TC.
YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

SONLU CİSİMLER ÜZERİNDE AYRIK FOURİER DÖNÜŞÜMÜ VE CEBİRSEL KODLAMA TEORİSİNDEKİ BAZI UYGULAMALARI

SULTAN SELÇUK

YÜKSEK LİSANS TEZİ
MATEMATİK ANABİLİM DALI
MATEMATİK PROGRAMI

DANIŞMAN
YRD. DOÇ. DR. AYTEN ÖZKAN

İSTANBUL, 2011

YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

SONLU CİSİMLER ÜZERİNDE AYRIK FOURİER DÖNÜŞÜMÜ VE CEBİRSEL KODLAMA TEORİSİNDEKİ BAZI UYGULAMALARI

Sultan SELÇUK tarafından hazırlanan tez çalışması 01.07.2011 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Matematik Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Tez Danışmanı
Yrd. Doç. Dr. Ayten ÖZKAN
Yıldız Teknik Üniversitesi

Jüri Üyeleri
Yrd. Doç. Dr. Ayten ÖZKAN
Yıldız Teknik Üniversitesi _____

Prof. Dr. A. Göksel Ağargün
Yıldız Teknik Üniversitesi _____

Doç. Dr. Ünsal Tekir
Marmara Üniversitesi _____

ÖNSÖZ

Bu çalışma temelleri matematiğe dayanan, bilimin birçok alanında aktif olarak kullanılan Fourier dönüşümü ve bunun kodlama teorisinde karşımıza çıkan bazı uygulamalarını incelemeye yönelik bir çalışmadır.

Öncelikle çalışmam boyunca desteği ve yardımları için danışman hocam Yrd. Doç. Dr. Ayten Özkan'a teşekkürlerimi sunarım. Ayrıca bu konu ile tanışmamı sağlayan Prof. Dr. Çetin Kaya Koç'a teşekkürü bir borç bilirim.

Son olarak, tüm eğitim hayatım boyunca bana büyük sabır, sevgi ve ilgi gösteren aileme ve destekleri ile yanımda olan çok değerli arkadaşım Tolga Sütlü'ye sonsuz teşekkürler ederim.

Mayıs, 2011

Sultan SELÇUK

İÇİNDEKİLER

Sayfa

SİMGE LİSTESİ

$\mathrm{GF(q)}$	q elemanlı Galois cismi
$\mathrm{F_q}$	q elemanlı sonlu cisim
$\mathrm{GF(q)^*}$	Sonlu cismin sıfırdan farklı elemanlarının oluşturduğu küme
$GF(q^m)$	Genişleme cismi
$F^{(m)}$	Genişleme cismi
F^n	n boyutlu vektör uzayı
ω	Cismin mertebesi n olan bir elemanı
$\boldsymbol{v}$	Sonlu cismin n uzunluklu bir vektörü
$\boldsymbol{V}$	$\boldsymbol{v}$ vektörünün Fourier dönüşümü
V_j	$j = 0,1, \dots, n-1$ olmak üzere $\boldsymbol{V}$ vektörünün bileşenleri
v_i	$i = 0,1, \dots, n-1$ olmak üzere $\boldsymbol{V}$'nin ters Fourier dönüşümünün bileşenleri
$((b))$	b'nin modulo p için değeri
$\Lambda(x)$	Bağlantı polinomu
$(\Lambda(x), L)$	L uzunluklu lineer yineleme
$\mathcal{L}(\mathbf{V})$	$\mathbf{V}$ vektörünün lineer kompleksitesi
Λ	Bileşenleri $\Lambda(x)$'in katsayıları olan vektör
λ	Λ vektörünün ters Fourier dönüşümü
$tr(\beta)$	β elemanının iz (trace) fonksiyonu
(n,k) blok kodu $\mathcal{C}$	n uzunluklu k boyutlu bir kod
$d_{min} = d$	(n,k,d) kodu için minimum uzaklık
G	Üreteç matrisi
H	Kontrol matrisi
$\mathcal{C}^{\perp}$	$\mathcal{C}$ kodunun ortogonal tümleyeni
$\mathcal{A}$	Bir kodun tanım kümesi
$\mathcal{A}_c$	Tam tanım kümesi
$c(x)$	Kodsöz polinomu
$C(x)$	Spektrum polinomu
$g(x)$	Üreteç polinomu
$w(x)$	İdempotent polinom
$h(x)$	Kontrol polinomu
$h(x)$	Veri polinomu
$E(x)$	Hata spektrum polinomu

S	Sendrom vektörü
$S(x)$	Sendrom polinomu
f_{β}	β elemanının minimal polinomu
c_+ ve c_-	Genişletilmiş Reed-Solomon koda eklenen bileşenler
$\Lambda(x)$	Kodlama ve dekodlama algoritmalarında hata yeri bulan polinom
M_{μ}	Sendromlar matrisi

KISALTMA LİSTESİ

GF	Galois Cismi (Galois Field)
DFT	Ayrık Fourier Dönüşümü (Discrete Fourier Transform)
BCH	BCH Kodu (Bose-Chaudhuri-Hochquenghem Kodu)

ŞEKİL LİSTESİ

Sayfa

ÇİZELGE LİSTESİ

ÖZET

SONLU CİSİMLER ÜZERİNDE AYRIK FOURİER DÖNÜŞÜMÜ VE CEBİRSEL KODLAMA TEORİSİNDEKİ BAZI UYGULAMALARI

Sultan SELÇUK

Matematik Anabilim Dalı

Yüksek Lisans Tezi

Tez Danışmanı: Yrd. Doç. Dr. Ayten ÖZKAN

Fourier dönüşümü, kompleks fonksiyonlar teorisinde Fourier analizi konu başlığı altında incelenen, bilimin ve mühendisliğin çeşitli alanlarında karşımıza çıkan matematikçiler için ilginç bir yöntem, mühendisler için kullanışlı bir araç olan çok yönlü bir kavramdır. Matematikte daha çok kompleks sayılar cisminde Fourier dönüşümü incelense de, bu çalışmada sadece sonlu cisimler üzereki dönüşümleri incelendi. Bu amaçla öncelikle sonlu cisimler, cisim genişlemeleri ve bunlar için örnekler verildi. Fourier dönüşümünün tanımıyla birlikte matris gösterimi, tez içinde tanım ve teoremlerde kullanacağımız özellikleri ve genellikle mühendislerin aşina oldukları fakat matematiksel olarak da çok anlamlı olan lineer yineleme ve lineer kompleksite kavramları incelendi.

Fourier dönüşümün kullanım alanlarından en önemlilerinden biri olan hata düzelten kodlar içinde devirli kodlar ve bunlara ait özellikler üzerinde duruldu. Bir devirli kodun Fourier dönüşümü sonucunda elde edilen frekans bölgesindeki spektral olarak ne anlam ifade ettiği incelendi. Devirli kodların klasik tanımının dışında, belirli spektral bileşenleri sıfıra eşit olan kodlar olarak tanımlanabileceği ve birçok kodun dekodlamasını da spektral olarak tanımlanabileceği gösterildi.

Devirli kodlar içinde özel olarak, günümüzde iletişim sistemlerinde, kompakt disklerde ve dijital video işlemlerinde en çok kullanılan Reed-Solomon kodları ve bunların bir

altcisim-alt kodu olarak BCH kodları araştırıldı. Bu kodların spektral tanımları ve bazı kodlama örnekleri incelendi. Ayrıca yine Reed-Solomon ve BCH kodlarının bir veya iki bileşen eklenerek genişletilmiş formlarının oluşturulması gösterildi.

Reed-Solomon kodlar için kullanılan algoritmaların en kullanışlı olanlarının Fourier dönüşümüne dayandığı görülmüştür. Bunlar hata yerini bulan dekodlama sınıfına girip cebirsel olarak önemlidirler. Bu algoritmalardan bazılarını tanıtmak amacıyla polinom gösterimleri, sendromlar ve bunların spektral tanımları incelendi. Hatanın yerini ve değerini cebirsel metodlar kullanarak bulmaya yarayan Peterson-Gorenstein-Zierler dekodlayıcısı ve Sugiyama algoritması ayrıntılarıyla incelenmiştir.

Anahtar Kelimeler: Fourier dönüşümü, hata düzelten kodlar, devirli kodlar, Reed-Solomon kodları, BCH kodları, hata yeri bulan dekodlama algoritmaları, Peterson-Gorenstein-Zierler dekodlayıcısı, Sugiyama algoritması.

ABSTRACT

DISCRETE FOURIER TRANSFORM OVER FINITE FIELDS AND ITS APPLICATIONS ON ALGEBRAIC CODING THEORY

Sultan SELÇUK

Department of Mathematics

MSc. Thesis

Advisor: Assist. Prof. Dr. Ayten ÖZKAN

The Fourier transform is a multidimensional concept which is investigated under the topic of Fourier analysis in theory of complex functions and which we see in various areas of science and engineering, as an interesting method for mathematicians and a useful tool for engineering. Although in mathematics, the Fourier transform in the field of complex numbers is a more widespread topic of investigation, this study only investigates the transforms on finite fields. With this aim, firstly finite fields, extension fields and examples of these have been given. Together with the definition of the Fourier transform, matrix form of the transform, its characteristics to be used within the thesis in definitions and theorems and the concepts of linear recursion and linear complexity –usually more familiar to engineers but also very meaningful in terms of mathematics- have been analyzed.

Error correcting codes in cyclic codes, which are one of the most important areas of use for the Fourier transform, and their characteristics have been studied. The spectral meaning of the codes in the frequency domain obtained as a result of the Fourier transform of a cyclic code has been investigated. It has been shown specifically for cyclic codes that the value of the code obtained as a result of the Fourier transform has certain specified spectral components equal to zero and that it may be used in coding and decoding algorithms.

Specifically for cyclic codes, Reed-Solomon codes presently used most widely in communication systems, compact discs and digital video procedures and BHC codes as a subfield-subcode of these have been researched. The spectral definitions and some coding examples have been analyzed. In addition, the formation of the extended forms of the Reed-Solomon and BCH codes by adding one or two components has been shown.

It has been seen that the most useful of the algorithms used for Reed-Solomon codes are based on the Fourier transform. These are part of the decoding class which finds error and algebraically important. In order to give information on some of these algorithms, polynomial representations, syndromes and their spectral definitions have been analyzed. The Peterson-Gorenstein-Zierler decoder and Sugiyama algorithm, which serve to find the location and value of the error using algebraic methods, have been put through detailed analysis.

Keywords: Fourier transform, error correcting codes, cyclic codes, Reed-Solomon codes, BCH codes, locator decoding algorithms, Peterson-Gorenstein-Zierler decoder, Sugiyama algorithm.

BÖLÜM 1

GİRİŞ

1.1 Literatür Özeti

Bilimin değişik alanlarında geniş kullanım alanlarına sahip olan Fourier dönüşümünün kökenleri 1805 yılına kadar uzanır. Bu yıllarda, ünlü matematikçi Carl Friedrich Gauss belirli bölgelerdeki bazı astroitlerin yörüngeleri üzerinde çalışırken ayrık Fourier dönüşümünü geliştirmiştir. Fakat bu konuyla ilgili herhangi bir yayın yapmamıştır. Fourier analizinin temellerini kuran Fransız matematikçi Jean Baptiste Joseph Fourier de, henüz bu tarihlerde kendi çalışmasını yayınlamamıştı. 1822 yılında Fourier bu konudaki ilk yayınını yaptı [1].

Bundan sonra Fourier dönüşümü, kısmi diferansiyel denklemlerin çözümlerinde, çok büyük tamsayıların ve polinomların çarpımında, kompleks analizde, veri iletim sistemlerinde, cebirsel kodlama teorisinde ve kriptolojide kullanılan, hem matematikçiler hem de mühendisler için çok yönlü bir araç haline gelmiştir.

Fourier dönüşümünün hata-kontrol kodlarındaki ilk uygulamaları ise 1960'lı yıllarda görülür. İlk spektral dekodlayıcı Reed ve Solomon [2] tarafından, Reed-Solomon kodlarının minimum uzaklığının ispatında kullanılmıştır. Fakat bu dekodlayıcı pek pratik olmadığından uzun bir süre spektral dekodlama teknikleri üzerinde çalışma olmadı. Kodlama teorisinde Fourier dönüşümünün rolü, bu ad altında olmasa bile, Mattson ve Solomon tarafından 1961 yılında ortaya kondu [3]. Teorik ilerleme için önemli bir rolü olan spektral polinom tanımını yaptılar, fakat yine bir Fourier dönüşümü olarak spektral polinomun önemi çok net anlaşılmadı.

Sonlu cisimler üzerinde Fourier dönüşümü ilk defa 1971'de Pollard [4] tarafından incelendi. Bunların hata-kontrol kodları üzerindeki kullanımını Gore [5] 1973 yılında

tanıttı. Ardından Chien ve Choy [6] ile Lempel ve Winograd [7] tarafından konu ele alındı. Bir dizinin ağırlığı ile Fourier dönüşümünün lineer kompleksitesi arasındaki ilişkiyi 1979'da Blahut [8] ortaya koydu. Schaub 1988'de yazdığı doktora tezinde devirli kodların minimum uzaklıkları üzerindeki sınırların yani kısaca devirli kodların, lineer kompleksite kavramı kullanılarak nasıl oluşturulabileceğini gösterdi [9].

Reed-Solomon kodları başta olmak üzere devirli kodların popülerliğini, bunlar üzerinde çok kullanışlı dekodlama algoritmalarının varlığına bağlayabiliriz. Devirli kodlar sınıfında çok etkili bir şekilde kullanılan hata yeri bulan dekodlama algoritmalarına ilk adımı Peterson [10], hata yeri bulan polinomu tanıtarak atmış oldu. Daha sonra Gorenstein ve Zierler [11] hata yeri bulma dekodlamasını ikili olmayan kodlarda incelediler. Forney [12] ve sonra 1989 yılında Horiguchi [13] ve 1997 yılında da Koetter [14] çalışmalarında hata değerini hesaplayan değişik metodlar geliştirdiler. Blahut [8] hata yeri bulma dekodlama algoritmaları ailesini Fourier dönüşüm metodlarına göre yeniden formüle etti. 1983'de Welch ve Berlekamp, sendromların hesabını elimine eden kod bölgesinde bir dekodlama algoritmasının pantentini aldılar [20]. Bundan sonra Berlekamp bu konudaki çalışmalarına devam ederek 1996'da çalışmalarını yayınladı.

1.2 Tezin Amacı

Matematikte Fourier dönüşümünün temelleri Fourier analizi başlığı altında atılmıştır. Fourier analizi ise başta mühendislik alanları olmak üzere fizik, akustik, istatistik, kriptografi, istatistik teorisi, kodlama teorisi, kombinatorik, optik, geometri, kısmi diferansiyel denklemler, kuantum mekaniği, deniz bilimi, telekomünikasyon ve haberleşme alanlarında kullanılan bir araçtır. Matematik alanında Fourier analizi kompleks sayılar cismi üzerinde incelenir. Bu çalışmada öncelikle Fourier analizinde matematikçiler tarafından çok fazla incelenilmemiş olan sonlu cisimler üzerindeki Fourier dönüşümünü incelemek amaçlanmıştır. Ayrıca diğer temel amaç, veri ve sinyal işleme konusunda sıklıkla karşımıza çıkan Fourier dönüşümünün ve özelliklerinin cebirsel kodlama teorisindeki uygulamalarını araştırarak, özellikle devirli kodların frekans bölgesindeki tanımlarını ve bunların kodlama ve dekodlama algoritmalarındaki kullanılışlarını açıklamak ve bu konunun altında yatan teorik matematiği incelemektektir. Ayrıca hata düzelten kodlar konusunda bilinen kavramların Fourier dönüşüm teknikleriyle alternatif bir şekilde ele alınıp yeni bakış açıları kazandırmak ve

dönüşüm tekniklerine yatkın olan fakat bunların kodlama teorisindeki yerini bilmeyen mühendisler için de faydalı bir çalışma yapmak amaçlanmıştır.

1.3 Hipotez

Ayrık Fourier dönüşümü sonlu cisimler üzerinde uygulanabilir. Hata düzelten kodlar konusunda bilinen kavramlar, Fourier dönüşümü sonucunda elde ettiğimiz frekans bölgesinde tanımlanabilir. Bir vektörün Hamming ağırlığı ters Fourier dönüşümünün lineer kompleksitesine eşittir. Özel olarak devirli kodlar için frekans bölgesinde alternatif bir tanım vardır. Devirli kodlar belirli spektral bileşenleri sıfır olan kodlar olarak da tanımlanabilir. Bu tanım kullanılarak kodlama ve dekodlama algoritmaları oluşturulabilir. BCH ve Reed-Solomon kodlarını da içeren birçok kodun dekodlaması spektral olarak açıklanabilir [8].

BÖLÜM 2

TEMEL KAVRAMLAR

Bir semboller kümesine alfabe denir. Bazı alfabeler sonsuzdur, örneğin reel sayılar kümesi veya kompleks sayılar kümesi gibi. Biz genellikle sonlu alfabeler ile ilgileneceğiz. Verilen bir alfabenin sembollerinin oluşturduğu ardışık dizilmiş kümeye dizi denir. Bir dizi sonsuz uzunluklu olabilir. Bizim ilgilendiğimiz sonsuz dizilerin belirli bir başlangıcı olup sonu olmayacak, fakat başlangıç ve sonu olmayan sonsuz diziler de vardır.

Verilen bir alfabeden sonlu uzunlukta sembollerden oluşan kümeye sonlu dizi denir. Dizideki sembollerin sayısına blok uzunluğu denir ve n ile gösterilir. Bazen blok uzunluğu açıkça belirtilmez, fakat belirli bir dizi verildikten sonra dizideki sembollerin sayısı sayılarak hesaplanabilir. Diğer durumlarda, blok uzunluğu n olarak verilir, ki biz sadece n blok uzunluğu ile ilgili çalışma yapacağız.

Dizileri bir çok açıdan inceleyebiliriz. Örneğin verilen bir sembol dizisinin yapısı üzerinde ve bunların çeşitliliği üzerinde çalışılabilir. Biz bu çalışmada genellikle sonlu uzunluklu diziler üzerinde, cisim olarak bilinen özel bir aritmetik yapıya sahip alfabeler üzerinde duracağız. Bu durumda sabit bir sonlu blok uzunluğuna sahip dizilere vektör diyeceğiz.

Cebirsel metodlar kullanarak cisimler üzerindeki dizilerle işlem yapabiliriz. Lineer yineleme, devirli konvolüsyon ve Fourier dönüşümü gibi kavramlar kullanarak diziler üzerinde çalışabiliriz. Bu çalışmada sadece sembol alfabesi cisim olan hatta sadece sonlu cisim olan, adına kod diyeceğimiz sonlu boyutlu dizi kümeleri üzerinde çalışacağız.

Bir cisim üzerinde tanımlı bir vektörün en önemli özelliklerinden biri Hamming ağırlığı ya da kısaca ağırlıktır. Hamming ağırlığı bir vektörde sıfırdan farklı bileşenlerin sayısı olarak tanımlanır. Bir cisim üzerinde tanımlı bir çift vektörün en önemli özelliklerinden biri ise iki vektör arasındaki Hamming uzaklığı ya da kısaca uzaklıktır. Hamming uzaklığı iki vektörün bileşenlerinden farklılık gösterenlerin sayısı olarak tanımlanır. Bu çalışmada vektörlerin ağırlıkları ve vektörler arasındaki uzaklık konusu bizim için önemli olacaktır.

2.1 Sonlu Cisim

Cebirsel olarak cisim adı verilen yapı kabaca; toplama, çıkarma, çarpma ve bölme yapabildiğimiz ve bu işlemlerle birleşme, değişme ve dağılma özelliklerini sağlayan aritmetik bir sistemdir. En çok bilinen cisim örnekleri rasyonel sayılar kümesi $\mathbb{Q}$, reel sayılar kümesi $\mathbb{R}$ ve kompleks sayılar kümesi $\mathbb{C}$'dir. Toplama, çıkarma, çarpma ve bölme işlemlerinin kuralları her cisimde bilindiği gibidir.

Bazı bilenen aritmetik sistemler cisim değildir. Örneğin tam sayılar kümesi $\mathbb{Z}$ toplama ve çarpma işlemleri göz önüne alındığında cisim değildir. Benzer şekilde doğal sayılar kümesi $\mathbb{N}$ de cisim değildir.

Bunlar dışında sonlu veya sonsuz elemanlı birçok cisim örneği vardır. Sonlu sayıda elemanı olan cisimlere sonlu cisim ya da Galois cismi denir. q elemanlı bir Galois cismi $GF(q)$ ya da F_q ile gösterilir. Bir sonlu cismin sıfırdan farklı elemanlarının oluşturduğu küme $GF(q)^*$ ile gösterilir. Galois cismi $GF(q)$, yanlızca q bir asal p'ye eşit ise ya da m birden büyük bir tamsayı olmak üzere, bir asalın kuvvetine (p^m) eşit ise vardır. q'nun diğer tüm değerlerinde toplama ve çarpma cisim aksiyomlarını sağlamaz.

Bir F cismini tanımlamak için, cismin her eleman çifti için iyi tanımlı olan iki işlem tanımlamalıyız. Bu işlemler toplama ve çarpma olup şu özellikleri sağlarlar.

Toplama Aksiyomları: F cismi toplama işlemi altında kapalıdır, birleşme ve değişme özelliklerini sağlar.

$$a + (b + c) = (a + b) + c \tag{2.1}$$

$$a + b = b + a \tag{2.2}$$

Toplamanın birim elemanı sıfır vardır ki, $a + 0 = a$ olur. Ayrıca her elemanın tek bir negatifi vardır ki $a + (-a) = 0$ olur. Çıkarma işlemi, $a + (-a)$ şeklinde tanımlanır.

Çarpma Aksiyomları: F cismi çarpma işlemi altında kapalıdır, birleşme ve değişme özelliklerini sağlar.

$$a(bc) = (ab)c \quad \text{ve} \quad ab = ba \tag{2.3}$$

Çarpmanın birim elemanı $1 \neq 0$ olup $1a = a$ ve sıfırdan farklı her elemanın tek bir tersi vardır ki a^{-1} ile gösterilir ve $aa^{-1} = 1$ olur. Bölme işlemi ise $a \div b$ (yada a/b) ile tanımlanıp ab^{-1} anlamındadır.

Ortak Aksiyom: Dağılma özelliği; her $a, b, c \in F$ için,

$$(a + b)c = ac + bc \tag{2.4}$$

sağlanır. Eğer q bir asal sayı p'ye eşit ise, $GF(q)$ sonlu cisminin yapısını açıklamak kolaydır. Bu durumda, $GF(q) = \{0,1,2, \dots, p-1\}$ olur. Toplama ve çarpma işlemleri modulo- p toplama ve modulo- p çarpma olarak yapılır. p elemanlı bir cisim tanımlamanın diğer bir yolu farklı bir yapı ile farklı notasyonlar kullanılarak oluşturulabilir fakat bu farklı bir bakış açısından başka birşey değildir, aynı sonuca varılır. Sonuçta her p asal için sonlu cisim $GF(p)$ notasyon farkı ile bir tektir. Bu bağlamda p elemanlı sadece bir cisim vardır diyebiliriz. Benzer şekilde $GF(p^m)$ cismi içinde aynı şeyi söyleyebiliriz [16].

2.2 Cisim Genişlemesi

$GF(2)$, $GF(3)$, $GF(5)$ cisimlerinin toplama ve çarpma tablolarını kolaylıkla oluşturabiliriz. (Çizelge 2.1) Fakat $GF(4)$ için aynı şeyi söyleyemeyiz. Modulo-4 de $2 \times 2 = 0$ olup, 2'nin modulo 4'te çarpımsal tersi olmadığından burada modulo-p aritmetiği yapamayız. Dolayısıyla $GF(4)$ için farklı bir yapı oluşturacağız. 4 elemanlı bir cisim oluşturmak için 2 elemanlı $GF(2)$ cisminin genişlemesini kullanacağız. Genel olarak, F cismini içeren herhangi bir cisme F cisminin cisim genişlemesi denir. Burada F cisminin kendisine temel cisim denir. $GF(p^m)$ şeklindeki bir cisim, $GF(p)$ cisminin bir genişlemesidir ve kompleks sayılar cisminin reel sayılar cisminden oluşturulma şekline benzer şekilde basit polinomlar yoluyla kurulur. Bu yapılanmanın en genel halini açıklamak istiyoruz, fakat önce $\mathbb{C}$ kompleks sayılar cismini, $\mathbb{R}$ reel sayılar cisminin bir genişlemesi olarak nasıl oluştuğunu anlayalım.

Cisim genişlemesi ile oluşan yeni cisim, toplama ve çarpma işlemlerinin tanımlarını yapabilmemiz için reel sayı çiftlerinden oluşmalıdır. Cisim genişlemesini geçiçi olarak $\mathbb{R}^{(2)}$ ile gösterelim ve $\mathbb{R}^{(2)} = \{(a, b)|a, b \in \mathbb{R}\}$ şeklinde tanımlayalım. Burada $\mathbb{R}^{(2)}$ cisim genişlemesini $\mathbb{R}^2$ vektör uzayı ile karıştırmamak gerekir. Ayrıca $\mathbb{R}^{(2)}$ üzerindeki toplama ve çarpma işlemlemlerini tanımlamanın birkaç yolu olduğunu söylemeliyiz. Cisim genişlemesi $\mathbb{R}^{(2)}$ üzerindeki aritmetiği tanımlamak için elemanlarını polinomlar olarak ifade edeceğiz. Bu amaç için kuracağımız polinomlarda x sembolünü başka yerlerde kullanmak için z kullanacağız. Böylece (a, b) yerine daha kullanışlı olan $a + bz$ yazarak cisim genişlemesini tekrar tanımlayalım;

$$\mathbb{R}^{(2)} = \{a + bz|a, b \in \mathbb{R}\} \tag{2.5}$$

Sonra $\mathbb{R}$'de çarpanlarına ayrılmayan 2. dereceden bir polinom bulalım. Örneğin $p(z) = z^2 + 1$ polinomu çarpanlarına ayrılmaz. Bunun yanında $\mathbb{R}$'de çarpanlarına ayrılmayan 2. dereceden birçok polinom vardır (örneğin $z^2 + z + 1$), fakat kullanım kolaylığı sağladığından genellikle $p(z)$ polinomu seçilir. Şimdi cisim genişlemesi olarak, katsayıları $\mathbb{R}$'den olmak üzere derecesi $p(z)$ den küçük olan polinomlar kümesini tanımlayalım. $\mathbb{R}^{(2)}$ üzerindeki toplama ve çarpma işlemleri modulo $p(z)$ toplama ve çarpma olarak tanımlanır. Böylece toplama ve çarpma işlemleri de

$$(a + bz) + (c + dz) = (a + c) + (b + d)z \tag{2.6}$$

$$\begin{aligned}(a + bz)(c + dz) &= ac + (ad + bc)z + bdz^2 \pmod{z^2 + 1} \\ &= (ac - bd) + (ad + bc)z \end{aligned} \tag{2.7}$$

şeklinde tanımlanır. Burada $z^2 = -1 \pmod{p(z)}$ olduğundan ve $z = \sqrt{-1}$ için genel olarak bilinen kompleks sayılarla çarpma işlemi formuna dönüştürmüş oluruz. Böylece cisim genişlemesi ile elde ettiğimiz yapı aslında kompleks sayılar cismi $\mathbb{C}$ olur. Ayrıca $\mathbb{R}^{(2)}$ cisim genişlemesini kurmak için başka yapılar kullanılsa bile sonuçta notasyon farkıyla yine kompleks sayılar cismi $\mathbb{C}$ elde edilir.

Benzer şekilde $GF(2)$ cismini $GF(4)$ cismine genişletmek için $p(z) = z^2 + z + 1$ polinomunu seçelim. Bu polinom $GF(2)$'de çarpanlarına ayrılmaz. Gerçektende $GF(2)$ cisminde birinci dereceden polinomlar sadece z ve $z + 1$ polinomlarıdır ve bunların hiçbiri $z^2 + z + 1$ 'nin bir çarpanı değildir. Böylece,

$$GF(4) = \{a + bz|a, b \in GF(2)\} \tag{2.8}$$

olarak tanımlanır ve $GF(4)$ cismindeki toplama ve çarpma işlemleri, modulo $p(z)$ polinom toplaması ve çarpmasıdır. Yani, $(a + bz) + (c + dz) = (a + c) + (b + d)z$

ve $GF(2)$'de "+" ve "-" işlemlerinin aynı olması kullanılarak

$$(a + bz)(c + dz) = ac + (ad + bc)z + bdz^2 \quad (\mathrm{mod}\ z^2 + z + 1)$$

$$= (ac + bd) + (ad + bc + bd)z \tag{2.9}$$

elde edilir. $GF(4)$ cisminin elemanları $0,1,z,z+1$ olup bunları $0,1,2,3$ ile gösterirsek, toplama ve çarpma islemlerinin tablosu şöyle olur.

Çizelge 2.1 Bazı küçük cisim örneklerinin aritmetik tabloları

GF(2)

+	0	1
0	0	1
1	1	0

.	0	1
0	0	0
1	0	1

GF(3)

+	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

.	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

GF(4)

+	0	1	2	3
0	0	1	2	3
1	1	0	3	2
2	2	3	0	1
3	3	2	1	0

.	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	3	1
3	0	3	2	1

GF(5)

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

.	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

Şimdi bunları genelleştirelim. Herhangi bir F cisminin $F^{(m)}$ cisim genişlemesini oluşturmak için öncelikle F cisminde çarpanlarına ayrılmayan m. dereceden bir $p(z)$ polinomu buluruz. Böyle bir polinoma F üzerinde indirgenemez polinom denir. Fakat F cisminde m. dereceden indirgenemez polinom bulunmak zorunda değildir. [1] Bu durumda $F^{(m)}$ cisim genişlemesi de yoktur. Fakat bir $GF(q)$ sonlu cisminde her m pozitif sayısı için m. dereceden bir indirgenemez polinom bulunur. Eğer böyle m. dereceden indirgenemez polinom sayısı birden çok ise bu durumda birden çok cisim genişlemesi mevcuttur. Sonlu cisimlerde m. dereceden indirgenemez polinomlar ile oluşturulan tüm cisim genişlemeleri notasyon farkı ile aynıdır. Bunlara aynı cismin izomorfik kopyaları denir.

$F^{(m)}$ cismini oluşturacak olursak, önce derecesi m'den küçük olan polinomlar kümesini yazalım;

$$F^{(m)} = \{a_{m-1}z^{m-1} + a_{m-2}z^{m-2} + \cdots + a_1 z + a_0 | a_i \in F\} \qquad (2.10)$$

$F^{(m)}$ cisim genişlemesinde toplama işlemi polinom toplaması ve çarpma işlemi de modulo $p(z)$'de polinom çarpması gibi yapılır. Bu yapıda F cismi q elemanlı sonlu cisim $GF(q)$ ve cisim genişlemesi de q^m elemanlı sonlu cisim olarak görülebilir. Yani bu cisim notasyon farkı ile $GF(q^m)$ cismidir. Her $GF(q)$ sonlu cismi aynı yollarla bir p asalı ve l tam sayısı için $GF(p^l)$ cisminden oluşturulabilir. Buradaki p asalına $GF(q)$ cisminin karakteristiği denir.

Örneğin $GF(16)$ cismini $GF(2)$'nin bir cisim genişlemesi olarak kuralım. $p(z) = z^4 + z + 1$ seçelim. Bu polinom hem $GF(2)$ üzerinde indirgenemezdir, hem de önemli bir özelliğe sahiptir. Eğer $p(z)$ polinomu $GF(16)$'yı inşaa etmek için kullanılırsa, bu durumda z'nin çarpma işlemine göre mertebesi 15 olur.[2] Bu durumda z polinomunun mertebesi $GF(16)$'daki sıfırdan farklı eleman sayısına eşit olduğundan, $GF(16)$'daki her eleman z'nin bir kuvveti olmak zorundadır.

