented in hardware because of the necessity of having modulo p adders (as oppose to exclusive-OR gates that do addition modulo two). For the rest of this section, we will illustrate how to obtain an extension field GF(2^m) from the base field GF(2). The concept of extension field will later be applied to coding theory, with each element in the extended field corresponding to an error position.

Let F[x] denote the set of irreducible polynomials with coefficients from a field F. A polynomial $f(x)$ is said to be *irreducible* if it can not be factored. For example, the polynomial $f(x) = x^2+x+1$ is irreducible in the binary field of 0 and 1, since neither 0 nor 1 is a root of $f(x)$. However, the polynomial $g(x) = x^2 +1$ can be factored into $g(x) = (x+1)^2$, since 1+1=0 in the binary field. Now the question is: given an arbitrary polynomial of positive degree in $F[x]$, is it possible to find an extension field F' containing F such that the irreducible polynomial has a root in F'? The answer is revealed by the nineteenth century German mathematician Kronecker, where he simply proved that: if $f(x)$ is an irreducible polynomial in $F[x]$, then there is an extension field of F in which $f(x)$ has a root. In fact, this theorem yields an extension of the field F in which the polynomial $f(x)$ can be factored into a product of linear factors; that is, the field F' is large enough to contain all the roots of $f(x)$.

Let us now work through a concrete example on a finite field and obtain its extension field; after that we will demonstrate how the concept of extension field can be applied to algebraic coding theory.

Let us consider the binary field GF(2) with only two elements: 0 and 1. The polynomial $f(x) = x^3 + x + 1$ is irreducible since neither of the elements 0 or 1 is a root. The Kronecker theorem guarantees the existence of an extension field, in which the given polynomial has a root. Let us denote this root by α; of course, $1 + \alpha + \alpha^3 = 0$. Since the field is closed under the addition, the extension field F' must then be:

$$F' = \{0, 1, \alpha, 1 + \alpha, \alpha^2, 1 + \alpha^2, \alpha + \alpha^2, 1 + \alpha + \alpha^2\}. \tag{4.4}$$

As an example of operating in this new field, let us calculate the inverse of $1 + \alpha^2$. As we will see soon, computing inverse is a common operation in most of the decoding algorithms. Before starting, observe that by using $\alpha^3 = 1+\alpha$, we have $\alpha^4 = \alpha+\alpha^2$, $\alpha^5 = \alpha^2+\alpha^3 = 1+\alpha+\alpha^2$ and so on. Now the problem is to determine elements *a*, *b*, *c* for which

$$(1+\alpha^2)*(a+b\alpha+c\alpha^2) = 1.$$

Carrying out the multiplication and substituting for α^3 and α^4 with lower order terms, we have:

$$a+b+c\alpha+a\alpha^2 = 1.$$

This yields a system of linear equations:

$a+b = 1, c = 0, a = 0,$

with solutions $a = c = 0$, $b = 1$. Therefore, $(1+\alpha^2)^{-1} = \alpha$. In addition, $f(x) = x^3+x+1$ factors completely into linear factors in the extension field and has three roots α, α^2 and $\alpha+\alpha^2$, i.e.

$$x^3+x+1 = (x-\alpha)(x-\alpha^2)(x-(\alpha+\alpha^2)).$$

$f(x)$ is the smallest degree polynomial with coefficients in the base field GF(2) that has α as a root in the extension field GF(2^3). Note $f(x)$ must also have α^2 and $\alpha+\alpha^2$ as roots. Therefore $f(x)$ is called the *minimal polynomial* of α, the field elements that are roots of the same minimal polynomial are called *conjugates*. As a remark, minimal polynomials will be used for constructing generator polynomials for encoding.

Another way to express this concept is to use the relation of $\alpha^3 = 1+\alpha$, $\alpha^4 = \alpha+\alpha^2$, and $\alpha^5 = \alpha^2+\alpha^3 = 1+\alpha+\alpha^2$ etc., and express the extension field F' as

$$F' = \{0, 1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6\}, \text{ and } \alpha^7 = 1. \qquad (4.5)$$

It can be verified that this is equivalent to the representation expressed in (4.4)). Consequently, the minimal polynomial $f(x)$ of α can be written as $f(x) = (x-\alpha)(x-\alpha^2)(x-\alpha^4)$, where $\alpha, \alpha^2, \alpha^4$ are conjugates. In general, the conjugate of any field element α in $GF(q^m)$ is the set $\left\{\alpha, \alpha^q, \alpha^{q^2}, \ldots, \alpha^{q^{r-1}}\right\}$, where r is the smallest integer such that $\alpha^{q^r} = \alpha$.

To summarize, suppose $f(x)$ is the minimal polynomial over GF(q) of α, where α is an element of GF(q^m), then $f(x)$ is also the minimal polynomial of α^q. Consequently, the minimal polynomial of any field element α can be determined as:

$$f(x) = (x-\alpha)(x-\alpha^q)\ldots(x-\alpha^{q^{r-1}}) \qquad (4.6)$$

where r is the smallest integer such that $\alpha^{q^r} = \alpha$.

A *primitive element* of GF(q) is an element α that has the property that the first $q^m - 1$ powers of α are exactly all the $q^m - 1$ non-zero field elements of the extension field GF(q^m), such as shown in (4.5). An irreducible polynomial $p(x)$ in GF(q) that has α as a root in the entension field is called the *primitive polynomial.* Just as a remark, the primitive polynomial $p(x)$ of the smallest degree over GF(q) with $p(\alpha) = 0$ is the minimal polynomial $f(x)$ of α.

Let us illustrate the concept of extension field and its elements with one more example. Again, the binary field, GF(2), is chosen as the base field, we will extend it to GF(2^4). Let

α denote a root of the equation $x^4 + x + 1 = 0$; this happens to be a primitive element. Then the 15 non-zero field elements are given as:

α^0	1	(1000)
α^1	α	(0100)
α^2	α^2	(0010)
α^3	α^3	(0001)
α^4	$1+\ \alpha$	(1100)
α^5	$\alpha+\ \alpha^2$	(0110)
α^6	$\alpha^2+\ \alpha^3$	(0011)
α^7	$1+\ \alpha+\ \alpha^3$	(1101)
α^8	$1+\ \alpha^2$	(1010)
α^9	$\alpha+\ \alpha^3$	(0101)
α^{10}	$1+\ \alpha+\ \alpha^2$	(1110)
α^{11}	$\alpha+\ \alpha^2+\ \alpha^3$	(0111)
α^{12}	$1+\ \alpha+\ \alpha^2+\ \alpha^3$	(1111)
α^{13}	$1+\ \alpha^2+\ \alpha^3$	(1011)
α^{14}	$1+\ \alpha^3$	(1001)
α^{15}	1	

Table 4-1. Representations of field elements for GF(2^4).

Addition of two elements is done by bit-wise exclusive-OR. Multiplication in the finite field can be done with the powers of the primitive element α, according to:

$$\alpha^m * \alpha^n = \alpha^k \text{, where } k = m + n \mod 15 .$$

For example, (0111)*(0011) = $\alpha^{11} * \alpha^6 = \alpha^{17} = \alpha^2$ = (0010). Later we will show the implementation of the multiplication using shift registers. This table will be referred often when describing the decoding algorithm.

4.5. Cyclic Codes

A linear block code *C* is called a cyclic code if every cyclic shift of a codeword in *C* is also a codeword in *C*. In other words, if $[c_0, c_1, c_2, \ldots, c_{n-1}]$ is a codeword, then the shifted version $[c_{n-1}, c_0, c_1, \ldots, c_{n-2}]$ is also a codeword. Any (n, k) cyclic code can be completely specified by its *generator polynomial* $g(x)$ of degree n-k, or the *parity-check polynomial* $h(x)$ degree k, where $g(x) \cdot h(x) = x^n - 1$.

A cyclic code with block length $n = q^m - 1$, where q is a prime number is called *primitive cyclic code*. Since the field GF(q^m) is an extension field of GF(q), by the unique factorization theorem, the factorization

$$x^{q^m - 1} = f_1(x) f_2(x) \ldots f_s(x)$$

is unique over the field GF(q), where $f_l(x)$ denotes the minimal polynomial of a field element. Because the generator polynomial $g(x)$ divides $x^{q^m - 1} - 1$, it must be a product of some of these minimal polynomials. Therefore, if we wish to construct a generator polynomial $g(x)$ that has $\alpha_1, \alpha_2, \ldots, \alpha_l$ as zeros (which corrects l number of errors), then

$$g(x) = lcm[f_1(x), f_2(x), \ldots, f_l(x)],$$

where $f_i(x)$ is the minimal polynomial of α_i. The minimal polynomial of any field element α can be obtained according to (4.6).

4.5.1. Example: Bose-Chaudhuri-Hocquenghem (BCH) Codes

One prominent subclass of cyclic code is the BCH codes. These codes have been widely used in applications such as deep space communications, computer memory modules, and CD players. BCH codes are popular in practice mostly because of their relatively simple decoding algorithms discovered by Berlekamp [3][30].

By definition, a code generated by $g(x)$ is a BCH code if $g(x)$ is a polynomial of lowest degree over GF(q) for which $\alpha^{m_0}, \alpha^{m_0+1}, \ldots, \alpha^{m_0+d_0-2}$ are zeros. As a result, the minimum distance of the codes is at least d_0. One common type of BCH codes are the binary codes obtained by letting $m_0 = 1$, $d_0 = 2t_0 + 1$, and α be a primitive element of

GF(2^m); therefore, $c(x)$ is a codeword if and only if $\alpha, \alpha^2, \alpha^3, \ldots \alpha^{2t_0}$ are zeros. Since the field is binary, every even power of α is a conjugate as some previous odd power of α, i.e. they are roots of the same minimal polynomial. For example, $\{\alpha, \alpha^2, \alpha^4, \ldots\}$ is a set of conjugates, and so is the set $\{\alpha^3, \alpha^6, \ldots\}$. Therefore, the previous definition of binary BCH codes can be reduced to: $c(x)$ is a codeword if and only if $\alpha, \alpha^3, \ldots, \alpha^{2t_0}$ are zeros of the generator polynomial. Thus the generator polynomial of a binary BCH code which guaranteed to correct up to t_0 random errors can be obtained as:

$$g(x) = lcm[f_1(x), f_3(x), \ldots, f_{2t_0-1}(x)]. \tag{4.7}$$

For illustrative purposes, let us construct GF(2^4) from GF(2) using the primitive polynomial $p(x) = x^4 + x + 1$. The field elements are listed in Table 4-1. If we wish to obtain a double error correction code of block length 15, then $g(x)$ must have two roots, α and α^3 such that:

$$\begin{aligned} g(x) &= lcm[f_1(x), f_3(x)] \\ &= (x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1) = x^8 + x^7 + x^6 + x^4 + 1 \end{aligned}$$

Another commonly practiced subclass of BCH codes is the Reed-Solomon (RS) codes. These are the BCH codes which the symbol field GF(q) is the same as the error locator field GF(q^m), i.e. $m = 1$. If α is a primitive element, then the block length $n = q^m - 1 = q - 1$ symbols, which is the largest possible value.

The minimal polynomial of α^j in GF(q) is simply $x - \alpha^j$. Because the generator polynomial $g(x)$ of a t-error-correcting RS code which must have $\alpha, \alpha^2, \ldots, \alpha^{2t}$ as roots, it therefore equals to:

$$g(x) = (x - \alpha)(x - \alpha^2)\ldots(x - \alpha^{2t}).$$

$g(x)$ always have degree $2t$. For example, for a (15, 11) $t = 2$ RS code over GF(16), it can be verified (with the help of Table 4-1) that

$$g(x) = (x-\alpha)(x-\alpha^2)(x-\alpha^3)(x-\alpha^4) = x^4+\alpha^{13}x^3+\alpha^6x^2+\alpha^3x+\alpha^{10}.$$

Also note that each symbol in GF(16) is four bits in binary representation. For block length of 15 symbols, it is equivalent to bits in binary.

A RS code has minimum distance $d = 2t+1 = n-k+1$, so it is a maximum-distance separable code. In other words, for any fixed (n, k) where a RS code exists, no other code can have a larger minimum distance. Of course, when there are (n, k) where no RS codes exist, other codes are preferred.

Since RS codes are character oriented code, they have natural advantage in terms of burst error correction, but have disadvantage with random errors. Therefore, in the case of a binary symmetric channel, a binary BCH code would offer better error correction capability than the RS code of the same length and redundancy. However if the channel is bursty, then RS codes are often preferred.

4.5.2. Encoding

For an (n, k) cyclic code, suppose we want to encode the information sequence $[i_0, i_1, \ldots, i_{k-1}]$. This sequence can be expressed with a polynomial $i(x)$ of degree of $k-1$, where the coefficients of $i(x)$ are the information symbols. The generator polynomial is obtained by (4.7). One simple way to encode is to multiply the information polynomial with the generator polynomial such that the codeword polynomial $c(x) = i(x) \cdot g(x)$. Figure 4-6 shows a shift register circuit which performs this type of encoding.

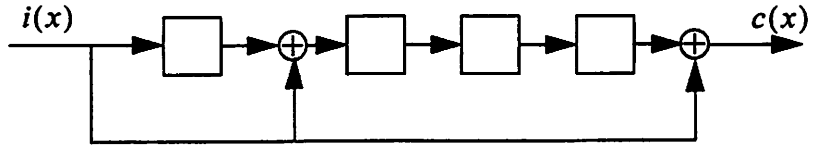


Figure 4-6. A non-systematic encoder for (15, 11) Hamming code.

Recall that a code is called systematic first k symbols are information symbols and the last $n-k$ symbols are the parity check symbols. Since the information sequence $i(x)$ is not directly a part of $c(x)$, the above encoding process does not produce a systematic code. It can be shown that every linear block code is equivalent to a systematic code. A cyclic code, $c(x) = i(x) \cdot g(x)$, can be made systematic by the codeword of the form $c(x) = x^{n-k} \cdot i(x) + r(x)$, where $r(x)$ is the negative of the remainder of $x^{n-k} \cdot i(x)$ dividing $g(x)$. Figure 4-7 illustrates a systematic encoder for (15, 11) Hamming code, for which the generator polynomial is $g(x) = x^4 + x + 1$.

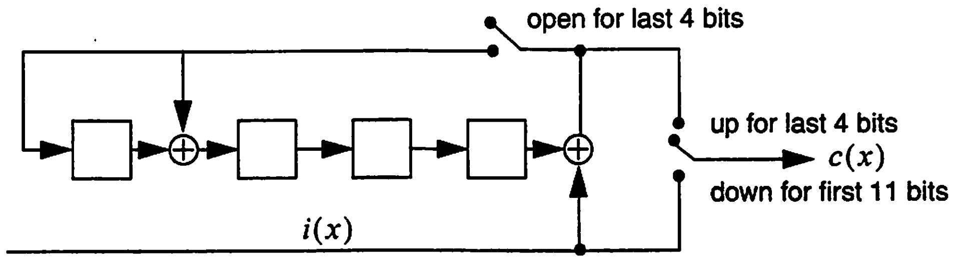


Figure 4-7. A systematic encoder for (15, 11) Hamming code.

Since $g(x) = (x^n - 1)/h(x)$, encoding is also feasible by employing an $n-k$ stage shift register that performs division of the *parity check polynomial* $h(x)$, i.e. $c(x) = i(x) \cdot g(x) = i(x) \cdot (x^n - 1)/h(x)$. The decision on which encoding method to use depends on the relative size of n and k. In the case when a codeword has fewer information symbols, i.e. $k < n - k$, then a k-stage shift register is used for multiplying the information polynomial with the generator polynomial. On the other hand, if a codeword has more information symbols than parity check symbols, i.e. $k > n - k$, an $n - k$ stage shift register is preferred.

4.5.3. Decoding

Assume the codeword is transmitted through a noisy channel and the received vector is $[r_{n-1}, r_{n-2}, \ldots, r_0]$. Using the polynomial notation, the received polynomial is then: $r(x) = r_{n-1}x^{n-1} + r_{n-2}x^{n-2} + \ldots + r_0$. To decode, $r(x)$ is first divided by $g(x)$; the remainder $s(x)$ is called the syndrome polynomial, and the corresponding coefficient vector is the syndrome vector. If the syndrome is a zero vector, it indicates that the received vector is a codeword; if the syndrome is non-zero, it says there exist error(s) in the received vector. To correct the errors, the procedure involves multiple steps and has two outcomes: if a syndrome is associated with a correctable error pattern, then the corresponding error pattern is added to the receiving vector to recover the original transmitted codeword so these errors are corrected; if a syndrome is associated with an un-correctable error pattern, then this un-correctable-error result can only yield received word error. We will discuss decoding algorithms starting with the binary Hamming codes and then moving on to the BCH codes.

4.5.4. Hamming code

Hamming code is a single error correction code. The generator polynomial $g(x)$ has one root which we denote as α. At the receiver's end, the received vector can be written as the polynomial $r(x)$, where $r(x)$ it is the sum of the codeword polynomial $c(x)$ and the error polynomial $e(x)$, i.e. $r(x) = c(x) + e(x)$. The syndrome is calculated by evaluating the received vector at α. Since α is a root of $g(x)$, it is also a root of $c(x)$ because the codeword is the product of the information polynomial and the generator polynomial. Evaluating $r(x)$ at α yields $r(\alpha) = c(\alpha) + e(\alpha) = e(\alpha)$.

Suppose we are decoding a (n, k) Hamming code. The error polynomial evaluated at α is $e(\alpha) = e_0\alpha^0 + e_1\alpha^1 + e_2\alpha^2 + \ldots + e_n\alpha^n$, where e_i is 0 when there is no error at ith bit and 1 when there is an error. Thus $r(\alpha) = e(\alpha) = 0$ when the received codeword is correct. If there is a single error, then $e_i = 1$ for some i. The syndrome calculated by the decoder then equals to: syndrome $= r(\alpha) = e(\alpha) = \alpha^i$. By consulting with a look-up table, we can identify α^i thus know it is the ith bit that has been corrupted. However, if two bits are corrupted, i.e. $e_i = 1, e_j = 1$ for some i and j, this decoding algorithm would fail because one equation, $e(\alpha)$, is not sufficient to reveal multiple unknowns. The decoding algorithm for a Hamming code is outlined in Figure 4-8.

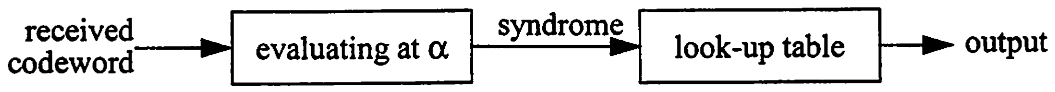


Figure 4-8. Syndrome decoder for Hamming codes.

4.5.5. Binary BCH codes

The BCH codes is a class of powerful multiple error correction code. As a consequence of its ability to correct multiple errors, the decoding algorithm is significantly more complicated than the Hamming code. In this section, we will review two decoding algorithms. The first one is conceptually clear but less efficient; it was first developed by Peterson [35]. The second decoding algorithm is more commonly used but harder to understand, it was first discovered by Berlekamp [3].

4.5.5.1. The Peterson Algorithm

The received polynomial for an (n, k, t) binary BCH code is $r(x)$ where $r(x) = c(x) + e(x)$. The error polynomial $e(x) = e_0 + e_1x + e_2x^2 + \ldots + e_nx^n$ with the error magnitude $e_i \in \{0, 1\}$. Suppose there are m errors occurred inside the block where

$m \leq t$, and these errors are at positions $l_1, l_2, \ldots, l_m$, then the error polynomial can be simplified to $e(x) = x^{l_1} + x^{l_2} + \ldots + x^{l_m}$.

For each received polynomial, the first $2t$ syndromes are calculated in preparation to correct up to t errors. The jth syndrome is defined as:

$$S_j = r(\alpha^j) = c(\alpha^j) + e(\alpha^j) = e(\alpha^j).$$

which does evaluation of the received polynomial at the jth root of $g(x)$.

For example, $S_1 = e(\alpha) = \alpha^{l_1} + \alpha^{l_2} + \ldots + \alpha^{l_m}$, $S_2 = e(\alpha^2) = \alpha^{2l_1} + \alpha^{2l_2} + \ldots + \alpha^{2l_m}$ and so on. To make things less confusing, we will adopt the notation $X_i = \alpha^{l_i}$, thus the syndromes $S_1, S_2, \ldots, S_{2t}$ are:

$$\begin{aligned} S_1 &= X_1 + X_2 + \ldots + X_m \\ S_2 &= X_1^2 + X_2^2 + \ldots + X_m^2 \\ &\vdots \\ S_{2t} &= X_1^{2t} + X_2^{2t} + \ldots + X_m^{2t} \end{aligned}$$

Because the codes are binary, it can be proved that $S_{2j} = S_j^2$, which says only the odd numbered syndromes have to be computed.

The goal here is to solve the above syndrome equations for error locations X_i. However, as these are non-linear equations, no systematic method is available for solving them directly. Fortunately, a less direct method has been discovered by introducing an *error locator polynomial* $\Lambda(x)$ in x, where

$$\Lambda(x) = 1 + \Lambda_1 x + \ldots + \Lambda_{m-1} x^{m-1} + \Lambda_m x^m = (1 - xX_1)(1 - xX_2)\ldots(1 - xX_m) \quad (4.8)$$

The zeros of $\Lambda(x)$ are the inverse of the error locations X_i, for $i = 1, \ldots, m$. Therefore, if we can obtain the error locator polynomial, then we can find its zeros thus the error locations. Now the decoding problem can be reformulated as finding the coefficients, Λ_i for the error locator polynomial.

To proceed, let us first set $x = X_i^{-1}$ so the left side of (4.8) equals to zero. Next we multiply both side by X_i^{j+m} so we have:

$$0 = X_i^{j+m} + \Lambda_1 X_i^{j+m-1} + \ldots + \Lambda_{m-1} X_i^{j+1} + \Lambda_m X_i^j$$

which hold for all X_i and j. Summing these equations up with the index i, we have:

$$\sum_{i=1}^{m} (X_i^{j+m} + \Lambda_1 X_i^{j+m-1} + \ldots + \Lambda_{m-1} X_i^{j+1} + \Lambda_m X_i^j) = 0$$

or

$$\sum_{i=1}^{m} X_i^{j+m} + \Lambda_1 \sum_{i=1}^{m} X_i^{j+m-1} + \quad + \Lambda_m \sum_{i=1}^{m} X_i^j = 0$$

Notice the syndromes $S_{j+m} = \sum_{i=1}^{m} X_i^{j+m}$; therefore, the above equation can be written in the form:

$$\Lambda_1 S_{j+m-1} + \Lambda_2 S_{j+m-2} + \quad + \Lambda_m S_j = -S_{j+m} \qquad \text{for all } j = 1, \ldots, m \qquad (4.9)$$

Writing out (4.9) for all j using the matrix notation, we have:

$$\begin{bmatrix} S_1 & S_2 & \dots & S_{m-1} & S_m \\ S_2 & S_3 & \dots & S_m & S_{m+1} \\ S_3 & S_4 & & S_{m+1} & S_{m+2} \\ & & & & \\ S_m & S_{m+1} & \dots & S_{2m-2} & S_{2m-1} \end{bmatrix} \begin{bmatrix} \Lambda_m \\ \Lambda_{m-1} \\ \Lambda_{m-2} \\ \\ \Lambda_{2m-1} \end{bmatrix} = \begin{bmatrix} -S_{m+1} \\ -S_{m+2} \\ -S_{m+3} \\ \\ -S_{2m} \end{bmatrix} \tag{4.10}$$

Now Λ_i can be solved by taking the inverse of the syndrome matrix and multiply it at both sides. The $l \times l$ syndrome matrix

$$M = \begin{bmatrix} S_1 & S_2 & \dots & S_l \\ S_2 & S_3 & \dots & S_{l+1} \\ & & & \\ S_l & S_{l+1} & \dots & S_{2l-1} \end{bmatrix}$$

is non-singular if l equals to the number of errors occurred, i.e. $l = m$. This property will help us to determine the exact number of errors occurred within a block: first we let $l = t$, which is the maximum number of correctable errors and test if the syndrome matrix is singular. If the syndrome matrix is singular, then l is decreased by 1 and the test is performed again. This process repeats itself until the syndrome matrix becomes non-singular, in which case it will be inverted to compute the error locator polynomial.

To summarize, the decoding algorithm involves three steps:

- for received block $r(x)$, compute the syndromes, where $S_j = r(\alpha^j)$ for $j = 1, \dots, t$
- use the syndrome matrix to determine the number of errors within the block and then compute the error locator polynomial with the inverse of the syndrome matrix
- substitute each field element into the error locator polynomial and determine the actual error location.

Next we will demonstrate the above decoding algorithm with a (15, 5) BCH triple error correction code. Its generator polynomial is $g(x) = x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1$. Let us assume information bits are all zero thus the information polynomial $i(x) = 0$. Suppose three bits are corrupted at locations 2, 3, and 4 (counting starts at 0), so the received polynomial $r(x) = x^2 + x^3 + x^4 = e(x)$.

Using the finite field representation of GF(2^4) shown in Table 4-1, we have:

Step 1: compute the syndromes.

$$S_1 = \alpha^2 + \alpha^3 + \alpha^4 = \alpha^{12}$$

$$S_2 = S_1^2 = \alpha^9$$

$$S_3 = \alpha^6 + \alpha^9 + \alpha^{12} = \alpha^{14}$$

$$S_4 = S_2^2 = \alpha^3$$

$$S_5 = \alpha^{10} + \alpha^{15} + \alpha^{20} = 0$$

$$S_6 = S_3^2 = \alpha^{13}$$

Step 2: compute the coefficients $(\Lambda_1, \Lambda_2, \Lambda_3)$ for the error locator polynomial $\Lambda(x) = 1 + \Lambda_1 x + \Lambda_2 x^2 + \Lambda_3 x^3$.

The syndrome matrix

$$M = \begin{bmatrix} \alpha^{12} & \alpha^9 & \alpha^{14} \\ \alpha^9 & \alpha^{14} & \alpha^3 \\ \alpha^{14} & \alpha^3 & 0 \end{bmatrix}$$

is non-singular, so we know the received block has three errors. Taking inverse of M to compute Λ_i:

$$\Lambda_1 = S_1 = \alpha^{12}$$

$$\Lambda_2 = \frac{S_1^2 S_3 + S_5}{S_1^3 + S_3} = 1 \text{, and}$$

$$\Lambda_3 = \frac{S_1 S_5 + S_3^2 + S_1^3 S_3 + S_1^6}{S_1^3 + S_3} = \alpha^9 .$$

As a result, $\Lambda(x) = 1 + \alpha^{12}x + x^2 + \alpha^9 x^3 = (1 + \alpha^2 x)(1 + \alpha^3 x)(1 + \alpha^4 x)$.

Step 3: the inverse of zeros for $\Lambda(x)$ are $\alpha^2, \alpha^3, \alpha^4$, so the errors occurred at the 2nd, 3rd, 4th bits.

Notice the step 2 is the most computational intensive step of the entire algorithm, for it calculates the inverse of the syndrome matrix. In general, the complexity of inverting an $l \times l$ matrix is in the order l^3. Clearly, for codes that correct a large number of errors, this step can be very expensive. Fortunately, the syndrome matrix is not any arbitrary matrix, and Berlekamp has discovered a recursive method that uses linear shift back registers for computing the error locator polynomial. For the rest of this section, we will discuss Massy's interpretation of Berlekamp's algorithm [30].

4.5.5.2. The Berlekamp Algorithm

The (4.9)) from the previous section stated that

$$\Lambda_1 S_{j+m-1} + \Lambda_2 S_{j+m-2} + \quad + \Lambda_m S_j = -S_{j+m} \qquad \text{for } j = 1, \ldots, m .$$

For a fixed set of Λ_i, this is an equation of an autoregressive filter, as shown in Figure 4-9.