Temel cisim $GF(q)$ üzerinde herhangi bir $p(z)$ polinomu için $z \pmod{p}$'nin mertebesi $(q^m - 1)$ ise bu polinoma $GF(q)$ üzerinde primitif polinom denir ve z elemanına da $GF(q^m)$ genişleme cisminin primitif elemanı denir. $GF(q)$ inşaa etmek için primitif

[1] Örneğin $\mathbb{R}$'de kübik indirgenemez polinom yoktur.

[2] Bir γ elemanının mertebesi $\gamma^n = 1$ olacak şekilde en küçük n pozitif tam sayıdır.

polinom kullanmanın sebebini $GF(16)$ cisminin elemanlarını z'nin kuvvetleri olarak yazarak görebiliriz. Bu elemanlar şunlardır: $\{1, z, z+1, z^2, z^2+1, z^2+z, z^2+z+1, z^3, z^3+1, z^3+z, z^3+z+1, z^3+z^2, z^3+z^2+1, z^3+z^2+z, z^3+z^2+z+1\}$.

Bu durumda cismin sıfırdan farklı her elemanı z'nin bir kuvveti olduğundan primitif eleman z cismi üretir denir. Bu rolü ile z'yi primitif eleman olarak vurgulamak istersek α olarak kullanırız. Burada α'yi teorik olarak bir cisim elemanı, z'yi ise α'nın polinom olarak gösterilişi olarak düşünebiliriz. $GF(16)$'da sıfırdan farklı cisim elemanlarını α'nın (ya da z'nin) kuvvetleri şeklinde yazarsak şöyle buluruz:

$$\alpha^1 = z$$
$$\alpha^2 = z^2$$
$$\alpha^3 = z^3$$
$$\alpha^4 = z+1 \quad z^4 = z+1 \pmod{z^4+z+1}$$
$$\alpha^5 = z^2+z$$
$$\alpha^6 = z^3+z^2$$
$$\alpha^7 = z^3+z+1$$
$$\alpha^8 = z^2+1$$
$$\alpha^9 = z^3+z$$
$$\alpha^{10} = z^2+z+1$$
$$\alpha^{11} = z^3+z^2+z$$
$$\alpha^{12} = z^3+z^2+z+1$$
$$\alpha^{13} = z^3+z^2+1$$
$$\alpha^{14} = z^3+1$$
$$\alpha^{15} = 1 = \alpha^0 \qquad (2.11)$$

$GF(16)$ aritmetiği şu şekilde işler; cisimde iki elemanı toplamak için bunları iki polinom gibi toplar ve katsayılarını modulo 2 yaparız. Örneğin z^3+z^2 ve z^2+z+1 alalım. Sadece katsayıları yazarsak, $1100+0111=1011$ elde ederiz. Benzer şekilde bunları çarpmak için;

$$\begin{aligned}(1100)(0111) = (z^3+z^2)(z^2+z+1) &= \alpha^6\alpha^{10} = \alpha^{16} = \alpha\alpha^{15} \\ &= \alpha.1 = \alpha = z \\ &= (0010) \end{aligned} \qquad (2.12)$$

işlemleri yapılır. Benzer şekilde bölme için ise;

$$(1100)/(0111) = (z^3 + z^2)/(z^2 + z + 1) = \alpha^6/\alpha^{10} = \alpha^6\alpha^5$$
$$= \alpha^{11} = z^3 + z^2 + z = (1110) \quad (2.13)$$

işlemleri ile sonuca varılır. $GF(256)$ cismini inşaa etmek için de aynı yolu izleriz. İndirgenemez polinom olarak,

$$p(z) = z^8 + z^4 + z^3 + z^2 + 1 \quad (2.14)$$

polinomunu alabiliriz ki bu aslında bir primitif polinomdur ya da $GF(2)$'de herhangi bir 8. dereceden polinom alabiliriz [16].

Herhangi bir cisim üzerinde, matris cebri ve vektör uzayı teorisi de dahil olmak üzere birçok temel cebir metodlarını kullanabiliriz. Ayrıca herhangi bir F cismi üzerinde n uzunluklu bir vektörün ayrık Fourier dönüşümü de tanımlıdır. $GF(q)$ sonlu cismi $(q-1)$'i bölen her n için, mertebesi n olan bir eleman içerir, çünkü $GF(q)$ her zaman mertebesi $(q-1)$ olan bir primitif elemana sahiptir. Eğer n, $(q-1)$'i bölüyor ise cismin her sıfırdan farklı elemanı α'nın bir kuvveti olacak ve her zaman α'nın bir kuvvetinin mertebesi n olacaktır. Eğer n, $(q-1)$ 'i bölmüyorsa mertebesi n olan eleman yoktur. Şimdi bu bilgileri kullanarak sonlu cisimler üzerinde Fourier dönüşümünü inceleyelim.

BÖLÜM 3

AYRIK FOURİER DÖNÜŞÜMÜ ve ÖZELLİKLERİ

3.1 Ayrık Fourier Dönüşümü (DFT)

Ayrık Fourier dönüşümü (discrete Fourier Transform-DFT) kompleks sayılar cisminde tanımlanınca hata-kontrol kodları konusu ve mühendisler için oldukça faydalı ve kullanışlı bir araç olur. Ama Fourier dönüşümü herhangi bir cisim üzerinde de varlık gösterir. Fourier dönüşümünün birçok özelliği, belirli bir cismin belirli yapısal özelliklerine değilde, genel olarak bir cismin soyut özelliklerine dayandığından, Fourier dönüşümünün en bilinen özellikleri herhangi bir cisim üzerinde de geçerli olur.

Fourier dönüşümü, F^n şeklinde gösterilen n boyutlu vektör uzayı üzerinde tanımlanır. F^n vektör uzayında bir $\boldsymbol{v}$ vektörü, F cisminin n elemanından oluşur ve şöyle yazılır:

$$\boldsymbol{v} = [v_0, v_1, \dots, v_{n-1}]. \tag{3.1}$$

Vektörün F cisminin bir γ elemanı ile çarpılması, v'nin her bileşeninin γ ile çarpılması anlamına gelir. Yani;

$$\gamma\boldsymbol{v} = [\gamma v_0, \gamma v_1, \dots, \gamma v_{n-1}] \tag{3.2}$$

olur ve buradaki cisim elemanı γ'ya skaler denir. İki vektör $\boldsymbol{u}$ ve $\boldsymbol{v}$ vektörlerinin toplaması ise,

$$\boldsymbol{u} + \boldsymbol{v} = [u_0 + v_0, u_1 + v_1, \dots, u_{n-1} + v_{n-1}] \tag{3.3}$$

şeklinde bileşen bileşen toplamadır.

Tanım 3.1 F cismi üzerinde n uzunluklu bir vektör $\boldsymbol{v}$ olsun. ω ise F'nin mertebesi n olan bir elemanı olsun. $\boldsymbol{v}$ vektörünün Fourier dönüşümü, F üzerinde yine n uzunluklu bir $\boldsymbol{V}$ vektörüdür ve bileşenleri şöyledir:

$$V_j = \sum_{i=0}^{n-1} \omega^{ij} v_i \qquad j = 0,1,\dots,n-1 \tag{3.4}$$

Burada bulunan $\boldsymbol{V}$ vektörüne $\boldsymbol{v}$'nin spektrumu da denir ve $\boldsymbol{V}$'nin bileşenlerine de spektral bileşenler denir. Bir vektörün Fourier dönüşüm vektörünün bileşenlerini her zaman j ile, orijnal vektörün bileşenlerini ise i ile indeksleyeceğiz. Ayrıca Fourier dönüşüm ilişkisini de $\boldsymbol{v} \leftrightarrow \boldsymbol{V}$ ile göstereceğiz.

Fourier dönüşümü bir polinomun değerinin hesaplanması gibi de düşünülebilir.

$$v = [v_i | i = 0,1,\dots,n-1] \tag{3.5}$$

vektörünün polinom olarak gösterimi;

$$v(x) = \sum_{i=0}^{n-1} v_i x^i \tag{3.6}$$

şeklindedir. Buradan bir $x = \beta$ cisim elemanı için $v(\beta)$'yi hesaplamak istersek,

$$v(\beta) = \sum_{i=0}^{n-1} v_i \beta^i \tag{3.7}$$

buluruz. Böylece Fourier dönüşümü, $v(x)$ polinomunu n mertebeli ω elemanının n kuvvetlerinde hesaplamak anlamına gelir. Yani $j = 0,1,\dots,n-1$ için V_j bileşeni $v(\omega^j)$'e eşit olur. Eğer F sonlu cisim $GF(q)$'ya eşit ise ve ω bir primitif eleman ise bu durumda Fourier dönüşümü ile sıfırdan farklı tüm $(q-1)$ elemanın $v(x)$ değerini hesaplamış oluruz [16].

Fourier dönüşümü bir çok kullanışlı özelliği ile bizim için çok güçlü bir araçtır. Bu özellikleri vermeden önce birkaç tane Fourier dönüşümü örnekleri verelim.

1. $\mathbb{Q}$ ya da $\mathbb{R}$: $\omega = 1$ elemanının mertebesi 1 ve $\omega = -1$ elemanının mertebesi 2'dir. $\mathbb{Q}$'da ya da $\mathbb{R}$'de bunlar dışında mertebesi n olacak şekilde bir ω elemanı yoktur. Böylece $\mathbb{Q}$ ve $\mathbb{R}$'de trivial Fourier dönüşümü vardır. $\mathbb{R}$ üzerinde uzunluğu 2'den büyük bir Fourier dönüşümü elde etmek için $\mathbb{R}$'yi $\mathbb{C}$ içine gömmek gerekir. Fakat $\mathbb{Q}$ ya da $\mathbb{R}$ üzerinde 2^m eleman içeren çok boyutlu Fourier dönüşümü vardır.

2. $\mathbb{C}$: $i = \sqrt{-1}$ olmak üzere $\omega = e^{-2\pi i/n}$ elemanı n mertebelidir. Dolayısı ile her n için $\mathbb{C}$'de Fourier dönüşümü mevcuttur.

3. $GF(5)$: $\omega = 2$ elemanının mertebesi 4'tür. Buradan,

$$V_j = \sum_{i=0}^{3} 2^{ij} v_i \qquad j = 0,1,2,3 \tag{3.8}$$

dönüşümü $GF(5)$'te 4 uzunluklu bir Fourier dönüşümüdür.

4. $GF(31)$: $\omega = 2$ elemanının mertebesi 5'tir. Buradan,

$$V_j = \sum_{i=0}^{4} 2^{ij} v_i \qquad j = 0,1,2,3,4 \tag{3.9}$$

dönüşümü $GF(31)$'te 5 uzunluklu bir Fourier dönüşümüdür. Ayrıca, $\omega = 3$ elemanının mertebesi 30 olup Fourier dönüşümü,

$$V_j = \sum_{i=0}^{29} 3^{ij} v_i \qquad j = 0,1,\dots,29 \tag{3.10}$$

şeklindedir.

5. $GF(2^{16}+1)$: $2^{16}+1$ asal bir sayı olduğundan ancak n, $2^{16}+1-1$ sayısını bölerse mertebesi n olan bir ω elemanı bulunur. Dolayısıyla $k = 1,2,\dots,16$ için mertebesi 2^k olan elemanlar mevcuttur. Yani $k = 1,2,\dots,16$ iken her 2^k için $GF(2^{16}+1)$ 'de uzunluğu 2^k olan Fourier dönüşümü bulunur.

6. $GF((2^{17}-1)^2)$: Bu cisim, $GF(2^{17}-1)$ üzerinde indirgenemez olan 2. dereceden bir polinom kullanılarak $GF(2^{17}-1)$ cisminin bir cisim genişlemesi olarak inşaa edilmiştir.

$$(2^{17}-1)^2 - 1 = (2^{17}-1+1)(2^{17}-1-1) = 2^{17}(2^{17}-2) = 2^{18}(2^{16}-1)$$

olduğundan $2^{18}(2^{16}-1)$'i bölecek şekilde bir n için genişleme cisminde n mertebeli bir ω elemanı vardır. Özellikle $GF((2^{17}-1)^2)$'de 2'nin 2^{18}'e kadar her kuvveti için 2'nin o kuvvetine eşit uzunlukta olan bir Fourier dönüşümü vardır.

7. $\mathbb{Q}^{(16)}$: $p(z) = z^{16}+1$ polinomunu göz önüne alalım. Bu polinom $\mathbb{Q}$ üzerinde indirgenemez bir polinomdur. $z^{16} = -1$ yazarak modulo $p(z) = z^{16}+1$ çarpması indirgenir. $\mathbb{Q}^{(16)}$'nin bir elemanı 2 yerine 16 parçalı bir "süperkomleks" sayı gibi düşünülebilir. Bu paralellik ile düşündüğümüzde z sembolü i ile değiştirilebilir. Fakat bir kompleks sayı $a_0, a_1 \in \mathbb{Q}$ olmak üzere $a_0 + a_1 i$ formatında iken, bir "süperkompleks" sayı her $k = 1,2,\dots,15$ için $a_k \in \mathbb{Q}$ ve $i^{16} = -1$ olmak üzere

$a_0 + a_1 i + a_2 i^2 + \cdots + a_{14} i^{14} + a_{15} i^{15}$ formundadır. $\mathbb{Q}(16)$ cisminde 32 blok uzunluğunda bir Fourier dönüşümü vardır. Çünkü $z^{16} = -1 \quad (mod\, z^{16} + 1)$ olup z elemanının mertebesi 32'dir. Bu Fourier dönüşümü 32 uzunluklu bir vektörü yine 32 uzunluklu başka bir vektöre dönüştürür. Vektörün bileşenleri $\mathbb{Q}$ üzerinde 15. dereceden polinomlardır ve Fourier dönüşümü;

$$V_j(z) = \sum_{i=0}^{31} z^{ij} v_i(z) \qquad (mod\ z^{16} + 1) \tag{3.11}$$

şeklinde hesaplanır. Bu işlemi, 32'ye 16'lık rasyonel sayılardan oluşan bir matristen başka bir 32'ye 16'lık rasyonel sayılar matrisi oluşturma işlemi gibi düşünebiliriz. Çünkü z ile çarpma işlemi bir indeksleme işlemi olarak uygulanabilir, $\mathbb{Q}^{(16)}$'da Fourier dönüşümü $\mathbb{Q}$'da hiç çarpma yapmadan da hesaplanabilir [16].

3.2 Fourier Dönüşümün Matris Gösterimi

$\boldsymbol{v}$ ve $\boldsymbol{V}$, F^n cismi üzerinde n-liler ve ω, n. primitif birim kök olmak üzere Fourier dönüşümünü hesaplamak için bunların matris gösteriminden faydalanabiliriz. Öncelikle $\boldsymbol{v}$ ve $\boldsymbol{V}$ n-lilerini sütun vektörleri olarak yazalım [9].

$$\boldsymbol{v} = \begin{bmatrix} v_0 \\ v_1 \\ \vdots \\ v_{n-1} \end{bmatrix} \qquad ve \qquad \boldsymbol{V} = \begin{bmatrix} V_0 \\ V_1 \\ \vdots \\ V_{n-1} \end{bmatrix} \tag{3.12}$$

Buradan $\boldsymbol{v}$'nin Fourier dönüşümü olan $\boldsymbol{V}$'yi DFT matrisi yardım ile şöyle hesaplayabiliriz:

$$\boldsymbol{V} = F\boldsymbol{v}. \tag{3.13}$$

Buradaki F, DFT matrisi ise şöyledir,

$$F = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega^1 & \omega^2 & \dots & \omega^{n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \omega^{n-1} & \omega^{2(n-1)} & \dots & \omega^{(n-1)(n-1)} \end{bmatrix} \tag{3.14}$$

Örnekler

1. $GF(16)$ cismini göz önüne alalım. Eğer $GF(16)$ cismi $p(z) = z^4 + z + 1$ primitif polinomu ile inşaa edilmiş ise, z elemanının mertebesi 15 olur. Dolayısıyla $\omega = z$'de 15 mertebeli bir eleman olup,

$$V_j = \sum_{i=0}^{14} z^{ij} v_i \qquad j = 0,1,\dots,14 \tag{3.15}$$

şeklinde $n = 15$ uzunluklu Fourier dönüşümü elde ederiz.

v_i ve V_j'nin bileşenleri $GF(16)$'nin elemanları olarak $GF(2)$'de $z^4 = z + 1$ polinomu ile indirgendikten sonra en fazla 3. dereceden polinomlar şeklinde ifade edilebilirler.

Başka bir yöntemle, $GF(16)$'da $\omega = z^3$ elemanının mertebesi 5 olup, boyutu 5 olan bir Fourier dönüşümü oluşturabiliriz.

$$\begin{bmatrix} V_0(z) \\ V_1(z) \\ V_2(z) \\ V_3(z) \\ V_4(z) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & z^3 & z^6 & z^9 & z^{12} \\ 1 & z^6 & z^{12} & z^{18} & z^{24} \\ 1 & z^9 & z^{18} & z^{27} & z^{36} \\ 1 & z^{12} & z^{24} & z^{36} & z^{48} \end{bmatrix} \begin{bmatrix} v_0(z) \\ v_1(z) \\ v_2(z) \\ v_3(z) \\ v_4(z) \end{bmatrix} \tag{3.16}$$

Burada $GF(16)$ cisminin elemanlarının polinomlar olarak gösterildiğini vurgulamak için $\boldsymbol{v}$ ve $\boldsymbol{V}$ vektörlerinin bileşenleri de polinomlar şeklinde ifade edildi. Ayrıca z'nin 3. dereceden büyük bütün kuvvetleri $z^4 = z + 1$ polinomu kullanılarak küçültüldüler.

2. $GF(256)$ cismini ele alalım. 255'i bölecek şekilde bir n için mertebesi n olan bir ω mevcuttur. Eğer $GF(256)$ cismini inşaa etmek için $p(z) = z^8 + z^4 + z^3 + z^2 + 1$ primitif polinomunu kullanırsak z 'nin mertebesi 255 olur. Böylece,

$$\begin{bmatrix} V_0(z) \\ V_1(z) \\ \vdots \\ V_{254}(z) \end{bmatrix} = \begin{bmatrix} 1 & 1 & \cdots \\ 1 & & \\ \vdots & & z^{ij} \end{bmatrix} \begin{bmatrix} v_0(z) \\ v_1(z) \\ \vdots \\ v_{254}(z) \end{bmatrix} \tag{3.17}$$

$GF(256)$ üzerinde 255 boyutlu Fourier dönüşümü elde ederiz. Her bileşen $GF(2)$ üzerinde bir polinom ile gösterilir ve z'nin kuvvetleri $z^8 = z^4 + z^3 + z^2 + 1$ ile sadeleştirilmiştir [16].

3.3 Fourier Dönüşümünün Özellikleri

Fourier dönüşümü birçok kullanışlı ve faydalı özellikleri olduğundan önemli bir araçtır. Bu doğrultuda bizim için faydalı olacak olan özellikleri sıralayalım. $\boldsymbol{V} = [V_j]$ vektörü, $\boldsymbol{v} = [v_i]$ vektörünün Fourier dönüşümü olsun. Bu durumda şu özellikler sağlanır:

Lineerlik: $\delta\boldsymbol{v} + \mu\boldsymbol{v}' \leftrightarrow \delta\boldsymbol{V} + \mu\boldsymbol{V}'$ Açıkça yazacak olursak,

$$\sum_{i=0}^{n-1} \omega^{ij}(\delta v_i + \mu v'_i) = \delta \sum_{i=0}^{n-1} \omega^{ij} v_i + \mu \sum_{i=0}^{n-1} \omega^{ij} v'_i = \delta V_j + \mu V'_j. \tag{3.18}$$

Ters Dönüşüm: Herhangi bir cisimde $n = 1 + 1 + \cdots + 1$ (n tane) şeklinde tanımlanmak üzere,

$$v_l = \frac{1}{n} \sum_{j=0}^{n-1} \omega^{-ij} V_j \quad i = 0,1, \dots, n-1 \tag{3.19}$$

Açıkça yazacak olursak,

$$\frac{1}{n} \sum_{j=0}^{n-1} \omega^{-ij} \sum_{l=0}^{n-1} \omega^{lj} v_l = \frac{1}{n} \sum_{l=0}^{n-1} v_l \sum_{j=0}^{n-1} \omega^{(l-i)j}$$

$$= \frac{1}{n} \sum_{l=0}^{n-1} v_l \begin{cases} n & l = i \\ \dfrac{1-\omega^{(l-i)n}}{1-\omega^{(l-i)}} = 0 & l \neq i \end{cases} = v_i \tag{3.20}$$

Modülasyon-Değişme: Çift parantezler modulo n olduğunu göstermek üzere,

$[v_i \omega^{ij}] \leftrightarrow \left[V_{((j+l))}\right]$ dönüşümü vardır. Açıkça yazacak olursak,

$$\sum_{i=0}^{n-1} (v_i \omega^{(il)}) \omega^{ij} = \sum_{i=0}^{n-1} v_i \omega^{i(j+l)} = V_{((j+l))} \tag{3.21}$$

Öteleme: $[v_{((i-l))}] \leftrightarrow [V_j \omega^{lj}]$. Açıkça yazacak olursak,

$$\frac{1}{n} \sum_{j=0}^{n-1} (V_j \omega^{lj}) w^{-ij} = \frac{1}{n} \sum_{j=0}^{n-1} V_j \omega^{-(i-l)j} = v_{((i-l))}. \tag{3.22}$$

Konvolüsyon özelliği: Çarpmanın konvolüsyonu

$$e_i = \sum_{l=0}^{n-1} f_{((i-l))} g_l \leftrightarrow E_j = F_j G_j \tag{3.23}$$

Konvolüsyonların çarpımı:

$$e_i = f_i g_i \leftrightarrow E_j = \frac{1}{n} \sum_{l=0}^{n-1} F_{((j-l))} G_l \tag{3.24}$$

Açıkça yazacak olursak,

$$e_i = \sum_{l=0}^{n-1} f_{((i-l))} g_l = \sum_{l=0}^{n-1} f_{((i-l))} \frac{1}{n} \sum_{j=0}^{n-1} \omega^{-lj} G_j$$

$$= \frac{1}{n} \sum_{j=0}^{n-1} \omega^{-\mathbb{Z}j} G_j \sum_{l=0}^{n-1} \omega^{(i-l)} j f_{((i-l))} = \frac{1}{n} \sum_{j=0}^{n-1} \omega^{-ij} G_j F_j \quad (3.25)$$

Polinomların sıfırları: $v(x) = \sum_{i=0}^{n-1} v_i x^i$ polinomunun bir ω^j noktasında sıfırı olması için gerek ve yeter koşul $V_j = 0$ olmasıdır. $V(y) = \sum_{i=0}^{n-1} V_j y^j$ polinomunun bir ω^{-i} noktasında sıfır olması için gerek ve yeter şart $v_i = 0$ olmasıdır. Kısaca;

$$v(\omega^j) = \sum_{i=0}^{n-1} v_i(\omega^j) = V_j \quad (3.26)$$

Lineer kompleksite: Bir $\boldsymbol{v}$ vektörünün ağırlığı onun Fourier dönüşümü $\boldsymbol{V}$'nin devirli kompleksitesine eşittir. Sonraki bölümde ayrıntılı anlatılacaktır.

Karşıtlık (reciprocation) Özelliği: Bir $[v_i]$ vektörünün ters vektörü $[v_{n-i}]$ vektörüdür. $\boldsymbol{v}$'nin tersinin Fourier dönüşümü, Fourier dönüşümü $\boldsymbol{V}$'nin tersine eşittir.

$$[v_{((n-i))}] \leftrightarrow [V_{((n-j))}]$$

Burada $\omega^n = 1$ olmak üzere,

$$\sum_{i=0}^{n-1} v_{((n-1))} \omega^{ij} = \sum_{i=0}^{n-1} v_i \omega^{(n-i)j}$$

$$= \sum_{i=0}^{n-1} v_i \omega^{i(n-j)} = V_{((n-j))} \quad (3.27)$$

Devirli permütasyon: b ve n aralarında asal tam sayılar olsun. Yani en büyük ortak bölenleri 1 olsun. Bu durumda $Bb = 1 \pmod n$ olmak üzere,

$$[v_{((b_i))}] \leftrightarrow [V_{((B_j))}].$$

Burada temel sayılar teorisi bilgimizden biliyoruz ki, öyle B ve N sayılar vardır ki, $Bb + Nn = 1$'dir. Böylece bunu kullanarak

$$\sum_{i=0}^{n-1} \omega^{ij} v_{((bi))} = \sum_{i\prime=0}^{n-1} \omega^{(Bb+Nn)ij} v_{((bi))} = \sum_{i=0}^{n-1} \omega^{(bi)(Bj)} v_{((bi))} \quad (3.28)$$

yazabiliriz. $i' = ((bi))$ olsun. b ve n aralarında asal olduğundan bu permütasyondan toplam değişmez. Böylece,

$$\sum_{i=0}^{n-1} \omega^{ij} v_{((bi))} = \sum_{i\prime=0}^{n-1} \omega^{i\prime Bj} V_{i\prime} = V_{((B_j))}. \tag{3.29}$$

Desimasyon (decimation) Özelliği:

Teorem 3.1 $\mathrm{obeb(b,n)} = 1$ olmak üzere eğer $\mathbf{v}'$ n-lisi, $\mathbf{v}$ n-lisinden b kadar desimasyon ile elde edilmiş ise bu durumda, $((\mathrm{b}^{-1}))$ b'nin modulo n tersi olmak üzere spektral n-li $\mathbf{V}'$, $\mathbf{V}$ n-lisinden $((\mathrm{b}^{-1}))$ kadar desimasyon ile elde edilmiştir.

Burada $\mathbf{v}$'nin b kadar desimasyonu ile anlatılmak istenen, $\mathrm{i} = 0,1,\dots,\mathrm{n}-1$ için $\mathbf{v}$'nin bileşenleri ile $\mathrm{v'_i} = \mathrm{v}_{((\mathrm{ib}))}$ dönüşümü yaparak $\mathbf{v}'$ bileşenlerini elde etmektir. Burada $\mathbf{v}'$'nün bileşenleri ise şöyledir: $\mathbf{v}' = [\mathrm{v}_0, \mathrm{v}_{((\mathrm{b}))}, \dots, \mathrm{v}_{(([\mathrm{n}-1]\mathrm{b}))}]$

İspat: $obeb(b,n) = 1$ ile $\boldsymbol{v}$'nin tüm elemanlarının $\boldsymbol{v}'$'de bulunmasını garantilemiş oluyoruz. Frekans bölgesinde $\boldsymbol{V}'$'nün bileşenleri

$$V'_j = \sum_{i=0}^{n-1} v'_i \omega^{ij} \tag{3.30}$$

şeklindedir ya da bunlar $\boldsymbol{v}$'nin terimleri cinsinden yazarsak

$$V'_j = \sum_{i=0}^{n-1} v_i \omega^{i((b^{-1}))j} = V_{\left(\left(\left((b^{-1})\right)\right)\right)} \tag{3.31}$$

olur. Bu özellik benzer şekilde zaman bölgesinde desimasyon ile de uygulanabilir. [9]

Örnek: Fourier dönüşümünün desimasyon özelliğini bir örnekte gösterelim. $n = 7$ uzunluğunda elemanları $GF(2^3)$ üzerinde olan bir kodsöz alalım.

$$\boldsymbol{v} = [1110011] \tag{3.32}$$

olsun. Bu kodsözün 3 ile desimasyonu sonucunda

$$\boldsymbol{v}' = [1011110] \tag{3.33}$$

elde ederiz. Şimdi $\boldsymbol{v}$'nin Fourier dönüşümünü alırsak

$$\boldsymbol{V} = [1, \omega^6, \omega^5, \omega^3, \omega^3, \omega^5, \omega^6] \tag{3.34}$$

olur. Benzer şekilde $\boldsymbol{v}'$'nün Fourier dönüşümü de

$$\boldsymbol{V}' = [1, \omega^5, \omega^3, \omega^6, \omega^6, \omega^3, \omega^5] \tag{3.35}$$

bulunur. Buradan 3'ün modulo 7 tersi 5 olduğundan $\boldsymbol{V}$'nin 5 ile desimasyonu ile

$$[1, \omega^5, \omega^3, \omega^6, \omega^6, \omega^3, \omega^5] = \boldsymbol{V}' \tag{3.36}$$

olduğunu görürüz.

Teorem 3.2 *(Konjugelik (conjugacy) Kısıtı) Spektral bölgenin n-lisi $\boldsymbol{V}$'nin elemanları $GF(q)$ cisminin genişleme cismi olan $GF(q^m)$'ye ait olsun. Bu durumda zaman bölgesi n-lisi $\boldsymbol{v}$'nin elemanlarının $GF(q)$'ya ait olması için gerek ve yeter koşul spektral bileşenler konjugelik kısıtı olan*

$$V_j^q = V_{((qj))} \quad j = 0,1,\dots,n-1 \tag{3.37}$$

eşitliğini sağlıyor olmasıdır. Benzer şekilde zaman bölgesi elemanları genişleme cismi $GF(q^m)$'de ise spektral elemanların altcisim $GF(q)$'da olması için gerek ve yeter koşul zaman bölgesi n-lilerin konjugelik kısıtı $v_j^q = v_{((qj))}$ $j = 0,1,\dots,n-1$ eşitliğini sağlamasıdır [18].

İspat: Tanımdan $\mathrm{V_j} = \sum_{\mathrm{i}=0}^{\mathrm{n}-1} \omega^{\mathrm{ij}} \mathrm{v_i}$ $\mathrm{j} = 0,1,\dots,\mathrm{n}-1$ olduğunu biliyoruz. Buradan

$$\mathrm{V_j^q} = \left(\sum_{\mathrm{i}=0}^{\mathrm{n}-1} \omega^{\mathrm{ij}} \mathrm{v_i} \right)^{\mathrm{q}} \tag{3.38}$$

değerini hesaplamalıyız. Karakteristiği p olan herhangi bir cisimde

$$\mathrm{p^s}_{\mathrm{l}} = \frac{\mathrm{p^s!}}{(\mathrm{p^s}-\mathrm{l})!\,\mathrm{l!}} = 0 \pmod{\mathrm{p}} \quad 0 < \mathrm{l} < \mathrm{p^s} \tag{3.39}$$

olduğunu biliyoruz. Bu ise $\mathrm{GF(q^m)}$'de eğer q, p'nin bir kuvveti ise $(\mathrm{a}+\mathrm{b})^{\mathrm{q}} = \mathrm{a^q} + \mathrm{b^q}$ olur. Çünkü burada diğer tüm terimler $\mathrm{q}_{\mathrm{l}}\ \mathrm{a^{q-l}b^q}$ şeklindedir ve q_{l} p'nin bir katı olduğundan modulo p'de sıfıra eşittir. Böylece

$$\mathrm{V_j^q} = \sum_{\mathrm{i}=0}^{\mathrm{n}-1} \omega^{\mathrm{qij}} \mathrm{v_i^q} \tag{3.40}$$

yazabiliriz. Ayrıca buradan $\mathrm{GF(q)}$'daki her a elemanı için $\mathrm{a^q} = \mathrm{a}$ olduğunu da kullanarak

$$\mathrm{V_j^q} = \sum_{\mathrm{i}=0}^{\mathrm{n}-1} \omega^{\mathrm{iqj}} \mathrm{v_i} = \mathrm{V_{((qj))}} \tag{3.41}$$

elde ederiz. Benzer şekilde ikinci kısımı da gösterebiliriz. Ayrıca bir önceki örneğimizde $\mathbf{V}$ ve $\mathbf{V}'$ eşleniklik kısıtı özelliğini sağladığına da bir örnek teşkil ederler.