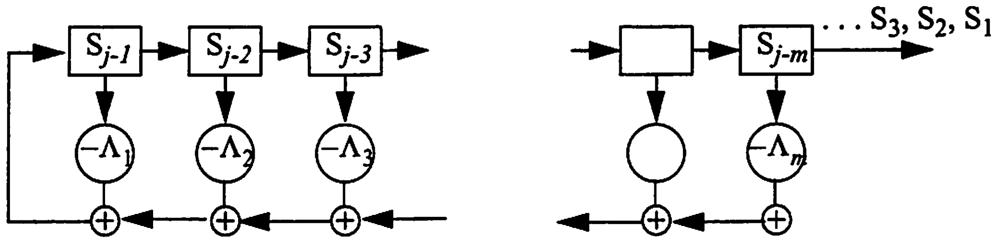


Figure 4-9. Error locator polynomial as in an autoregressive filter

The problem of finding the error locator polynomial $\Lambda(x)$ is now reformulated as designing the autoregressive filter (with unknown taps Λ_i) that produces the right syndromes. The procedure for such filter design is also the procedure for solving the matrix equation in (4.10)), which was first discovered by Berlekamp.

The steps for finding the right tap values are recursive. Let us assume the syndromes $S_1, S_2, \ldots, S_{2t}$ are already given. We start with iteration $k = 1$ with the initial conditions $\Lambda^{(0)}(x) = 1$, $B^{(0)}(x) = 1$, $L_0 = 0$. At any later iteration k, let L_k be the length of the shift registers, and let Δ_k, $B(x)$, δ be the temporary variables, the following computation assures the shortest shift register that provides the correct syndromes.

$$\Delta_k = \sum_{j=0}^{n-1} \Lambda_j^{(k-1)} S_{k-j}$$

$$L_k \leftarrow \delta(k - L_{k-1}) + (1-\delta)L_{k-1}$$

$$\begin{bmatrix} \Lambda^{(k)}(x) \\ B^{(k)}(x) \end{bmatrix} = \begin{bmatrix} 1 & -\Delta_k x \\ \Delta_k^{-1}\delta & (1-\delta)x \end{bmatrix} \begin{bmatrix} \Lambda^{(k-1)}(x) \\ B^{(k-1)}(x) \end{bmatrix}$$

where $k = 1, \ldots, 2t$, and $\delta = 1$ if $\Delta_k \neq 1$ and $2L_{k-1} \leq k-1$, and $\delta = 0$ otherwise. The algorithm terminates with the correctable error patterns when $k = 2t$ and $\deg\Lambda(x) = L_{k-1}$. Otherwise if $k = 2t$ and $\deg\Lambda(x) \neq L_{k-1}$, it indicates more than t errors have occurred, thus they are un-correctable. In the first case, the resulting error locator polynomial is passed on to the next step for finding the corresponding error locations, where the received bits are then inverted accordingly. In the case when the errors are not correctable, instead of potentially introduce more errors, the data are left unchanged as they are received; in the mean while, a flag will indicate the errors being not correctable. The step-by-step Berlekamp's algorithm is outlined in the following flow chart.

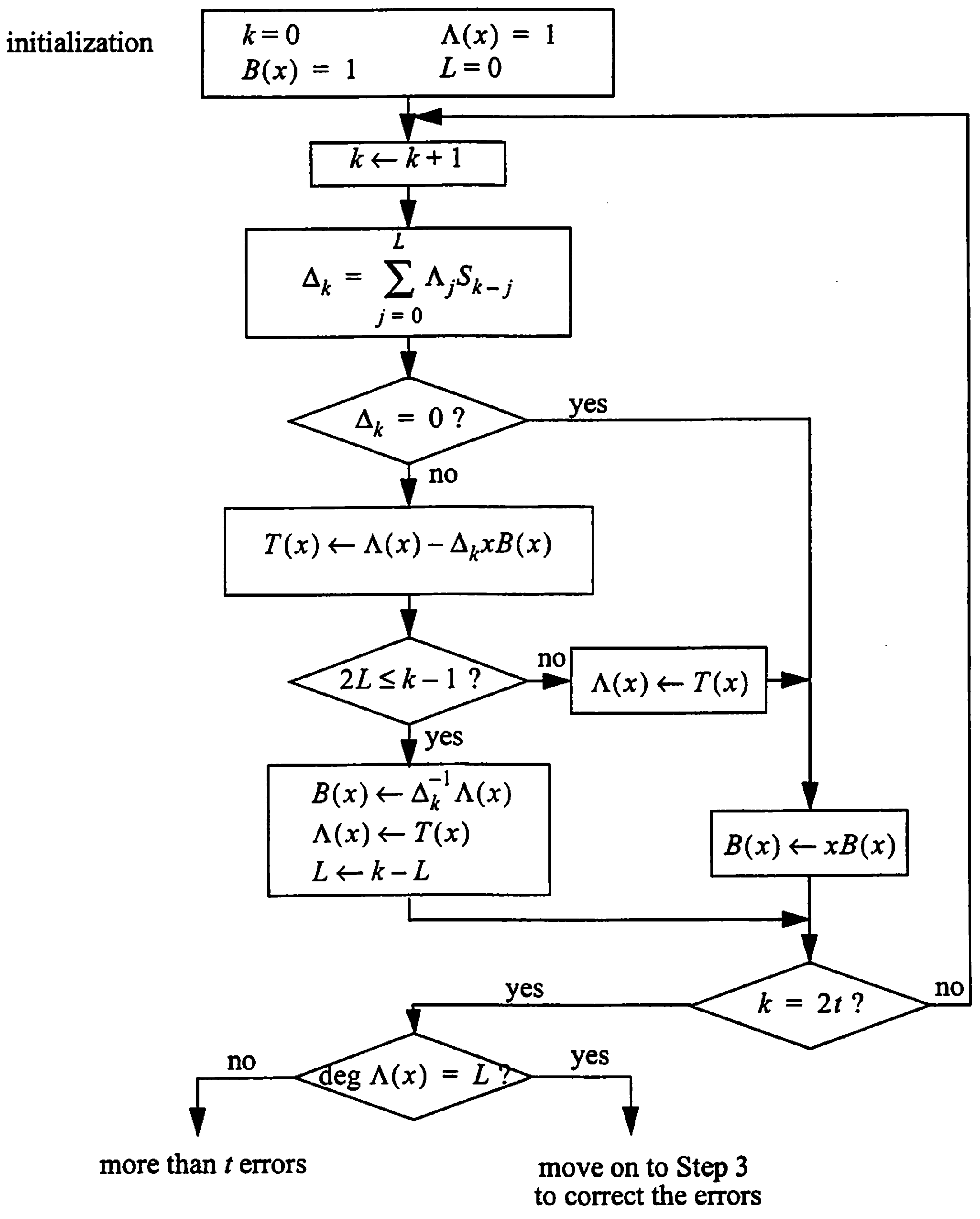

initialization
$k=0$
$\Lambda(x) = 1$
$B(x) = 1$
$L=0$
$k \leftarrow k+1$
$\Delta_k = \sum_{j=0}^{L} \Lambda_j S_{k-j}$
$\Delta_k = 0$?
yes
no
$T(x) \leftarrow \Lambda(x) - \Delta_k x B(x)$
$2L \leq k-1$?
no
$\Lambda(x) \leftarrow T(x)$
yes
$B(x) \leftarrow \Delta_k^{-1} \Lambda(x)$
$\Lambda(x) \leftarrow T(x)$
$L \leftarrow k-L$
$B(x) \leftarrow xB(x)$
yes
$k = 2t$?
no
no
deg $\Lambda(x) = L$?
yes
more than t errors
move on to Step 3 to correct the errors

Figure 4-10. The Berlekamp algorithm.

Let us walk through the steps of this iterative algorithm using the same example on $(15, 5)$ BCH code as we had earlier. In that example, the generator polynomial $g(x) = x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1$, the information bits are all zero thus the information polynomial $i(x) = 0$. If three bits errors are received at locations 2, 3, and 4, then the received polynomial $r(x) = x^2 + x^3 + x^4$. As a remark, please refer to Table 4-1 for finite field addition and multiplication.

The corresponding syndromes for this received polynomial are: $S_1 = \alpha^{12}$, $S_2 = \alpha^9$, $S_3 = \alpha^{14}$, $S_4 = \alpha^3$, $S_5 = 0$, and $S_6 = \alpha^{13}$. Going through the recursive steps described in Figure 4-10, the values for all the parameters in each step are summarized below:

k	Δ_k	T(x)	B(x)	Λ(x)	L
0			1	1	0
1	α^{12}	$1+\alpha^{12}x$	α^3	$1+\alpha^{12}x$	1
2	0	$1+\alpha^{12}x$	$\alpha^3 x$	$1+\alpha^{12}x$	1
3	α^8	$1+\alpha^{12}x+\alpha^{11}x^2$	$\alpha^7+\alpha^4 x$	$1+\alpha^{12}x+\alpha^{11}x^2$	2
4	0	$1+\alpha^{12}x+\alpha^{11}x^2$	$\alpha^7 x+\alpha^4 x^2$	$1+\alpha^{12}x+\alpha^{11}x^2$	2
5	α^5	$1+\alpha^{12}x+x^2+\alpha^9 x^3$	$\alpha^{10}+\alpha^7 x+\alpha^6 x^2$	$1+\alpha^{12}x+x^2+\alpha^9 x^3$	3
6	0	$1+\alpha^{12}x+x^2+\alpha^9 x^3$	$\alpha^{10}x+\alpha^7 x^2+\alpha^6 x^3$	$1+\alpha^{12}x+x^2+\alpha^9 x^3$	3

Table 4-2. Recursive steps in Berlekamp's algorithm.

The error-correcting procedure terminates at the sixth iteration, where $k = 6 = 2t$ and $\deg\Lambda(x) = L_{k-1}$; thus errors are known to be correctable. The error locator polynomial $\Lambda(x) = 1 + \alpha^{12}x + x^2 + \alpha^9 x^3$, which agrees with the result derived using Peterson's algorithm.

Observe that $\Delta_k = 0$ for all k even. This is true in general for binary BCH codes due to the fact that $S_{2j} = S_j^2$ in binary field. As a consequence, the iterative algorithm can skip all even numbered iterations, so the total number of iterations required to decode a (n, k, t) binary BCH code is t (as oppose to $2t$ for general BCH codes).

To summarize, the complete iterative decoding algorithm consists of three steps:

- for received block $r(x)$, compute the syndromes $S_1, S_2, \ldots, S_{2t}$;
- apply Berlekamp's algorithm to obtain the error locator polynomial; this step takes $2t$ iterations for the general BCH codes and t iterations for the binary BCH codes;
- substitute each field element into the error locator polynomial and determine the actual error location.

In the next chapter, we will present a decoder implementation of binary BCH codes for correcting variable number of errors. The basic architecture will be based on Berlekamp's algorithm.

5 Architecture for Variable FEC Decoder

5.1. Goal

Extensive work has been done in implementing Reed-Solomon (RS) codes which is a subclass of BCH codes [3][9][42]. In this chapter, we will propose an variable forward error correction decoder architecture based on binary BCH code. As will be shown later, this design can be mapped efficiently to a custom VLSI implementation or programmable logic devices. We will first present an architecture and its control logic for a single binary $(63, k)$ BCH decoder. This design will then be extended to a VFEC decoder design.

Referring back to the simulation results obtained in Chapter 3, the family of 63-bit binary BCH codes appear to be the most suitable choice for VFEC. This is because these codes offer a good compromise between implementation complexity and variable error correction capabilities. These are the codes generated by the finite field GF(2^6), with the primitive polynomial $p(x) = x^6 + x + 1$; and thus have the same block length. The range of error correction capability for codes within this family is $t = 0, 1, 2, 3, 4, 5, 6, 7, 10, 11, 13, 15, 31$, the corresponding number of information bits within the block is $k = 63, 57, 51, 45, 39, 36, 30, 24, 18, 16, 10, 7, 1$. These codes have the further implementation advantage that a majority of the decoder hardware can be shared after exploiting redundancies among codes. An overview of the subsystem incorporating VFEC is shown in Figure 5-1.

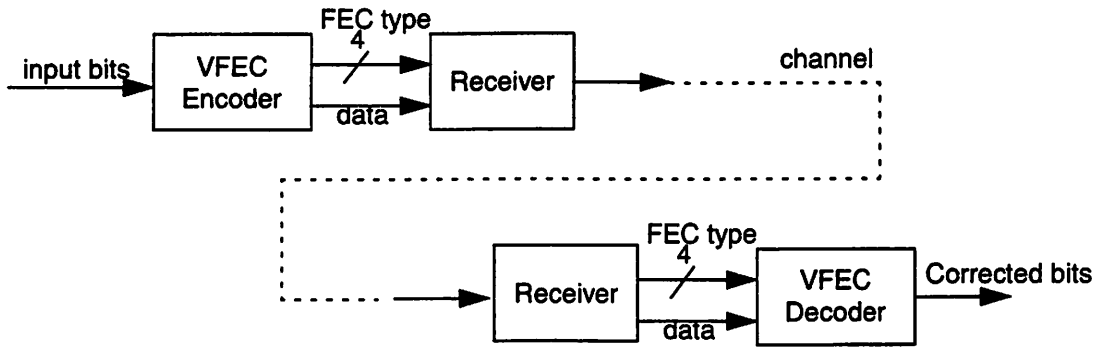


Figure 5-1. An overview of a VFEC subsystem.

5.2. Basic circuits for GF algebra

Before presenting the decoder structure, we will first illustrate the basic circuits for performing finite field arithmetic, such as addition and multiplication. Our examples will be carried out in GF(2^6); however, the concept can be applied to any field.