3.4 Lineer Yineleme ve Lineer Kompleksite

F cismi üzerinde bir lineer yineleme, V_j ve Λ_j terimleri *F* cisminin elemanları olmak üzere

$$V_j = -\sum_{k=1}^{L} \Lambda_k V_{((j-k))} \qquad \text{j} = L, L+1, \ldots \tag{3.42}$$

şeklinde bir gösterime sahiptir. Λ_j'lere *bağlantı katsayısı* diyelim. $j = 1, \ldots, L$ için L adet bağlantı katsayısı Λ_j verildiğinde lineer yineleme fonksiyonu, $j = 0,1, \ldots, L-1$ için V_j terimlerinden $j = L, L+1, \ldots$ için V_j terimlerini üretir. Burada L tamsayısı yinelemenin uzunluğunu verir. Yinelemenin L adet katsayıları, $\Lambda_0 = 1$ olmak üzere

$$\Lambda(x) = 1 + \sum_{j=1}^{L} \Lambda_j x^j = \sum_{j=0}^{L} \Lambda_j x^j \tag{3.43}$$

şeklinde tanımlı ve *bağlantı polinomu (connection polynomial)* olarak adlandırılan bir polinom oluşturmak için kullanılırlar. Lineer yineleme kısaca $(\Lambda(x), L)$ ile gösterilir. Bu gösterimde $\Lambda(x)$ bir polinom ve L bir tamsayıdır [15].

Elemanları F cisminden olmak üzere (sonlu veya sonsuz) $V = (V_0, V_1, \ldots,)$ dizisi verilsin. Bunu bir vektörün Fourier dönüşümü olarak düşünebiliriz. Bu dizinin *lineer kompleksitesi*, bu dizi için lineer yenileme var olacak şekilde en küçük L değeridir. Başka bir ifade ile bu diziyi oluşturan en küçük uzunluklu lineer yinelemeye **V**'nin lineer kompleksitesi denir. **V**'nin lineer kompleksitesi $\mathcal{L}(\mathbf{V})$ ile gösterilir. Eğer sıfırdan farklı sonsuz **V** dizisi için böyle bir yineleme yoksa bu durumda $\mathcal{L}(\mathbf{V}) = \infty$ olur. Tüm bileşenleri sıfır olan bir dizinin lineer kompleksitesi sıfırdır. Uzunluğu r olan sonlu bir dizi için $\mathcal{L}(\mathbf{V})$ her zaman mevcuttur ve r'den daha büyük değildir. n periyodlu periyodik bir dizi için de $\mathcal{L}(\mathbf{V})$ her zaman mevcuttur ve n'den daha büyük değildir.

Örnek: Elemanları $GF(2)$'de olan 6 uzunluklu $v = [010101]$ dizisi verilsin. Bu diziyi oluşturan en küçük lineer yineleme $i = 2,3,4,5$ için $v_i = v_{i-1}$ olduğundan v dizisinin lineer kompleksitesi $\mathcal{L}(v) = 2$ olur.

Eğer **V**, n uzunluğundaysa bu durumda **V**, n uzunluklu bir vektörün Fourier dönüşümü olarak düşünülebilir. Şimdi **V**'nin lineer kompleksitesi ile ters Fourier dönüşümü **v**'nin özellikleri arasındaki ilişkiyi inceleyelim. Bu durumda daha sonra göreceğimiz sebeplerden dolayı **V**'yi üreten en küçük lineer yineleme için elde ettiğimiz bağlantı polinomuna hata yerini bulan polinom (locator polinom) denir. Yani hata yerini bulan polinom bir periyodik dizi için minimal uzunlukta bir bağlantı polinomudur [9].

n uzunluğunda ve ağırlığı 1 olan bir **v** vektörü düşünelim ve **v**'nin sıfırdan farklı olan tek bileşeni v_m olsun. Bunun Fourier dönüşümü

$$V_j = v_m \omega^{jm} \tag{3.44}$$

olur. Spektrum lineer yineleme ile $V_0 = v_m$ ilk değerleri için

$$V_j = \omega^m V_{j-1} \tag{3.45}$$

şeklinde üretilir. Yani 1 ağırlıklı bir vektörün spektrumu 1 uzunluklu bir lineer yineleme ile üretilebilir. Bu durum lineer kompleksitenin aşağıdaki teoremde verilen özel bir durumudur.

Teorem 3.3 *Devirli olarak tekrar eden, sonlu uzunluklu bir **V** vektörünün lineer kompleksitesi, ters Fourier dönüşümü **v**'nin Hamming ağırlığına eşittir.*

İspat: $i_1, i_2, \dots, i_d$ indisleri $\boldsymbol{v} = (v_0, v_1, \dots, v_{n-1})$'nin sıfır olmayan d adet bileşeninin indislerini göstersin. Şu polinomu düşünelim:

$$\Lambda(x) = \prod_{l=1}^{d} \left(1 - x\omega^{i_l}\right) = \sum_{k=0}^{d} \Lambda_k x^k \tag{3.46}$$

Λ, bileşenleri $\Lambda(x)$'in katsayıları olan bir vektör olsun. λ da Λ'nin ters Fouirer dönüşümü olsun.

$$\lambda_i = \frac{1}{n}\sum_{k=0}^{n-1} \Lambda_k = \frac{1}{n}\Lambda\left(\omega^{-i}\right) = \prod_{l=1}^{d} \left(1 - \omega^{-i}\omega^{i_l}\right) \tag{3.47}$$

Burada $\lambda_i = 0$ olması için gerek ve yeter koşul $i \in \{i_1, i_2, \dots, i_d\}$ olmasıdır. Yani $\lambda_i = 0$ ancak ve ancak $v_i \neq 0$. Sonuç olarak her i için $\lambda_i v_i = 0$ dir. Buradan konvolüsyon teoremine göre $\Lambda * \mathbf{V} = 0$ ya da

$$\sum_{j=0}^{n-1} \Lambda_j V_{((k-j))} = 0 \tag{3.48}$$

olur. Fakat $k > d$ ise $\Lambda_0 = 1$ ve $\Lambda_k = 0$ olur. Böylece

$$V_k = -\sum_{j=1}^{d} \Lambda_j V_{((k-j))} \tag{3.49}$$

elde edilir ve lineer kompleksite d'den büyük olmaz. Daha küçük olmadığını göstermek için varsayalım ki

$$\sum_{j=0}^{L} \Lambda_j V_{((k-j))} = 0 \tag{3.50}$$

olsun. Ters Fourier dönüşümü $\lambda_i = 0$ için $v_i \neq 0$ olduğu anlamına gelen $\lambda_i v_i = 0$ eşitliğini sağlamak zorundadır. $\Lambda(x)$ en fazla L adet sıfıra sahip olacağından, L en fazla $\boldsymbol{v}$'nin ağırlığı kadar olabilir diyebiliriz. Böylece ispat tamamlanır.

Sonuç: Bir **V** vektörünün Hamming ağırlığı ters Fourier dönüşümü **v**'nin lineer kompleksitesine eşittir.

3.5 Vektörlerin Ağırlıkları Üzerindeki Sınırlar

Lineer kompleksite özelliği bir vektörün ağırlığını, onun Fourier dönüşümünü oluşturan lineer yinelemenin uzunluğuna bağladığını gördük. Bu özellik kullanılarak bir vektörün Fourier dönüşümü ile vektörün istenilen ağırlıkta olup olmadığı kontrol edilebilir. Bu bölümde, bir vektörün Fourier dönüşümündeki sıfırların yapısının o vektörün ağırlığındaki sınırları nasıl belirlediğini açıklayan teoremler verilmiştir. Bu teoremler cebirin temel teoreminin sonuçları olarak da elde edilebilir.

Teorem 3.4 *(BCH Sınırı) Ağırlığı $(d-1)$'e eşit veya daha küçük olan, Fourier dönüşümünde $(d-1)$ (devirli) ardışık bileşeni sıfıra eşit olan n uzunluklu tek vektör sıfır vektördür.*

İspat: Lineer kompleksite özelliğine göre **v** vektörünün ağırlığı d'den küçük olduğundan Fourier dönüşümü olan **V** şu yinelemeyi sağlar;

$$V_j = -\sum_{k=1}^{d-1} \Lambda_k V_{((j-k))} \tag{3.51}$$

Bu yineleme gösteriyor ki **V**'nin sıfıra eşit olan herhangi $d-1$ adet devirli ardışık bileşeni, **V**'nin sıfıra eşit olan başka bir bileşeni ile devam edecek ve bu işlem böyle devam edecek. Sonuçta **V**'nin bileşenleri her yerde sıfıra eşit olup **v** sıfır vektör olacaktır.

Teorem 3.5 *(Devirli Permütasyon ile BCH Sınırı) b ve n aralarında asal ve a keyfi bir sabit olsun. Ağırlığı $(d-1)$ veya daha az olan ve Fourier dönüşümü*

$$V_{((a+bl))} = 0 \quad , \quad l = 0,1,\dots,d-1 \tag{3.52}$$

*eşitliğini sağlayan tek **v** vektörü tüm bileşenleri sıfır olan vektördür.*

İspat: Fourier dönüşümünün modülasyon özelliğinden dolayı, spektrum **V**'nin a kadar yer değiştirmesi **v**'nin ağırlığını değiştirmez. Devirli permütasyon özelliği ile dönüşüm **V**'nin $B = b^{-1}(\mathrm{mod}\ n)$ kadar devirli permütasyonu **v**'nin ağırlığını değiştirmediğini söyleyebiliriz. Bu ise spektral **V**'nin verilen $(d-1)$ sıfırının yerini değiştiren, **v**'nin ağırlığı koruyan bir permütasyonunu elde ettiğimizi gösterir. Bir önceki BCH sınırı teoremi ile ispat tamamlanmış olur [16].

3.6 Alt cisimler, Konjugeler ve İdempotentler

F cismi mertebesi n olan bir ω elemanı içeriyor ise n uzunluğunda bir Fourier dönüşümü de vardır. Eğer F, n mertebeli bir elemana sahip değilse bu cisim üzerinde n uzunluklu Fourier dönüşümü yoktur demektir. Eğer genişleme cismi E mertebesi n olan bir ω elemanı içeriyor ise bu durumda E üzerinde n uzunluklu bir Fourier dönüşümü mevcuttur ve bu dönüşüm yine,

$$V_j = \sum_{i=1}^{n-1} \omega^{ij} v_i \qquad j = 0,\dots,n-1 \tag{3.53}$$

şeklinde tanımlanır. Ayrıca ***v*** vektörünün bileşenleri sadece F'de olsa bile ***V*** vektörü genişleme cismi E de bileşenlere sahiptir. Şimdi F^n vektör uzayında tanımlı bir ***v*** vektörünün Fourier dönüşümü olan E^n vektör uzayında tanımlı herhangi bir ***V*** vektörünün yapısını inceleyelim. Teorem 3.2'de konjugelik kısıtı ile verdiğimiz eşitliğe göre konjugelik ilişkisi $V_j^q = V_{((qj))}$ ile belirlidir ve bize genişleme cismi $GF(q^m)$ ile altcisim $GF(q)$ arasındaki konjugelik ilişkisini gösterir. $GF(q^m)$ sonlu cisminde bir β elemanının $i = 1,2,\dots,r-1$ için q^i. kuvvetleri β'nın q'lu konjugeleri olarak tanımlanır.

($r, \beta^{q^r} = \beta$ olacak şekilde en küçük pozitif tam sayıdır.) Ayrıca $\{\beta, \beta^q, \beta^{q^2}, \dots, \beta^{q^{r-1}}\}$ kümesi β'nın q'lu konjugeleridir. Genellikle bir eleman birden çok q'lu konjugeye sahip olur. Eğer $GF(q^m)$'in bir elemanı kendisi dahil r adet konjugeye sahipse, bu durumda eleman $GF(q^r) \subset GF(q^m)$ şeklinde bir alt cismin elemanıdır ve r, m'i böler. Böylece konjugelik altında cisim birbirinden ayrık altkümelere ayrılmış olur ve bu kümelere *konjugelik sınıfları* denir. q'lu konjugelerin kümesi aynı zamanda bir primitif elemanın kuvvetlerinin kümesi olarak da ifade edilebilir.

İkili cisim $GF(2^m)$'de bir β elemanının tüm ikili kuvvetleri β'nın ikili konjugeleri olarak adlandırılır. Örneğin $GF(16)$'da ikili konjugelik sınıfları:

$$\{0\}, \{1,2,4,8\}, \{3,6,12,9\}, \{5,10\}, \{7,14,13,11\}.$$

Konjugelik sınıfları aynı zamanda α'nın kuvvetleri ile de tanımlanabilir, öyle ki $\{\alpha^0\}, \{\alpha^1, \alpha^2, \alpha^4, \alpha^8\}, \{\alpha^3, \alpha^6, \alpha^{12}, \alpha^9\}, \{\alpha^5, \alpha^{10}\}, \{\alpha^7, \alpha^{14}, \alpha^{13}, \alpha^{11}\}$.

Bunlar ikili gösterim ile de,

$$\{0000\}, \{0001\}, \{0011\}, \{0101\}, \{0111\}$$

şeklinde gösterilir. Burada dikkat edilirse her 4 bitlik elemanın devirli olarak yer değiştirmesi sonucu aynı konjugelik sınıfına ait diğer bir eleman elde edilir.

$GF(q^m)$ içinde boyutu 1 olan q'lu konjuge sınıfları $GF(q)$ altcismini meydana getirir. Alt cismin elemanlarını tanımlamak için $GF(q)$'nun her elemanının $\beta^q = \beta$ eşitliğini sağladığını unutmayalım. Ayrıca $x^q - x$ polinomunun $GF(q^m)$ içinde sadece q tane sıfır olabilir ve bunlar 1 uzunluğunda q'lu konjugelik sınıfları içindeki elemanlarıdır. Örneğin $GF(64)$ cisminin $\beta^4 = \beta$ eşitliğini sağlayan dört elemanı altcisim $GF(4)$'ün dört elemanıdır [16].

Tanım 3.2 $GF(q^m)$'in bir q'lu konjugelik sınıfının tüm elemanlarını toplarsak

$$\beta + \beta^q + \beta^{q^2} + \cdots + \beta^{q^{r-1}}$$

elde ederiz. Bu toplam *β'nın q'lu izi (trace)* olarak adlandırılır ve $tr(\beta)$ ile gösterilir. Ayrıca β'nın q'lu izi $GF(q)$'nun bir elemanıdır, çünkü

$$(tr(\beta))^q = (\beta + \beta^q + \beta^{q^2} + \cdots + \beta^{q^{r-1}})^q$$

$$= \beta^q + \beta^{q^2} + \cdots + \beta^{q^{r-1}} + \beta = tr(\beta) \qquad (3.54)$$

$GF(2^m)$'de β'nın tüm ikili konjugelerinin toplamı β'nın ikili izini verir. Aynı konjuge sınıfında olan elemanlar aynı ikili ize sahiptir. $GF(16)$ cisminde, $\alpha^0, \alpha^1, \alpha^3, \alpha^5, \alpha^7$'nin konjugelik sınıflarındaki elemanların ikili izleri sırasıyla 1,0,1,1,1'dir.

Tanım 3.3 $GF(2)$ üzerinde tanımlı Fourier dönüşümünün bileşenleri W_j'ler sadece 0 ve 1 değerlerini alan $w(x)$ polinomuna *ikili idempotent polinom* denir.

$W_j^2 = W_j$ olduğundan konvolüsyon teoremine göre bir idempotent polinom

$$w(x)^2 = w(x) \pmod{x^n - 1} \tag{3.55}$$

eşitliğini sağlamalıdır. Konjugelik ilişkisi $W_j^2 = W_{((2j))}$ olduğundan, $w(x)$ bir idempotent eleman ise her j için α^j aynı konjugelik sınıfında olmak üzere W_j aynı değeri alır. Yani 0 veya 1.

Örneğin $GF(8)$'in ikili konjugelik sınıfları $\{\alpha^0\}, \{\alpha^1, \alpha^2, \alpha^4\}, \{\alpha^3, \alpha^6, \alpha^5\}$'dir. 3 tane konjugelik sınıfı olduğundan ve bunların birleşimlerini almanın 2^3 yolu olduğundan 2^3 idempotent polinom vardır. Bunlardan iki tanesi trivialdir. Trivial olmayanların idempotent polinomlarının spektralleri,

$\mathbf{W} = (1,0,0,0,0,0,0), (0,1,1,0,1,0,0), (0,0,0,1,0,1,1)$ ve bunların ikişerli bileşen bileşen toplamlarıdır. Böyle altı tane trivial olmayan spektral vardır. Bunlar ise

$w(x) = x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$, $x^4 + x^2 + x$, $x^6 + x^5 + x^3$ ve bunların ikişerli tüm toplamları ile oluşan idempotent polinomlara karşılık gelir. Her idempotent polinom $w(x)^2 = w(x) \pmod{x^7 - 1}$ eşitliğini sağlar. Bu eşitliğin tam 6 tane trivial olmayan çözümü vardır ve bunların hepsini bulmuş olduk.

Katsayıları $GF(q^m)$ cisminde olan bir $v(x)$ polinomu hesaplanarak elde edilen $GF(q^m)$ cisminde $j = 0,1,n-1$ için bir V_j dizisi konjugelik özelliğini sağlamak zorundadır. Böyle bir dizinin bağlantı polinomu, lineer yinelemesi ve konjugelik ilişkisi ile ilgili olarak ne söylenebilir? Sıradaki teorem bununla ilgili bir ifade vermektedir.

Teorem 3.6 *Karakteristiği 2 olan bir cisim üzerinde,* $V_j^2 = V_{((2j))}$ *eşitliğini sağlayan herhangi bir* $V_0, V_1, \dots, V_{n-1}$ *dizisi ve herhangi bir* $(\Lambda(x), L)$ *lineer yinelemesi için*

$$V_j = -\sum_{i=1}^{L} \Lambda_i V_{j-i} \qquad j = L, \dots, 2r-1 \tag{3.56}$$

eşitliği sağlanıyorsa bu durumda,

$$V_{2r} = -\sum_{i=1}^{L} \Lambda_i V_{2r-i} \tag{3.57}$$

olur.

İspat: Hipotezden $V_{2r} = V_r^2$ seçelim. İspat, aynı terim için iki farklı gösterimin olduğunu içermektedir. Öncelikle karakteristiği 2 olan bir cisimde $1 + 1 = 0$ olduğunu kullanarak

$$V_r^2 = \left(\sum_{i=1}^{L} \Lambda_i V_{r-i}\right)^2 = \sum_{i=1}^{L} \Lambda_i^2 V_{r-i}^2 = \sum_{i=1}^{L} \Lambda_i^2 V_{2r-2i} \tag{3.58}$$

elde ederiz. İkinci olarak

$$V_{2r} = -\sum_{k=1}^{L} \Lambda_k V_{2r-k} = -\sum_{k=1}^{L} \Lambda_k \left(-\sum_{i=1}^{L} \Lambda_i V_{2r-i}\right) = \sum_{k=1}^{L} \sum_{i=1}^{L} \Lambda_k \Lambda_i V_{2r-k-i} \tag{3.59}$$

yazabiliriz. Simetri özelliğinden $i \neq k$ için her terim iki kere oluştuğu için karakteristiği 2 olan bir cisimde bu iki terimin toplamı 0 olur. Yani sadece köşegendeki ($i = k$ için) elemanlar kalır. Böylece

$$V_{2r} = -\sum_{k=1}^{L} \Lambda_k V_{2r-k} = \sum_{i=1}^{L} \Lambda_i^2 V_{2r-2i} \tag{3.60}$$

elde ederiz. Bu ise ilk bulduğumuz ifade ile aynı olur ve teoremin ispatı biter. Teoremin bir sonucu, eğer $V_0, V_1, \dots, V_{n-1}$ dizisi ikili değerli bir vektörün Fourier dönüşümü ise bu durumda $(\Lambda(x), L)$ lineer yinelemesinin bu diziyi oluşturup oluşturmadığını test etmek için, j'nin sadece tek değerleri için hesaplanan yinelemenin değerlerinin doğrulanması yeterlidir. Teoreme göre j'nin çift değerleri için yineleme otomatik olarak sağlanır [15].

BÖLÜM 4

DEVİRLİ KODLAR ve FOURİER DÖNÜŞÜMÜ

Hata kontrol kodları günümüzde birçok kullanım alanına sahiptir. Bunların başında iletişim sistemleri, manyetik ve optik kayıt sistemleri gelir. Kompakt disk ve dijital video diskleri ise bu uygulamaların en bilindik örnekleridir.

Bu çalışmada hata kodları içinde sadece blok kodlardan bahsedeceğiz. Hata kontrolü için bahsettiğimiz bir blok kod, sonlu bir alfabeden genellikle $GF(q)$ sonlu cisminden, alınan n'liler kümesinden oluşur. Alfabe olarak bir cisim seçilmesinin sebebi ise, kodlama ve dekodlama algoritmalarının yapısını oluşturmak için zengin bir aritmetik yapıya sahip olması ve böylece kullanışlı ve pratik kodların oluşmasına yardımcı olmasıdır.

En popüler blok kodlar lineer kodlardır. Bir kodun lineer olması ise, iki kodsözün bileşen bileşen toplamlarının yine bir kodsöz olması ve bir kodsözün herhangi bir skaler ile çarpılması ile yine bir kodsöz oluşturması anlamındadır. Böylece kodsözler birbirinden farklı olacağı düşünüldüğünden çok sayıda hata düzeltilebilir. Kod sözler arasındaki bu farklılık ise Hamming uzaklığı ile ölçülür.

En önemli blok kod sınıfı *Reed-Solomon kodları*dır. Diğer bir önemli kod sınıfı Reed-Solomon kodların bir alt sınıfı olarak açıklanacak olan *BCH kodları*dır. BCH ve Reed-Solomon kodları, lineer blok kodlar sınıfının içinde bir alt sınıf oluşturan devirli kodlara iki örnek oluşturur.

4.1 Lineer Kodlar, Ağırlık ve Uzaklık

F cismi üzerinde bir lineer (n, k) blok kodu $\mathcal{C}$, *F* üzerinde n'lilerin oluşturduğu vektör uzayının k-boyutlu bir altuzayıdır. Biz genellikle *F* cismi olarak $GF(q)$ sonlu cismini

alacağız. Böylece $GF(q)^n$ cisminin k boyutlu alt uzayı n uzunluklu q^k vektör içerir diyebiliriz. Bir (n, k) kodu, her k sembollü *verisözü* için n sembollü *kodsöz* temsil edecek şekilde kullanılır. Bu durumda kodsözde $n - k$ sembol fazladan olur ki bu ise alınan sözdeki hataların düzeltilmesi için kullanılır. Altuzay için seçilen uygun bir baz ile fazlalık $n - k$ sembol ile sınırlandırılır ve bunlara da *kontrol sembolleri* denir. Kodun oranı ise k/n ile bulunur. Kodun blok uzunluğu *n*, veri uzunluğu ya da boyutu *k*'dır[3].

Tanım 4.1 Bir vektörün *Hamming ağırlığı*, vektörün sıfırdan farklı bileşenlerinin sayısı olarak tanımlanır. Aynı blok uzunluğundaki iki vektör arasındaki *Hamming uzaklığı* iki vektörün farklı bileşenlerinin sayısı olarak tanımlanır. Bir kodun *minimum uzaklığı* ise d_{min} ile gösterilir ve koddaki herhangi iki farklı kodsöz arasındaki minimum Hamming uzaklığı şeklinde tanımlanır. Bir lineer kod için herhangi kodsöz çiftlerinin aralarındaki minimum Hamming uzaklığı, sıfırdan farklı herhangi bir kodsözün Hamming ağırlığına eşittir.

$d = d_{min}$ olmak üzere bir (n, k) kodu (n, k, d) kod olarak da yazabiliriz. Bu notasyon minimum uzaklık bilinmediği zamanlarda $d \leq d_{min}$ olacak şekilde kullanılır. Burada d her zaman d_{min} üzerinden alt sınırdır.

$GF(q)^n$'de bir noktanın t yarıçaplı Hamming küresi, o noktadan en fazla t kadar uzaklıkta olan noktalar kümesidir. Bir kodun her kodsözü için aynı yarıçapa (bir tamsayı) sahip Hamming küreleri olduğunu düşünelim. Bu kürelerin yarıçaplarının hiçbirinin kesişmeyecek şekilde tamsayı olarak büyütüldüğünü varsayalım. Bu durumda elde ettiğimiz değer kodun düzeltebileceği hata sayısıdır. Bu sayıya kodun *paketleme yarıçapı (packing radius)* adı verilir ve t ile gösterilir. Aynı zamanda $d_{min}/2$'den daha küçük en büyük tam sayı olarak tanımlanır. Benzer şekilde bu kürelerin yarıçaplarını tüm noktalar içine alana kadar büyütürsek, bu durumda kodun *örtü yarıçapını (covering radius)* elde ederiz. Örtü yarıçapı ise ρ ile gösterilir. Bir kodda kodsözlere ait, eşit yarıçapa sahip Hamming küreleri ayrık ve tüm uzayı kaplıyor ise bu koda *mükemmel kod* denir.Kodsöz merkezli, Hamming yarıçapı t olan küreler birbirinden ayrık olduğundan, böyle herhangi bir küredeki $GF(q)^n$'in eleman sayısı ile böyle kürelerin eleman sayısı q^k çarpılırsa $GF(q)^n$'in toplam eleman sayısından büyük olamaz. Yani

[3] Buradaki blok uzunluğu ve veri uzunluğu lineer olmayan kodlarda da vardır fakat boyut kavramı yoktur.

$$q^k \sum_{l=0}^{t} (q-1)^l n_l \leq q^n \quad (4.1)$$

dir. Bu eşitsizlik Hamming sınırı olarak bilinir. Mükemmel kodlar bu Hamming sınırını eşitlik olarak sağlarlar ve bunlar çok nadir bulunurlar.

Bir lineer kod $\mathcal{C}$, $GF(q)^n$ vektör uzayının k boyutlu bir alt uzayı olduğundan bu altuzayın herhangi bir bazı ile belirlenebilir. k satırı altuzay için bir baz oluşturan bir matrise kodun *üreteç matrisi* denir ve **G** ile gösterilir. $(n-k)$ satırı ortogonal tümleyen için bir baz oluşturan bir matrise kodun *kontrol matrisi* denir ve **H** ile gösterilir. Sonuç olarak **c**, $\mathcal{C}$'nin bir elemanı olması için gerek ve yeter koşul **c**'nin şu iki eşitlikten birini sağlıyor olmasıdır.

$$\mathbf{c} = \mathbf{aG} \; ya \; da \; \mathbf{cH^T} = \mathbf{0}. \quad (4.2)$$

İlk eşitlik **c** kodsözünün **a** veri sözü cinsinden ifade edildiğini gösterir. İkinci eşitlik ise $\mathcal{C}$'nin $\mathbf{H^T}$'nin *boş uzayı* olduğunu söyler. $\mathcal{C}$ kodunun *w* ağırlığında bir kodsöze sahip olması için gerek ve yeter şart **H** kontrol matrisinin *w* tane lineer bağımlı sütunun olmasıdır.

$\mathcal{C}$ lineer kodunun boyutu **G** üreteç matrisinin rankına eşittir. Çünkü **G** üreteç matrisinin satır sayısı ile **H** kontrol matrisinin satır sayıları toplamı vektör uzayının boyutu olan n'e eşittir. Bir kodun boyutu da kontrol matrisi **H**'nin rankının n'den farkına eşittir.

Bir matrisin rankı onun hem satır hemde sütun rankına eşit olduğundan bir $\mathcal{C}$ lineer kodunun boyutu, üreteç matrisi **G**'nin lineer bağımsız sütunlar kümesinin en büyüğünün kardinalitesine eşittir. Tersine, $\mathcal{C}$'nin minimum uzaklığı ise, **H**'nin $(d-1)$ sütunlu her kümesinin lineer bağımsız olduğu en büyük d sayısına eşittir.

Tanım 4.2 (Singleton Sınırı) Herhangi bir lineer (n, k) blok kodun minimum uzaklığı

$$d_{min} \leq n - k + 1 \quad (4.3)$$

eşitsizliğini sağlar. Bu eşitsizliği eşitlik olarak sağlayan kodlara maksimum uzaklıklı kod denir.

Teorem 4.1 *Bir (n, k) maksimum uzaklıklı kodda, herhangi k tane bileşen veri kısmı olarak seçilebilir ve $GF(q)$'dan keyfi değerler olarak atanabilir, böylece bir tek olan kodsözler belirlenir.*

İspat: d_{min} uzaklığa sahip bir kod $d_{min}-1$ kadar hatayı düzeltebilir. Maksimum uzaklıklı kodlarda $d_{min}=n-k+1$ olduğundan $(n-k)$ atanmamış bileşeni hatalı olarak düşünürsek teorem ispatlanır.

Bir lineer $\mathcal{C}$ kodu $GF(q)^n$ vektör uzayının bir lineer altuzayıdır. Alt uzay olarak $\mathcal{C}$'nin bir ortogonal tümleyeni $\mathcal{C}^{\perp}$ vardır. Bu ortogonal, $\mathcal{C}$'nin her elemanı için $GF(q)^n$'de bunlara ortogonal olan bütün **v** vektörlerini içerir. Bu ise her $c \in \mathcal{C}$ için $\sum_{i=0}^{n-1} c_i v_i$ *iç çarpımının* sıfıra eşit olması anlamına gelir. Yani

$$\mathcal{C}^{\perp}=\left\{v|\sum_{i=0}^{n-1} c_i v_i \; her\; c \in \mathcal{C}\; için\right\} \tag{4.4}$$

şeklinde tanımlanır. Ortogonal tümleyen $\mathcal{C}^{\perp}$ bir lineer koddur ve $\mathcal{C}$'nin *dual kodu* denir. Sırasıyla **H** ve **G** matrisleri $\mathcal{C}^{\perp}$ için üreteç ve kontrol matrisleridir. Sonlu bir cisim üzerinde $\mathcal{C}$ ve $\mathcal{C}^{\perp}$'nin kesişimi boş olmayabilir, yani sıfırdan farklı bir **c** vektörü hem $\mathcal{C}$ de hem de $\mathcal{C}^{\perp}$ de bulunabilir. Hatta $\mathcal{C}=\mathcal{C}^{\perp}$ olabilir. Bu durumda bu koda *self-dual (kendine dual) kod* denir. Ayrıca $dim(\mathcal{C})+dim(\mathcal{C}^{\perp})=n'$dir ve bu eşitlik $\mathcal{C}$ ve $\mathcal{C}^{\perp}$ kümelerinin kesişimi sıfırdan oluşsa dahi doğrudur.

$GF(q^m)$ üzerinde bir $\mathcal{C}$ kodu $GF(q^m)$ cismi üzerinde bileşenlere sahiptir. Aynı zamanda tüm bileşenleri $GF(q)$ alt cisminde olan kodsözleri de olabilir. $\mathcal{C}$ kodun bu şekilde tüm kodsözlerinin oluşturduğu küme $GF(q)$ üzerinde bir kod oluşturur. Bu koda *altcisim-altkod* denir ve

$$\mathcal{C}'=\mathcal{C}\cap GF(q)^n \tag{4.5}$$

şeklinde yazılır. $\mathcal{C}'$ kodunun minimum uzaklığı $\mathcal{C}$ kodunun minimum uzaklığından küçük değildir. Bir altcisim-altkodu altkodların özel bir durumu olarak düşünebiliriz. Genel olarak bir *altkod* bir kodun herhangi bir alt kümesidir. Bir lineer altkod ise kod için tanımlı işlemler altında lineer olan altkoddur [15], [16].

4.2 Devirli Kodlar

Devirli kodlar, bu bölümde ele alacağımız Reed-Solomon ve BCH kodları olarak bilinen kodları da içeren, hata düzelten blok kodlar arasında en önemli sınıfı oluşturur.

Bir F cismi üzerinde tanımlı n uzunluğunda bir *devirli kod*, spektral bileşenleri sıfıra eşit olan belirli bir kümeye sahip n vektörlü tüm **c** kod sözler kümesi olarak tanımlanır.

Böyle vektörlerin kümesi lineer kombinasyonlar altında kapalı olduğundan, tanımdan bir devirli kodun lineer kod olduğunu söyleyebiliriz. Spektral bileşenler ancak Fourier dönüşümün var olduğu durumda var olacağından, bir F cisminde n uzunluklu bir devirli kodun var olması için, aynı cisimde ya da genişleme cisminde n uzunluklu Fourier dönüşümün var olması gerekir.

Kodun *tanım kümesi* olarak adlandırılan spektral indekslerden bir $\mathcal{A} = \{j_1, j_2, \dots, j_r\}$ kümesi alalım. $\mathcal{C}$ kodu, F cismi üzerinde, $l = 1,2, \dots, r$ için Fourier dönüşümü $\mathbf{C}_{j_l} = 0$ olan n uzunluklu tüm **c** vektörlerinin kümesidir. Yani ω, F cismi ya da genişleme cisminde mertebesi n olan bir eleman olmak üzere ve $C_j = \sum_{i=0}^{n-1} \omega^{ij} c_i$ iken $\mathcal{C} = \{c | C_{j_l} = 0 \quad l = 1,2, \dots, r\}$ olur. Burada **C** spektrumuna *kodsöz spektrumu* denir. Ayrıca ters Fourier dönüşümü da, $c_i = \frac{1}{n} \sum_{j=0}^{n-1} \omega^{-ij} C_j$ şekilde tanımlıdır. Eğer F cismi $GF(q)$ sonlu cismi ise, bu durumda n, $(q^m - 1)$`i böler ve ω'da $GF(q^m)$'nin bir elemanıdır. Eğer, $n = q^m - 1$ ise ω, $GF(q^m)$'nin bir primitif elemanıdır ve bu durumda oluşan devirli koda *primitif devirli kod* denir.

Bir primitif devirli kodun $(q^m - 1)$ kodsözünün bileşenlerini indekslemek için her bileşen, $GF(q^m)$'nin bir primitif elemanının $(q^m - 1)$ kuvvetleri olarak tanımlanabilen, sıfırdan farklı bir elemanına atanır. Benzer sekilde n uzunluğunda bir devirli kodun n kodsöz bileşenini indekslemek için, her bileşen n mertebeli ω elemanının n farklı kuvvetinden birine atanır. Kodsözün bileşenleri $i = 0,1, \dots, n-1$ için c_{ω^i} ile gösterilebilir. Fakat bu gösterim tarzı gereksiz şekilde karmaşık olduğundan i'yi ω^i ile tanımlayarak, $i = 0,1, \dots, n-1$ için her bileşeni c_{ω^i} yerine c_i ile gösteririz. Cismin sıfır elemanı bir devirli kod için indeks olarak kullanılmaz. Bir devirli kodun bir **c** kodsözü aynı zamanda bir *kodsöz polinomu* olarak

$$c(x) = \sum_{i=0}^{n-1} c_i x^i \tag{4.6}$$

şeklinde ifade edilebilir. Bir devirli kodun bir kodsöz spektrumu da **C** de *spektrum polinomu* olarak

$$C(x) = \sum_{j=0}^{n-1} C_j x^j \tag{4.7}$$

şeklinde ifade edilebilir. Fourier dönüşümü $C_j = c(w^j)$ ile, ters Fourier dönüşümü ise $c_i = n^{-1}C(w^{-i})$ ile hesaplanır. Eğer ω, $GF(q)$'nun bir elemanı ise bu durumda her C_j spektral bileşeni $GF(q)$'nun bir elemanıdır. Eğer j tanım kümesinde değilse C_j diğer spektral bileşenlerden bağımsız ve keyfi şekilde belirlenebilir. Eğer ω, $GF(q)$'nun elemanı değilse bu durumda bir m için genişleme cismi $GF(q^m)$'nin elemanıdır ve spektral bileşenlerin $C_j^q = C_{((qj))}$ konjugelik kısıtını sağlamaları gerekir. Bu ise, j tanım kümesi $\mathcal{A}$'da iken $qj \pmod n$ de tanım kümesinde olması anlamındadır. Bu durumda tanım kümesi $\mathcal{A}$, her konjuge sınıfından sadece bir veya birkaç üye ile kısaltılabilir ve bu durumda tanım kümesine *tam tanım kümesi* denilir ve $\mathcal{A}_c$ ile gösterilebilir.

Bir kodsöz her zaman, *temel idempotent* denilen ve tek şekilde tanımlı $w(x)$ kodsöz polinomunu içerir. Bu $w(x)$ polinomu, herhangi kodsöz polinomu $c(x)$ için $w(x)c(x) = c(x) \pmod{x^n - 1}$ özelliğini sağlar. Temel idempotent Fourier dönüşümü ile tanımlanabilir. Konvolüsyon özelliğinden, herhangi kodsöz spektrumu **C** için tüm j'lerde $W_jC_j = C_j$ olur. Bu özelliği sağlayan kodsöz spektrumu **W**

$$W_j = \begin{cases} 0 & j \in \mathcal{A}_c \\ 1 & j \notin \mathcal{A}_c \end{cases} \tag{4.8}$$

ile verilir ve bu spektrum tek bir kodsöz için geçerlidir.

Bir kodsöz her zaman *üreteç polinomu* denilen $g(x)$ polinomu içerir. Bu $g(x)$ polinomu minimum derecede monik kodsöz polinomudur. Böyle bir monik polinom bir tekdir. Gerçekten eğer böyle mininum derecede monik iki tane polinom olsa bunların farkı daha az derecede yeni bir kodsöz polinomu oluşturur. Bu ise bir skalerle tekrar monik hale getirilebilir. Her $c(x)$ kodsöz polinomu, $g(x)$ ile bölününce kalan sıfıra eşit olan bir polinom olmak zorundadır. Yoksa kalan polinom, derecesi $g(x)$'den daha küçük olan bir kodsöz polinomu olur. Yani her kodsöz $g(x)$'in bir katı olmalıdır ve $c(x) = a(x)g(x)$ şeklinde yazılmalıdır. Böylece kodun boyutu $k = n - \deg(g(x))$ olur.

Fourier dönüşümünün öteleme(translation) özelliğinden dolayı, eğer **c** devirli bir şekilde b kadar öteleniyorsa C_j de C_jw^{jb} ile yer değiştirir ki C_j sıfır olunca o da sıfır olur. Bir devirli kodun herhangi bir kodsözünün devirli şekilde ötelenmesi sonucu oluşan kod, yine aynı devirli kodun başka bir kodsözünü oluşturur. Bu özellik *devirli olma özelliği* olarak bilinir. Biz bu çalışmada kodların bu özelliği ile ilgilenmesek de devirli

kodlar adlarını bu özelliklerinden almışlardır ve önemli bir özelliktir. Fakat devirli kodlar sadece bu özellikleri ile değil Fourier dönüşümü özellikleri ile minimum uzaklıklarının bulunması ve bu sayede kodlama ve dekodlama işlemlerinin yapılması açısından da çok önemlidirler.

$\mathcal{C}$ kodu devirli olduğundan $xc(x) \pmod{x^n - 1}$ de aynı zamanda kodun içindedir. Benzer şekilde herhangi bir $a(x)$ polinomu için $a(x)c(x) \pmod{x^n - 1}$ de koddadır. Bölme algoritmasından

$$x^n - 1 = Q(x)g(x) + r(x) \tag{4.9}$$

olur ve burada kalan polinom $r(x)$'in derecesi $g(x)$'in derecesinden küçük olup $r(x)$ sıfırdan farklı bir kodsöz olamaz. Fakat $r(x)$ kodsöz olması için spektral sıfırlarının olması gerekir ve dolayısı ile sıfır kodsözdür, yani $r(x) = 0$'dir ve böylece *kontrol polinomu* denen bir $h(x)$ polinomu için

$$g(x)h(x) = x^n - 1 \tag{4.10}$$

olur. Devirli kodlar konusundaki çalışmalarımızdaki en temel amaç kodun minimum uzaklığını bulmaktır. Devirli kodlar lineer olduğundan bir kodun minimum uzaklığını bulmak, kodun sıfırdan farklı herhangi kodsözleri için en küçük Hamming ağırlığını bulmak ile aynı şeydir. Kod tanım kümesiyle tamamen belirli olduğundan, minimum uzunluk da kodun tanım kümesinin bilinmesiyle hemen bilinir. Bu durumda bir vektörün ağırlığı ile vektörün Fourier dönüşümünün içerdiği sıfırlar arasındaki ilişki devirli kodların yapısının temelini oluşturur.

$GF(q)$ üzerinde bir $g(x)$ polinomu aynı zamanda $GF(q^m)$ üzerinde bir polinom olarak düşünülebilir. $g(x)$ üreteç polinomu olarak kullanıldığında $GF(q)$ veya $GF(q^m)$ üzerinde bir devirli kod tanımlayabilir.

Teorem 4.2: *$g(x)$ polinomu $GF(q)$ üzerinde tanımlı ($x^{q^m-1} - 1$)'i bölen bir polinom olsun. $GF(q)$ üzerinde tanımlı $g(x)$ tarafından üretilen devirli kod ve $GF(q^m)$ üzerinde tanımlı $g(x)$ tarafından üretilen devirli kod aynı minimum uzaklığa sahiptir.*

İspat: $\mathcal{C}_q$ ve $\mathcal{C}_{q^m}$ sırasıyla $GF(q)$ ve $GF(q^m)$ üzerinde tanımlı kodlar olsunlar. $\mathcal{C}_q \subset \mathcal{C}_{q^m}$ olduğundan bu durumda $d_{min}(\mathcal{C}_q) \geq d_{min}(\mathcal{C}_{q^m})$ olur. $c(x)$ polinomu $\mathcal{C}_{q^m}$ içindeki minimum ağırlıklı kodsöz polinomu olsun. Buradan $a(x)$ ve $c(x)$

polinomlarının katsayıları $GF(q^m)$'de olmak üzere $c(x) = a(x)g(x)$ olur. $\boldsymbol{c}$'nin bileşenleri

$$c_i = \sum_{j=0}^{k-1} g_{i-j}\mathrm{a_j} \tag{4.11}$$

olsun. i. bileşeni $\mathbf{c}$'nin q-lu izinin i. bileşenine eşit olan sıfırdan farklı bir vektör $\boldsymbol{c}'$ alalım. $\boldsymbol{c}'$'nün sıfır vektör olmadığını kabul edebiliriz. Çünkü eğer sıfır vektör olursa bunun yerine c_i'ler sıfır olmadığı sürece her γ için $tr(\gamma c_i)$ sıfır olmayan $\gamma\boldsymbol{c}$ kodsözünü gözönüne alırız. Bu durumda

$$c'_i = tr(c_i) = tr\sum_{j=0}^{k-1} g_{i-j}a_j = \sum_{j=0}^{k-1} tr\left(g_{i-j}a_j\right) \tag{4.12}$$

olur. g_{i-j}, $GF(q)$'nun bir elemanı olduğundan, q. kuvvetine eşittir ve izleri çarpanlarına ayrılabilir. Buradan

$$c'_i = \sum_{j=0}^{k-1} g_{i-j}tr\left(a_j\right) = \sum_{j=0}^{k-1} g_{i-j}a'_j \tag{4.13}$$

sonucuna varırız. Böylece $c'(x)$ polinomunun $g(x)a'(x)$ ile verildiğini ve $\mathcal{C}_q$'da bir kodsöze karşılık geldiğini görürüz. Fakat iz fonksiyonu c_i'nin bir sıfır bileşeninden, c_i'nün sıfır olmayan bir bileşenini oluşturamaz. Yani, $\mathbf{c}'$'nin ağırlığı $\mathbf{c}$'nin ağırlığından daha büyük olamaz. Sonuçta

$$d_{min}(\mathcal{C}_q) \leq d_{min}(\mathcal{C}_{q^m}) \tag{4.14}$$

bulunur ve iki eşitsizlikten eşitlik elde ederiz.

4.3 Reed-Solomon Kodları

Tanım 4.3 F, mertebesi n olan bir elemana sahip bir cisim olsun. F üzerinde tanımlı bir $(\mathrm{n}, \mathrm{k}, \mathrm{d})$ devirli Reed-Solomon kodu, Fourier dönüşüm kümesinde sıfıra eşit olan, $\{\mathrm{j}_0, \mathrm{j}_0 + 1, \dots, \mathrm{j}_0 + \mathrm{d} - 2\}$ indis kümesiyle belirli $(\mathrm{d} - 1)$ ardışık spektral bileşene sahip, n uzunluklu tüm vektörlerin kümesidir.

Bir Reed-Solomon kodunu lineer cebir dilinde de ifade edebiliriz. Fourier dönüşümü F^{n}'den F^{n}'e tersinir lineer bir dönüşümdür. Fourier dönüşümü sonucunda dönüşüm herhangi bir belirli ardışık $(\mathrm{d} - 1)$ bileşene kısıtlanırken, bu durumda kısıtlanmış Fourier

dönüşümü, $n-k=d-1$ olmak üzere n boyutlu vektör uzayından $(n-k)$ boyutlu vektör uzayına lineer bir dönüşüm olarak düşünülebilir. Reed Solomon kodu bu dönüşümün boş uzayı olarak tanımlanır. Benzer şekilde, ters Fourier dönüşümü da F^n'den F^n'e tersinir lineer bir dönüşümdür. Tamamı sıfıra eşit olan $(d-1)$ tane ardışık bileşenden oluşan tüm vektörleri içeren F^n'in k boyutlu bir alt uzayına uygulanırsa ters Fourier dönüşümü k boyutlu vektör uzayından n boyutlu vektör uzayına bir dönüşüm olarak düşünülebilir. Reed-Solomon kod bu dönüşümün görüntü kümesidir. Yani boyutu k'dır.

n, $(q-1)$'i böldüğünden kodsöz bileşenleri ve spektrum bileşenleri aynı cisim $GF(q)$'nun elemanlarıdır. Eğer **c** kodsözü devirli olarak bir yer değiştiriyorsa onun spektral bileşeni C_j, ω^j ile çarpılır. C_j sıfır olduğunda $C_j\omega_j$ de sıfır olacağından **c**'nin devirli yer değiştirmesi de bir kodsözdür. Yani $\mathcal{C}$ bir devirli koddur. Reed Solomon kod bir devirli kod olduğundan bir üreteç polinomu vardır. $GF(q)$'da bir β elemanının aynı cisim üzerinde minimal polinomu $f_\beta = x-\beta$ olur.

Sembollerin cismi ile dönüşüm aynı cisim üzerinde olduğundan tüm minimal polinomlar birinci derecedendir. Bu durumda $\{\omega^{j_0}, \omega^{j_0+1}, \dots, \omega^{j_0+d-2}\}$ kümesi kod için tek tanım kümesidir ve aynı zamanda tam tanım kümesidir.

Bir Reed-Solomon kod için herhangi bir j_0 seçebiliriz. Bu durumda üreteç polinomu

$$g(x) = (x-\omega^{j_0})(x-\omega^{j_0+1})\cdots(x-\omega^{j_0+d-2}) \qquad (4.15)$$

olur. Bu polinomun derecesi her zaman $(d-1)$'dir. Buradan bir Reed Solomon kod için

$$n-k=d-1 \qquad (4.16)$$

olduğunu söyleyebiliriz. Eğer ω'nın mertebesi $(q-1)$ ise $GF(q)$ üzerinde bir Reed Solomon kodun blok uzunluğu maksimum olur. Bu durumda ω bir primitif eleman olur ki bu durumda α ile gösterilir, kodun blok uzunluğu $n=q-1$ olur. Bu durumda Reed Solomon kod primitif Reed Solomon kod olarak adlandırılır.

Sıradaki teorem, cebirin temel teoremi ile ispatlanır, Singleton sınırına göre Reed Solomon kodun bir optimal kod olduğunu göstermektedir. Böylece spektrumunda $(d-1)$ tane ardışık sıfır olan bir Reed-Solomon kod $d_{min}=d$ olacak şekilde minimum uzaklığa sahiptir.

Teorem 4.3 *Bir Reed-Solomon kodu maksimum uzaklıklı koddur ve minimum uzaklığı*

$$d_{min} = n - k + 1 \tag{4.17}$$

eşitliğini sağlar.

İspat: Tanım kümesi olarak ***C***'nin son $(\mathrm{d}-1)$ bileşeninin sıfıra eşit olduğunu düşünmek yeterlidir. Başka şekilde, tanım kümesini son $(\mathrm{d}-1)$ bileşene taşımak için modülasyon teoremini kullanarak her kodsöz bileşenini ω'nın bir kuvveti ile çarparsak bir kodsözün ağırlığı değişmez, çünkü sıfır olmayan bileşenler yine sıfır olmayacaktır.

$$\mathrm{C}(\mathrm{x}) = \sum_{\mathrm{j}=0}^{\mathrm{n}-\mathrm{d}} \mathrm{C}_{\mathrm{j}}\mathrm{x}^{\mathrm{j}} \tag{4.18}$$

olsun. Bu durumda buna karşılık gelen kodsöz bileşenleri $\mathrm{c}_{\mathrm{i}} = \frac{1}{\mathrm{n}}\mathrm{C}(\omega^{-\mathrm{i}})$ olur. $\mathrm{C}(\mathrm{x})$ derecesi en fazla $(\mathrm{n}-\mathrm{d})$ olan bir polinom olduğundan en fazla $(\mathrm{n}-\mathrm{d})$ tane sıfırı olabilir. Bu durumda eğer tamamen sıfır değil ise ***c*** en az d bileşende sıfır değildir. Minimum uzaklık $\mathrm{d}_{\min}$, Reed-Solomon kodu için $\mathrm{d}-1 = \mathrm{n}-\mathrm{k}$ olduğu için

$$\mathrm{d}_{\min} \geq \mathrm{d} = \mathrm{n} - \mathrm{k} + 1 \tag{4.19}$$

eşitsizliğini sağlar. Aynı zamanda herhangi bir lineer kod Singleton sınırından

$$\mathrm{d}_{\min} \leq \mathrm{n} - \mathrm{k} + 1 \tag{4.20}$$

eşitsizliğini sağlar. Böylece $\mathrm{d}_{\min} = \mathrm{n} - \mathrm{k} + 1$ yani $\mathrm{d}_{\min} = \mathrm{d}$ olup ispat biter.

Bu teoremden, sabit n ve k tam sayıları için hiçbir kodun Reed-Solomon koddan daha büyük bir minimum uzaklığa sahip olamayacağını çıkartabiliriz. Bu ise bize Reed-Solomon kodu kullanmak için güçlü bir gerekçe verir. Fakat bu olduğundan daha çok şey söylediği anlamına gelmemelidir. Çünkü örneğin herhangi $(\mathrm{n}', \mathrm{k}')$ parametrelerine sahip bir kod, bu parametreler için var olmayan fakat bir (n, k) Reed-Solomon kodundan daha tercih edilebilir olabilir.

Reed-Solomon kodlar diğer devirli kodlara kıyasla aynı alfabe üzerinde daha kısa blok uzunluğuna sahiptirler. Eğer alfabe küçük ise bu durumda pek işe yaramazlar. Örneğin $\mathrm{GF}(2)$ üzerindeki ikili alfabedeki Reed-Solomon kodların blok uzunluğu 1 olup hiç enteresan değildir [15], [16].

4.4 Reed-Solomon Kodları İçin Kodlayıcılar

Bir kodlayıcı bir k sembollü veri sözünü n sembollü kodsöze dönüştüren bir kural veya bu dönüşümü yapan bir araçtır. Bir kodun çok çeşitli kodlayıcıları olabilir. Herhangi bir kodlayıcı onu dizayn edenin isteğine veya gerek duyduğu özelliklere göre değişebilir.

$\cdots$	0	0	0	$\cdots$	0	a_0	a_1	a_2	$\cdots$

Şekil 4.1 Spektral sıfırların yerleşmesi [16]

Basit bir kodlama kuralını, Şekil 4.1'de gösterildiği gibi

$$C_j = \begin{cases} a_{j-j_0-n+k}, & j = j_0 + n - k, \dots, j_0 + n - 1 \\ 0, & j = j_0, j_0 + 1, \dots, j_0 - 1 + n - k \end{cases} \tag{4.21}$$

spektrumun k bağımsız bileşenine $l = 0,1, \dots, k-1$ için k adet veri sembolü a_l'leri ekleme şeklinde tanımlayabiliriz. Ters Fourier dönüşümü ile kodlama işlemi tamamlanır. Kodsözleri

$$c_i = \frac{1}{n}\sum_{i=0}^{n-1} \omega^{-ij} C_j \qquad i = 0,1, \dots, n-1 \tag{4.22}$$

ile elde ederiz. Burada ***c*** kodsözünün n bileşeni içinden k veri sembolünü hemen bulmak mümkün değildir. Veri sözünü elde etmek için ***c***'nin Fourier dönüşümünü hesaplamak gerekir. Bu dekodlayıcıya dönüşüm bazlı kodlayıcı denir.

Daha popüler bir kodlayıcı ise kod bazlı kodlayıcı olarak bilinen kodlayıcıdır. Kodun üreteç matrisini şöyle tanımlayalım:

$$\begin{aligned} g(x) &= (x - \omega^{j_0})(x - \omega^{j_0+1}) \cdots (x - \omega^{j_0+n-k-1}) \\ &= x^{n-k} + g_{n-k-1}x^{n-k-1} + g_{n-k-2}x^{n-k-2} + \cdots + g_1 x + g_0. \end{aligned} \tag{4.23}$$

$g(x)$'in katsayıları **g** vektörünün bileşenleridir. Bu vektörün Fourier dönüşümü **G** olup, bu dönüşümde $(j_0, j_0 + 1, \dots, j_0 + n - k - 1)$ için bileşenleri sıfırdır. Yani $g(x)$'in kendisi de ağırlığı $(n - k + 1)$'den büyük olmayan bir kodsözdür. (BCH sınırına göre minimum ağırlık $(n - k + 1)$'den küçük olamayacağından, bu Reed-Solomon kodların minimum ağırlığının $(n - k + 1)$ olduğunu göstermek için alternatif bir yoldur.)

Diğer bir kodlayıcıyı şöyle tanımlayabiliriz. k tane veri sembolü veri polinomu oluşturmak için kullanılarak

$$a(x) = \sum_{i=0}^{k-1} a_i x^i \qquad (4.24)$$

şeklinde polinom oluşturulur ve $c(x) = g(x)a(x)$ olur ve $c(x)$ 'in derecesi de

$$\deg\big(c(x)\big) = \deg(g(x)) + \deg(a(x)) \qquad (4.25)$$

şeklinde bulunur. Eğer $a(x)$ maksimum dereceye sahipse yani derecesi $k-1$ ise

$$\deg\big(c(x)\big) = (n-k) + (k-1) = n-1 \qquad (4.26)$$

olup $c(x)$'de maksimum dereceye sahip olur. Böylece $a(x)$ ve $g(x)$ polinomlarının çarpımı ile kodsözün tüm n bileşenleri tamamlanmış olur. Burada yine $c(x)$ içinde k veri sembolü hemen bulunamaz. Bunun için

$$a(x) = \frac{c(x)}{g(x)} \qquad (4.27)$$

polinom bölmesi yaparak kolaylıkla kodsözü bulabiliriz. Bu kod bazlı kodlayıcı, dönüşüm bazlı kodlayıcı ile tam olarak aynı kodsöz kümesini verir, fakat veri sözleri ile kodsözler arasındaki ilişki farklıdır.

Çizelge 4.1 $GF(2^3)$ Cisminin Gösterimi

Üstel Notasyon	Polinom Notasyonu	İkili Notasyon	Sekizlik Sistemde	Minimal Polinom
0	0	000	0	-
α^0	1	001	1	$x+1$
α^1	z	010	2	x^3+x^2+1
α^2	z^2	100	4	x^3+x^2+1
α^3	z^2+1	101	5	x^3+x+1
α^4	z^2+z+1	111	7	x^3+x^2+1
α^5	$z+1$	011	3	x^3+x+1
α^6	z^2+z	110	6	x^3+x+1

t hata düzelten bir Reed-Solomon kod için $n-k = d-1 = 2t$ seçelim. $j_0 = 1$ seçimi ile üreteç polinom

$$g(x) = (x-\omega)(x-\omega^2)\cdots(x-\omega^{2t}) \qquad (4.28)$$

olur ve bu polinomun derecesi her zaman $2t$'dir. Örnek olarak, $t = 1$ olmak üzere $GF(8)$ üzerinde bir $(7,5)$ Reed-Solomon kod için $g(x)$ üreteç polinomunu bulalım. Herhangi bir j_0 için bu yapılabilir. $j_0 = 1$ ve $\omega = \alpha$ seçelim. $GF(8)$'in elemanları Çizelge 4.1 deki gibi z'nin polinomları olarak ifade edilmek üzere, üreteç polinom

$$g(x) = (x - \alpha^1)(x - \alpha^2) = x^2 + (z^2 + z)x + (z^2 + 1) \tag{4.29}$$

şeklindedir.

Bir veri polinomu, 5 tane sekizli(oktal) sembollerin dizisinden oluşur. Varsayalım ki $a(x) = (z^2 + z)x^4 + x^3 + (z + 1)x^2 + zx + (z^2 + 1)$ polinomu veri polinomu olsun. Sistematik olmayan kodsöz ise 7 tane sekizli sembollerin dizisinden oluşur ve polinomu

$$\begin{aligned} c(x) = a(x)g(x) &= (\alpha^4 x^4 + x^3 + \alpha^3 x^2 + \alpha x + \alpha^6)(x^2 + \alpha^4 x + \alpha^3) \\ &= \alpha^4 x^6 + \alpha^3 x^5 + \alpha^2 x^4 + 0x^3 + \alpha^5 x^2 + \alpha^6 x + \alpha^2 \end{aligned} \tag{4.30}$$

şeklindedir. Bu kodsöz aynı zamanda şu notasyon ile de yazılabilir;

$$c(x) = (z^2 + z)x^6 + (z + 1)x^5 + z^2 x^4 + 0x^3 + (z^2 + z + 1)x^2 + (z^2 + 1)x + z^2$$

x'in kuvvetlerinin katsayıları sekizli kod sembolleridir ve her kod sembolünde z'nin kuvvetlerinin katsayıları bu sekizli sembolün üç bileşenlik pozisyonlarını indeksler.

$a(x)$'in sekizli veri sembolleri $8^5 = 32768$ yolla belirlenebilir. Bu durumda $(7,5)$ Reed-Solomon kodunda 32768 tane kodsöz vardır. Bu kodsözlerden bazıları Çizelge 4.2'de gösterilmiştir. Fakat görüldüğü gibi bu küçük kodda bile listemek için çok fazla kodsöz vardır. Bu listede kodsözler sistematik kodlayıcı ile üretilmiş gibi düzenlenmiştir, ilk beş sembol veri sembolleri son iki sembol ise kontrol sembolleri gibi düşünülebilir.

İkinci bir örnek olarak, $GF(8)$ üzerinde $t = 2$ ile $(7,3)$ Reed-Solomon kod için $g(x)$ polinomunumu bulalım. Herhangi bir j_0 ile yapabiliriz. $j_0 = 4$ ve $\omega = \alpha$ seçelim. Bu durumda $GF(8)$ cisminin elemanları z'nin polinomları olarak gösterilmek üzere

$$\begin{aligned} g(x) &= (x - \alpha^4)(x - \alpha^5)(x - \alpha^6)(x - \alpha^0) \\ &= x^4 + (z^2 + 1)x^3 + (z^2 + 1)x^2 + (z + 1)x + z \end{aligned} \tag{4.31}$$

şeklinde bulunur.

Bir veri polinomu üç tane sekizli sembolün dizisini gösterir. Varsayalım ki veri polinomu

$$a(x) = (z^2 + z)x^2 + x + (z + 1) \tag{4.32}$$

olsun. Sistematik olmayan kodsöz

$$c(x) = a(x)g(x)$$

$$= (\alpha^4 x^2 + x + \alpha^3)(x^4 + \alpha^6 x^2 + \alpha^3 x + \alpha)$$

$$= \alpha^4 x^6 + \alpha x^5 + \alpha^6 x^4 + 0x^3 + 0x^2 + \alpha^5 x + \alpha^4 \qquad (4.33)$$

şeklinde bulunur ve yedi tane sekizli sembolden oluşur.

Çizelge 4.2 $(7,5)$ Reed Solomon kodu

0	0	0	0	0	0	0
0	0	0	0	1	6	3
0	0	0	0	2	7	6
0	0	0	0	3	1	5
			⋮			
0	0	0	1	0	1	1
0	0	0	1	1	7	2
0	0	0	1	2	6	7
0	0	0	1	3	0	4
			⋮			
0	0	0	7	0	7	7
0	0	0	7	1	1	4
0	0	0	7	2	0	1
0	0	0	7	3	6	2
			⋮			
0	0	1	0	0	7	3
0	0	1	0	1	1	0
0	0	1	0	2	0	5
0	0	1	0	3	6	6
			⋮			
0	0	1	1	0	6	2
0	0	1	1	1	0	1
0	0	1	1	2	1	4
0	0	1	1	3	7	7
			⋮			

Üçüncü bir örnek olarak $GF(16)$ üzerinde $t = 2$ ile $(15,11)$ Reed-Solomon kodu için bir $g(x)$ polinomu bulalım. $j_0 = 1$ ve $\omega = \alpha$ seçelim.

$$g(x) = (x - \alpha)(x - \alpha^2)(x - \alpha^3)(x - \alpha^4) \tag{4.34}$$

olur. Çizelge 4.3 den yararlanarak $p(z) = z^4 + z + 1$ primitif polinomu ile $GF(2)$'nin bir genişleme cismi olarak $GF(16)$'nın oluşturulmasıyla üreteç polinomu,

$$g(x) = x^4 + (z^3 + z^2 + 1)x^3 + (z^3 + z^2)x^2 + z^3x + (z^2 + z + 1)$$

$$= x^4 + \alpha^{13}x^3 + \alpha^6x^2 + \alpha^3x + \alpha^{10} \tag{4.35}$$

şeklinde bulunur. $g(x)$ dördüncü dereceden olduğundan $n - k = 4$ ve $k = 11$ olur. Veri polinomu onbir tane 16'lı (hexadesimal) sembolden oluşur.

Son olarak tanımlayacağımız kodlayıcı daha kullanışlıdır, çünkü kodsöz içerisinde veri sembolleri açıkça bellidir. Bu özelliği taşıyan kodlayıcılara sistematik kodlayıcı denir.

Öncelikle $x^{n-k}a(x)$ polinomunun $a(x)$ polinomu ile aynı katsayılara sahip olduğuna ve sadece bu katsayıları $n - k$ kadar kaymış durumda olduğuna dikkat edelim. $x^{n-k}a(x)$ polinomunun $n - k - 1, n - k - 2, \ldots, 0$ indisli $(n - k)$ tane katsayısı sıfır olur. Reed-Solomon kodun bir kodsözünü üretmek için bu $(n - k)$ tane pozisyona kontrol sembollerini ekleyeceğiz. Özel olarak, $R_{g(x)}[x^{n-k}a(x)]$ polinomu $x^{n-k}a(x)$'in $g(x)$'le bölümünden kalan polinom olmak üzere

$$c(x) = x^{n-k}a(x) - R_{g(x)}\left[x^{n-k}a(x)\right] \tag{4.36}$$

olsun. Kalan polinomunun katsayıları tam olarak $x^{n-k}a(x)$ polinomunun katsayılarının sıfır olduğu yerlerdedir. Yani $c(x)$ polinomunu oluşturan parçalar çakışmaz. Böylece veri polinomu $a(x)$, $c(x)$ içinde hemen görünür.

$c(x)$'in geçerli bir kodsöz polinomu olduğunu görmek için, kalan işleminin toplama üzerine dağılma özelliği ve kalanın kalanı da kalandır gerçeği de gözönünde bulundurularak $c(x)$ polinomunun $g(x)$ ile bölümünden kalan hesaplanır.

$$R_{g(x)}[c(x)] = R_{g(x)}[x^{n-k}a(x) - R_{g(x)}[x^{n-k}a(x)]]$$

$$= R_{g(x)}[x^{n-k}a(x)] - R_{g(x)}[x^{n-k}a(x)] = 0 \tag{4.37}$$

Böylece $c(x)$'in $g(x)$'in bir katı olduğunu görürüz. Bu ise $c(x)$'in doğru spektral sıfırlara sahip olduğu, yani Reed-Solomon kodun bir kodsözü olduğu anlamına gelir. Sistematik

kodlayıcı diğer iki kodlayıcı ile aynı kodsöz kümesini elde eder, fakat veri sözleri ile kod sözler arasıdaki ilişki farklıdır.