5.2.1. Addition of two field elements

Any field element in GF(2^6) is represented using 6 bits. Therefore, addition of two field elements can be made using six parallel bit-wide exclusive-OR gates. Let α be the primitive element of GF(2^6). Then any field element can be expressed in terms of α. Let

$$a = a_0 + a_1\alpha + a_2\alpha^2 + a_3\alpha^3 + a_4\alpha^4 + a_5\alpha^5, \text{ and}$$

$$b = b_0 + b_1\alpha + b_2\alpha^2 + b_3\alpha^3 + b_4\alpha^4 + b_5\alpha^5.$$

Then $a + b = c$ is achieved by:

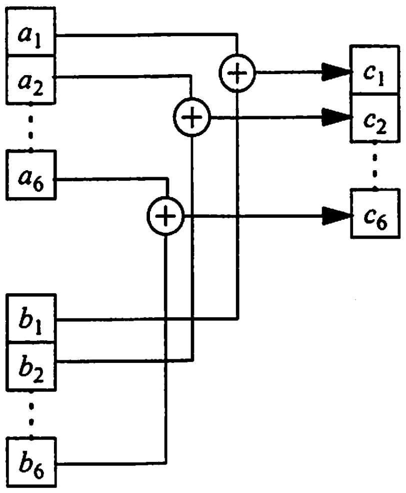


Figure 5-2. Adding two field elements.

5.2.2. Multiplication of two field elements

The multiplication of two field elements is more involved. The steps can be seen by explicitly carrying out the multiplication of two field elements, a and b. Let

$a = a_0 + a_1\alpha + a_2\alpha^2 + a_3\alpha^3 + a_4\alpha^4 + a_5\alpha^5$ and

$b = b_0 + b_1\alpha + b_2\alpha^2 + b_3\alpha^3 + b_4\alpha^4 + b_5\alpha^5$, where α is the primitive element, i.e. $\alpha^6 + \alpha + 1 = 0$. Carrying out the algebra by first letting each term of a multiplying with each term of b and then adding the terms of the same order, the result is:

$$\begin{aligned}
ab &= (a_0 + a_1\alpha + a_2\alpha^2 + a_3\alpha^3 + a_4\alpha^4 + a_5\alpha^5)(b_0 + b_1\alpha + b_2\alpha^2 + b_3\alpha^3 + b_4\alpha^4 + b_5\alpha^5) \\
&= (a_0b_0 + a_1b_5 + a_2b_4 + a_3b_3 + a_4b_2 + a_5b_1) + \\
&\quad (a_0b_1 + a_1b_0 + a_1b_5 + a_2b_4 + a_2b_5 + a_3b_3 + a_3b_4 + a_4b_2 + a_4b_3 + a_5b_1 + a_5b_2)\alpha + \\
&\quad (a_0b_2 + a_1b_1 + a_2b_0 + a_2b_5 + a_3b_4 + a_3b_5 + a_4b_3 + a_4b_4 + a_5b_2 + a_5b_3)\alpha^2 + \\
&\quad (a_0b_3 + a_1b_2 + a_2b_1 + a_3b_0 + a_3b_5 + a_4b_4 + a_4b_5 + a_5b_3 + a_5b_4)\alpha^3 + \\
&\quad (a_0b_4 + a_1b_3 + a_2b_2 + a_3b_1 + a_4b_0 + a_4b_5 + a_5b_4 + a_5b_5)\alpha^4 + \\
&\quad (a_0b_5 + a_1b_4 + a_2b_3 + a_3b_2 + a_4b_1 + a_5b_0 + a_5b_5)\alpha^5
\end{aligned}$$

Again, a_i and b_i are binary, so the multiplication between any pair is done using an AND gate, and addition of two binary numbers is done with a XOR gate.

5.2.3. Inverse of a field element

One approach for obtaining the inverse of a field element is to work out the arithmetic that for a given a, we wish to find a field element b such that $ab = 1$. Multiplying out the cross terms like before and apply this constraint, we can express all the b_i in terms of a_i. One example using this method has been described in Chapter two. Another approach is more straight forward which requires a small lookup table with six input and six output (384 bit ROM). Even though this later method may require more logic gates to implement, the circuit is however simpler and faster. As we will see later, only one field element inverter is needed in designing VFEC decoder, so later approach will be adopted in our architecture.

5.3. VFEC encoder design

The design of any error correcting system involves two parts: encoding and decoding. Generally, the FEC encoding process is relatively simple and is implemented by a set of shift registers. One such design is shown in Figure 5-3 where all encoders process data simultaneously. The final output is selected by the output multiplexer according to the FEC type.

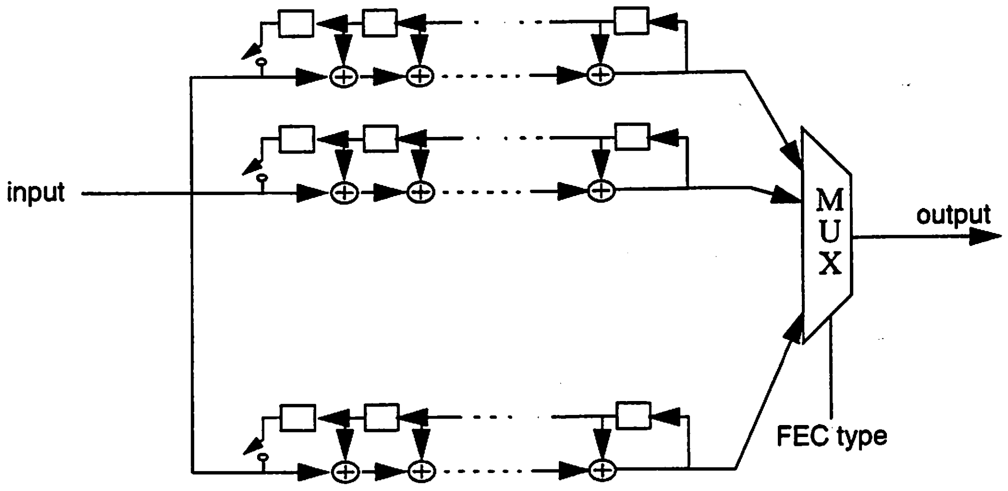


Figure 5-3. A VFEC encoder.

The implementation challenges of a FEC system often come from the decoder design. The decoder for VFEC is especially critical since multiple decoders have to be implemented simultaneously. We will first present an architecture and the control logic for a single binary $(63, k)$ BCH decoder; this single decoder will then serve as the building block for VFEC design.

5.4. Implementation of a single binary (63, *k*) BCH decoder

5.4.1. Review of error correction procedure

Let $r(x)$ denote the received codeword. As stated in the previous chapter, the procedure to decode an (n, k) t-error-correcting BCH code consists of three steps:

1. Calculate the first $2t$ syndromes, $S_1, S_2, \ldots, S_{2t}$, where $S_i = r(\alpha^i)$ and α is the primitive element.
2. Compute the error locator polynomial $\Lambda(x)$; this can be done through the Berlekamp's algorithm.
3. Compute the error magnitude and then correct the errors.

These three steps can be seen in an overall decoder structure illustrated below:

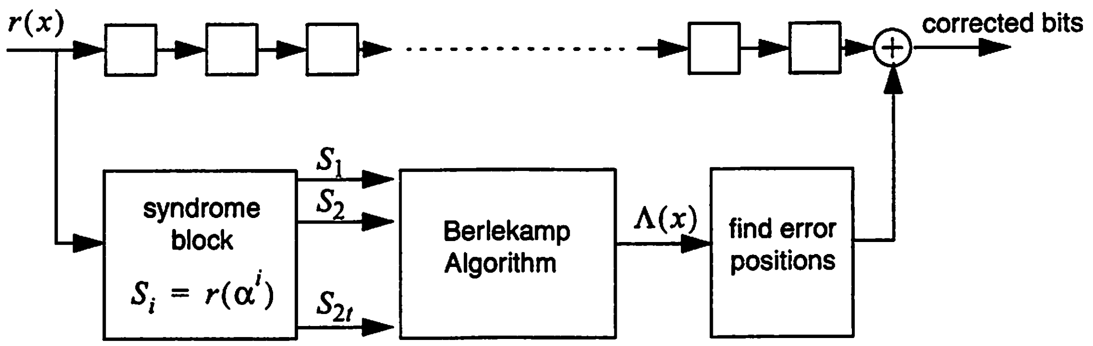


Figure 5-4. Outline for error correction procedure.

Since VFEC is implemented using binary BCH codes, the decoding algorithm is significantly simpler than a general decoding procedure. First, the received symbols are binary, the error magnitude is always 1. In addition, since even syndromes $S_{2j} = S_j^2$ for all j, the even numbered iterations of the Berlekamp algorithm are redundant because the discrepancy factor, Δ, equals zero during these iterations. Therefore, instead of taking $2t$ iterations to complete the Berlekamp algorithm, as for the general BCH codes, the error locator polynomial $\Lambda(x)$ for binary BCH codes can be obtained in t iterations.

5.4.2. Syndrome block

To decode an arbitrary $(63, k)$ BCH code, the syndrome block computes the first $2t$ syndromes, $S_1, S_2, \ldots, S_{2t}$, for each received block of 63 bits. The resulting syndromes are field elements of GF(2^6). One way to compute the ith syndrome is to employ a division circuit that divides $r(x)$ by the minimum polynomial of α^i, denoted as $m^{(i)}(x)$. The remainder $R^{(i)}(x)$ evaluated at α^i is the desired syndrome S_i. For example, to calculate S_1, we first divide the $r(x)$ by the minimum polynomial of α, $m^{(1)}(x)$. Suppose the remainder is $R^{(1)}(x)$, then $S_1 = R^{(1)}(\alpha)$. Similarly, to compute S_5, we divide the $r(x)$

by the minimum polynomial of α^5, $m^{(5)}(x)$ to get the remainder $R^{(5)}(x)$, then $S_5 = R^{(5)}(\alpha^5)$.

Another approach to obtain S_i is evaluating $r(x)$ at the ith power of the primitive element α, i.e. $S_i = r(\alpha^i)$. This can be done by observing the relation that:

$$r(\beta) = r_{62}\beta^{62} + r_{61}\beta^{61} + r_{60}\beta^{60} + \ldots + r_0 = (((r_{62}\beta + r_{61})\beta + r_{60})\beta + \ldots)\beta + r_0$$

where β is an arbitrary field element. Therefore, $r(\beta)$ can be obtained through an iterative operation which first multiplies β and then adds the result to the coefficient of the next order. A shift register is a perfect choice for multiplying α^i repeatedly. The circuit for $S_1 = r(\alpha) = r_{62}\alpha^{62} + r_{61}\alpha^{61} + \ldots + r_0$ is relatively simple. This can be seen by expressing a field element $a = a_0 + a_1\alpha + a_2\alpha^2 + a_3\alpha^3 + a_4\alpha^4 + a_5\alpha^5$. Because α is primitive, then $\alpha^6 + \alpha + 1 = 0$. Calculating αa, we have:

$$\begin{aligned}
\alpha a &= \alpha(a_0 + a_1\alpha + a_2\alpha^2 + a_3\alpha^3 + a_4\alpha^4 + a_5\alpha^5) \\
&= a_0\alpha + a_1\alpha^2 + a_2\alpha^3 + a_3\alpha^4 + a_4\alpha^5 + a_5\alpha^6 \\
&= a_0\alpha + a_1\alpha^2 + a_2\alpha^3 + a_3\alpha^4 + a_4\alpha^5 + a_5(1 + \alpha) \\
&= a_5 + (a_0 + a_5)\alpha + a_1\alpha^2 + a_2\alpha^3 + a_3\alpha^4 + a_4\alpha^5
\end{aligned}$$

The new value in the register 0 is the content of old register 5, and the new value in register 1 is the sum of old content in register 0 and 5, the new value of register 2 is what was in register 1, etc. Therefore, a shift register with connection shown in Figure 5-5 computes S_1. After shifting this circuit for 63 clock cycles, the register contents are the desired

S_1. Just as a side note, this shift register can also be used for counting in finite field, from α to α^{62}, with 1 placed in the lowest order register and 0s elsewhere.

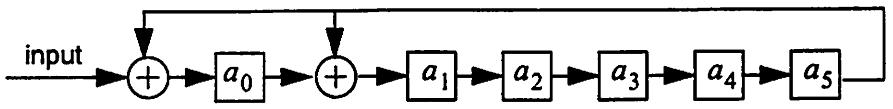


Figure 5-5. A circuit for calculating $r(\alpha)$.

Evaluating polynomial $r(x)$ at α^i for $i \neq 1$ to obtain $S_i = r(\alpha^i)$ is more complicated. We will illustrate the concept by implementing $S_3 = r(\alpha^3)$. Calculating $\alpha^3 a$, we have:

$$\begin{aligned}\alpha^3 a &= \alpha^3(a_0 + a_1\alpha + a_2\alpha^2 + a_3\alpha^3 + a_4\alpha^4 + a_5\alpha^5)\\ &= a_3 + (a_3 + a_4)\alpha + (a_4 + a_5)\alpha^2 + (a_0 + a_5)\alpha^3 + a_1\alpha^4 + a_2\alpha^5\end{aligned}$$

This can then be mapped into a shift register with connections shown in Figure 5-6. The other syndromes can be obtained by similar circuits.

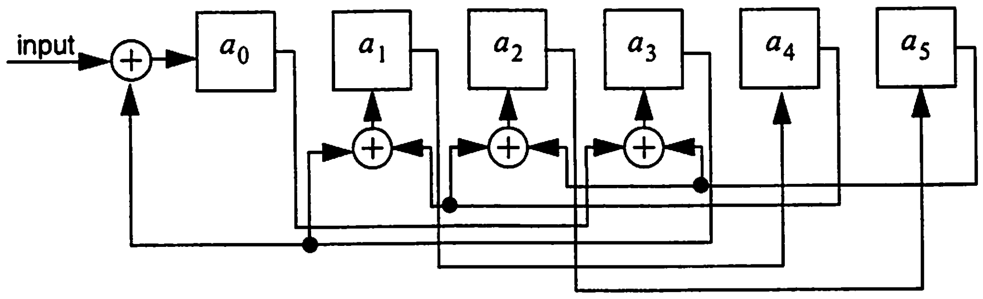


Figure 5-6. A circuit for calculating $r(\alpha^3)$.

This syndrome computation therefore results in a set of syndromes at the end of every 63 clock cycles, these are then provided as input into the error locator module, where the Berlekamp algorithm is performed.