Çizelge 4.3 $GF(2^4)$ Cisminin Gösterimi

Üstel Notasyon	Polinom Notasyon	İkili Notasyon	Onluk Sistemde	Minimal Polinom
0	0	0000	0	-
α^0	1	0001	1	$x+1$
α^1	z	0010	2	x^4+x+1
α^2	z^2	0100	4	x^4+x+1
α^3	z^3	1000	8	$x^4+x^3+x^2+x+1$
α^4	$z+1$	0011	3	x^4+x+1
α^5	z^2+z	0110	6	x^2+x+1
α^6	z^3+z^2	1100	12	$x^4+x^3+x^2+x+1$
α^7	z^3+z+1	1011	11	x^4+x^3+1
α^8	z^2+1	0101	5	x^4+x+1
α^9	z^3+z	1010	10	$x^4+x^3+x^2+x+1$
α^{10}	z^2+z+1	0111	7	x^2+x+1
α^{11}	z^3+z^2+z	1110	14	x^4+x^3+1
α^{12}	z^3+z^2+z+1	1111	15	$x^4+x^3+x^2+x+1$
α^{13}	z^3+z^2+1	1101	13	x^4+x^3+1
α^{14}	z^3+1	1001	9	x^4+x^3+1

4.5 BCH Kodları

d minimum uzaklığına, $(q-1)$'i bölen n blok uzunluğuna sahip bir devirli Reed-Solomon kodu, $GF(q)$ üzerinde, $\{\omega^{j_0}, \omega^{j_0+1}, \dots, \omega^{j_0+d-2}\}$ tanım kümesi ile belirli

$(d-1)$ ardışık bileşeninin spektrumu sıfır olan tüm sözler kümesi olduğunu hatırlayalım. Bir Reed-Solomon kodda, kod sembolleri ve spektrum sembolleri aynı $GF(q)$ cismi üzerindedir. d tasarlanmış uzaklığına ve (q^m-1)'i bölen n blok uzunluğuna sahip bir BCH kodu ise, $GF(q)$ üzerinde belirli $(d-1)$ ardışık bileşeninin spektrumu sıfıra eşit olan tüm sözler kümesidir. BCH sınırına göre BCH kodunun minimum uzaklığı d_{min} en az tasarlanmış uzaklık d kadar olması gerekir. Fakat bazen bir BCH kodun minimum uzaklığı tasarlanmış uzaklığından daha büyük olabilir. Bir BCH kodun her kodsözünün spektrumu genişleme cismi $GF(q^m)$ üzerindedir ve $\{\omega^{j_0}, \omega^{j_0+1}, \dots, \omega^{j_0+d-2}\}$ tanım kümesi ile belirli $(d-1)$ ardışık bileşeninin spektrumu sıfıra eşit olmak zorundadır. Böylece $GF(q)$ üzerinde BCH kodun kodsözleri $GF(q^m)$ üzerinde Reed-Solomon kodun kodsözleridir. Bunlar sadece Reed-Solomon kodun $GF(q)$ üzerinde değerler alan kodsözleridir. Böylece her BCH kod bir Reed-Solomon kodun altcisim-alkodudur.

Çizelge 4.4'de $GF(8)$ üzerinde $(7,5,3)$ Reed-Solomon kodun 8^5 adet kodsözlerinden bir kısmı listelenmiştir. Bu listeyi incelediğimizde bazı kodsözlerin sadece ikili değerler, yani 0 ve 1 değerlerini aldığını görürüz. $(7,5,3)$ Reed-Solomon kodun tüm kodsözlerini incelersek böyle onaltı tane kodsöz olduğunu görürüz. Bu kodsözler $GF(2)$ üzerinde bir lineer devirli kod oluştururlar. Bu koda $GF(2)$ üzerinde $(7,4,3)$ BCH kodu, aynı zamanda $(7,5,3)$ Hamming kodu denir. Bu örnekte $GF(2^3)$ üzerindeki Reed-Solomon kodun bir altcisim-altkodu olarak $GF(2)$ üzerinde BCH kodunu görmüş olduk.

$GF(q)$ üzerinde n uzunluklu bir kod aradığımızda, $GF(q)$'da mertebesi n olan bir ω elemanı yokken $GF(q^m)$'da böyle bir eleman varsa bu durumda BCH kod işimize yarar. Böyle bir durumda kodsöz bileşenlerimiz $GF(q)$'da iken bunların spektral bileşenleri $GF(q^m)$'de olur. Kısaca $GF(q^m)$ üzerinde bir (n,k,d) Reed-Solomon $\mathcal{C}$ kodu ile elde edilen BCH kodunu şöyle tanımlayabiliriz: $\mathcal{C}' = \mathcal{C} \cap GF(q)^n$.

BCH kodun minimum uzaklığı en az tasarlanmış uzaklık d kadar olur. Reed-Solomon kodun paketleme yarıçapı her zaman $t = \lfloor (d-1)/2 \rfloor$ olsa da d_{min}, d'den daha büyük olabileceğinden BCH kodun paketleme yarıçapı $\lfloor (d-1)/2 \rfloor$'den daha büyük olabilir. Bu noktada tasarlanmış uzaklığı d^* ile gösterip, t'yi de BCH yarıçapı olarak adlandırabiliriz. BCH sınırı ile paketleme yarıçapının en az BCH yarıçapı t kadar olacağı garantilenir.

Çizelge 4.4 $(7,5)$ Reed-Solomon kodundan bir altcisim-altkodun çıkartılması

Reed-Solomon Kod							Altcisim-Altkod						
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	6	3							
0	0	0	0	2	7	6							
0	0	0	0	3	1	5							
			⋮										
0	0	0	1	0	1	1	0	0	0	1	0	1	1
0	0	0	1	1	7	2							
0	0	0	1	2	6	7							
0	0	0	1	3	0	4							
			⋮										
0	0	0	7	0	7	7							
0	0	0	7	1	1	4							
0	0	0	7	2	0	1							
0	0	0	7	3	6	2							
			⋮										
0	0	1	0	0	7	3							
0	0	1	0	1	1	0	0	0	1	0	1	1	0
0	0	1	0	2	0	5							
0	0	1	0	3	6	6							
			⋮										
0	0	1	1	0	6	2							
0	0	1	1	1	0	1	0	0	1	1	1	0	1
0	0	1	1	2	1	4							
0	0	1	1	3	7	7							
			⋮										

Konjugelik kısıtı $C_j^q = C_{((qj))}$, ters Fourier dönüşümü $GF(q)$'da olan spektrumu tanımlamak için tam bir formül verir. BCH kodun her kodsözü spektrumunda bu kısıtı

sağlamak zorundadır. BCH sınırı, minimum uzaklığı en az tasarlanmış uzaklık kadar olan bir kod oluşturmak için bir koşul verir. Hem BCH sınırı hem de $GF(q^m)$'de spektrum bileşenlerinin konjugelik kısıtı kullanılarak oluşturulan $GF(q)$ üzerinde bir kod BCH koddur.

$(7,4)$ ikili BCH kodu BCH sınırı teoremini kullanarak da açıklayalım. Her spektral C_j bileşeni kodsöz bileşeninin ikili değerler aldığını garantilemek için $C_j^2 = C_{2j}$ eşitliğini sağlamak zorundadır. Bu nedenle tek bileşendeki bir hata düzeltebilmek için C_1 ve C_2 bileşenleri kontrol frekansları olarak ve sıfıra eşit seçilir. Yine BCH sınırı teoremine göre C_4 de sıfır olmalıdır. Yine bu teoremden $C_0^2 = C_0$ olur ki bu da C_0'in ya 0 ya da 1 olduğunu gösterir. Ve $C_5 = C_6^2 = C_3^4$ olup bu da C_3'ün $GF(8)$'de herhangi bir eleman, C_5 ve C_6'nin ise C_3'e bağlı olduğu anlamına gelir. Kodsözler, spektrumunun ters Fourier dönüşümü ile bulunur. Çizelge 4.5'de bunlar gösterilmektedir. Yine bu çizelgeden de görüleceği gibi tek hata düzelten BCH kodu $(7,4)$ Hamming koduna eşittir.

Zaman bölgesinde onaltı kodsözü daha net bir şekilde belirlemek için gerekli spektral sıfırlara sahip olan bir $g(x)$ üreteç polinomu oluşturalım. Bir $v(x)$ polinomu bir ω^j noktasında sıfırı olması için j. spektral bileşen V_j'in sıfıra eşit olması gerekir. Çizelge 4.5'e baktığımızda bu sıfırların $j = 1,2,4$ noktalarında olduğunu görürüz. Yani $\alpha^1, \alpha^2, \alpha^4$ noktalarında polinom sıfırdır. İlk iki sıfıra BCH sınırını, üçüncü sıfıra ise konjugelik kısıtını sağlamak için ihtiyaç vardır. Böylece üreteç polinom $\alpha^3 = \alpha + 1$ eşitliği kullanılarak

$$g(x) = (x - \alpha^1)(x - \alpha^2)(x - \alpha^4)$$

$$= x^3 + x + 1 \qquad (4.38)$$

şeklinde bulunur. $g(x)$'in bir çarpanı $(x - \alpha^4)$ olduğundan, sıfırları $GF(8)$ de olmasına rağmen $g(x)$'in bileşenleri ikilidir.

Çizelge 4.5 $(7,4)$ Hamming kodu

Frekans Bölgesi Kodsözleri							Zaman Bölgesi Kodsözleri						
C_6	C_5	C_4	C_3	C_2	C_1	C_0	c_6	c_5	c_4	c_3	c_2	c_1	c_0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
α^0	α^0	0	α^0	0	0	0	1	1	1	0	1	0	0
α^2	α^4	0	α^1	0	0	0	0	0	1	1	1	0	1
α^4	α^1	0	α^2	0	0	0	0	1	0	0	1	1	1
α^6	α^5	0	α^3	0	0	0	1	1	0	1	0	0	1
α^1	α^2	0	α^4	0	0	0	0	1	1	1	0	1	0
α^3	α^6	0	α^5	0	0	0	1	0	0	1	1	1	0
α^5	α^3	0	α^6	0	0	0	1	0	1	0	0	1	1
0	0	0	0	0	0	1	1	1	1	1	1	1	1
α^0	α^0	0	α^0	0	0	1	0	0	0	1	0	1	1
α^2	α^4	0	α^1	0	0	1	1	1	0	0	0	1	0
α^4	α^1	0	α^2	0	0	1	1	0	1	1	0	0	0
α^6	α^5	0	α^3	0	0	1	0	0	1	0	1	1	0
α^1	α^2	0	α^4	0	0	1	1	0	0	0	1	0	1
α^3	α^6	0	α^5	0	0	1	0	1	1	0	0	0	1
α^5	α^3	0	α^6	0	0	1	0	1	0	1	1	0	0

Genel olarak modulo n tam sayıları denklik sınıfına ayrılmıştır:

$$\mathcal{B}_j = \{j, jq, jq^2, \dots, jq^{m_j-1}\}. \tag{4.39}$$

Eğer C_j'in spektral bileşenleri belirlenmiş ise, indeksi j denklik sınıfında olan diğer her spektral bileşen C_j'in bir kuvveti olmak zorundadır ve ondan farklı bir şekilde belirlenemez. Ayrıca eğer j'nin denklik sınıfı r elemandan oluşuyor ise bu durumda $C_j^{q^r} = C_j$ ve $C_j^{q^r-1} = 1$ olur. Sonuç olarak $GF(q^m)$'nin herhangi bir elemanını C_j olarak seçemeyiz. Sadece sıfır elemanı ya da mertebesi $(q^r - 1)$'i bölen elemanları seçebiliriz.

Böylece j 'nin denklik sınıfı r elemanlı ise bu durumda $C_j \in GF(q^r)$ olduğunu söyleyebiliriz.

Frekans bölgesinde bir kodu belirlemek için ise öncelikle $(q^m - 1)$ tamsayıyı denklik sınıflarına ayırırız ve her sınıf için bir temsilci seçeriz. Bir BCH kodu oluşturmak için ardışık $(d - 1)$ spektral bileşen kontrol frekansı olarak seçilir ve sıfır olarak düşünülür. Diğer semboller veri sembolü olarak kalır.

Örneğin $n = 63$ için bunu uygulayalım. Denklik sınıflarını şu şekilde buluruz:

$\{C_0\}$
$\{C_1, C_2, C_4, C_8, C_{16}, C_{32}\}$
$\{C_3, C_6, C_{12}, C_{24}, C_{48}, C_{33}\}$
$\{C_5, C_{10}, C_{20}, C_{40}, C_{17}, C_{34}\}$
$\{C_7, C_{14}, C_{28}, C_{56}, C_{49}, C_{35}\}$
$\{C_9, C_{18}, C_{36}\}$
$\{C_{11}, C_{22}, C_{44}, C_{25}, C_{50}, C_{37}\}$
$\{C_{13}, C_{26}, C_{52}, C_{41}, C_{19}, C_{38}\}$
$\{C_{15}, C_{30}, C_{60}, C_{57}, C_{51}, C_{39}\}$
$\{C_{21}, C_{42}\}$
$\{C_{23}, C_{46}, C_{29}, C_{58}, C_{53}, C_{43}\}$
$\{C_{27}, C_{54}, C_{45}\}$
$\{C_{31}, C_{62}, C_{61}, C_{59}, C_{55}, C_{47}\}$

Her kümenin ilk elemanını kümenin temsilcisi olarak seçelim. Temsilci belirlendikten sonra kümenin diğer elemanları da bellidir. Eğer kontrol frekansı olarak $C_1, C_2, C_3, C_4, C_5, C_6$ seçersek bunlar üç sınıf oluşturduğundan üçlü hata düzelten, 63 blok uzunluğunda ikili bir BCH kodu elde etmiş oluruz. Bu durumda geriye kalan tüm elemanları $C_0, C_7, C_9, C_{11}, C_{13}, C_{15}, C_{21}, C_{23}, C_{27}, C_{31}$ veri sembolü olarak kabul ederiz.

C_9 ve C_{27} denklik sınıfları $r = 3$ elemanlı olduğundan ve ayrıca $C_9^8 = C_9$, $C_{27}^8 = C_{27}$ olduğundan bunlar $GF(8)$'in elemanlarıdır. Benzer şekilde C_{21}, $r = 2$ elemanlı bir denklik sınıfına sahip olduğundan ve ayrıca $C_{21}^4 = C_{21}$ olduğundan $GF(4)$)'ün elemanıdır. Yine aynı sebeplerle C_0'in $GF(2)$, $C_7, C_{11}, C_{13}, C_{15}, C_{23}, C_{31}$ değerlerinin de $GF(64)$'ün elemanları olduğunu söyleyebiliriz. Sonuçta bu veri sembollerini göstermek için $3 + 3 + 2 + 1 + 6 + 6 + 6 + 6 + 6 + 6 = 45$ bileşene ihtiyacımız vardır. Böylece üç hata düzelten $(63,45)$ BCH kodunu elde etmiş oluruz. Frekans bölgesinde kodlama yaparak elde edilen 2^{45} tane kodsöz ile zaman bölgesinde bir üreteç polinom ile kodlama yapılarak elde edilen 2^{45} tane kodsöz aynıdır. Veri elde edilene kadar,

dekodlayıcı verinin nasıl kodlandığı ile ilgilenmez. Fakat son aşamada veri düzeltilmiş kodsözden çıkartıldığında dekodlayıcı verinin kodsözde nasıl saklandığını bilmesi gerekir. Eğer veri bileşenleri frekans bölgesinde kodlanmış ise yine frekans bölgesinde okunmaları gerekir.

15 kod uzunluğunda, çift hata düzelten bir ikili kod elde etmek için tanım kümesi olarak spektrumun dört ardışık bileşenini seçelim. $j = 1,2,3,4$ için $C_1 = C_2 = C_3 = C_4 = 0$ seçebiliriz. Spektrumun diğer bileşenleri $GF(16)$'nın elemanlarıdır ve birbirlerine konjugelik kısıtı ile $C_j^2 = C_{((2j))}$ şeklinde bağlıdırlar. Bu, $\{\alpha^j, \alpha^{2j}, \dots\}$ konjugelik sınıfına karşılık gelen j indeksli bir spektral bileşenin, aynı konjugelik sınıfına karşılık gelen diğer tüm spektral bileşenleri belirlediği anlamına gelir. Modulo 15'deki konjugelik sınıfları, spektral bileşenler kümesinin parçalanışı ile aşağıdaki gibi ifade edilir;

$$\{C_0\}, \{C_1, C_2, C_4, C_8\}, \{C_3, C_6, C_{12}, C_9\}, \{C_5, C_{10}\}, \{C_7, C_{14}, C_{13}, C_{11}\}$$

5 uzaklıklı bir kodun BCH sınırını sağlaması için, tanım kümesini $\mathcal{A} = \{1,2,3,4\}$ şeklinde seçelim. Bu elemanların tüm konjugelik sınıflarının elemanlarını ekleyerek bütün tanım kümesini $\mathcal{A} = \{1,2,3,4,6,8,9,12\}$ elde ederiz. $C_0^2 = C_0$ olduğundan C_0 bileşeni $GF(2)$'nin bir bileşenidir ve 1 bileşen ile belirlenebilir. $C_0^4 = C_{10}^2 = C_5$ olduğundan C_5 bileşeni $GF(4)$'ün bir elemanıdır ve 2 bileşen ile belirlenebilir. C_7 bileşeni $GF(16)$'nin herhangi bir elemanıdır ve 4 bileşen ile belirlenebilir. Toplamda kodsöz spektrumunu belirlemek için 7 bileşene ihtiyaç vardr. Böylece kod $GF(2)$ üzerinde bir $(15,7,5)$ BCH kodudur.

İkili $(15,7,5)$ BCH kodu kod kümesinde üreteç polinomu cinsinden de ifade edilebilir. Bu polinomun tam tanım kümesinin her elemanına karşılık gelen bir sıfırı α_j olmalıdır. Bu nedenle katsayıları $GF(2)$'de olan

$$g(x) = (x - \alpha^1)(x - \alpha^2)(x - \alpha^4)(x - \alpha^8)(x - \alpha^3)(x - \alpha^6)(x - \alpha^{12})(x - \alpha^9)$$
$$= x^8 + x^7 + x^6 + x^4 + 1 \qquad (4.40)$$

polinomu vardır. $\deg(g(x)) = 8$ olduğundan $n - k = 8$ ve $k = 7$ 'dir.

Üç hata düzelten, 63 uzunluğunda bir ikili kod elde etmek için tanım kümesi olarak altı ardışık spektral indis seçelim. $\mathcal{A} = \{1,2,3,4,5,6\}$ olsun. Buna uygun gelen konjugelik sınıfları $\{1,2,4,8,16,32\}, \{3,6,12,24,48,33\}, \{5,10,20,40,17,34\}$ 'dır ve tüm tanım

kümesi bu üç kümenin bileşimidir. Böylece üreteç polinomunun derecesi 18'dir. Bu devirli kod $(63,45,7)$ BCH kodudur.

Bir BCH kodunun minimum uzaklığı tasarlanmış uzaklığından daha büyük olabilir. 23 kod uzunluğunda iki hata düzelten bir ikili kod elde etmek için tanım kümesinden 4 ardışık spektral bileşen seçelim. $j = 1,2,3,4$ için $C_1 = C_2 = C_3 = C_4 = 0$ olsun. $23^9 = 2^{11} - 1$ olduğundan spektral bileşenler $GF(2^{11})$ cismindedir. Spektrumun diğer bileşenleri konjugelik şartıyla bağlı olup $GF(2^{11})$'dedirler. Modulo 23'de üç adet konjugelik sınıfı vardır ve Fourier spektrumun bu bölünüşü şöyle olur,

$\{C_0\}$

$\{C_1, C_2, C_4, C_8, C_{16}, C_9, C_{18}, C_{13}, C_3, C_6, C_{12}\}$

$\{C_{22}, C_{21}, C_{19}, C_{15}, C_7, C_{14}, C_5, C_{10}, C_{20}, C_{17}, C_{11}\}$

Sadece C_1 elemanını içeren küme sıfıra sınırlanmıştır. Bu nedenle kod bir $(23,12,5)$ BCH koddur. Fakat bu kodun minimum uzaklığı tasarlanmış uzaklığından daha büyüktür.

Zaman bölgesinde kodlama yapmak için kodun üreteç polinomunun bulunması gerekir. Üreteç polinom ise $1, \dots, d-1$ aralığında bir eşleniği olan tüm j'ler için

$$g(x) = \prod \left(x - \omega^j\right) \tag{4.41}$$

şeklinde verilir. Kodsözler ise $c(x) = a(x)g(x)$ formunda olduğundan $j = 1, \dots, d-1$ için $g(x)$ yapısı da kullanılarak

$$C_j = c\left(\omega^j\right) = a\left(\omega^j\right)g\left(\omega^j\right) = 0 \qquad j = 1, \dots, d-1 \tag{4.42}$$

elde ederiz. Buna eş olarak $j = 1, \dots, d-1$ için $f_j(x)$'ler ω^j'lerin minimal polinomları olmak üzere $g(x) = \text{ekok}[f_1(x), \dots, f_{d-1}(x)]$ yazabiliriz. 15 blok uzunluğunda iki hata düzelten BCH kodun üreteç polinomunu bulmak istersek,

$$\begin{aligned} g(x) &= \text{ekok}[f_1(x), f_2(x), f_3(x), f_4(x)] \\ &= \text{ekok}[x^4 + x + 1, x^4 + x + 1, x^4 + x^3 + x^2 + x + 1, x^4 + x + 1] \\ &= (x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1) \\ &= x^8 + x^7 + x^6 + x^4 + 1 \end{aligned} \tag{4.43}$$

elde ederiz. $g(x)$ sekizinci dereceden olduğundan $n-k=8$ olur. Buradan da $k=7$ olup $(15,7)$ iki hata düzelten BCH kod için üreteç polinom bulmuş oluruz. Burada BCH kodların belirlenmiş n ve t değerleri ile oluşturulduğunun altını çizelim. k değeri ise $g(x)$ polinomu bulunmadan bilinemez. Yine aynı yolla devam ederek 15 uzunluğunda primitif BCH kodlar için üreteç polinomlar bulabiliriz. Örneğin $t=3$ alırsak,

$$\begin{aligned} g(x) &= \text{ekok}[f_1(x), f_2(x), f_3(x), f_4(x), f_5(x), f_6(x)] \\ &= (x^4+x+1)(x^4+x^3+x^2+x+1)(x^2+x+1) \\ &= x^{10}+x^8+x^5+x^4+x^2+x+1 \end{aligned} \tag{4.44}$$

polinomu $(15,5)$ üç hata düzelten BCH kodu için üreteç polinom olur. $t=4$ için üreteç polinom hesaplamak istersek

$$\begin{aligned} g(x) &= \text{ekok}[f_1(x), f_2(x), f_3(x), f_4(x), f_5(x), f_6(x), f_7(x), f_8(x)] \\ &= (x^4+x+1)(x^4+x^3+x^2+x+1)(x^2+x+1)(x^4+x^3+1) \\ &= x^{14}+x^{13}+x^{12}+x^{11}+x^{10}+x^9+x^8+x^7+x^6+x^5+x^4+x^3+x^2+x+1 \end{aligned} \tag{4.45}$$

elde ederiz ki bu da $(15,1)$ BCH kodu için üreteç polinomdur. Bu kod 7 hata düzelten basit tekrar kodudur. $t=5,6,7$ için de $t=4$ için bulduğumuz aynı üreteç polinomu buluruz. $t=7$'den sonra ise BCH kodu tanımsızdır çünkü cisimde sıfırdan farklı eleman sayısı 15'dir.

Çizelge 4.6'da $GF(16)$ cismini $GF(4)$'ün bir genişleme cismi olarak $p(z)=z^2+z+2$ primitif polinomu ile nasıl oluşturulduğu verilmiştir. Bu tablo aynı zamanda $\alpha=z$ primitif kök olmak üzere $GF(16)$'daki tüm cisim elemanlarının $GF(4)$ üzerindeki minimal polinomlarını da içerir.

$GF(4)$ üzerinde 15 uzunluklu tek hata düzelten BCH kod için üreteç polinomu şöyle elde edilir.

$$\begin{aligned} g(x) &= \text{ekok}[f_1(x), f_2(x)] \\ &= (x^2+x+2)(x^2+x+3) = x^4+x+1 \end{aligned} \tag{4.46}$$

Bu polinom $GF(4)$ üzerinde 15 uzunluklu tek hata düzelten $(15,11)$ BCH kod için üreteç polinomudur. Bu bir Hamming kod değildir ve 15 dörtlü (quaternary) sembol içinden 11 dörtlü sembol kodlar.

Çizelge 4.6 $\mathrm{GF}(4^2)$ Cisminin Gösterimi

Üstel Notasyon	Polinom Notasyon	Dörtlü Notasyon	Onluk Sistemde Notasyon	Minimal Polinom
0	0	00	0	-
α^0	1	01	1	x+1
α^1	z	10	4	x^2+x+2
α^2	$z+2$	12	6	x^2+x+3
α^3	$3z+2$	32	14	x^2+3x+1
α^4	$z+1$	11	5	x^2+x+2
α^5	2	02	2	$x+2$
α^6	2z	20	8	x^2+2x+1
α^7	$2z+3$	23	11	x^2+2x+2
α^8	$z+3$	13	7	x^2+x+3
α^9	$2z+2$	22	10	x^2+2x+1
α^{10}	3	03	3	$x+3$
α^{11}	3z	30	12	x^2+3x+3
α^{12}	$3z+1$	31	13	x^2+3x+1
α^{13}	$2z+1$	21	9	x^2+2x+2
α^{14}	$3z+3$	33	15	x^2+3x+3

Yine benzer şekilde $\mathrm{GF}(4)$ üzerinde 15 uzunluklu BCH kodlar için üreteç polinomlar bulabiliriz. Örneğin $t=2$ için:

$$g(x)=\mathrm{ekok}[f_1(x),f_2(x),f_3(x),f_4(x)]$$
$$=(x^2+x+2)(x^2+x+3)(x^2+3x+1)$$
$$=x^6+3x^5+x^4+x^3+2x^2+2x+1 \qquad (4.47)$$

olur ki bu polinom $GF(4)$ üzerinde iki hata düzelten $(15,9)$ BCH kodu için üreteç polinomdur. $t = 3$ alırsak:

$$g(x) = x^9 + 3x^8 + 3x^7 + 2x^6 + x^5 + 2x^4 + x + 2 \quad (4.48)$$

polinomu $GF(4)$ üzerinde üç hata düzelten $(15,6)$ BCH kodu için üreteç polinomdur. $t = 4$ alırsak

$$g(x) = x^{11} + x^{10} + 2x^8 + 3x^7 + 3x^6 + x^5 + 3x^4 + x^3 + x + 3 \quad (4.49)$$

polinomu $GF(4)$ üzerinde dört hata düzelten $(15,4)$ BCH kodu için üreteç polinomdur. $t = 4$ için

$$g(x) = x^{12} + 2x^{11} + 3x^{10} + 2x^9 + 2x^8 + x^7 + 3x^6 + 3x^4 + 3x^3 + x^2 + 2 \quad (4.50)$$

polinomu $GF(4)$ üzerinde beş hata düzelten $(15,3)$ BCH kodu için üreteç polinomdur. Son olarak $t = 6$ için

$$g(x) = x^{14} + x^{13} + x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1 \quad (4.51)$$

polinomu $GF(4)$ üzerinde 6 hata düzelten $(15,1)$ BCH kodu için üreteç polinomdur ve aslında 7 hata düzeltebilir çünkü aynı zamanda basit tekrar kodudur.

BCH kodun tanımı bizim yukarda primitif uzunluk ile yaptığımız tanımından daha geneldir. Genelde BCH kod, üreteç polinomu $g(x)$'in sıfırları olarak, primitif olması gerekmeyen herhangi bir ω'nın $2t$ ardışık kuvvetinin değerlerini alır. Bu kodun uzunluğu ω'nın mertebesidir yani $\omega^n = 1$ eşitliğini sağlayan en küçük n sabitidir [15], [16].

4.6 Genişletilmiş Reed-Solomon Kodları

Eğer bir kod, bir Reed-Solomon kodun kodsözlerine bir veya iki ekstra sembol eklenerek elde ediliyor ise bu koda genişletilmiş Reed-Solomon kodu denir. Sadece bir sembol ekleniyorsa tek genişletilmiş Reed-Solomon kod, iki sembol ekleniyorsa çift genişletilmiş Reed-Solomon kod adı verilir. Karakteristiği 2 olan bir cisim üzerinde en önemli genişletilmiş Reed-Solomon kodu tek genişletilmiş primitif Reed-Solomon kodudur. Çünkü bu kodun blok uzunluğu 2^m olup bir çok uygulamada kolaylıkla kullanılabilir.

Eklenen her sembol, veri sembolü ya da kontrol sembolü olarak düşünülebilir. Yani bir kodun oranını arttırarak genişletilmişi elde edebilir veya minimum uzaklığını arttırarak

uzatabiliriz. Aynı genişletilmiş kod, minimum uzaklığı d_{min} olan bir Reed-Solomon kod genişletilerak veya minimum uzaklığı $d_{min} - 2$ olan bir Reed-Solomon kodun uzatılması ile elde edilebilir.

Eklenen iki sembolün genişletilmiş kodsözde tanımlanması gerekir. Bunun için birkaç indeksleme notasyonu kullanılabilir. Eğer $(d-1)$ blok uzunluğundaki genişletilmiş olmayan primitif kodsözün bileşenleri sıfır olmayan cisim elemanları ile indekslenmiş ise, bu durumda kodun yeni bileşenlerinde cisim elemanı sıfır indeks olarak kullanılabilir. Genellikle ∞[4] kullanılır ve böylece $q+1$ uzunluğunda genişletilmiş kodsöz

$$\mathbf{c} = (c_0, c_{\omega^0}, , c_{\omega^1}, \dots, c_{\omega^{q-2}}, c_\infty) \tag{4.52}$$

olur. Eğer genişletilmiş olmayan kodun bileşenleri primitif elemanın $(q-1)$ kuvveti ile indekslenmiş ise bu durumda sıfır yeni sembollerde indeks olarak kullanılmak için uygun değildir. Yani iki yeni sembole ihtiyaç vardır. Bu durumda indeks olarak $+$ ve $-$ kullanabiliriz. Böylece $n = q+1$ blok uzunluğuna sahip bir genişletilmiş kodsöz bir yeni bileşen kodsözün en başında ve bir yeni bileşende en sonunda olmak üzere

$$\mathbf{c} = \left(c_-, c_0, , c_1, \dots, c_{q-2}, c_+\right) \tag{4.53}$$

şeklinde yazılır. Burada c_- ve c_+ çıkartılarak elde edilen $(d-1)$ uzunluğundaki vektöre iç (interior) vektör denir. Bir kodsözün spektrumundan bahsederken iç vektörün spekturumu anlaşılmalıdır.

Tanım 4.4 j_0 ve t rastgele tamsayılar olsunlar. Çift genişletilmiş Reed-Solomon kodu blok uzunluğu $(q+1)$ ve kodsözleri $(c_-, c_0, , c_1, \dots, c_{q-2}, c_+)$ olan $GF(q)$ üzerinde bir lineer koddur. Bu kodun kodsözlerinin spektrumu $(q-1)$ uzunluğunda olup şu özellikleri sağlar:

$$1.\ C_j = 0, j = j_0 + 1, \dots, j_0 + d - 3 \tag{4.54}$$

$$2.\ C_{j_0} = c_- \tag{4.55}$$

$$3.\ C_{j_0+d-2} = c_+ \tag{4.56}$$

[4] Reel sayılar cismini genişletmek için en uygun yol $+\infty$ ve $-\infty$ sembollerini eklemek ve bunlar için aritmetik işlemleri tanımlamaktır. Genişletilmiş reel sayılar bir cisim oluşturmaz. Bu işlemin aynısını bir Galois cismine de uygulayabiliriz. ∞ sembolünü ekleyerek ve işlemleri $a + \infty = \infty$ ve $a\infty = \infty$ şeklinde tanımlayarak cisim olmayan bir küme oluştururuz.

Genişletilmiş Reed-Solomon kodun tasarlanmış uzaklığı d'dir. Tanımdan $(\mathrm{d}-3)$ ardışık spektral bileşen sıfıra eşittir ve bu $(\mathrm{d}-3)$ bileşenin her iki tarafındaki elemanlar c_ ve c_{+} olur.

Teorem 4.4 *$GF(q)$ üzerinde bir genişletilmiş Reed-Solomon kod, minimum uzaklığı $d = n - k + 1 = q - k + 2$ olan bir $(q+1, k)$ koddur.*

İspat: Kolaylık sağlamak için $\mathrm{j}_0 = 0$ alabiliriz. Eğer j_0 sıfıra eşit değilse, kontrol matrisinde ilk ve son sütundaki elemanlar hariç diğer tüm elemanlar ω^{j_0} ile çarpılmış olacaktır. Fakat bu sonucumuzu etkilemez. Kontrol matrisi

$$\begin{bmatrix} -1 & 1 & 1 & \cdots & 1 & 1 & 1 \\ 0 & \omega & \omega^2 & \cdots & \omega^{\mathrm{q}-2} & \omega^{\mathrm{q}-1} & 0 \\ 0 & \omega^2 & \omega^4 & \cdots & \omega^{2(\mathrm{q}-2)} & \omega^{2(\mathrm{q}-1)} & 0 \\ \vdots & & & & & \vdots & \\ 0 & \omega^{\mathrm{q}-\mathrm{k}-1} & \omega^{(\mathrm{q}-\mathrm{k}-1)2} & \cdots & \omega^{(\mathrm{q}-\mathrm{k}-1)(\mathrm{q}-2)} & \omega^{(\mathrm{q}-\mathrm{k}-1)(\mathrm{q}-1)} & 0 \\ 0 & \omega^{\mathrm{q}-\mathrm{k}} & \omega^{(\mathrm{q}-\mathrm{k})2} & \cdots & \omega^{(\mathrm{q}-\mathrm{k})(\mathrm{q}-2)} & \omega^{(\mathrm{q}-\mathrm{k})(\mathrm{q}-1)} & -1 \end{bmatrix} \tag{4.57}$$

şeklindedir. Eğer kontrol matrisinin her $(\mathrm{d}-1)$ sütununun kümesi bir lineer bağımsız vektörler kümesi oluşturuyor ise kodun minimum uzaklığı en az d olur. Eğer ilk ve son sütun silinirse bu durumda matrisin herhangi $\mathrm{q}-\mathrm{k}+1$ sütununun kümesi, sütunları farklı olan bir Vandermonde matrisi oluşturur ve nonsingüler olur. Böylece içteki tüm sütunların $\mathrm{q}-\mathrm{k}+1$ elemanlı kümeleri lineer bağımsız olur. Bunun yerine ilk ve son sütunu da içeren $\mathrm{q}-\mathrm{k}+1$ sütunlar kümesini alırsak determinant, sırası ile bir kere ilk sütuna bir kere de son sütuna açılarak alınabilir. Bu ise matrisin ilk ve son satırlarını yok ederek Vandermonde matrisinin determinantını sıfırdan farklı kılar. Böylece herhangi $\mathrm{q}-\mathrm{k}+1$ sütun lineer bağımsız ve minimum uzaklık en az $\mathrm{q}-\mathrm{k}+2$ olarak bulunur.

Şimdi genişletilmiş Reed-Solomon kod için bir kodlayıcı verelim. Kodlayıcı spektrumun $(d-3)$ ardışık bileşenini sıfır olarak atar. Spektrumun diğer bileşenleri $GF(q)$'dan rastgele veri bileşenleri içerir. Spektrumun kontrol frekans bloğunun iki yanındaki değerler de yine $GF(q)$'da değerler alıp kodsöze eklenmiş c_+ ve c_- kodsözleri gibi düşünülür. Bu ise $(q+1)$ uzunluğunda d_{min} minimum uzaklığına sahip genişletilmiş Reed-Solomon kodu verir.

Çift genişletilmiş bir (n, k) Reed-Solomon kodu zaman bölgesinde kodlamak için bir üreteç polinomu ele alalım. $g(x)$ polinomu $\omega^2, \ldots, \omega^{2t-1}$ noktalarında sıfırları olan

üreteç polinomu olsun. Bu polinomu $(c_0, c_1, \dots, c_{n'-1})$ içinde k veri sembolünü kodlamak için kullanalım. Buradan genişlemede oluşan semboller

$$c_- = \sum_{i=0}^{n'-1} c_i \omega^i \qquad \text{ve} \qquad c_+ = \sum_{i=0}^{n'-1} c_i \omega^{2ti} \tag{4.58}$$

şeklinde olur ve bunlar $(c_0, c_1, \dots, c_{n'-1})$ kodsözlerine eklenerek yeni kodsöz oluşturulur.

Genişletilmiş Reed-Solomon kodu dekodlamak için Reed-Solomon kodun herhangi bir dekodlayıcısını kullanabiliriz. Ya eklenen bileşenler hatalı değildir ki bu durumda iki ekstra sendrom

$$S_{j_0} = V_{j_0} = c_- \quad \text{ve} \qquad S_{j_0+d-2} = V_{j_0+d-2} - c_+ \tag{4.59}$$

oluşturmak için kullanılabilirler, ya da en az bir hata barındırabilirler. İkinci durumda iç vektörde en fazla $(t-1)$ hata vardır ve ekstra iki sendroma ihtiyaç kalmaz. Sadece bir kodsöz gelensözden t kadar uzaklıktadır, yani bu iki sendrom kümesi ile dekodlama sonucu iki farklı kodsöz oluşmaz [15].

4.7 Genişletilmiş BCH Kodu

$GF(q^m)$ üzerinde $n = q^m + 1$ blok uzunluğuna sahip bir genişletilmiş Reed-Solomon kodu, genişletilmiş Reed-Solomon kodun altcisim-altkodunu alarak $GF(q)$ üzerinde $n = q^m + 1$ blok uzunluğunda bir kod oluşturmak için kullanılabilir. Alternatif olarak, n uzunluğunda bir genişletilmiş Reed-Solomon kod, her q^m-li sembolü m tane q-lu sembole değiştirerek mn blok uzunluğuna sahip bir kod oluşturmak için kullanılabilir.

Bu başlıkta $GF(q)$ üzerinde bir genişletilmiş kodun biraz farklı bir şekilde oluşturulmasından bahsedeceğiz. Burada yaklaşımımız, genişletilmiş Reed-Solomon kodda $(q^m - 1)$ iç sembolden sadece $GF(q)$ içindeki bileşenleri almak olacaktır. Genişlemeyi sağlayacağımız iki sembol ise $GF(q^m)$'de herhangi bileşenler olabilirler. Bu şekilde oluşturulan kodlara genişletilmiş BCH kodları denir.

$(q^m - 1)$ uzunluğunda $GF(q^m)$ üzerinde bir $\boldsymbol{C}$ vektörü, $C_j^q = C_{((qj))}$ konjugelik ilişkileri sağlanıyorsa $GF(q)$'da bir kodun kodsöz spektrumudur. Tasarlanmış uzaklığı d olan bir genişletilmiş BCH kod elde etmek için, j_0 indeksiyle başlamak üzere ardışık $(d-1)$ spektral bileşeni seçelim ve $c_-, c_+ \in GF(q^m)$ olmak üzere

$$C_{j_0} = c_- \tag{4.60}$$

$$C_j = 0 \qquad j = j_0 + 1, \dots, j_0 + d - 3 \tag{4.61}$$

$$C_{j_0+d-2} = c_- + \tag{4.62}$$

elemanlarını tanımlayalım. Bu semboller spekturumdaki mümkün olan konjugelik ilişkileri dışında rastgele elemanlardır. Benzer şekilde geri kalan spektral bileşenlerde q'lu konjugelik ilişkisini sağlamak koşulu ile rastgeledirler. Bu durumda kodsözün içi bu spektrumun ters Fourier dönüşümü olur. Kodsözün içi olarak söz ettiğimiz kısım, c_- ile gösterdiğimiz en fazla m tane $GF(q)$'lu sembol ile başlayan ve c_+ ile devam eden sembollerdir.

Çizelge 4.7 $j_0 = 0$ için genişletilmiş BCH kodları

d_{BCH}	m=4	m=5	m=6	m=7	m=8
3	(20,15)	(37,31)	(70,63)	(135,127)	(264,255)
5	(20,11)	(37,26)	(70,57)	(135,120)	(264,247)
7	(18,7)	(37,21)	(70,51)	(135,113)	(264,239)
9		(37,16)	(70,45)	(135,106)	(264,231)
11			(67,39)	(135,99)	(264,223)
13		(37,11)	(70,36)	(135,92)	(264,215)
15			(70,30)	(135,85)	(264,207)
17			(70,24)	(135,78)	(264,199)
19					(264,191)
21				(135,71)	(264,187)
23			(66,18)	(135,64)	(264,179)
25			(70,16)	(135,57)	(264,171)
27					(264,163)
29				(135,50)	(264,155)

Genişletilmiş BCH kodun nasıl inşaa edildiğini göstermek için $j_0 = 0$ seçelim ve modulo 63 konjugelik sınıflarını kullanarak spektrumu $GF(64)$'de olan bazı kodlar oluşturalım. Çizelge 4.7 de, blok uzunluğu 255 kadar olan primitif BCH kodundan elde edilen tek minimum uzaklığa sahip tüm genişletilmiş kodlar verilmiştir. $C_0 = C_0^2$ olduğundan sıfır veya bir olmak zorundadır yani c_- tek bir sembolden oluşur. $2t-1 = 9,18,36,27,54,45$ olduğu durumlarda c_+ üç bileşen içerir ve $2t-1 = 21$ veya 42 olduğu durumlarda da c_+, 2 bileşen içerir. Bu durumlar haricinde ise C_{2t-1} bileşeni yani c_+, $GF(64)$'nun herhangi bir elemenıdır. Ayrıca $2t-1$ konjugelik

sınıfında en küçük sayı olmalıdır çünkü eğer konjugelik sınıfında başka bir sayı varsa spektrumun o bileşeni de sıfır olmak zorunda olup bu da C_{2t-1}'in sıfır olması anlamına gelir.

$t = 4$ alalım. $2t - 1 = 7$ olup $C_0, C_1, \dots, C_7$ kontrol frekanslarımız olur. $C_0 = c_-$, $GF(2)$'de herhangi bir eleman ve $C_7 = c_+$ de $GF(64)$'de herhangi bir elemandır. Ayrıca C_0 tek bite, C_1, C_3, C_5, C_7 bileşenleri 6 bileşene karşılık gelmektedir. Bu durumda $n = 63 + 1 + 6$ olup minimum uzaklığı en az 9 olan bir $(70,24)$ kod elde etmiş oluruz.

$t = 8$ olsun. Buradan $2t - 1 = 15$ ve kontrol frekansları $C_0, C_1, \dots, C_1$ olur. $C_0 = c_-$, $GF(2)$'de herhangi bir eleman ve $C_1 = c_+$ de $GF(64)$'de herhangi bir elemandır. Ayrıca C_0 tek bileşene, $C_1, C_3, C_5, C_7, C_{11}, C_{13}, C_{15}$ bileşenleri 6 bileşene ve C_9 üç bileşene karşılık gelmektedir. Bu durumda minimum uzaklığı en az 17 olan bir $(70,24)$ kod elde etmiş oluruz [15].

4.7 Bir BCH Kod Örneği

Bu bölümü bitirmeden önce bir ikili $(7,4)$ BCH kodunu inceleyelim. Bu kodun $(7,4)$ Hamming kodu ile denk olduğunu, ayrıca BCH kodun bir devirli kod olduğunu biliyoruz. Kodun elemanları şunlardır:

0 0 0 0 0 0 0
1 1 1 0 1 0 0

0 0 1 1 1 0 1

0 1 0 0 1 1 1

1 1 0 1 0 0 1

0 1 1 1 0 1 0

1 0 0 1 1 1 0

1 0 1 0 0 1 1

1 1 1 1 1 1 1

0 0 0 1 0 1 1

1 1 0 0 0 1 0

1 0 1 1 0 0 0

0 0 1 0 1 1 0

1 0 0 0 1 0 1

0 1 1 0 0 0 1

0 1 0 1 1 0 0

Bu elemanların Fourier dönüşümlerini (spektrumlarını) hesaplayalım. Öncelikle $n = q^m - 1$ için $GF(q^m)$'de bir primitif elemanı bulmalıyız. $n = 7$ olduğundan $m = 3$ olup $\omega, p(x) = x^3 + x + 1$ polinomunun köküdür ve $\omega^7 = 1$ olup birimin 7. köküdür. Gerçekten,

$$p(\omega) = \omega^3 + \omega + 1 = 0$$
$$\omega^3 = \omega + 1$$
$$\omega^4 = \omega^2 + \omega$$
$$\omega^5 = \omega^3 + \omega^2 = \omega^2 + \omega + 1$$
$$\omega^6 = \omega^3 + \omega^2 + \omega = \omega^2 + 1$$
$$\omega^7 = 1 \qquad (4.64)$$

Şimdi kodsözlerin spektrallerini hesaplayalım. Bunun için

$$C_j = \sum_{i=0}^{6} \omega^{ij} c_i \quad j = 0,1, \dots ,6 \qquad (4.65)$$

toplamlarını hesaplayalım.

$$C_0 = c_0 + c_1 + c_2 + c_3 + c_4 + c_5 + c_6$$
$$C_1 = c_0 + \omega c_1 + \omega^2 c_2 + \omega^3 c_3 + \omega^4 c_4 + \omega^5 c_5 + \omega^6 c_6$$
$$C_2 = c_0 + \omega^2 c_1 + \omega^4 c_2 + \omega^6 c_3 + \omega^8 c_4 + \omega^{10} c_5 + \omega^{12} c_6$$
$$C_3 = c_0 + \omega^3 c_1 + \omega^6 c_2 + \omega^9 c_3 + \omega^{12} c_4 + \omega^{15} c_5 + \omega^{18} c_6$$
$$C_4 = c_0 + \omega^4 c_1 + \omega^8 c_2 + \omega^{12} c_3 + \omega^{16} c_4 + \omega^{20} c_5 + \omega^{24} c_6$$
$$C_5 = c_0 + \omega^5 c_1 + \omega^{10} c_2 + \omega^{15} c_3 + \omega^{20} c_4 + \omega^{25} c_5 + \omega^{30} c_6$$
$$C_6 = c_0 + \omega^6 c_1 + \omega^{12} c_2 + \omega^{18} c_3 + \omega^{24} c_4 + \omega^{30} c_5 + \omega^{36} c_6 \qquad (4.66)$$

Bu toplamları kullanarak Fourier dönüşüm matrisini oluşturalım.

$$\begin{bmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \\ C_6 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \omega^{12} & \omega^{15} & \omega^{18} \\ 1 & \omega^4 & \omega^8 & \omega^{12} & \omega^{16} & \omega^{20} & \omega^{24} \\ 1 & \omega^5 & \omega^{10} & \omega^{15} & \omega^{20} & \omega^{25} & \omega^{30} \\ 1 & \omega^6 & \omega^{12} & \omega^{18} & \omega^{24} & \omega^{30} & \omega^{36} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \end{bmatrix} \qquad (4.67)$$

Şimdi her bir kodsöz için bileşenleri yerine koyarak Fourier dönüşümünü hesaplayabiliriz.

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \omega^{12} & \omega^{15} & \omega^{18} \\ 1 & \omega^4 & \omega^8 & \omega^{12} & \omega^{16} & \omega^{20} & \omega^{24} \\ 1 & \omega^5 & \omega^{10} & \omega^{15} & \omega^{20} & \omega^{25} & \omega^{30} \\ 1 & \omega^6 & \omega^{12} & \omega^{18} & \omega^{24} & \omega^{30} & \omega^{36} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1+\omega+\omega^2+\omega^4 \\ 1+\omega^2+\omega^4+\omega^8 \\ 1+\omega^3+\omega^6+\omega^{12} \\ 1+\omega^4+\omega^8+\omega^{16} \\ 1+\omega^5+\omega^{10}+\omega^{20} \\ 1+\omega^6+\omega^{12}+\omega^{24} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \omega^{12} & \omega^{15} & \omega^{18} \\ 1 & \omega^4 & \omega^8 & \omega^{12} & \omega^{16} & \omega^{20} & \omega^{24} \\ 1 & \omega^5 & \omega^{10} & \omega^{15} & \omega^{20} & \omega^{25} & \omega^{30} \\ 1 & \omega^6 & \omega^{12} & \omega^{18} & \omega^{24} & \omega^{30} & \omega^{36} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ \omega^2+\omega^3+\omega^4+\omega^6 \\ \omega^4+\omega^6+\omega^8+\omega^{12} \\ \omega^6+\omega^9+\omega^{12}+\omega^{18} \\ \omega^8+\omega^{12}+\omega^{16}+\omega^{24} \\ \omega^{10}+\omega^{15}+\omega^{20}+\omega^{30} \\ \omega^{12}+\omega^{18}+\omega^{24}+\omega^{36} \end{bmatrix} = \begin{bmatrix} 0 \\ \omega^2 \\ \omega^4 \\ 0 \\ \omega \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \omega^{12} & \omega^{15} & \omega^{18} \\ 1 & \omega^4 & \omega^8 & \omega^{12} & \omega^{16} & \omega^{20} & \omega^{24} \\ 1 & \omega^5 & \omega^{10} & \omega^{15} & \omega^{20} & \omega^{25} & \omega^{30} \\ 1 & \omega^6 & \omega^{12} & \omega^{18} & \omega^{24} & \omega^{30} & \omega^{36} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ \omega+\omega^4+\omega^5+\omega^6 \\ \omega^2+\omega^8+\omega^{10}+\omega^{12} \\ \omega^3+\omega^{12}+\omega^{15}+\omega^{18} \\ \omega^4+\omega^{16}+\omega^{20}+\omega^{24} \\ \omega^5+\omega^{20}+\omega^{25}+\omega^{30} \\ \omega^6+\omega^{24}+\omega^{30}+\omega^{36} \end{bmatrix} = \begin{bmatrix} 0 \\ \omega^4 \\ \omega \\ 0 \\ \omega^2 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \omega^{12} & \omega^{15} & \omega^{18} \\ 1 & \omega^4 & \omega^8 & \omega^{12} & \omega^{16} & \omega^{20} & \omega^{24} \\ 1 & \omega^5 & \omega^{10} & \omega^{15} & \omega^{20} & \omega^{25} & \omega^{30} \\ 1 & \omega^6 & \omega^{12} & \omega^{18} & \omega^{24} & \omega^{30} & \omega^{36} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1+\omega+\omega^3+\omega^6 \\ 1+\omega^2+\omega^6+\omega^{12} \\ 1+\omega^3+\omega^9+\omega^{18} \\ 1+\omega^4+\omega^{12}+\omega^{24} \\ 1+\omega^5+\omega^{15}+\omega^{30} \\ 1+\omega^6+\omega^{18}+\omega^{36} \end{bmatrix} = \begin{bmatrix} 0 \\ \omega^6 \\ \omega^5 \\ 0 \\ \omega^3 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \omega^{12} & \omega^{15} & \omega^{18} \\ 1 & \omega^4 & \omega^8 & \omega^{12} & \omega^{16} & \omega^{20} & \omega^{24} \\ 1 & \omega^5 & \omega^{10} & \omega^{15} & \omega^{20} & \omega^{25} & \omega^{30} \\ 1 & \omega^6 & \omega^{12} & \omega^{18} & \omega^{24} & \omega^{30} & \omega^{36} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ \omega+\omega^2+\omega^3+\omega^5 \\ \omega^2+\omega^4+\omega^6+\omega^{10} \\ \omega^3+\omega^6+\omega^9+\omega^{15} \\ \omega^4+\omega^8+\omega^{12}+\omega^{20} \\ \omega^5+\omega^{10}+\omega^{15}+\omega^{25} \\ \omega^6+\omega^{12}+\omega^{18}+\omega^{30} \end{bmatrix} = \begin{bmatrix} 0 \\ \omega \\ \omega^2 \\ 0 \\ \omega^4 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \omega^{12} & \omega^{15} & \omega^{18} \\ 1 & \omega^4 & \omega^8 & \omega^{12} & \omega^{16} & \omega^{20} & \omega^{24} \\ 1 & \omega^5 & \omega^{10} & \omega^{15} & \omega^{20} & \omega^{25} & \omega^{30} \\ 1 & \omega^6 & \omega^{12} & \omega^{18} & \omega^{24} & \omega^{30} & \omega^{36} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1+\omega^3+\omega^4+\omega^5 \\ 1+\omega^6+\omega^8+\omega^{10} \\ 1+\omega^9+\omega^{12}+\omega^{15} \\ 1+\omega^{12}+\omega^{16}+\omega^{20} \\ 1+\omega^{15}+\omega^{20}+\omega^{25} \\ 1+\omega^{18}+\omega^{24}+\omega^{30} \end{bmatrix} = \begin{bmatrix} 0 \\ \omega^3 \\ \omega^6 \\ 0 \\ \omega^5 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \omega^{12} & \omega^{15} & \omega^{18} \\ 1 & \omega^4 & \omega^8 & \omega^{12} & \omega^{16} & \omega^{20} & \omega^{24} \\ 1 & \omega^5 & \omega^{10} & \omega^{15} & \omega^{20} & \omega^{25} & \omega^{30} \\ 1 & \omega^6 & \omega^{12} & \omega^{18} & \omega^{24} & \omega^{30} & \omega^{36} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1+\omega^2+\omega^5+\omega^6 \\ 1+\omega^4+\omega^{10}+\omega^{12} \\ 1+\omega^6+\omega^{15}+\omega^{18} \\ 1+\omega^8+\omega^{20}+\omega^{24} \\ 1+\omega^{10}+\omega^{25}+\omega^{30} \\ 1+\omega^{12}+\omega^{30}+\omega^{36} \end{bmatrix} = \begin{bmatrix} 0 \\ \omega^5 \\ \omega^3 \\ 0 \\ \omega^6 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \omega^{12} & \omega^{15} & \omega^{18} \\ 1 & \omega^4 & \omega^8 & \omega^{12} & \omega^{16} & \omega^{20} & \omega^{24} \\ 1 & \omega^5 & \omega^{10} & \omega^{15} & \omega^{20} & \omega^{25} & \omega^{30} \\ 1 & \omega^6 & \omega^{12} & \omega^{18} & \omega^{24} & \omega^{30} & \omega^{36} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} =$$

$$= \begin{bmatrix} 1 \\ 1+\omega+\omega^2+\omega^3+\omega^4+\omega^5+\omega^6 \\ 1+\omega^2+\omega^4+\omega^6+\omega^8+\omega^{10}+\omega^{12} \\ 1+\omega^3+\omega^6+\omega^9+\omega^{12}+\omega^{15}+\omega^{18} \\ 1+\omega^4+\omega^8+\omega^{12}+\omega^{16}+\omega^{20}+\omega^{24} \\ 1+\omega^5+\omega^{10}+\omega^{15}+\omega^{20}+\omega^{25}+\omega^{30} \\ 1+\omega^6+\omega^{12}+\omega^{18}+\omega^{24}+\omega^{30}+\omega^{36} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \omega^{12} & \omega^{15} & \omega^{18} \\ 1 & \omega^4 & \omega^8 & \omega^{12} & \omega^{16} & \omega^{20} & \omega^{24} \\ 1 & \omega^5 & \omega^{10} & \omega^{15} & \omega^{20} & \omega^{25} & \omega^{30} \\ 1 & \omega^6 & \omega^{12} & \omega^{18} & \omega^{24} & \omega^{30} & \omega^{36} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ \omega^3+\omega^5+\omega^6 \\ \omega^6+\omega^{10}+\omega^{12} \\ \omega^9+\omega^{15}+\omega^{18} \\ \omega^{12}+\omega^{20}+\omega^{24} \\ \omega^{15}+\omega^{25}+\omega^{30} \\ \omega^{18}+\omega^{30}+\omega^{36} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ \omega^6 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \omega^{12} & \omega^{15} & \omega^{18} \\ 1 & \omega^4 & \omega^8 & \omega^{12} & \omega^{16} & \omega^{20} & \omega^{24} \\ 1 & \omega^5 & \omega^{10} & \omega^{15} & \omega^{20} & \omega^{25} & \omega^{30} \\ 1 & \omega^6 & \omega^{12} & \omega^{18} & \omega^{24} & \omega^{30} & \omega^{36} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1+\omega+\omega^5 \\ 1+\omega^2+\omega^{10} \\ 1+\omega^3+\omega^{15} \\ 1+\omega^4+\omega^{20} \\ 1+\omega^5+\omega^{25} \\ 1+\omega^6+\omega^{30} \end{bmatrix} = \begin{bmatrix} 1 \\ \omega^2 \\ \omega^4 \\ 0 \\ \omega \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix}1&1&1&1&1&1&1\\1&\omega&\omega^2&\omega^3&\omega^4&\omega^5&\omega^6\\1&\omega^2&\omega^4&\omega^6&\omega^8&\omega^{10}&\omega^{12}\\1&\omega^3&\omega^6&\omega^9&\omega^{12}&\omega^{15}&\omega^{18}\\1&\omega^4&\omega^8&\omega^{12}&\omega^{16}&\omega^{20}&\omega^{24}\\1&\omega^5&\omega^{10}&\omega^{15}&\omega^{20}&\omega^{25}&\omega^{30}\\1&\omega^6&\omega^{12}&\omega^{18}&\omega^{24}&\omega^{30}&\omega^{36}\end{bmatrix}\begin{bmatrix}1\\0\\1\\1\\0\\0\\0\end{bmatrix}=\begin{bmatrix}1\\1+\omega^2+\omega^3\\1+\omega^4+\omega^6\\1+\omega^6+\omega^9\\1+\omega^8+\omega^{12}\\1+\omega^{10}+\omega^{15}\\1+\omega^{12}+\omega^{18}\end{bmatrix}=\begin{bmatrix}1\\\omega^4\\\omega\\0\\\omega^2\\0\\0\end{bmatrix}$$

$$\begin{bmatrix}1&1&1&1&1&1&1\\1&\omega&\omega^2&\omega^3&\omega^4&\omega^5&\omega^6\\1&\omega^2&\omega^4&\omega^6&\omega^8&\omega^{10}&\omega^{12}\\1&\omega^3&\omega^6&\omega^9&\omega^{12}&\omega^{15}&\omega^{18}\\1&\omega^4&\omega^8&\omega^{12}&\omega^{16}&\omega^{20}&\omega^{24}\\1&\omega^5&\omega^{10}&\omega^{15}&\omega^{20}&\omega^{25}&\omega^{30}\\1&\omega^6&\omega^{12}&\omega^{18}&\omega^{24}&\omega^{30}&\omega^{36}\end{bmatrix}\begin{bmatrix}0\\0\\1\\0\\1\\1\\0\end{bmatrix}=\begin{bmatrix}1\\\omega^2+\omega^4+\omega^5\\\omega^4+\omega^8+\omega^{10}\\\omega^6+\omega^{12}+\omega^{15}\\\omega^8+\omega^{16}+\omega^{20}\\\omega^{10}+\omega^{20}+\omega^{25}\\\omega^{12}+\omega^{24}+\omega^{30}\end{bmatrix}=\begin{bmatrix}1\\\omega^6\\\omega^5\\0\\\omega^3\\0\\0\end{bmatrix}$$

$$\begin{bmatrix}1&1&1&1&1&1&1\\1&\omega&\omega^2&\omega^3&\omega^4&\omega^5&\omega^6\\1&\omega^2&\omega^4&\omega^6&\omega^8&\omega^{10}&\omega^{12}\\1&\omega^3&\omega^6&\omega^9&\omega^{12}&\omega^{15}&\omega^{18}\\1&\omega^4&\omega^8&\omega^{12}&\omega^{16}&\omega^{20}&\omega^{24}\\1&\omega^5&\omega^{10}&\omega^{15}&\omega^{20}&\omega^{25}&\omega^{30}\\1&\omega^6&\omega^{12}&\omega^{18}&\omega^{24}&\omega^{30}&\omega^{36}\end{bmatrix}\begin{bmatrix}1\\0\\0\\0\\1\\0\\1\end{bmatrix}=\begin{bmatrix}1\\1+\omega^4+\omega^6\\1+\omega^8+\omega^{12}\\1+\omega^{12}+\omega^{18}\\1+\omega^{16}+\omega^{24}\\1+\omega^{20}+\omega^{30}\\1+\omega^{24}+\omega^{36}\end{bmatrix}=\begin{bmatrix}1\\\omega\\\omega^2\\0\\\omega^4\\0\\0\end{bmatrix}$$

$$\begin{bmatrix}1&1&1&1&1&1&1\\1&\omega&\omega^2&\omega^3&\omega^4&\omega^5&\omega^6\\1&\omega^2&\omega^4&\omega^6&\omega^8&\omega^{10}&\omega^{12}\\1&\omega^3&\omega^6&\omega^9&\omega^{12}&\omega^{15}&\omega^{18}\\1&\omega^4&\omega^8&\omega^{12}&\omega^{16}&\omega^{20}&\omega^{24}\\1&\omega^5&\omega^{10}&\omega^{15}&\omega^{20}&\omega^{25}&\omega^{30}\\1&\omega^6&\omega^{12}&\omega^{18}&\omega^{24}&\omega^{30}&\omega^{36}\end{bmatrix}\begin{bmatrix}0\\1\\1\\0\\0\\0\\1\end{bmatrix}=\begin{bmatrix}1\\\omega+\omega^2+\omega^6\\\omega^2+\omega^4+\omega^{12}\\\omega^3+\omega^6+\omega^{18}\\\omega^4+\omega^8+\omega^{24}\\\omega^5+\omega^{10}+\omega^{30}\\\omega^6+\omega^{12}+\omega^{36}\end{bmatrix}=\begin{bmatrix}1\\\omega^3\\\omega^6\\0\\\omega^5\\0\\0\end{bmatrix}$$

$$\begin{bmatrix}1&1&1&1&1&1&1\\1&\omega&\omega^2&\omega^3&\omega^4&\omega^5&\omega^6\\1&\omega^2&\omega^4&\omega^6&\omega^8&\omega^{10}&\omega^{12}\\1&\omega^3&\omega^6&\omega^9&\omega^{12}&\omega^{15}&\omega^{18}\\1&\omega^4&\omega^8&\omega^{12}&\omega^{16}&\omega^{20}&\omega^{24}\\1&\omega^5&\omega^{10}&\omega^{15}&\omega^{20}&\omega^{25}&\omega^{30}\\1&\omega^6&\omega^{12}&\omega^{18}&\omega^{24}&\omega^{30}&\omega^{36}\end{bmatrix}\begin{bmatrix}0\\1\\0\\1\\1\\0\\0\end{bmatrix}=\begin{bmatrix}1\\\omega+\omega^3+\omega^4\\\omega^2+\omega^6+\omega^8\\\omega^3+\omega^9+\omega^{12}\\\omega^4+\omega^{12}+\omega^{16}\\\omega^5+\omega^{15}+\omega^{20}\\\omega^6+\omega^{18}+\omega^{24}\end{bmatrix}=\begin{bmatrix}1\\\omega^5\\\omega^3\\0\\\omega^6\\0\\0\end{bmatrix}$$

Bütün kodsözlerin DFT'lerini hesapladıktan sonra bunları sıralayarak inceleyelim.

0	0	0	0	0	0	0
1	1	0	1	0	0	0
ω^2	ω^4	0	ω	0	0	0
ω^4	ω	0	ω^2	0	0	0
ω^6	ω^5	0	ω^3	0	0	0
ω	ω^2	0	ω^4	0	0	0
ω^3	ω	0	ω^5	0	0	0
ω^3	ω	0	ω^5	0	0	0
0	0	0	0	0	0	1
1	ω^6	0	1	0	0	1
ω^2	ω^4	0	ω	0	0	1
ω^4	ω	0	ω^2	0	0	1
ω^6	ω^5	0	ω^3	0	0	1
ω	ω^2	0	ω^4	0	0	1
ω^3	ω^6	0	ω^5	0	0	1
ω^5	ω^3	0	ω^6	0	0	1

Sonuçta kodsözlerin spektra0llarini incelediğimizde C_1, C_2, C_4 bileşenlerinin sıfır olduğunu görürüz.