5.4.3. Berlekamp's algorithm

Our main contribution to BCH decoder design is efficiently implementing the error locator module which computes the error locator polynomial, $\Lambda(x)$. This is often the most challenging task of any BCH decoder design. For a t-error-correcting code, $\Lambda(x)$ has degree t and can be expressed as $\Lambda(x) = \Lambda_t x^t + \Lambda_{t-1} x^{t-1} + \ldots + \Lambda_0$, with the roots of $\Lambda(x)$ corresponding to the actual error locations. One efficient algorithm to obtain $\Lambda(x)$ is an iterative algorithm known as Berlekamp's algorithm[3]. The even numbered iterations of the Berlekamp algorithm will be skipped because the codes are binary which yields a total of t iterations for obtaining $\Lambda(x)$ for a t-error-correcting code.

Suppose the computed syndromes are $S_1, S_2, S_3, \ldots, S_{2t-1}$. With the initial conditions $\Lambda^{(0)}(x) = 1$ and $B^{(0)}(x) = 1$, the Berlekamp algorithm for binary BCH codes is:

$$\Delta^{(2k)} = \sum_{j=0}^{n-1} \Lambda_j^{(2k)} S_{k-j}$$

$$\Lambda^{(2k+2)}(x) = \Lambda^{(2k)}(x) + \Delta^{(2k)} x B^{(2k)}(x)$$

$$B^{(2k+2)}(x) = \begin{cases} x^2 B^{(2k)}(x) & \text{if } \Delta^{(2k)} = 0, \text{ or } \deg \Lambda^{(2k)}(x) > k \\ \dfrac{x\Lambda^{(2k)}(x)}{\Delta^{(2k)}} & \text{if } \Delta^{(2k)} \neq 0, \text{ or } \deg \Lambda^{(2k)}(x) \leq k \end{cases}$$

Conceptually, this algorithm can be carried out with four sets of shift registers: one for the syndromes $S_1, S_2, S_3, \ldots, S_{2t-1}$, one for the estimated coefficients of $\Lambda(x)$, one for the coefficients of the intermediate polynomial $B(x)$, and one for the product term $\Lambda_j S_{k-j}$. This design is shown in Figure 5-7.

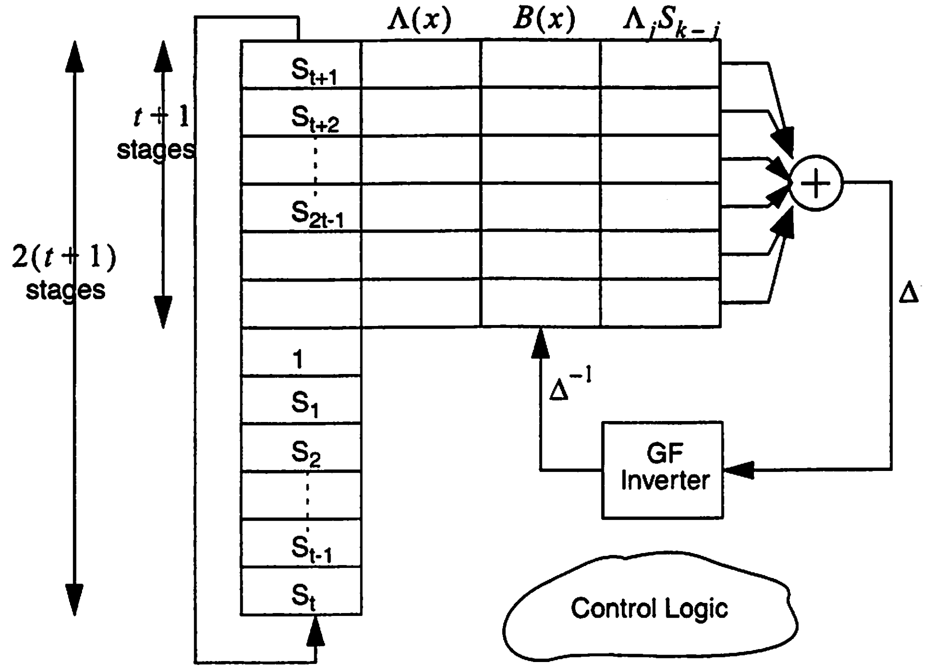


Figure 5-7. Shift register design for Berlekamp's algorithm.

The syndrome register has size $2t$ to store all the syndromes; both $\Lambda(x)$ and $B(x)$ registers have size t since these polynomials have the maximum degree t; and the $\Lambda_j S_{k-j}$ register also has the size t which are used for computing the discrepancy factor Δ. All the register contents are field elements in GF(2^6) and are thus represented by 6 bits. To simplify the overall control logic, we add one extra layer of registers to store initial values. The adder at the right end of the figure is a finite field adder that adds the field elements and the discrepancy Δ is also a finite field element that goes into a finite field inverter. The inverse of Δ is denoted as Δ^{-1} which is used for updating $B(x)$.

As will be shown in Section 5.4.3, this design is highly desirable for VLSI implementation as the updates of Δ, $B(x)$ and $\Lambda(x)$ are all performed in parallel. Furthermore, this parallel structure introduces no extra hardware (except the additional finite field adders and multipliers) and also meets the low power design guideline. The later character is important when decoders are incorporated inside portable units for decoding received downlink data, in which case battery life is usually a crucial implementation constraint.

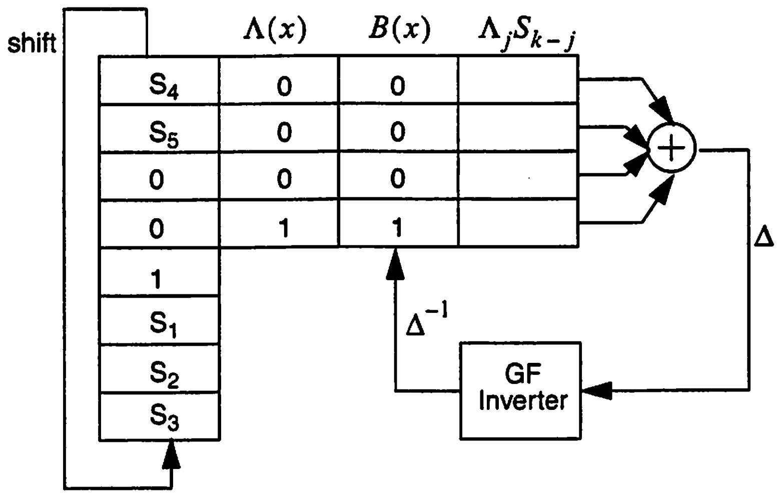
5.4.3.1. An example using this decoding structure

To fully understand how the algorithm operates under this frame work, we will work through a concrete example and show the register contents at each iteration. Since there are no even numbered iterations, we numerate all the non-trivial iterations as 1, 2, 3, 4, ... (instead 1, 3, 5, 7, ...). To illustrate the approach of implementation, a shorter block code is used in our example for the sake of clarity.

Once again we use the (15, 5) BCH code with generator polynomial $g(x) = x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1$; this is a triple error correcting code. Suppose the transmitted bits are all zero, and the received bits are $r(x) = x^7 + x^5 + x^2$. Then three bits errors have occurred at positions 2, 5, 7 (position count starts from 0). The first 6 syndromes are calculated to be: $S_1 = \alpha^{14}$, $S_2 = \alpha^{13}$, $S_3 = 1$, $S_4 = \alpha^{11}$, $S_5 = \alpha^5$, $S_6 = 1$.

To differentiate the updated $\Lambda(x)$ and the current $\Lambda(x)$ (which is needed for computing $B(x)$), an intermediate term called $\Lambda^{\text{old}}(x)$ is introduced. The algorithm is initialized as $\Lambda(x) = 1$ and $B(x) = 1$. The register content at the beginning of the first iteration is shown below:

<u>initial stage</u>:



At the first iteration:

(1) shifting the Syn-register two times:

(2) multiplying Syn-register and Λ-register pair-wise to obtain Δ:
$\Delta^{(1)} = \text{Syn} \cdot \Lambda = S_1 = \alpha^{14}$

Syn	$\Lambda(x)$	$B(x)$	$\Lambda_j S_{k-j}$
0	0	0	0
0	0	0	0
1	0	0	0
S_1	1	1	α^{14}
S_2			
S_3			
S_4			
S_5			

(3) updating $\Lambda(x)$:
$\Lambda^{\text{update}}(x) = \Lambda^{\text{old}}(x) + \Delta^{(1)} x B(x) = 1 + \alpha^{14} x$

(4) updating $B(x)$:
since $\Delta^{(1)} \neq 0$, therefore $B(x) = \dfrac{x\Lambda^{\text{old}}(x)}{\Delta^{(1)}} = \alpha x$

At the second iteration:

(1) shifting the Syn-register two times:

(2) multiplying Syn-register and Λ-register pair-wise:

$$\Delta^{(2)} = \text{Syn} \cdot \Lambda = S_2\alpha^{14} + S_3 = \alpha^{12} + 1 = \alpha^{11}$$

Syn	$\Lambda(x)$	$B(x)$	$\Lambda_j S_{k-j}$
1	0	0	0
S_1	0	0	0
S_2	α^{14}	α	α^{12}
S_3	1	0	1
S_4			
S_5			
0			
0			

(3) updating $\Lambda(x)$:

$$\Lambda^{\text{update}}(x) = \Lambda^{\text{old}}(x) + \Delta^{(2)}xB(x) = 1 + \alpha^{14}x + \alpha^{12}x^2$$

(4) updating $B(x)$:

since $\Delta^{(2)} \neq 0$, therefore: $B(x) = \dfrac{x\Lambda^{\text{old}}(x)}{\Delta^{(2)}} = \alpha^4 x + \alpha^3 x^2$

At the third iteration:

(1) shifting the Syn-register two times:

(2) multiplying Syn-register and Λ-register pair-wise:

$$\Delta^{(3)} = \text{Syn} \cdot \Lambda = S_3\alpha^{12} + S_4\alpha^{14} + S_5$$
$$= \alpha^{12} + \alpha^{10} + \alpha^{5} = \alpha^{11}$$

Syn	$\Lambda(x)$	$B(x)$	$\Lambda_j S_{k-j}$
S_2	0	0	0
S_3	α^{12}	α^3	α^{12}
S_4	α^{14}	α^4	α^{10}
S_5	1	0	α^5
0			
0			
1			
S_1			

(3) updating $\Lambda(x)$:

$$\Lambda^{\text{update}}(x) = \Lambda^{\text{old}}(x) + \Delta^{(3)} x B(x)$$
$$= 1 + \alpha^{14}x + \alpha^{11}x^2 + \alpha^{14}x^3$$

(4) updating $B(x)$:

$$\text{since } \Delta^{(3)} \neq 0, \text{ therefore: } B(x) = \frac{x\Lambda^{\text{old}}(x)}{\Delta^{(3)}} = \alpha^4 x + \alpha^3 x^2 + \alpha x^3$$

Now we have obtained an error locator polynomial $\Lambda(x)$ which has degree 3. This implies that three errors have occurred in the received block. In fact, $\Lambda(x)$ can be factored as $\Lambda(x) = 1 + \alpha^{14}x + \alpha^{11}x^2 + \alpha^{14}x^3 = (1+\alpha^7 x)(1+\alpha^5 x)(1+\alpha^2 x)$, indicating the error positions are 2nd, 5th, and 7th bit, respectively.

5.4.3.2. Data path

The data path for Berlekamp's algorithm of a (63, *k*, *t*) binary BCH code consists of five set of registers: Syndrome, Λ^{updated}, Λ^{old}, $B(x)$, and $\Lambda_j S_{k-j}$ register. The register initialization is shown in Figure 5-8. The mark "X"s are the don't-cares.

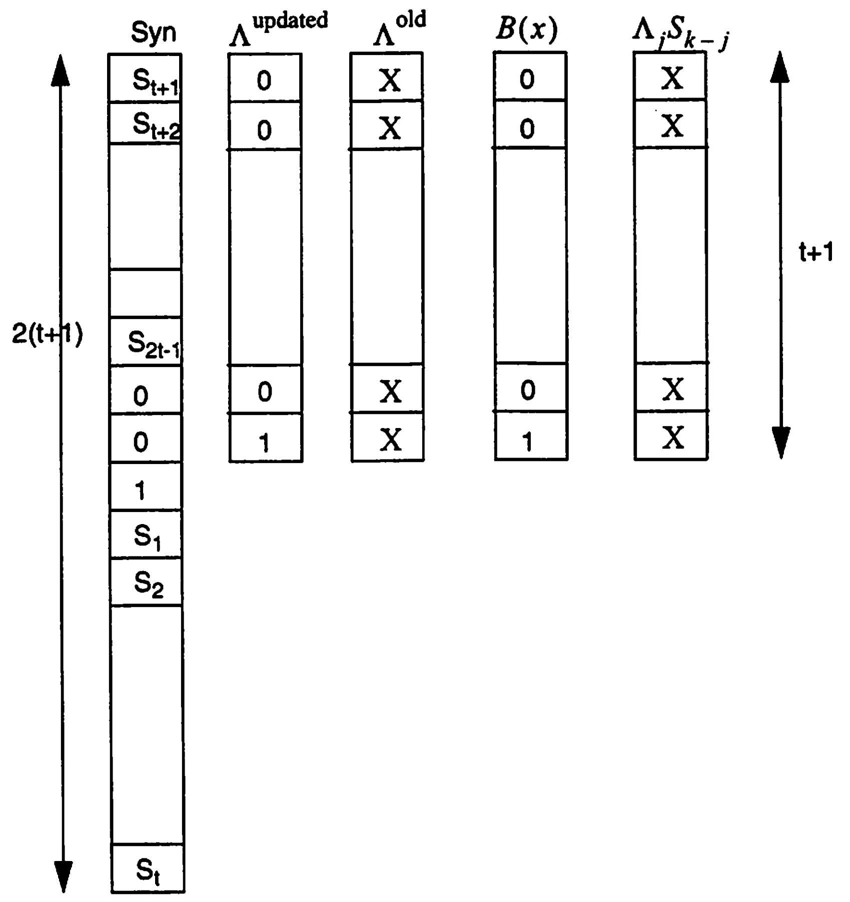


Figure 5-8. Register initialization for Berlekamp's algorithm.

The complete data path for Berlekamp's algorithm is shown in Figure 5-9. The connections between modules are all 6-bit buses, as data passed between modules are elements of GF(2^6). A shift in Syn-register corresponds to a shift of two positions, because all even iterations are skipped. Parallel adders and multipliers are adopted for updating $\Lambda(x)$, $B(x)$, and for computing $\Lambda_j S_{k-j}$, so that these operations can be completed in one clock cycle. Notice the algorithm requires multiplying x with a polynomial when updating $\Lambda(x)$ and

$B(x)$, this is equivalent to a single shift of the shift register content. For example, the ith element of $\Lambda^{\text{updated}}(x)$ is $\Lambda_i^{\text{updated}} = \Delta B_{i-1} + \Lambda_i^{\text{old}}$ where i is the ith element of the corresponding register.

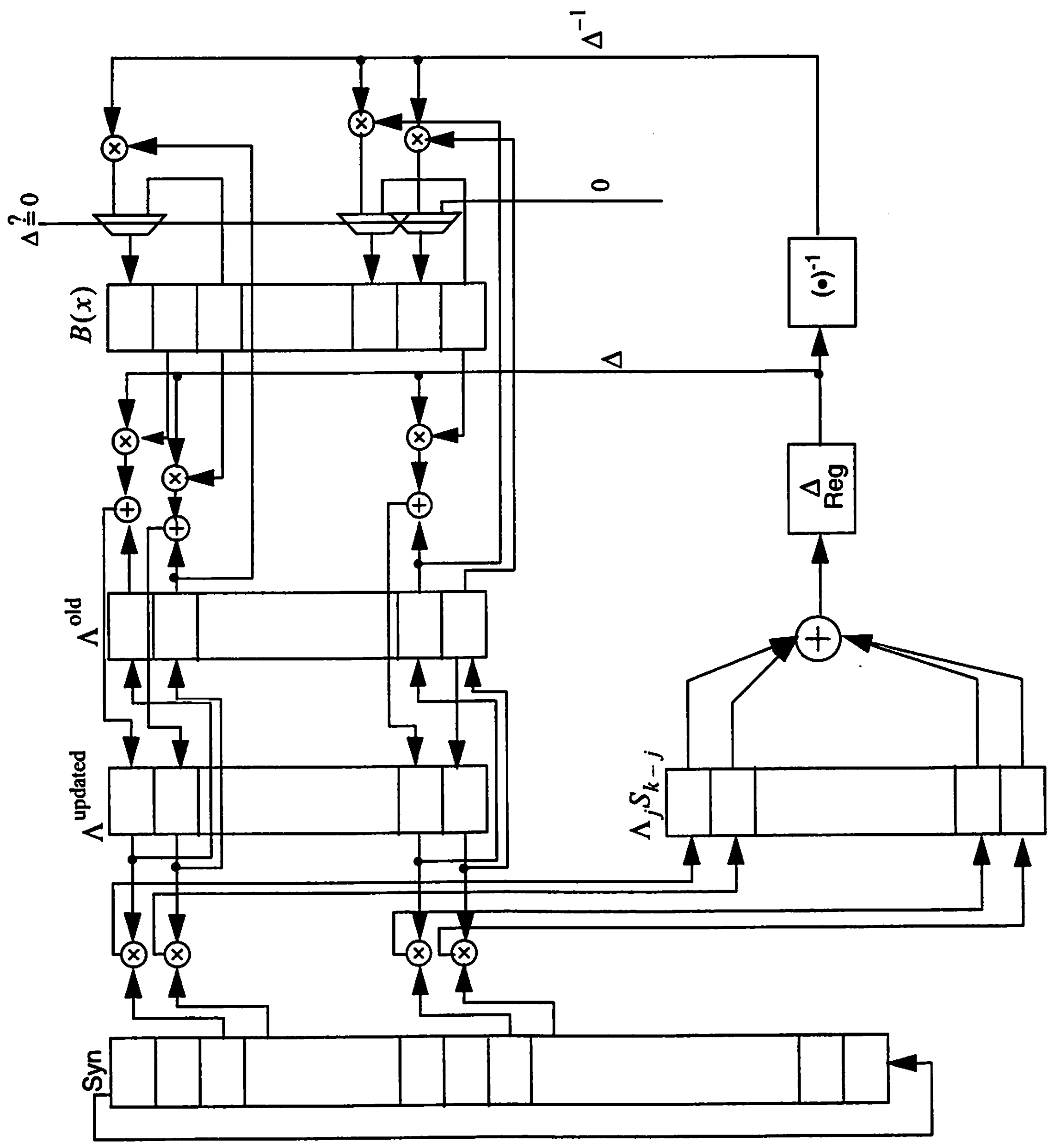


Figure 5-9. Data path for (63, *k*, *t*) Berlekamp's algorithm as a part of the binary BCH decoder.

5.4.3.3. Scheduling

Assume we have a finite field adder that performs t additions in one clock cycle. We also assume the inverse of a finite field element can be obtained in one clock cycle which can be simply implemented as a small 6-bit-input 6-bit-output lookup table (384 bit ROM). A complete iteration of Berlekamp's algorithm can be expressed sequentially as:

- shifting Syn-register by two blocks,
- multiplying each syndrome with coefficients of $\Lambda(x)$, i.e. $\Lambda_i^{\text{updated}} \cdot S_{k-i}$,
- summing the products: $\sum \Lambda_i^{\text{updated}} \cdot S_{k-i} = \Delta$,
- load: register $\Lambda^{\text{updated}} \to$ register Λ^{old},
- updating $\Lambda(x)$: $\Lambda^{\text{updated}}(x) = \Delta x B(x) + \Lambda^{\text{old}}(x)$,
- updating $B(x)$:

$$B(x) = \begin{cases} \dfrac{x \cdot \Lambda^{\text{old}}(x)}{\Delta} & \text{if } \Delta \neq 0 \\ x^2 B(x) & \text{otherwise} \end{cases}$$

Some of the operations can be done in parallel, for example, $\sum \Lambda_i^{\text{updated}} \cdot S_{k-i}$ and shifting Syn-reg, since these two operations are completely independent. Therefore, after incorporating parallelism into our design, each iteration of Berlekamp's algorithm can be completed in three sequential clock cycles:

- shifting Syn-register by two blocks,
  multiplying each syndrome with coefficients of $\Lambda(x)$, i.e. $\Lambda_i^{\text{updated}} \cdot S_{k-i}$,
- summing the products: $\sum \Lambda_i^{\text{updated}} \cdot S_{k-i} = \Delta$,
  load: register $\Lambda^{\text{updated}} \to$ register Λ^{old},
- updating $\Lambda(x)$: $\Lambda^{\text{updated}}(x) = \Delta x B(x) + \Lambda^{\text{old}}(x)$,
  updating $B(x)$:

$$B(x) = \begin{cases} \dfrac{x \cdot \Lambda^{\text{old}}(x)}{\Delta} & \text{if } \Delta \neq 0 \\ x^2 B(x) & \text{otherwise} \end{cases}$$

In addition to parallelism, we can also introduce pipeline to our design to allow either fastest possible operation or to allow lowest possible power[6]. When designing pipeline stages, one has to be particularly careful of *data hazards* in the implementation. These hazards occur when the pipeline changes the order of accesses to operands so that the order differs from the order seen by sequentially executing instructions on an un-pipelined machine. In our case, hazards can occur in situations such as when loading Λ^{old}. Our pipeline consists of two stages, it is shown together with the rest of the scheduling in Figure 5-10. Notice after the three cycles to complete the first iteration, each additional iteration takes two clock cycles. As Berlekamp's algorithm requires a total of t iterations, the algorithm therefore takes $3 + 2(t-1) = 2t + 1$ clock cycles.

clock cycle 1	clock cycle 2	clock cycle 3	clock cycle 4	clock cycle 5	clock cycle 6
Shift Syn-reg $\Lambda_i^{\text{updated}} \cdot S_{k-i}$	$\sum \Lambda_i^{\text{updated}} \cdot S_{k-i}$ load Λ^{old}	$\Lambda^{\text{updated}}(x) = \Delta x B(x) + \Lambda^{\text{old}}(x)$ $B(x) = \begin{cases} \frac{x \cdot \Lambda^{\text{old}}(x)}{\Delta} \\ x^2 B(x) \end{cases}$			
		Shift Syn-reg $\Lambda_i^{\text{updated}} \cdot S_{k-i}$	$\sum \Lambda_i^{\text{updated}} \cdot S_{k-i}$ load Λ^{old}	$\Lambda^{\text{updated}}(x) = \Delta x B(x) + \Lambda^{\text{old}}(x)$ $B(x) = \begin{cases} \frac{x \cdot \Lambda^{\text{old}}(x)}{\Delta} \\ x^2 B(x) \end{cases}$	
				Shift Syn-reg $\Lambda_i^{\text{updated}} \cdot S_{k-i}$	$\sum \Lambda_i^{\text{updated}} \cdot S_{k-i}$ load Λ^{old}

Figure 5-10. Scheduling of operating cycles.

5.4.4. Chien's search

The last stage of the decoding is to determine the roots of the error locator polynomial $\Lambda(x)$, where these roots correspond the actual error locations. Since only a finite number of field elements have to be checked, the most straight forward way to compute these roots is by trial and error, a method known as Chien's search. The procedure for evaluating $\Lambda(x)$ at $1, \alpha, \alpha^2, \alpha^3, \ldots, \alpha^{62}$ is similar to the one used for computing syndromes, except the polynomial coefficients are now field elements Λ_i (instead of 0 or 1 of received bits). The following example illustrates the concept for computing $\Lambda(\alpha^3)$.

Expressing the field element Λ_i as $\Lambda_i = a_0 + a_1\alpha + a_2\alpha^2 + a_3\alpha^3 + a_4\alpha^4 + a_5\alpha^5$, where α is the primitive field element, thus $\alpha^6 + \alpha + 1 = 0$. Then $\Lambda_i\alpha^3 = a_3 + (a_3 + a_4)\alpha + (a_4 + a_5)\alpha^2 + (a_0 + a_5)\alpha^3 + a_1\alpha^4 + a_2\alpha^5$. This can be mapped into the circuit shown in Figure 5-11. The $\Lambda(\alpha^3)$ is obtained after t shifts.

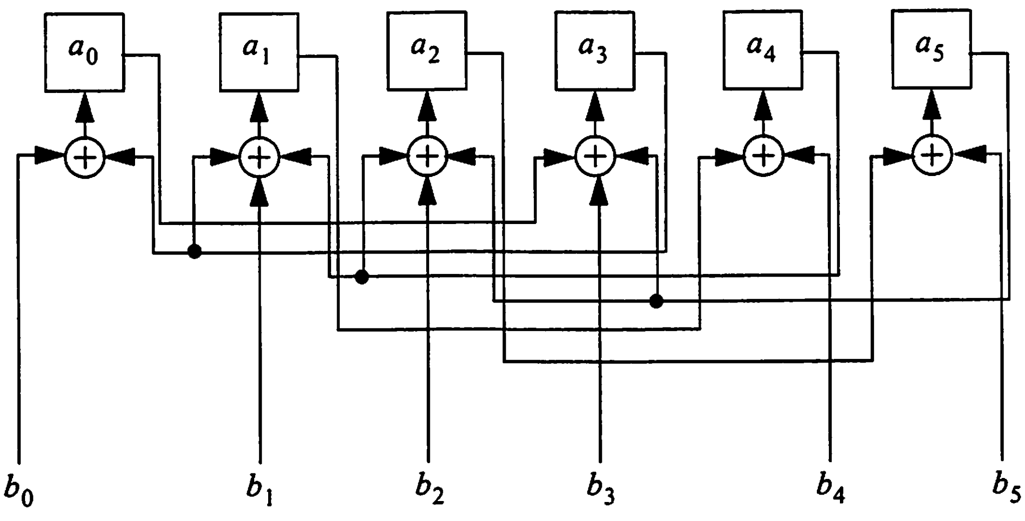


Figure 5-11. Circuit for evaluating polynomial $\Lambda(x)$ at α^3 over GF(2^6).

The polynomial evaluation $\Lambda(\alpha^i) \overset{?}{=} 0$ for $i = 0, 1, 2, \ldots, k-1$ can be done in parallel with k circuits. Since $\Lambda(x)$ has degree t, each evaluation takes t clock cycles to complete. In order to synchronize this module with the serial output of the decoder, each circuit starts one clock cycle apart. After the initial t cycles for correcting the first information bit, each additional cycle produces one corrected bit. Therefore, it takes a total of $t+k-1$ cycles to complete the Chien's search and output the data.

5.4.5. Control Logic

After the initial $2t$ cycles to compute the syndromes, only Berlekamp's algorithm and Chien's search contribute to the required clock cycles for decoding one block of data. With our pipeline approach, Berlekamp's algorithm takes $2t+1$ clock cycles, and Chien's search combined with serial output takes $t+k-1$ cycles. Adding them together, we need $3t+k$ cycles to decode a single block of data. For all 63-bit binary BCH codes, $3t+k < 63$, which means that decoding can be done in one block period. Consequently, only one register is needed for buffering the data while decoding; this register is shown as R2 in Figure 5-12, and it has size k to store the information bits.