BÖLÜM 5

FOURİER DÖNÜŞÜMÜNE DAYANAN DEKODLAMA ALGORİTMALARI

Bir hata kontrol kodu sadece oranı ve minimum uzaklığı açısından incelenmez, ayrıca uygun bir şekilde dekodlama yapılıp yapılmadığı açısından da incelenmelidir. Genellikle bir kodu dekodlamak için birçok yol vardır. Fakat bir dizayn yapan kişi belirli bir uygulamada hangisinin en uygun olacağına karar vermek için çeşitli dekodlama algoritmalarına aşina olmalıdır. En uygun algoritmayı seçmek ise sadece blok uzunluğu ve minimum uzaklık gibi kod parametrelerine bağlı değildir. Aynı zamanda uygulamanın yazılım ve donanım açısından nasıl çalıştığı, ihtiyaç duyulacak dekodlama hızı ve tüm dekodlama sürecinin maliyeti algoritmayı seçerken etkilidir.

Genellikle büyük lineer kodları dekodlamak zor bir iştir. Bu yüzden bir kodu seçerken, o kod için pratik bir dekodlama algoritmasının var olması önemli bir faktördür. Reed-Solomon kodları ve diğer devirli kodlar Fourier dönüşümünün özellikleri ile yakından alakalı bir uzaklık yapısına sahiptir. Bu nedenle Reed-Solomon kodlar için iyi algoritmaların bir çoğu Fourier dönüşümüne dayalıdır.

Bu bölümde anlatılan dekodlama algoritmaları "hata yerini bulan dekodlama (locator decoding) algoritmaları" olarak bilinen dekodlama algoritmaları sınıfındandır. Bu ise cebirsel dekodlama algoritmaları içinde en zengin, en ilginç ve en önemli olandır. Hata yerini bulan dekodlama algoritmaları matematiksel olarak daha ilginç ve sofistikedir. Bu dekodlama algoritmalarının ilginçliği, görünüşte çözülmesi zor görünen nonlineer problemleri lineer problemlere veya daha basit nonlineer problemlere dönüştürmesidir. Genel hata yerini bulan dekodlama algoritmaları sınıfına baktığımızda çok çeşitli algoritmalar olduğunu görürüz.

Hata bulan dekodlama algoritmaları, bir devirli kodun tanım kümesi ardışık sıfırlar olduğu her durumda kullanılabilir. Algoritma dekodlama yapmak için bu ardışık sıfırları kullanır ve böylece bu algoritma gerçek minimum uzaklık yerine daha çok BCH sınırı ile yakından ilgilidir. Hata bulan dekodlamada, BCH sınırının yarısından küçük olan en büyük tam sayı olarak tanımlı BCH yarıçapına ulaşılır, fakat kodun paketleme yarıçapına sadece paketleme yarıçapı BCH yarıçapına eşit olduğu durumda ulaşılır. Bir Reed-Solomon kod (ve birçok BCH kod) için minimum uzaklık BCH sınırına eşit olur ve böylece hata bulan dekodlama bu kodlar için paketleme sınırına ulaşır. Reed-Solomon kod için hata bulan dekodlama en çok kullanılan algoritmadır.

Hata bulan dekodlama algoritması *hata yerini bulan polinom (locator polynomial)* olarak adlandırılan $\Lambda(x)$ polinomunun kullanımına dayanır. Hata yerini bulan dekodlama, üzerinde çalışılan cismin cebirsel yapısını ve özelliklerini kullandığından özellikle büyük kodlar için uygun olan güçlü bir dekodlama algoritmaları ailesi oluşturur. Algoritmanın seçimi ise hem uygulamanın özel gereksinimlerine hem de dizayn eden kişinin istediğine bağlıdır. En önemli dekodlama algoritmaları Fourier dönüşümünün özelliklerine dayanır.

5.1 Sendromlar ve Hata Modelleri

Bir $\boldsymbol{c}$ kod sözü gönderilmiş olsun ve ulaşım esnasında hataya uğradığını varsayalım. Kodun paketleme yarıçapı $t = \lfloor (d_{min} - 1)/2 \rfloor$ olmak üzere, eğer oluşan hatalar t den daha az yerde ise dekodlayıcı kod sözü (ya da kod sözdeki veri sembollerini) düzeltir.

Bir veri iletişim sisteminde alınan söze *gelensöz* denir ve $\boldsymbol{v}$ vektörü ile gösterilir. Gelensöz $\boldsymbol{v}$, kodsöz $\boldsymbol{c}$'nin bir hata vektörü $\boldsymbol{e}$ ile bozulması sonucu oluşur. Bir gelensözün i. bileşeni

$$v_i = c_i + e_i \qquad i = 0,1,\dots,n-1 \tag{5.1}$$

şeklinde yazılır ve burada e_i vektörü i'nin en fazla t değerinde sıfırdan farklıdır. Eğer hata t'den daha fazla bileşende değil ise bu durumda $\boldsymbol{v}$ gelensözünden kodsözü ya da kodsözün içerdiği veri sembollerini *sınırlı-uzaklık dekodlayıcısı* düzeltir.

Biz sadece alfabesi cisim olan kodlarla ilgilendiğimizden bir kodsözün i. bileşenindeki hatayı $e_i = v_i - c_i$ şeklinde tanımlamak mantıklıdır. Yani gelensöz $\boldsymbol{v}$'yi, kodsöz $\boldsymbol{c}$ ile bir $\boldsymbol{e}$ hata vektörünün toplamı şeklinde düşünebiliriz. Burada $\boldsymbol{e}$ hata vektörünün en fazla t bileşeni sıfırdan farklıdır.

F cismi üzerinde, genellikle bir sonlu cisim $GF(q)$, bir lineer kod için kontrol matrisi $\boldsymbol{H}$ vardır, öyle ki her $\boldsymbol{c}$ kodsözü için $\boldsymbol{cH^T} = \boldsymbol{0}$ ilişkisi sağlanır. Böylece

$$\boldsymbol{vH^T} = (\boldsymbol{c} + \boldsymbol{e})\boldsymbol{H^T} = \boldsymbol{eH^T} \tag{5.2}$$

yazılabilir. Genel bir lineer kod için $\boldsymbol{s}$ *sendrom vektörü*, şöyle tanımlanır;

$$\boldsymbol{s} = \boldsymbol{vH^T} \ = \boldsymbol{eH^T} \tag{5.3}$$

ve sendrom vektörünün bileşenlerine *sendromlar* denir.

Lineer bir kodu dekodlamak için iki adım vardır. İlk adımda n boyutlu $\boldsymbol{e}$ vektörünü $(n-k)$ boyutlu $\boldsymbol{s}$ vektörüne dönüştüren $\boldsymbol{s} = \boldsymbol{vH^T}$ lineer işlemi ile sendromları hesaplarız. İkinci adımda ise ağırlığı t'den fazla olmamak üzere $\boldsymbol{e}$ vektörü için $\boldsymbol{s} = \boldsymbol{eH^T}$ eşitliğini çözeriz. Bu da minimum ağırlıklı n boyutlu $\boldsymbol{e}$ vektörü için $(n-k)$ eşitlik çözmek demektir. Böyle n boyutlu ve en fazla t ağırlıklı vektörlerin kümesi, $GF(q)^n$'in lineer bir altuzayı değildir. Yani sendrom vektörleri $\boldsymbol{s}$'lerden, hata vektörleri $\boldsymbol{e}$'lere olan dönüşüm bir lineer dönüşüm değildir. Bunun tersini almak için $(n-k)$ boyutlu bir uzaydan n boyutlu bir uzaya lineer olmayan bir işlem gereklidir. Her düzeltilebilir hata tek bir sendroma sahip olmak zorunda olduğundan, sendromlar uzayında çözümü olan vektörlerin sayısı

$$\sum_{m=0}^{t} (q-1)^m \binom{n}{m} \tag{5.4}$$

ile bulunur. q^{n-k}'den büyük olmayan bu sayı, $(n-k)$ vektörlü bir uzayda, ters görüntüleri düzeltilebilir hata modelleri olan eleman sayısıdır.

Küçük ikili kodlarda düzeltilebilir kodlar için ve bunlara karşılık gelen sendromları içeren bir sendrom tablosu çizilebilir. *Boolean-mantık dekodlayıcısı* denilen hızlı bir dekodlaycı, sendrom ve hata modelleri arasındaki ilişkiye bakmak için bir mantık ağacı içerir. Bir boolean-mantık dekodlaycısı çok çok hızlı olabilir fakat sadece çok basit ikili kodlarda kullanılır.

Devirli Reed-Solomon kodları ve diğer devirli kodlar için Fourier dönüşümü aracılığı ile alternatif farklı bir sendrom tanımı yapmak mümkündür ve bu çok daha kullanışlıdır. Gelensöz $\boldsymbol{v}$'nin Fourier dönüşümü

$$V_j = \sum_{i=0}^{n-1} \omega^{ij} v_i \qquad j = 0,1,\dots,n-1 \tag{5.5}$$

şeklinde hesaplandığını biliyoruz. Fourier dönüşümün lineerliğinden;

$$V_j = C_j + E_j \qquad j = 0,1,\dots,n-1 \tag{5.6}$$

yazılır ve Reed Solomon kodların yapısından

$$C_j = 0 \qquad j = 0,1,\dots,n-k-1 \tag{5.7}$$

olur. Buradan da

$$V_j = E_j \qquad j = 0,1,\dots,n-k-1 \tag{5.8}$$

elde ederiz. Burada bu elemanların $\boldsymbol{V}$ vektöründen kolaylıkla bulduğumuz hata spektrumu $\boldsymbol{E}$'nin $(n-k)$ tane bileşeni olduğunu vurgulamak gerekir. Bunlar genellikle S ile gösterilirler ve daha önce tanımlanan sendromlardan farklı olmalarına rağmen *(spektral) sendromlar* olarak adlandırılırlar. Bu iki tanımı birbirinden ayırmak için ilkini *kod-bölgesi sendromları* ikincisini de *dönüşüm-bölgesi sendromları* olarak adlandırabiliriz. Sonuç olarak dönüşüm-bölgesi sendromları

$$S_j = E_j = V_j \qquad j = 0,1,\dots,n-k-1 \tag{5.9}$$

şeklinde tanımlanır. Burada işlem kolaylığı olsun diye $j_0 = 0$ özel durumu kullanıyoruz. Fakat bunu kullanmakta bir sakınca yoktur çünkü Fourier dönüşümün modülasyon özelliğini bize, $\boldsymbol{C}$ devirli olarak değiştiğinde $\boldsymbol{c}$'ye ne olduğunu söyler. Modülasyon özelliğini kullanarak j_0 'ın diğer herhangi bir değeri ile de aynı işlemleri gerçekleştirebiliriz.

Polinomlar olarak ifade edersek *hata-spektrum polinomu* şöyledir;

$$E(x) = \sum_{j=0}^{n-1} E_j x^j \tag{5.10}$$

sendrom polinomu ise şöyledir;

$$S(x) = \sum_{j=0}^{n-k-1} S_j x^j \tag{5.11}$$

Hata-spektrum polinomu en fazla $(n-1)$ dereceli ve sendrom polinomu da en fazla $(n-k-1)$ derecelidir. Reed-Solomon kod için minimum uzaklık $d_{min} = n-k+1$ olduğundan $t = \lfloor (n-k)/2 \rfloor$ olmak üzere t hata düzeltir. Şimdi amacımız,

$$S_j = \sum_{i=0}^{n-1} \omega^{ij} e_i \qquad j = 0,1,\dots,n-k-1 \tag{5.12}$$

eşitliğini, ağırlık en fazla t olmak üzere, en küçük ağırlığa sahip hata vektörü $\boldsymbol{e}$ için hesaplamaktır. Bunun alternatifi ise n blok uzunluğunda hata vektörünün dönüşümü

$\boldsymbol{E}$'yi bulmaktır. Burada $\boldsymbol{e}$'nin ağırlığı en fazla $t = \lfloor (n-k)/2 \rfloor$'dir ve E_j'ler her $j = 0,1, \dots, n-k-1$ için bilinen S_j'lerdir. Fourier dönüşümü tanım kümesinde bu sendromları kullanan dekodlayıcılara *dönüşüm-bölgesi dekodlayıcı* denir.

İlk dekodlama adımında, *hata bulan polinom* olarak bilinen $\Lambda(x)$ yardımcı polinomu tanımlanır. Burada sendromlar kümesi ile hata spektrumu $\boldsymbol{E}$ arasındaki lineer olmayan ilişkinin, S_j sendromları ve hata bulan polinom $\Lambda(x)$'in katsayıları arasındaki lineer ilişkiye dönüştüğünü görürüz. Dekodlayıcıda bir yerde görünen lineer olmayan işlemler $\Lambda(x)$ ile $\boldsymbol{E}$'nin geriye kalan bileşenleri arasında toplanmışlardır ve lineer olmayan ilişki ise lineer yinelemenin basit bir formudur. Şimdi direk matris terslerinden elde edilen sendromlardan lineer yinelemenin katsayılarını bulmak için bir algoritma açıklayalım.

Ağırlığı v olan bir $\boldsymbol{e}$ hata vektörü verilsin. (Burada v en fazla t'dir ve bilinmez) $l = 0,1, \dots, v$ için indisler i_l olmak üzere şu polinomu düşünelim;

$$\Lambda(x) = \prod_{l=1}^{v} \left(1 - x\omega^{i_l}\right). \tag{5.13}$$

Burada $v \leq t$ noktaları hata olan konumları gösterir ve bunlar $\boldsymbol{e}$'nin sıfır olmayan bileşenlerine karşılık gelir. Bu durumda $\delta_i = (1/n)\Lambda(\omega^{-i})$'nin sıfır olması için gerek ve yeter şart i indeksli bileşende bir hata olması ve bu daha küçük dereceli bir $\Lambda(x)$ için sağlanmamasıdır. Böylece her i için $e_i\delta_i = 0$ olur. Fourier dönüşümünün konvolüsyon özelliğinden,

$$\Lambda(x)E(x) = 0 \quad \mod (x^n - 1) \tag{5.14}$$

olur ki bu da $\Lambda(x)$'in gerçekten hata bulan polinom olduğunu doğrular. Bu polinomu katsayılarına göre yazarsak, çift parantezler modulo n alındığını göstermek üzere,

$$\sum_{k=0}^{v} \Lambda_k E_{((j-k))} = 0 \qquad j = 0,1, \dots n-1 \tag{5.15}$$

elde ederiz. $\Lambda_0 = 1$ olduğundan bu eşitlik

$$E_j = -\sum_{k=1}^{v} \Lambda_k E_{((j-k))} = 0 \qquad j = 0,1, \dots n-1 \tag{5.16}$$

şeklinde tekrar yazılabilir ve bu eşitlik, hata spektrumunun bileşenlerini sağlayan bir basit lineer yinelemedir. Lineer yinelemenin uzunluğu olan v, $\boldsymbol{e}$'nin ağırlığına eşittir. Bu ise daha önce bahsettiğimiz lineer kompleksite özelliğine bir örnektir. Hata vektörü $\boldsymbol{e}$'nin ağırlığı v olup bu en fazla t kadardır ve böylece lineer kompleksite özelliğinden

E'nin bütün bileşenleri en fazla t uzunluklu lineer yineleme ile devirli bir şekilde üretilebilir. Yani $\Lambda(x)$, $\boldsymbol{e}$ hata vektörünün hata bulan polinomu olmak üzere,

$$E_j = -\sum_{k=1}^{t} \Lambda_k E_{((j-k))} \qquad j = 0,1,\dots,n-1 \tag{5.17}$$

yazılır. Bu devirli yinelemeyi oluşturmanın nedeni, bunun Λ_k'nin bilinmeyen katsayıları ve $\boldsymbol{E}$'nin bileşenleri ile ilgili bir lineer eşitlikler kümesi olmasıdır. Yukarıdaki yinelemenin içerdiği n denklem için, E'nin sadece $2t$ bilinen bileşenini içeren ve Λ'nin t bilinmeyen bileşenini içeren t denklemi vardır. Bunlar şunlardır:

$$E_t = -(\Lambda_1 E_{t-1} + \Lambda_2 E_{t-2} + \cdots + \Lambda_t E_0)$$
$$E_t + 1 = -(\Lambda_1 E_t + \Lambda_2 E_{t-1} + \cdots + \Lambda_t E_1)$$
$$\vdots$$
$$E_{2t-1} = -(\Lambda_1 E_{2t-2} + \Lambda_2 E_{2t-3} + \cdots + \Lambda_t E_{t-1}) \tag{5.18}$$

Bunları matris formunda yazarsak,

$$\begin{bmatrix} E_{t-1} & E_{t-2} & \cdots & E_0 \\ E_t & E_{t-1} & & E_1 \\ \vdots & & & \vdots \\ E_{2t-2} & E_{2t-3} & \cdots & E_{t-1} \end{bmatrix} \begin{bmatrix} \Lambda_1 \\ \Lambda_2 \\ \vdots \\ \Lambda_t \end{bmatrix} = - \begin{bmatrix} E_t \\ E_{t+1} \\ \vdots \\ E_{2t-1} \end{bmatrix} \tag{5.19}$$

elde ederiz. Bu matris eşitliği Λ_j'nin bağlantı katsayıları için uygun bir hesaplama metodu ile çözülebilir. Bu metodlardan biri Gauss eliminasyon yöntemidir. Hata vektörü e'nin en fazla t ağırlığında olduğunu varsaydığımızdan matris denkleminin bir çözümü vardır. Eğer matrisin determinantı sıfır ise bu durumda t'den daha az hata vardır. Bu ise Λ_t'nin başkatsayısı sıfır demektir. Eğer determinantı sıfır ise, matris denkleminde t'yi $(t-1)$ ile değiştirip problemi küçültmüş oluruz ve aynı yöntemle çözebiliriz. Bu yolla son olarak matris determinantı sıfır olmayan $v \times v$'lik bir matrise indirgenmiş olur.

Λ biliniyorsa hata spektrumu E'nin diğer bileşenleri,

$$E_j = -\sum_{k=0}^{t} \Lambda_k E_{((j-k))} \qquad j = 2t,\dots,n-1 \tag{5.20}$$

yinelemesi ile teker teker hesaplanabilir. Bu yineleme, dekodlama algoritmasının bir parçası olması gereken lineer olmayan fonksiyonların oluşmasını sağlar. Ters Fourier dönüşümü hata vektörü $\boldsymbol{e}$'yi verir. Buradan da,

$$c_i = v_i - e_i \qquad i = 0,1,\dots,n-1 \tag{5.21}$$

işlemiyle bileşen bileşen çıkarma yaparız. Sonuçta, dekodlayıcının kullandığı işlemin tersi ile basit bir hesaplama kullanılarak veri sembolleri koddan çıkartılmış olur.

Böylece Reed-Solomon ve BCH kodlarının sınırlı uzaklık dekodlaması için en temel dekodlama algoritması verilmiş oldu. Fakat bu kısım sadece bir başlangıçtır. Hata yerini bulma algoritması, dekodlama algoritmasının hesaplanmasını kolaylaştırmak için şuan çok daha sofistike şekilde gelişmiştir.

Çizelge 5.1 *GF(16)'nin bir gösterilimi*

$\alpha^0=$	1
$\alpha^1=$	z
$\alpha^2=$	z^2
$\alpha^3=$	z^3
$\alpha^4=$	$z+1$
$\alpha^5=$	z^2+z
$\alpha^6=$	z^3+z^2
$\alpha^7=$	z^3+z+1
$\alpha^8=$	z^2+1
$\alpha^9=$	z^3+z
$\alpha^{10}=$	z^2+z+1
$\alpha^{11}=$	z^3+z^2+z
$\alpha^{12}=$	z^3+z^2+z+1
$\alpha^{13}=$	z^3+z^2+1
$\alpha^{14}=$	z^3+1

Bu algoritmanın işleyişine bir örnek olması için $GF(16)$ üzerindeki $(15,9,7)$ Reed-Solomon kodunu dekodlamaya çalışalım. $n=15$ olduğundan α, $GF(16)$ bir primitif eleman olmak üzere $\omega=\alpha$ seçebiliriz. α'yi da $\alpha^4+\alpha+1=0$ şeklinde seçelim. $GF(16)$ cisminin elemanlarının gösterimi Çizelge 5.1 de görüldüğü gibi olsun. $(15,9,7)$ Reed-Solomon kodunu $\{1,2,3,4,5,6\}$ tanım kümesi ile seçelim. Ayrıca bu örnekte tanım kümesini $j_0=0$ yerine $j_0=1$ olarak seçtiğimize dikkat etmek gerekir. Şimdi sırasıyla verisözü, kodsöz ve gelensöz şu şekilde verilmiş olduğunu varsayalım;

$d=0,0,0,0,0,0,0,0,0$

$c=0,0,0,0,0,0,0,0,0,0,0,0,0,0,0$

$v=0,0,0,0,0,0,0,\alpha,0,\alpha^5,0,0,\alpha^{11},0,0$ (5.22)

Dekodlamanın ilk adımı $\boldsymbol{v}$'nin Fourier dönüşümünü hesaplamak ki burada sadece altı bileşeni hesaplamamız yeterlidir. Bunu hesapladımızda;

$$V = -, \alpha^{12}, 1, \alpha^{14}, \alpha^{13}, 1, \alpha^{11}, -, -, -, -, -, -, -, -. \quad (5.23)$$

elde ederiz. Bu altı bileşen ***E***'nin karşılık gelen altı bileşenine eşittir.

$$E_4 = -(\Lambda_1 E_3 + \Lambda_2 E_2 + \Lambda_3 E_1)$$
$$E_5 = -(\Lambda_1 E_4 + \Lambda_2 E_3 + \Lambda_3 E_2)$$
$$E_6 = -(\Lambda_1 E_5 + \Lambda_2 E_4 + \Lambda_3 E_3) \quad (5.24)$$

denklemerinden,

$$\begin{bmatrix} E_3 & E_2 & E_1 \\ E_4 & E_3 & E_2 \\ E_5 & E_4 & E_3 \end{bmatrix} \begin{bmatrix} \Lambda_1 \\ \Lambda_2 \\ \Lambda_3 \end{bmatrix} = - \begin{bmatrix} E_4 \\ E_5 \\ E_6 \end{bmatrix} \quad (5.25)$$

matris denklemini oluşturup buradan Λ'yı çözersek;

$$\begin{bmatrix} \Lambda_1 \\ \Lambda_2 \\ \Lambda_3 \end{bmatrix} = \begin{bmatrix} \alpha^{14} & 1 & \alpha^{12} \\ \alpha^{13} & \alpha^{14} & 1 \\ 1 & \alpha^{13} & \alpha^{14} \end{bmatrix}^{-1} \begin{bmatrix} \alpha^{13} \\ 1 \\ \alpha^{11} \end{bmatrix} = \begin{bmatrix} \alpha^{14} \\ \alpha^{11} \\ \alpha^{14} \end{bmatrix} \quad (5.26)$$

elde ederiz ve buradan da hata bulan polinomu;

$$\Lambda(x) = 1 + \alpha^{14} x + \alpha^{11} x^2 + \alpha^{14} x^3 \quad (5.27)$$

şeklinde buluruz. Bu polinom ise,

$$\Lambda(x) = (1 + \alpha^2 x)(1 + \alpha^5 x)(1 + \alpha^7 x) \quad (5.28)$$

şeklinde çarpanlarına ayrılır. Bu ise hatanın, $i = 2,5,7$ bileşenlerinde olduğu anlamına gelir.

5.2 Hata değerlerinin hesabı

Önceki bölümde açıklanan algoritma hata düzeltme problemini, hata yerini bulan polinomu oluşturma ve hata değerlerini hesaplama olarak iki kısıma ayrıştırır. Hata yerini bulan polinom bilindikten sonra, geriye sadece polinomdan ve sendromlardan hata değerlerini hesaplamak kalır. Bu hesap birkaç yolla yapılabilir. Şimdi bunun için üç farklı yaklaşım vereceğiz.

Hata değerlerinin hesaplanması için ilk metod, *tekrarlanan uzantı (recursive extension)* adı verilen, tam hata spektrumunu oluşturmak için

$$E_j = -\sum_{k=1}^{t} \Lambda_k E_{((j-k))} \quad (5.29)$$

yinelemesini kullanır. Örneğimizde bunu uygularsak,

$$E_7 = \Lambda_1 E_6 + \Lambda_2 E_5 + \Lambda_3 E_4$$
$$= \alpha^{14} \cdot \alpha^{11} + \alpha^{11} \cdot 1 + \alpha^{14} \cdot \alpha^{13} = \alpha^5 \quad (5.30)$$

elde ederiz. Benzer şekilde,

$$E_8 = \alpha^{14} \cdot \alpha^5 + \alpha^{11} \cdot \alpha^{11} + \alpha^{14} \cdot 1 = 1 \quad (5.31)$$

$$E_9 = \alpha^{14} \cdot 1 + \alpha^{11} \cdot \alpha^5 + \alpha^{14} \cdot \alpha^{11} = \alpha^6 \quad (5.32)$$

şeklinde bulunur ve bu işleme aynı yolla ***E***'nin tüm bileşenleri buluna kadar devam edilir ve sonuçta,

$$E = (\alpha^9, \alpha^{12}, 1, \alpha^{14}, \alpha^{13}, 1, \alpha^{11}, \alpha^5, 1, \alpha^6, \alpha^7, 1, \alpha^{10}, \alpha^3, 1) \quad (5.33)$$

bulunur. Buradan da ters Fourier dönüşümü ile

$$e = 0,0,0,0,0,0,0,0, \alpha, 0, \alpha^5, 0,0, \alpha^{11}, 0,0 \quad (5.34)$$

hata vektörü bulunur.

Hata değerlerini hesaplamanın bir diğer metodu ise *Forney algoritması* olarak bilinen metottur. Forney algoritması, ***e*** hata vektörünü hesaplamak için yeni bir yardımcı polinom $\Gamma(x)$'yı kullanır. Bu polinoma hata hesaplayan (error evaluator) polinom denir. Ayrıca

$$\Lambda'(x) = \sum_{j=1}^{t} j\Lambda_j x^{j-1} \quad (5.35)$$

olarak verilen $\Lambda(x)$'in türevini de burada kullanırız. Forney algoritmasını çalıştırmak için, $\Lambda(x)E(x) = 0 \pmod{x^n - 1}$ eşitliğini kullanırız. Bunu bir $\Gamma(x)$ polinomu için

$$\Lambda(x)E(x) + \Gamma(x)(x^n - 1) = 0$$

$$\Lambda(x)E(x) = -\Gamma(x)(x^n - 1) \quad (5.36)$$

şeklinde yazabiliriz. deg($\Lambda(x)) \leq t$ ve deg($E(x)) \leq n - 1$ olduğundan sonuçta $\Lambda(x)E(x)$ çarpımının derecesi en fazla $t + n - 1$ olabilir. Buradan deg $(\Gamma(x)) < t$ olması gerektiğini ve her $j = t, \dots, n - 1$ için $\Lambda(x)E(x)$'nin j. katsayısının 0 olduğunu çıkartırız. Sonuç olarak, $t \leq l \leq n - 1$ eşitsizliğini sağlayan her l için

$$\Gamma(x) = \Lambda(x)E(x) \pmod{x^l} \quad (5.37)$$

yazabiliriz. Fakat eğer l, $2t$'den daha büyük seçilirse bu ifade E'nin bilinmeyen bileşenlerini de içerecektir, çünkü E_j sadece $j < 2t$ için bilinmektedir.

$l = 2t$ seçelim ve $\Gamma(x) = \Lambda(x)E(x) \pmod{x^{2t}}$ yazalım. Bu seçim $v \leq t$, $\Lambda(x)$'in gerçek derecesi olmak üzere denklemin matris formunda yazılmasını sağlar.

$$\begin{bmatrix} E_0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ E_1 & E_0 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & & & & & & \vdots \\ E_{t-1} & E_{t-2} & E_{t-3} & \cdots & E_1 & E_0 & 0 \\ E_t & E_{t-1} & E_{t-2} & \cdots & E_2 & E_1 & E_0 \\ E_{t+1} & E_t & E_{t-1} & \cdots & E_3 & E_2 & E_1 \\ \vdots & & & & & & \vdots \\ E_{2t-1} & E_{2t-2} & E_{2t-3} & \cdots & E_{t+1} & E_t & E_{t-1} \end{bmatrix} \begin{bmatrix} 1 \\ \Lambda_1 \\ \Lambda_2 \\ \vdots \\ \Lambda_v \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} \Gamma_0 \\ \Gamma_1 \\ \Gamma_2 \\ \vdots \\ \Gamma_{v-1} \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{5.38}$$

Bu matris denklemi $\Lambda(x)$'in tersinin monik formu için ($\widetilde{\Lambda}(x)$) yazılabilir. Bu alternatif, $\widetilde{\Lambda}(x) = \Lambda_v^{-1} x^v \Lambda(x^{-1})$ ile verilir ve matris denklemi;

$$\begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 0 & E_0 \\ 0 & 0 & 0 & \cdots & 0 & E_0 & E_1 \\ \vdots & & & & & & \vdots \\ 0 & E_0 & E_1 & \cdots & E_{t-3} & E_{t-2} & E_{t-1} \\ E_0 & E_1 & E_2 & \cdots & E_{t-2} & E_{t-1} & E_t \\ E_1 & E_2 & E_3 & \cdots & E_{t-1} & E_t & E_{t+1} \\ \vdots & & & & & & \vdots \\ E_{t-1} & E_t & E_{t+1} & \cdots & E_{2t-3} & E_{2t-2} & E_{2t-1} \end{bmatrix} \begin{bmatrix} 1 \\ \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \vdots \\ \sigma_v \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} \Gamma_0 \\ \Gamma_1 \\ \Gamma_2 \\ \vdots \\ \Gamma_{v-1} \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{5.39}$$

şeklinde olur.

Teorem 5.1 ***(Forney)*** *Hata bulan polinom $\Lambda(x)$ verilsin. Hata vektörü* ***e****'nin sıfırdan farklı bileşenleri $\Lambda(\omega^{-i})$ sıfıra eşit iken oluşur ve $\Gamma(x) = \Lambda(x)E(x) \pmod{x^{2t}}$* olmak üzere

$$e_i = -\frac{\Gamma(\omega^{-i})}{w^{-i}\Lambda'(\omega^{-i})} \tag{5.40}$$

eşitliği sağlanır.

İspat: Öncelikle $\Lambda(x)E(x) = -\Gamma(x) \quad (x^n - 1)$ eşitliğinin türevine bakalım.

$$\Lambda'(x)E(x) + \Lambda(x)E'(x) = \Gamma'(x)(1 - x^n) - nx^{n-1}\Gamma(x) \tag{5.41}$$

$\omega^{-n} = 1$ olmak üzere $x = \omega^{-i}$ alalım. Buradan;

$$\Lambda'(\omega^{-i})E(\omega^{-i}) + \Lambda(\omega^{-i})E'(\omega^{-i}) = -n\omega^{i}\Gamma(\omega^{-i}) \tag{5.42}$$

olur. Bu eşitliğe baktığımızda sadece $\Lambda(\omega^{-i}) = 0$ ise e_i sıfır olmadığını görürüz ve böylece

$$e_i = \frac{1}{n}\sum_{j=0}^{n-1} E_j \omega^{ij} = \frac{1}{n}E(\omega^{-i}) \tag{5.43}$$

eşitliğini de kullanarak teoremdeki eşitlik elde edilir.