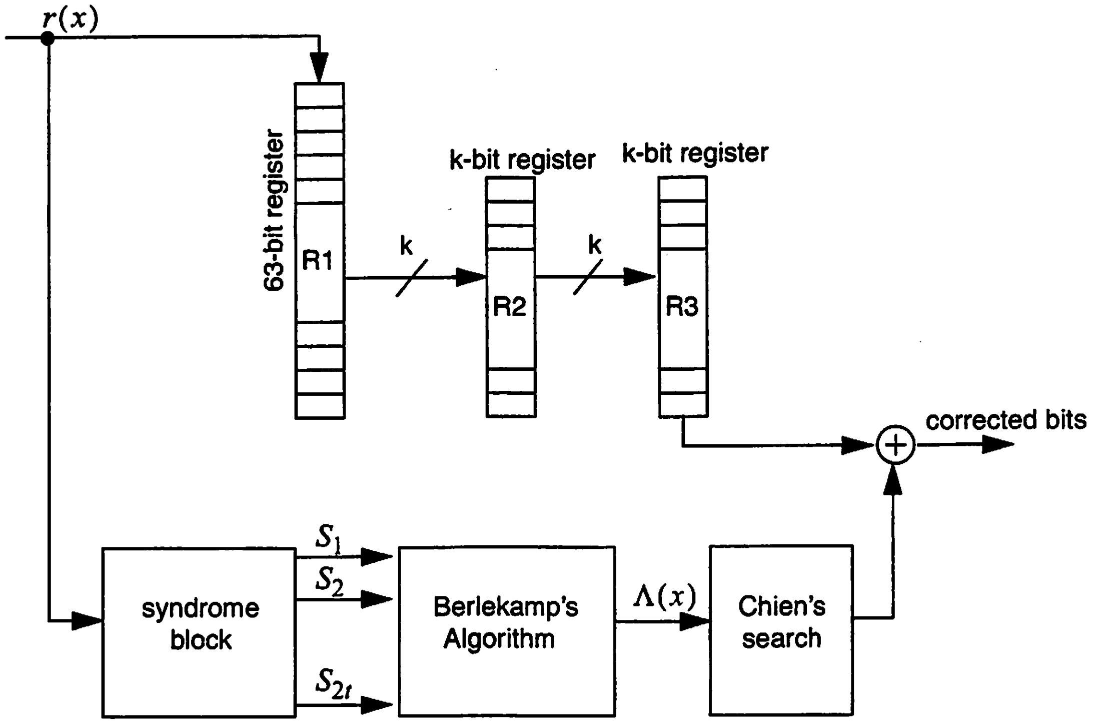


Figure 5-12. High level schematics for a BCH decoder.

The control logic is implemented using a finite state machine. Three counters are employed: 63-count, $2t + 1$ -count, and 2-count. The 63-count is the main controller of the decoding algorithm, and both $2t + 1$ -count and 2-count will be used during the Berlekamp's algorithm. All counters operates at the data rate. Using the same notations from of Figure 5-12, the finite state machine is implemented as follows:

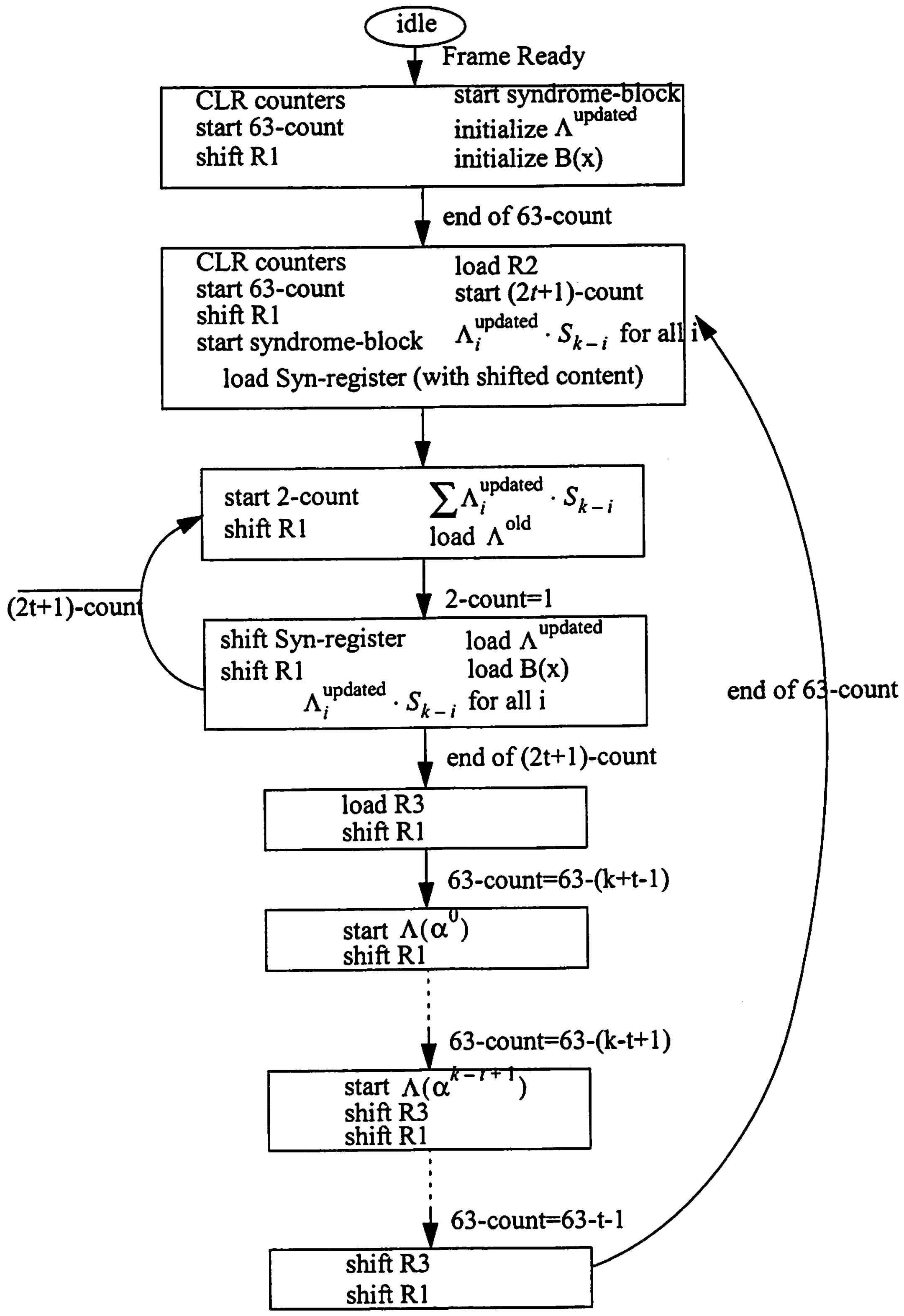


Figure 5-13. Control diagram for the decoding algorithm.

5.5. Variable FEC decoder implementation

An architecture for a single $(63, k, t)$ binary BCH decoder has now been presented. The design consists of a set of registers with their sizes proportional to the error correction capability t. This implies that if a decoder is designed to correct the largest possible number of errors within the 63-bit family BCH codes ($t_{\max} = 15$), then the majority of the hardware can be used again as a decoder for correcting fewer errors. With this observation, the architecture of the VFEC decoder is therefore based on (63, 7) $t = 15$ decoder design.

The high level schematics for VFEC decoder is shown in Figure 5-14. The FEC code type is input into the decoder and stored in the FEC-type register. This register output is connected to a lookup table which outputs the number of information bits and the number of correctable errors of this FEC code, i.e. k and t; these values are used as a part of the finite state machine for controlling the counters. The input register, R1, is still 63 bits long. Both information registers, R2 and R3 store the largest number of information bits among all codes, so they each has size 57 bits.

In the arithmetic portion of the decoder, the syndrome block always compute the first $2t_{\max} = 30$ syndromes, and the results are input into the Syn-register. Inside the error locator module, the registers Λ^{old}, $\Lambda^{\text{updataed}}$, $B(x)$, and $\Lambda_j S_{k-j}$ all have the size $t_{\max} + 1 = 16$ symbols, where each symbol is represented by 6 bits. Finally in Chien searcher, $k_{\max} = 57$ circuits are used to perform polynomial evaluations. One possible method to reduce the size of Chien searcher is to eliminate the case of single error correction that takes up 57 circuits; this may be done through a more sophisticated control algorithm that corrects the single error. In that case, the Chien searcher is reduced to $k = 51$ circuits for double error correction.

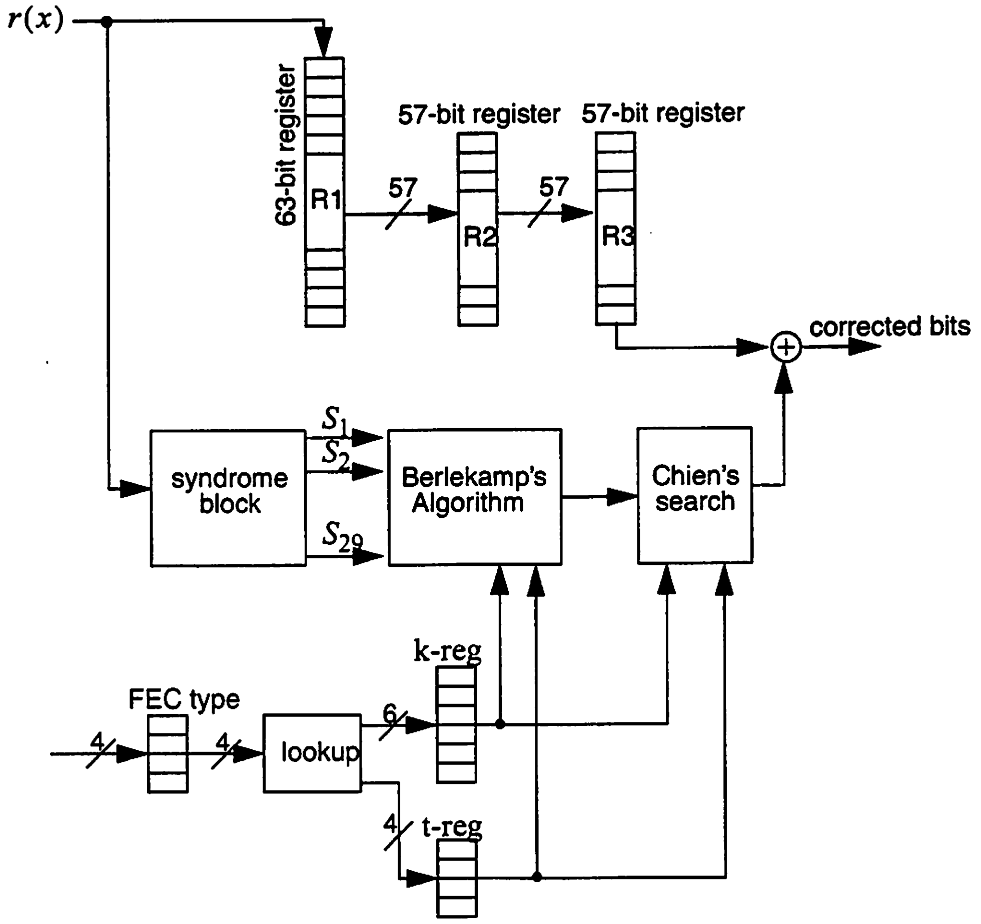


Figure 5-14. High level schematics for the VFEC decoder.

As far as the overall implementation complexity, all the multipliers shown in the datapath are finite field multipliers, and adders are finite field adders. Since codes are generated by GF(2^6), each multiplier can be implemented using a shift register of 6 bits. Each syndrome is also computed using a shift register of 6 bits. A summary of total hardware required by the VFEC decoder is shown below:

data registers	2 4-bit reg
	1 6-bit reg
	2 57-bit reg
	1 63-bit reg
finite field adders	15 two-input adder
	1 15-input adder
6-bit shift registers	30 for syndrome
	45 for multiplying
ROM (lookup table)	2

The scheduling and the control logic for VFEC is the same as the single decoder case, except the values for k and t used in 63-count are now read from the k-reg and t-reg. However, if one prefers simple control, we can apply the maximum Berlekamp's algorithm ($t_{max} = 15$) and the maximum Chien's search ($k_{max} = 57$) for all received data, disregarding the coding type. With this approach, more pipeline stages are needed to buffer the input data while decoding takes place. This approach obviously takes more hardware, the control logic, however, is indeed simplified. Therefore, the final decision on which approach will be used is made by the hardware designer, after considering the trade-off between area and the implementation complexity.

5.6. Conclusions

This chapter presents an architecture for a VFEC decoder. The VFEC is achieved by employing the family of 63-bit BCH codes which consists of 11 non-trivial error correction codes. These codes all have the same block length and are generated by the same finite field, GF(2^6). The difference between them is the wide range of coding rate. The decision for adopting this approach for VFEC is made by exploiting the redundancies shared by decoders. With the syndrome module, Berlekamp's algorithm, and Chien's search in common for all codes, our VFEC decoder is only slightly more complicated than a single decoder design.

6 Summary and Future Work

6.1. Summary

This thesis is devoted to issues directly related to network resource management for multimedia wireless networks. The goal of the resource allocation is to maximize the overall user satisfaction, which we call "utility". The major advantage of this proposed framework is that it is platform independent; in other words, it does not assume any prior knowledge on specifics such as multiple access scheme, traffic sources, user demands, or user distribution. These implementation details are introduced into this framework as a set of constraints, expressed in bandwidth, transmit power, and interference models.

Chapter 2 focused on maximizing system utility for a downlink DS CDMA system. Two key results are derived: first of all, we presented a framework which integrates power control, variable forward error correction, and scheduling; second, the algorithm is proved to be fully distributed with complexity independent of the number of cells in the system; in other words, this algorithm is scalable. The algorithm complexity depends on the number of interfering cells and the number of users reside in a cell. The first result is important as it explores the design space of the available network resources. The second result is important as it links the theoretical optimum with the implementation practicality, since any large wireless network has to be scalable to be practical. In the second half of Chapter 2, we presented a different approach to this maximization problem: using congestion pricing, which is a concept borrowed from economics. The basic idea behind the congestion pricing is setting the price for each user as an approximation of the total marginal cost for the rest of the system users; this marginal cost measures the congestion level of the system, so the price is zero if a user introduces no congestion to the system. In wireless communications, we measure congestion by signal to noise ratio. If increasing one user's power level introduces very little interference to rest of the users, then there is a strong incentive to improve this user's received signal quality by increasing the power level. On the other hand, if other users experience a lot of interference as a result of increasing someone's power, then this user should not increasing her transmit power from an overall system's perspective.

The flexibility of the utility approach also makes the evaluation of this framework difficult. Therefore, in Chapter 3, we assumed a particular form of the utility function for simulation purpose. The system performance is studied in detail as a function of the parameters such as: user population, user distribution, hand-off techniques, power control fairness, and FEC coding types. In addition, we also compared the effectiveness of the power control algorithm with variable forward error correction, with result indicating variable forward error correction is completely superior than power control. This is because variable error correction has the ability to quickly adapt to the highly varying multimedia traffic, thus better utilizing the network resources. In the study of hand-off, we concluded the system performance can be improved significantly when the "hot spot" is alleviated by letting a user communicate with his closest neighbor basestation.

The last two chapters of the thesis is focused on the implementation aspect of the proposed system. Since power control and scheduling are often not very performance critical, they are normally implemented in software. The variable error correction, on the other hand, operates at the bit rate thus requires high performance custom IC or special hardwares. The goal of Chapter 4 is to provide readers with sufficient background on coding theory, with emphasis on block codes. Chapter 5 presented a custom architecture for variable forward error correction decoder based on the 63-bit family BCH codes. The core of this design is an iterative algorithm called Berlekamp's algorithm. After sufficient pipelining, the decoding of a data block can be completed in one block period.

6.2. Future Work

6.2.1. Introducing delay as a part of the utility function

Delay is often a very important measure of the application quality, but it has not been considered in our formulation of the application utility. If delay becomes the third parameter of the application utility function (besides bandwidth and BER), it is most likely that it will either be a step function (Figure 6-1(a)) – indicating the data is valueless if its delay is beyond a certain threshold (d_T), or delay as a function shown in Figure 6-1(b) – indicating a gradual decrease in user satisfaction as delay increases. In any case, utility with respect to delay is a monotone non-increasing function. The addition of delay parameter to the framework will nevertheless influence the scheduling decision, the FEC code selection, and the power control output.

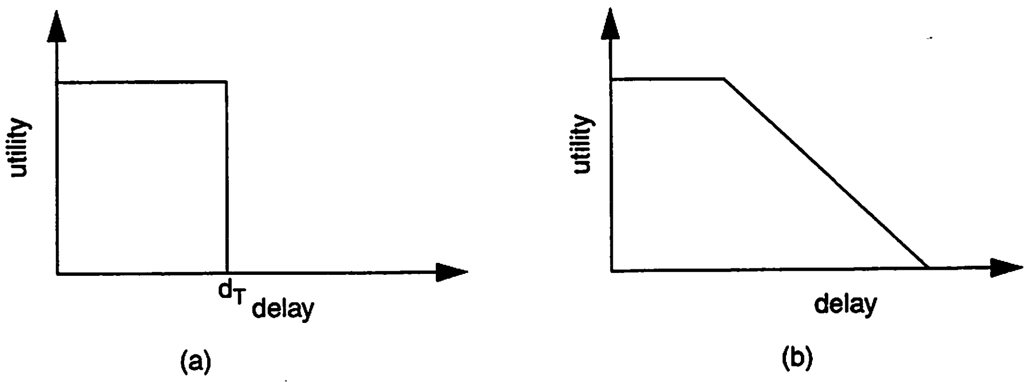


Figure 6-1. Application utility as a function of delay.

6.2.2. Introducing ARQ as an additional control variable

Our resource allocation framework mostly operates at the physical layer and link layer. The system reliability may be further improved by considering a higher layer protocol such as automatic repeat-request (ARQ). The basic idea behind ARQ is that at the receiver, if erroneous coded data are detected, the received word is discarded and the receiver requests a retransmission though some retransmission protocols. There are three basic types of ARQ protocols: Stop-and-Wait, Go-Back-N, and Selective-Repeat [53][28][44]. The main drawbacks for an ARQ scheme are the retransmission delay, bandwidth redundancy, and the additional control logic. However, wireless channels sometimes have long fade in which case the entire packet is corrupted. Since FEC has a limited error correcting capability, ARQ may be necessary for some high quality data as it achieves a greater time diversity.

6.2.3. Joint source and channel coding

As we have briefly mentioned in Chapter 2, a system should be designed to gracefully degrade its quality of service (instead of dropping packets) when there is excessive interference. This can be achieved through adopting a layered coding algorithm. In this case, data are compressed and separated into multiple hierarchies; during a transmission, the layer with the highest priority is always first transmitted. Similar ideas have been adopted for Mbone multicast video [31]. In this example, a receiver-oriented layered transmission system is combined with a layered compression algorithm for heterogeneous transmission of video data. As a result, the scheme adapts to variable bandwidth constraints imposed by different networks. In addition to layered coding, a system may also benefit from tightly coupled source and channel coding algorithm. This algorithm trades off data compression rate with channel coding rate, and is worth further research for understanding the true capacity gain.

References

[1] J. Acton and B. Mitchell, "Evaluating Time-of-day electricity rates for residential customers," in B. Mitchell and P. Kleindorfer, *Regulated industries and public enterprise*, Lexington Books, p 248, 1980.

[2] N. Bambos and G. J. Pottie, "Power control based admission policies in cellular radio networks," *Proceedings of IEEE Global Telecommunication Conference, GLOBECOM-92*, 1992, pp863-867.

[3] E. Berlekamp, *Algebraic Coding Theory*, Aegean Park Press, 1984.

[4] G. Birkhoff and S. MacLane, *A Survey of Modern Algebra*, The Macmillan Co., 1941.

[5] R. E. Blahut, *Theory and Practice of Error Control Codes*, Addison-Wesley, 1983.

[6] A. Chandrakasan and R. Brodersen, *Low Power Digital CMOS Design*, Kluwer Academic Publishers, 1995.

[7] S. C. Chen, N. Bambos, and G. J. Pottie, "On distributed power control for radio networks," *Proceedings of International Conference on Communications, ICC'94*, 1994.

[8] A. Cohen, Rate of Convergence for Root Finding and Optimization Algorithms, Ph.D. Dissertation, University of California at Berkeley, 1970.

[9] E. T. Cohen, "On the implementation of Reed-Solomon decoders", Ph.D. Dissertation, University of California, Berkeley, 1983.

[10]D. C. Cox, "Universal Digital Portable Radio Communications", *Proceedings of the IEEE*, Vol. 75, No. 4, pp. 436-477, April 1987.

[11]R. Edell, N. McKeown and P. Varaiya, "Billing users and pricing for TCP," *IEEE Journal on Selected Areas in Communications*, September 1995.

[12]M. Eyuboglu and S. Qureshi, "RSSE with Set Partitioning and Decision Feedback," *IEEE Transactions on Communications*, pp. 13-20, Jan. 1988.

[13]D. Falconer and F. Magee, “Adaptive Channel Memory Truncation for Maximum-Likelihood Sequence Estimation,” *Bell System Technical Journal*, vol. 52, pp. 1541-1562, Nov. 1973.

[14]G. David Forney, “The Viterbi algorithm,” *Proceedings of the IEEE*, pp. 268-278, March 1973.

[15]G. J. Foschini and Z. Milzanic, “A simple distributed autonomous power control algorithm and its convergence,” *IEEE Transactions on Vehicular Technology*, pp. 641-646, November 1993.

[16]R. G. Gallager, *Information Theory and Reliable Communication*, Joh Wiley, 1968.

[17]K. Gilhousen *et. al.*, “On the capacity of a cellular CDMA system,” *IEEE Transactions on Vehicular Technology*, pp. 303-312, May 1991.

[18]A. Goldsmith, “Design and performance of high-speed communication systems over time-varying radio channels,” Ph.D. Dissertation, University of California, Berkeley, 1994.

[19]S. A. Grandhi, R. Vijayan, D. J. Goodman, J. Zander, “Centralized power control in cellular radio systems,” *IEEE Transactions on Vehicular Technology*, pp 466-468, November, 1993.

[20]S. A. Grandhi, R. Vijayan, D. J. Goodman, J. Zander, “Distributed power control in cellular radio systems,” *IEEE Transactions on Communications*, vol 42, pp. 226-228, February-April, 19994.

[21]L. Guerra, M. Potkonjak, J. Rabaey, “System-Level Design Guidance Using Algorithm Properties,” *VLSI Signal Processing VII*, IEEE Press, pp. 73 - 82, 1994.

[22]R. Han, “Asymptotically reliable transport of multimedia/graphics over wireless channels,” *Proc. SPIE/IS&T Multimedia Computing and Networking*, Jan.28-Feb.2, San Jose, 1996.

[23]J. Hagenauer, “Rate-compatible punctured convolutional codes (RCPC codes) and their applications,” *IEEE Transactions on Communications*, pp. 380-400, September 1981.

[24]S. V. Hanly, “An algorithm for combined cell-site selection and power control to maximize cellular spread spectrum capacity”, *IEEE Journal on Selected Areas in Communications*, pp. 1332-1340, September 1995.

[25]M. D. Huang, F. Romeo, A. Sangiovanni-Vincentelli, “An efficient general cooling schedule for simulated annealing,” pp. 381-384, *IEEE ICCAD*, 1986.

[26]W. C. Jakes, Jr. Ed., *Microwave Mobile Communications*, IEEE Press, 1993.

[27]S. Lin and D. J. Costello, Jr., *Error Control Coding: Fundamentals and Applications*, Prentice-Hall, Inc., 1983.

[28]S. Lin, D. J. Costello, Jr., and M. J. Miller, "Automatic-Repeat-Request error-control schemes," *IEEE Communications Magazine*, pp5-17, December 1984.

[29]Y. Lu and R. Brodersen, "Unified power control, error correction coding and scheduling for a CDMA downlink system," ACM/Baltzer Journal on Wireless Networks, pp.83-90, 1997.

[30]J. L. Massy, "Shift-Register Synthesis and BCH Decoding," *IEEE Transactions on Information Theory*, pp 122-127, 1969.

[31]S. McCanne, S. Jacobson, and M. Vetterli, "Receiver-driven Layered Multicast," *Proceedings of ACM SIGCOMM*, August 1996, Stanford, CA, pp. 117-130.

[32]M. J. Miller and S. Lin, "The analysis of some selective-repeat ARQ schemes with finite receiver buffer," *IEEE Transactions on Communications*, pp. 1307-1315, September 1981.

[33]S. Nahar, S. Sahni, E. Shragowitz, "Simulated annealing and combinatorial optimization," pp. 293-299, *IEEE Design Automation Conference*, 1986.

[34]R. W. Nettleton and H. Alavi, "Power control for spread spectrum celluar mobile radio system," in *Proc. IEEE Vehicular Technology Conference*, pp242-246, 1983.

[35]W. W. Peterson, "Encoding and Error-Correction Procedures for the Bose-Chaudhuri Codes," *IEEE Transactions on Information Theory*, pp 459-470, September 1960.

[36]W. W. Peterson and E. J. Weldon, Jr., *Error-Correcting Codes*, the MIT press, 1972.

[37]E. Polak, *Computational Methods in Optimization*, Academic Press, 1971.

[38]J. G. Proakis, *Digital Communications*, NY: McGraw-Hill, 1989.

[39]J. Rabaey, L. Guerra, R. Mehra "Design Guidance in the Power Dimension," *Int'l. Conference on Acoustics, Speech, and Signal Processing*, 1995.

[40]M. K. Simon, J, K. Omura, R. A. Scholtz, and B. K. Levitt, *Spread Spectrum Communication, Volume I.* Rockville, MD: Computer Science, 1985.

[41]C. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, Vol. 27, pp. 379-423, 1948.

[42]Y. Shayan, T. Le-Ngoc, and V. Bhargava, "A versatile time-domain Reed-Solomon decoder," *IEEE Journal on Selected Areas in Communications*, pp. 1535-1542, October 1990.

[43]S. Sheng, A. Chandrakasan, R. W. Brodersen. "A Portable Multimedia Terminal," *IEEE Communications Magazine*, pp. 64-75, vol. 30, no. 12, Dec, 1992.

[44]A. S. Tanenbaum, *Computer Networks*, Second Edition, Prentice Hall, 1988.

[45]D. Taubman and A. Zakhor, "Multirate 3-D subband coding of video," *IEEE Transactions on Image Processing,* Vol. 3, No. 5, pp. 572-588, September 1994.

[46]S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, *Numerical Recipes in C*, pp. 444-455, Cambridge university Press, 1995.

[47]G. L. Turin, "The effects of multipath and fading on the performance of direct-sequence CDMA systems," *IEEE Journal on Selected Areas in Communications*, pp. 597-603, July 1984.

[48]G. L. Turin et al., "A statistical model of urban multipath propagation," *IEEE Transactions on Vehicular Technology*, January 1972.

[49]G. Ungerboeck, "Trellis-coded modulation with redundant signal sets, part II: state-of the art," *IEEE Communication Magazine*, Vol. 25, No. 2, pp. 12-21, February 1987.

[50]A. J. Viterbi, *Principle of Spread Spectrum Communication,* Addison-Wesley, 1995.

[51]A. J. Viterbi and J. K. Omura, *Principles of Digital Communication and Coding*, McGraw-Hill Book Company, 1979.

[52]A. J. Viterbi, "Convolutional codes and their performance in communication systems," *IEEE Transactions on Communications*, pp. 751-772, October 1971.

[53]J. Walrand and P. Varaiya, *High-Performance Communication Networks*, Morgan Kaufmann, 1996.

[54]E. J. Weldon, Jr., "An improved selective-repeat ARQ strategy," *IEEE Transactions on Communications,* pp. 480-486, March 1982.

[55]R. D. Yates, "A framework for uplink power control in cellular radio systems", *IEEE Journal on Selected Areas in Communications*, pp. 1341-1347, September 1995.

[56]R. D. Yates and C. Y. Huang, "Integrated power control and base station assignment," *IEEE Transactions on Vehicular Technology*, pp. 638-644, August 1995.

[57]Louis Yun, *Transport for Multimedia on Wireless Networks*, Ph.D thesis, University of California at Berkeley, 1995.

[58]Z. Zander, "Performance of optimum transmitter power control in cellular radio systems," *IEEE Transactions on Vehicular Technology*, February 1992.

[59]H. Zhang, *Service Disciplines for Packet-Switching Integrated-Services Networks*, Ph.D. dissertation, Electrical Engineering and Computer Sciences, University of California at Berkeley, 1993.