Forney formülü $i_0 = 0$ için türetilmiştir fakat Fourier dönüşümünün özelliği bize bunu herhangi bir j_0 için nasıl değiştireceğimizi gösterir. Sıfırların spektrumunu devirli bir şekilde j_0 ile değiştirilir ve böylece sıfırlar $j = 0$ da başlar. Fourier dönüşümünün modülasyon özelliğinden bu hata spektrumunu w^{ij_0} ile çarpar. Buradan hesaplanan hata modeli bu çarpım ile düzeltilmelidir. Böylece genel formül,

$$e_i = -\frac{\Gamma(w^{-i})}{w^{-i(j_0-1)}\Lambda'(w^{-i})} \tag{5.44}$$

olur. Daha önceki dekodlama örneğini hata miktarını hesaplamak için Forney algoritmasını kullanarak bitirelim. $j_0 = 0$ olduğundan matris denklemi;

$$\begin{bmatrix} E_1 & 0 & 0 & 0 \\ E_2 & E_1 & 0 & 0 \\ E_3 & E_2 & E_1 & 0 \\ E_4 & E_3 & E_2 & E_1 \\ E_5 & E_4 & E_3 & E_2 \\ E_6 & E_5 & E_4 & E_3 \end{bmatrix} \begin{bmatrix} 1 \\ \Lambda_1 \\ \Lambda_2 \\ \Lambda_3 \end{bmatrix} = \begin{bmatrix} \Gamma_0 \\ \Gamma_1 \\ \Gamma_2 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{5.45}$$

olur. Burada $\Lambda(x) = 1 + \alpha^{14}x + \alpha^{11}x^2 + \alpha^{14}x^3$olduğunu biliyoruz. Eşitliğin sol tarafı bilindiğinden Çizelge 5.1'deki $GF(16)$'nın açılımından faydalanarak sağ taraf da hesaplanabilir.

$$\begin{bmatrix} \alpha^{12} & 0 & 0 & 0 \\ 1 & \alpha^{12} & 0 & 0 \\ \alpha^{14} & 1 & \alpha^{12} & 0 \\ \alpha^{13} & \alpha^{14} & 1 & \alpha^{12} \\ 1 & \alpha^{13} & \alpha^{14} & 1 \\ \alpha^{11} & 1 & \alpha^{13} & \alpha^{14} \end{bmatrix} \begin{bmatrix} 1 \\ \alpha^{14} \\ \alpha^{11} \\ \alpha^{14} \end{bmatrix} = \begin{bmatrix} \alpha^{12} \\ \alpha^{12} \\ \alpha^{8} \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{5.46}$$

Buradan ise $\Gamma(\mathrm{x}) = \alpha^{12} + \alpha^{12}x + \alpha^{8}x^2$ ve $\Lambda'(x) = \alpha^{14} + \alpha^{14}x^2$ bulunur. Sonuçta $j_0 = 1$ için

$$e_i = \frac{\Gamma(w^{-i})}{\Lambda'(w^{-i})} \tag{5.47}$$

olduğunu elde ederiz. Hatalar $i = 8,10,13$ noktalarındadır. Yani w^{-i}'nin değerleri α^7, α^5 ve α^2 olur. Böylece $e_8 = \alpha$, $e_{10} = \alpha^5$ ve $e_{13} = \alpha^{11}$ bulunur [16].

5.3 Peterson-Gorenstein-Zierler Dekodlayıcısı

BCH kodlar diğer devirli kodların dekodlama yöntemlerinden herhangi biri ile dekodlanabilirler. Fakat özel olarak sadece BCH kodlar için geliştirilen daha iyi algoritmalar da vardır. Bu bölümde hataların yerlerini bulmak için kullanılan Peterson

algoritmasını ve hata miktarını bulmak için kullanılan Gorenstein-Zierler algoritmasını inceleyeceğiz.

Peterson-Gorenstein-Zierler dekodlayıcısının t hata düzelttiğini ispat ederek, bir BCH kodun minimum uzaklığının en az tasarlanmış uzaklığı kadar olduğunu ispatlamak için alternatif bir yol göstereceğiz. Herhangi bir j_0 seçmemiz birşeyi değiştirmeyeceğinden eşitlikleri basitleştirmek için $j_0 = 1$ alalım.

Bir BCH kodun primitif olması gerekmeyen bir cisim eleman ω tarafından oluşturulduğunu varsayalım. Buradan, en fazla t katsayısı sıfırdan farklı olan hata polinomu

$$e(x) = e_{n-1}x^{n-1} + e_{n-2}x^{n-2} + \cdots + e_1 x + e_0 \tag{5.48}$$

şeklinde olur. Varsayalım ki, $0 \leq v \leq t$ olmak üzere v tane hata bilinmeyen $i_1, i_2, \ldots, i_v$ noktalarında bulunsun. e_{i_l} $l.$ hatanın miktarını göstermek üzere (binary kodlar için $e_{i_l} = 1$) hata polinomu

$$e(x) = e_{i_1}x^{i_1} + e_{i_2}x^{i_2} + \cdots + e_{i_v}x^{i_v} \tag{5.49}$$

şeklinde yazılabilir. Burada hata indisleri $i_1, i_2, \ldots, i_v$ 'leri ve hata miktarları $e_{i_1}, e_{i_2}, \cdots, e_{i_v}$'leri bilmiyoruz. Hataları düzeltmek için bunların hepsi hesaplanmalıdır. Aslında v'nin değerini dahi bilmiyoruz. Başlangıçta tek bildiğimiz şey alınan polinom (received) $v(x)$.

Sendrom S_1, alınan polinom $v(x)$'in ω için değeri olarak tanımlanır. $c(\omega) = 0$ olduğunu hatırlayarak

$$\begin{aligned} S_1 &= v(\omega) = c(\omega) + e(\omega) \\ &= e_{i_1}\omega^{i_1} + e_{i_2}\omega^{i_2} + \cdots + e_{i_v}\omega^{i_v} \end{aligned} \tag{5.50}$$

şeklinde hesaplanır. Bu noktasyon işlemleri düzene koymak için biraz elverişsiz olduğundan bazı yeni tanımlar yapalım. $l = 1, \ldots, v$ için, i_l indeksleri $l.$ hatanın yerini göstermek üzere, hata değerlerini $Y_l = e_{i_l}$ ve hata yeri sayılarını da $X_l = \omega^{i_l}$ şeklinde tanımlayalım. Burada ω mertebesi n olan bir eleman olduğundan hata yeri sayısı hatanın her bileşeni için farklı olmak zorunda olduğunu dikkat edelim. Şimdi sendrom S_1'i bu notasyonlar ile yazacak olursak

$$S_1 = Y_1X_1 + Y_2X_2 + \cdots + Y_vX_v \tag{5.51}$$

elde ederiz. Benze şekilde $g(x)$'in tanımında kullanılan ω'nın her kuvvetinde alınan polinomu hesaplayabiliriz. $j = 1,2, \dots ,2t$ için sendromları şöyle tanımlayalım;

$$S_j = v(\omega^j) = c(\omega^j) + e(\omega^j) = e\left(\omega^j\right) \tag{5.52}$$

Bu sendromlar hatanın Fourier dönüşümünün bileşenleri formunda olduğundan zaman bölgesi sendromlardan ayırmak için *frekans bölgesi sendromlar* olarak adlandırılabilirler. Böylece bilinmeyen hata yerleri $X_1, \dots , X_v$ ve bilinmeyen hata değerleri $Y_1, \dots , Y_v$ ile $2t$ adet eşitlik elde ederiz:

$$S_1 = Y_1X_1 + Y_2X_2 + \cdots + Y_vX_v$$

$$S_2 = Y_1X_1^2 + Y_2X_2^2 + \cdots + Y_vX_v^2$$

$$S_3 = Y_1X_1^3 + Y_2X_2^3 + \cdots + Y_vX_v^3$$

$$\vdots$$

$$S_{2t} = Y_1X_1^{2t} + Y_2X_2^{2t} + \cdots + Y_vX_v^{2t} \tag{5.53}$$

Şimdi amacımız bilinen sendromlardan bilinmeyenleri bulmak olacak. Buradan sonra dekodlama problemi bir nonlineer denklem sistemini çözme problemine indirgenmiş olur. Sendromun tanımından bu denklem sisteminin en az bir çözümünün olması gerekir. Bu çözümün tek olduğunu görebiliriz. Çözümü bulmak için kullanılacak method herhangi bir cisim üzerinde böyle bir denklem sistemi için geçerlidir.

Bir nonlineer denklem sistemini direk olarak çözmek çok zordur. Bunun yerine sendromlardan hesapladığımız ve buradan da hata yerlerini hesapladığımız ara değerler tanımlayalım. Bu işleme *hata yeri bulma dekodlaması (error locator decoding)* denilir.

Hata yeri bulan polinom olarak bilinen ve $l = 1, \dots , v$ için X_l^{-1} ters hata yerlerinde sıfırları olan bir polinom tanımlayalım;

$$\Lambda(x) = \Lambda_v x^v + \Lambda_{v-1}x^{v-1} + \cdots + \Lambda_1 x + 1 \tag{5.54}$$

eşitliği sıfırlarıyla yazarsak;

$$\Lambda(x) = (1 - xX_1)(1 - xX_2)\cdots(1 - xX_v) \tag{5.55}$$

olur. Eğer $\Lambda(x)$'in katsayılarını bilirsek hataların yerini bulabilmek için $\Lambda(x)$'in sıfırlarını elde edebiliriz. Bu durumda öncelikle sendromlardan hata bulan polinomun katsayılarını $\Lambda_1, \dots, \Lambda_v$ bulalım. Bunu yapabilirsek problem neredeyse çözülmüş olur.

Öncelikle $\Lambda(x)$ için yazılan bu iki eşitliği $Y_l X_l^{j+v}$ ile çarpalım ve $x = X_l^{-1}$ koyalım. İkinci eşitlik sıfır olur ve

$$0 = Y_l X_l^{j+v}\left(1 + \Lambda_1 X_l^{-1} + \Lambda_2 X_l^{-2} + \cdots + \Lambda_{v-1} X_l^{-(v-1)} + \Lambda_v X_l^{-v}\right) \tag{5.56}$$

ya da

$$Y_l(X_l^{j+v} + \Lambda_1 X_l^{j+v-1} + \cdots + \Lambda_v X_l^{j}) = 0 \tag{5.57}$$

elde ederiz. Bu eşitlik her j ve her l için çalışır. Bu eşitlikleri her j için, $l = 1$'den $l = v$'ye kadar toplarsak

$$\sum_{l=1}^{v} Y_l\left(X_l^{j+v} + \Lambda_1 X_l^{j+v-1} + \cdots + \Lambda_v X_l^{j}\right) = 0 \tag{5.58}$$

ya da

$$\sum_{l=1}^{v} Y_l X_l^{j+v} + \Lambda_1 \sum_{l=1}^{v} Y_l X_l^{j+v-1} + \cdots + \Lambda_v \sum_{l=1}^{v} Y_l X_l^{j} = 0 \tag{5.59}$$

elde ederiz. Bu tek tek toplamlar sendromlar olarak biliniyor, dolayısıyla

$$S_{j+v} + \Lambda_1 S_{j+v-1} + \Lambda_2 S_{j+v-2} + \cdots + \Lambda_v S_j = 0 \tag{5.60}$$

olur. $v \le t$ olduğundan eğer j, $1 \le j \le 2t - v$ aralığında ise indisler her zaman bilinen sendromları belirlerler. Böylece eşitlikler kümemiz

$$\Lambda_1 S_{j+v-1} + \Lambda_2 S_{j+v-2} + \cdots + \Lambda_v S_v = -S_{j+v} \qquad j = 1, \dots, 2t - v \tag{5.61}$$

olur. Bu ise sendromlarla $\Lambda(x)$'in katsaylarını bağlayan bir lineer denklem sistemidir. Bu denklemlerin ilk v tanesini matris formunda yazarsak

$$\begin{bmatrix} S_1 & S_2 & S_3 & \cdots & S_v \\ S_2 & S_3 & S_4 & \cdots & S_{v+1} \\ S_3 & S_4 & S_5 & \cdots & S_{v+2} \\ \vdots & & & & \vdots \\ S_v & S_{v+1} & S_{v+2} & \cdots & S_{2v-1} \end{bmatrix} \begin{bmatrix} \Lambda_v \\ \Lambda_{v-1} \\ \Lambda_{v-2} \\ \vdots \\ \Lambda_1 \end{bmatrix} = - \begin{bmatrix} -S_{v+1} \\ -S_{v+2} \\ -S_{v+3} \\ \vdots \\ -S_{2v} \end{bmatrix} \tag{5.62}$$

Bu lineer denklem sistemi de matrisin tersi hesaplanarak çözülebilir. Tersi hesaplanabilmesi için matrisin nonsingüler olduğunu göstermek gerekir. Şimdi eğer v hata varsa matrisin nonsingüler olacağını ispatlayalım. Bu ispatla gerçekte oluşan hata sayısı hakkında bir fikir sahibi oluruz.

Teorem 5.2 *v gerçekte oluşan hata sayısı olmak üzere eğer $\mu = v$ ise*

$$M_\mu = \begin{bmatrix} S_1 & S_2 & \cdots & S_v \\ S_2 & S_3 & \cdots & S_{v+1} \\ \vdots & & & \vdots \\ S_v & S_{v+1} & \cdots & S_{2v-1} \end{bmatrix} \tag{5.63}$$

matrisi nonsingülerdir. Eğer μ, v'den büyük ise matris singülerdir.

İspat $\mu > v$ için $X_\mu = 0$ ve **A** matrisi elemanları $A_{ij} = X_j^{i-1}$ ile tanımlı

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ X_1 & X_2 & \cdots & X_\mu \\ \vdots & & & \vdots \\ X_1^{\mu-1} & X_2^{\mu-1} & \cdots & X_\mu^{\mu-1} \end{bmatrix} \tag{5.64}$$

şeklinde Vandermonde matrisi olsun. Ayrıca **B** matrisi de, $i = j$ ise $\delta_{ij} = 1$ yoksa $\delta_{ij} = 0$ olacak şekilde bileşenleri $B_{ij} = Y_i X_i \delta_{ij}$ olan aşağıdaki gibi bir köşegen matris olsun.

$$\mathbf{B} = \begin{bmatrix} Y_1 X_1 & 0 & \cdots & 0 \\ 0 & Y_2 X_2 & \cdots & 0 \\ \vdots & & & \vdots \\ 0 & 0 & \cdots & Y_\mu X_\mu \end{bmatrix} \tag{5.65}$$

Şimdi bunlardan $\mathbf{ABA^T}$ hesaplarsak, $\mathbf{M_\mu}$'nin ij elemanlar olmak üzere, bu matris çarpmının elemanları,

$$(\mathbf{ABA^T})_{ij} = \sum_{l=1}^{\mu} X_l^{i-1} \sum_{k=1}^{\mu} Y_l X_l \delta_{lk} X_k^{j-1}$$

$$= \sum_{l=1}^{\mu} X_l^{i-1} Y_l X_l X_l^{j-1} = \sum_{=1}^{\mu} Y_l X_l^{i+j-1} \tag{5.66}$$

şeklinde olur. Yani böylece $\mathbf{M_\mu} = \mathbf{ABA^T}$ olduğunu söyleyebiliriz. Buradan $\mathbf{M_\mu}$'nin determinantı

$$det(\mathbf{M_\mu}) = det(\mathbf{A})det(\mathbf{B})det(\mathbf{A^T}) \tag{5.67}$$

olur. Eğer μ, v'den büyükse bu durumda $\det(\mathbf{B}) = 0$ olup buradan $\det(\mathbf{M_\mu}) = 0$ bulunur ki bu da $\mathbf{M_\mu}$'nin singüler olması demektir. Eğer $\mu = v$ ise bu durumda $det(\mathbf{B}) \neq 0$ olur. Ayrıca $\mathbf{A}$ Vandermonde matrisinin sütunları birbirinden farklı ve sıfır değillerse determinantıda sıfırdan farklıdır. Bu durum ise $\mu = v$ olduğunda sağlanır. Yani bu durumda $\mu = v$ iken $\det(\mathbf{M_\mu}) \neq 0$ olup matris nonsingülerdir. Böylece ispat biter.

Bu teorem Peterson algoritmasının temellerini oluşturur. Öncelikle hata sayısı v'nin gerçek değerini bulunur. Bunun için $\mathbf{M_t}, \mathbf{M_{t-1}}, \ldots$ matrislerinin determinantları sırasıyla hesaplanarak ve sıfırdan farklı bir değer bulana kadar devam edelir. Buradan gerçekte oluşan hata sayısı bilinmiş olur. Sonra $\Lambda(x)$'i hesaplamak için $\mathbf{M_\mu}$ matrisinin tersi bulunur. Son olarak ise hata yerlerini bulmak için $\Lambda(x)$ polinomunun sıfırları hesaplanır. Böylece hatalarımızın yeri bulunmuş olur.

Eğer kod ikili bir kod ise hata değeri hemen hesaplanır. Her durumda hata değeri bir olur. Eğer kod ikili değilse hata miktarı da hesaplanmalıdır. Tekrar sendromları veren eşitliklere dönelim.

$$S_1 = Y_1X_1 + Y_2X_2 + \cdots + Y_vX_v$$

$$S_2 = Y_1X_1^2 + Y_2X_2^2 + \cdots + Y_vX_v^2$$

$$S_3 = Y_1X_1^3 + Y_2X_2^3 + \cdots + Y_vX_v^3$$

$$\vdots$$

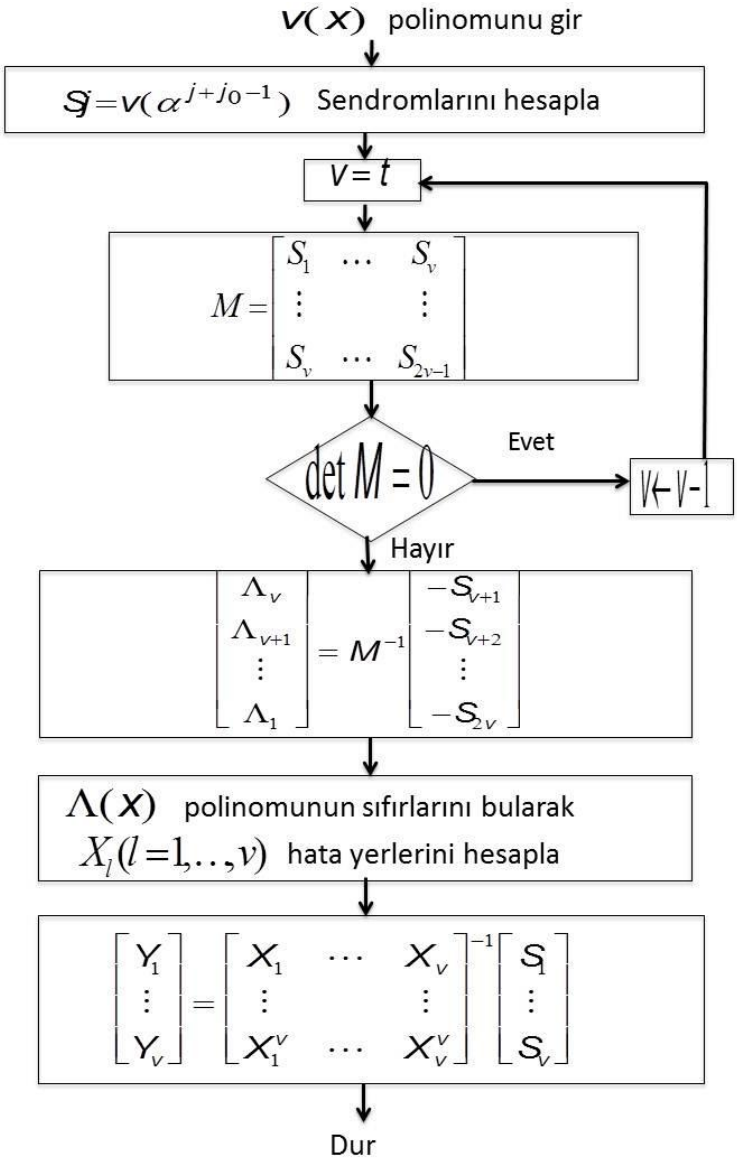
$$S_{2t} = Y_1X_1^{2t} + Y_2X_2^{2t} + \cdots + Y_vX_v^{2t} \tag{5.68}$$

Burada hata yerleri X_l'ler artık bilindiğinden eşitlikler bilinmeyen hata miktarları Y_l'ler ile bir lineer denklem sistemine dönüşmüş olur. Eğer katsayılar matrisinin determinantı sıfır değil ise ilk v denklem çözülebilir. Bu ise Gorenstein-Zierler algoritmasını oluşturur.

$$det\begin{bmatrix} X_1 & X_2 & \cdots & X_v \\ X_1^2 & X_2^2 & \cdots & X_v^2 \\ \vdots & \vdots & & \vdots \\ X_1^v & X_2^v & \cdots & X_v^v \end{bmatrix} = (X_1 X_2 \dots X_v)\ det\begin{bmatrix} 1 & 1 & \cdots & 1 \\ X_1 & X_2 & \cdots & X_v \\ \vdots & \vdots & & \vdots \\ X_1^{v-1} & X_2^{v-1} & \cdots & X_v^{v-1} \end{bmatrix} \quad (5.69)$$

eşitliğini yazabiliriz. Eğer v hata oluşmuş ise son elde ettiğimiz Vandermonde matrisinin determinantı sıfırdan farklıdır. Çünkü $X_1, X_2, \dots, X_v$ değerleri sıfırdan ve birbirinden farklıdırlar.

Şekil 5.1 de dekodlama algoritması verilmiştir. Özel durumlarda $j_0 = 1$ alınmasına rağmen burada j_0'ı rastgele koyduk.



Şekil 5.1 Peterson-Gorenstein-Zierler Dekodlayıcısı [15]

Genellikle, cisimde sonlu sayıda eleman olduğundan $\Lambda(x)$'in sıfırlarını bulmak için bir deneme yanılma yolu olan ve *Chien araması (Chien search)* olarak bilinen metodu kullanırız. Her j için $\Lambda(\omega^j)$ hesaplanarak sıfır olup olmadığı kontrol edilir. $\Lambda(x)$ polinomunu bir β noktasında hesaplamak için en basit yol *Horner kuralı*dır:

$$\Lambda(\beta) = (\dots(((\Lambda_v\beta + \Lambda_{v-1})\beta + \Lambda_{v-2})\beta + \Lambda_{v-3})\beta + \cdots + \Lambda_0) \tag{5.70}$$

Horner kuralı ile $\Lambda(x)$'i hesaplamak için sadece v tane çarpma ve v tane toplama yapmaya ihtiyaç vardır. Alternatif olarak, her j için işlemi şöyle de yazabiliriz:

$$\Lambda(\alpha^i) = \sum_{j=0}^{v} [\Lambda_j \alpha^{j(i-1)}]\alpha^j \,. \tag{5.71}$$

Şimdi bu dekodlama prosedürüne bir örnek verelim. Üç hata düzelten $(15,5,7)$ BCH kodunu göz önüne alalım. Kodun üreteç polinomu aşağıdaki gibidir:

$$g(x) = x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1 \tag{5.72}$$

Örneğin alınan polinomumuz da $v(x) = x^7 + x^2$ olsun. Üç veya daha az hata oluştuğundan kodsöz tüm elemanları sıfır olan bir söz olup, $v(x) = e(x)$ demektir. Fakat dekodlayıcı bu gözlemi ve yorumu yapamaz. Şimdi dekodlama algoritmasının tüm adımlarını uygulayacağız. Öncelikle, $x^4 + x + 1$ polinomu ile kurulan $GF(16)$ aritmetiği kullarak, $\omega = \alpha$ için sendromları hesaplayalım. Sendromlar:

$$S_1 = \alpha^7 + \alpha^2 = \alpha^{12}$$

$$S_2 = \alpha^{14} + \alpha^4 = \alpha^9$$

$$S_3 = \alpha^{21} + \alpha^6 = 0$$

$$S_4 = \alpha^{28} + \alpha^8 = \alpha^3$$

$$S_5 = \alpha^{35} + \alpha^{10} = \alpha^0$$

$$S_6 = \alpha^{42} + \alpha^{12} = 0 \tag{5.73}$$

$v = 3$ alalım. Bu durumda

$$\mathbf{M} = \begin{bmatrix} S_1 & S_2 & S_3 \\ S_2 & S_3 & S_4 \\ S_3 & S_4 & S_5 \end{bmatrix} = \begin{bmatrix} \alpha^{12} & \alpha^9 & 0 \\ \alpha^9 & 0 & \alpha^3 \\ 0 & \alpha^3 & 1 \end{bmatrix} \tag{5.74}$$

bulunur. Buradan ise **M**'nin determinantı sıfır olduğu görülür. Şimdi $v = 2$ alalım. Bu durumda ise

$$\mathbf{M} = \begin{bmatrix} S_1 & S_2 \\ S_2 & S_3 \end{bmatrix} = \begin{bmatrix} \alpha^{12} & \alpha^9 \\ \alpha^9 & 0 \end{bmatrix} \tag{5.75}$$

bulunur. Buradan ise determinantın sıfırdan farklı olduğunu görürüz. Yani iki hata oluştuğu anlamına gelir. Devam ederek **M**'nin tersini hesaplarsak:

$$\mathbf{M^{-1}} = \begin{bmatrix} 0 & \alpha^6 \\ \alpha^6 & \alpha^9 \end{bmatrix} \tag{5.76}$$

buluruz. Şimdi Λ'ları hesaplarsak:

$$\begin{bmatrix}\Lambda_2\\ \Lambda_1\end{bmatrix} = \mathbf{M}^{-1}\begin{bmatrix}-S_3\\ -S_4\end{bmatrix} = \begin{bmatrix}0 & \alpha^6\\ \alpha^6 & \alpha^9\end{bmatrix}\begin{bmatrix}0\\ \alpha^3\end{bmatrix} = \begin{bmatrix}\alpha^9\\ \alpha^{12}\end{bmatrix} \tag{5.77}$$

buluruz ve böylece

$$\Lambda(x) = \Lambda_2 x^2 + \Lambda_1 x + 1 = \alpha^9 x^2 + \alpha^{12} x + 1 \tag{5.78}$$

elde ederiz. Chien araması ile bunu çarpanlarına ayırırsak:

$$\Lambda(x) = (\alpha^7 x + 1)(\alpha^2 x + 1) \tag{5.79}$$

buluruz ki bu da $\Lambda(x)$'in sıfırlarının $x = \alpha^{-7}$ ve $x = \alpha^{-2}$ noktalarında olduğu anlamına gelir. Buradan hata yerlerinin bu sıfırlara karşılık gelen yerler olduğu sonucuna varırız. Yani hatalar ikinci ve yedinci bileşenlerde oluşmuştur. Kod ikili bir kod olduğundan hata miktarı 1 olup hata polinomumuz da $e(\mathrm{x}) = x^7 + x^2$ olur [15].

BÖLÜM 6

SONUÇ VE ÖNERİLER

Bu tez çalışmasında Fourier dönüşümünün sonlu cisimler üzerinde nasıl işlediği incelenmiştir. Dönüşümün hata düzelten kodlardaki rolü açıklanmış, özellikle Reed-Solomon ve BCH kodları gibi devirli kodların spektral tanımları yapılmıştır. Spektral tekniklerin kodlama ve dekodlama algoritmalarındaki kullanımları özetlenmiştir. Frekans bölgesi mantığıyla düşünerek alternatif kodlama ve dekodlama metodları açıklanmıştır. Bu yaklaşımla kodlama ve dekodlama algoritmalarına yeni bakış açıları geliştirilebilir. Var olan kodlar farklı bir yolla yeniden sınıflandırılabilir ve böylece kodlar arasında yeni ilişkiler bulunabilir. Ayrıca frekans bölgesindeki kodlama ve dekodlama algoritmaları bazı durumlarda uygulamada daha avantajlı olduğundan, bu konu dönüşüm tekniklerine aşina olan mühendislerin ilgisini çekebilir. Son olarak, dönüşüm tekniklerine dayanarak yeni kodlar, algoritmalar ve teknikler ortaya çıkartılabilir.

KAYNAKLAR

[1] Wörner S., (2008). "Fast Fourier Transform", Swiss Federal Institute of Technology Zurich Numerical Analysis Seminar, *www.math.ethz.ch/education/bachelor/seminars/.../nas/woerner.pdf, 09 Mayıs 2011.*

[2] Reed, I.S., (1960). "Polynomial Codes over Certain Finite Fields", Journal of Society of Industrial and Applied Mathematics, 8:300-304.

[3] Mattson, H.F., (1961). "A New Treatment of Bose-Chaudhuri Codes", Journal of Society of Industrial and Applied Mathematics, 9:654-669.

[4] Pollard, J.M., (1971). "The Fast Fourier Transform in a Finite Field", Mathematics of Computation, 25: 365-374.

[5] Gore, W.C., (1973). "Transmitting Binary Symbols with Reed-Solomon Codes", Proceeding of the Princeton Conference on Information Science System, Princeton, NJ, 495-497.

[6] Chien, R.T. ve Choy, D.M., (1975). "Algebraic Generalization of BCH-Goppa-Helgerts Codes", IEEE Transactions on Information Theory, 21:70-79.

[7] Lempel, A. ve Winograd, S., (1977). "A New Approach to Error-Correcting Codes, IEEE Transactions on Information Theory, 23:503-508.

[8] Blahut, R.E., (1979). "Transform Techniques for Error-Control Codes", IBM Journal of Research and Development, 23:299-315.

[9] Schaub T., (1988). "A Linear Complexity Approach to Cyclic Codes", Doctor of Technical Sciences Dissertation, Swiss Federal Institute of Technology, Zurich.

[10] Peterson, W.W., (1960). "Encoding and Error-Correction Procedures for the Bose-Chaudhuri Codes", IRE Transactions on Information Theory, 6:459-470.

[11] Gorenstein D.C. ve Zierler N., (1961). "A Class of Error Correcting Codes in p^m Symbols", Journal of the Society of Industrial and Applied Mathematics, 9:207-214.

[12] Forney, G.D., (1961). "On Decoding BCH Codes", IEEE Transactions on Information Theory, 11:549-557.

[13] Horiguchi, T., (1989). "High Speed Decoding of BCH Codes Using a New Error Evaluation Algorithm", Electronics and Communications in Japan, 72:12.

[14] Koetter, R., (1997). "On the Determination of Error Values for Codes from Class of Maximal Curves", Proceedings of the 35th Allerton Conference on Communication, Control, and Computing, University of Illiois, Monticello, Illinois.

[15] Blahut, R.E., (2003). Algebraic Codes for Data Transmission, Cambridge University Press, Cambridge.

[16] Blahut, R.E., (2008). Algebraic Codes on Lines, Planes, and Curves, Cambridge University Press, Cambridge.

[17] Michelson, A., (1976). "A Fast Transform in Some Galois Fields and an Application to Decoding Reed-Solomon Codes.", IEEE Abstracts of Papers-IEEE International Symposium on Information Theory, Ronneby, Sweden.

[18] Peterson, W.W. ve Weldon, E.J., (1972). Error-Correcting Codes, Second Edition, The M.I.T. Press, Cambridge, MA.

[19] Chien, R.T., (1972). "A New Proof of the BCH Bound", IEEE Trans. Info Theory, 18:541.

[20] Welch, L. ve Berlekamp, E.R., (1983). "Error Correction for Algebraic Block Codes", U. S. Patent 4 633 470.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Sultan SELÇUK

Doğum Tarihi ve Yeri : 16.10.1986 İstanbul

Yabancı Dili : İngilizce

E-posta : sultanselcuk@sehir.edu.tr

ÖĞRENİM DURUMU

Derece	Alan	Okul/Üniversite	Mezuniyet Yılı
Y. Lisans	Matematik	Yıldız Teknik Üniversitesi	2011
Lisans	Matematik	İstanbul Ticaret Üniversitesi	2009
Lise	Matematik-Fen	Bahçelievler Anadolu Lisesi	2004

İŞ TECRÜBESİ

Yıl	Firma/Kurum	Görevi
2009-devam	İstanbul Şehir Üniversitesi	Araştırma Görevlisi
2008-2009	Kültür Dersanesi	Stajyer Öğretmen