ANKARA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

DAĞITIK VERİ DEPOLAMA SİSTEMLERİNDE AĞ KODLAMA ALGORİTMALARININ UYGULAMALI KARŞILAŞTIRILMASI

Burak BEZİRCİ

ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

ANKARA
2018

TEZ ONAYI

Burak BEZİRCİ tarafından hazırlanan **"Dağıtık Veri Depolama Sistemlerinde Ağ Kodlama Algoritmalarının Uygulamalı Karşılaştırılması"** adlı tez çalışması **16/02/2018** tarihinde aşağıdaki jüri tarafından oy birliği ile 'Ankara Üniversitesi Fen Bilimleri Enstitüsü Elektrik-Elektronik Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Danışman : Prof. Dr. Asım Egemen YILMAZ
Ankara Üniversitesi Elektrik-Elektronik Mühendisliği Anabilim Dalı

Jüri Üyeleri :

Başkan : Prof. Dr. Erkan AFACAN
Gazi Üniversitesi Elektrik-Elektronik Mühendisliği Anabilim Dalı

Üye : Prof. Dr. Asım Egemen YILMAZ
Ankara Üniversitesi Elektrik-Elektronik Mühendisliği Anabilim Dalı

Üye : Yrd. Doç. Dr. Gökhan SOYSAL
Ankara Üniversitesi Elektrik-Elektronik Mühendisliği Anabilim Dalı

Yukarıdaki sonucu onaylarım.

Prof. Dr. Atila YETİŞEMİYEN
Enstitü Müdürü

ETİK

Ankara Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırladığım bu tez içindeki bütün bilgilerin doğru ve tam olduğunu, bilgilerin üretilmesi aşamasında bilimsel etiğe uygun davrandığımı, yararlandığım bütün kaynakları atıf yaparak belirttiğimi beyan ederim.

16.02.2018

Burak BEZİRCİ

ÖZET

Yüksek Lisans Tezi

DAĞITIK VERİ DEPOLAMA SİSTEMLERİNDE AĞ KODLAMA ALGORİTMALARININ UYGULAMALI KARŞILAŞTIRILMASI

Burak BEZİRCİ

Ankara Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik-Elektronik Mühendisliği Anabilim Dalı

Danışman: Prof. Dr. Asım Egemen YILMAZ

Bu tez çalışmasında, bir dağıtık veri depolama sistemi modeli oluşturulmuştur. Oluşturulan bu sistem modelinde verinin oluşturulması, kodlanması, ağ üzerinden düğümlerine ayrılması, hatalı verinin düzeltilmesi ve kod çözme ile orijinal verinin yeniden elde edilmesi sağlanmıştır. Sistem modelinde kullanılmak üzere, optimal kodlama yöntemini belirleyebilmek için farklı veri boyutlarında, Reed-Solomon, Hamming ve Parite dizi kodlama yöntemlerinin, belirlenen kriterlere (kodlama/kod çözme süresi, verimlilik, hata düzeltme oranı, genişletilebilirlik, verinin taşınabilme kapasitesi) göre uygulamalı kıyaslamaları yapılmıştır. Elde edilen bulguların sonucunda, incelenen kodlama yöntemlerinin kullanım alanları belirlenmiş olup, optimal bir kodlama yönteminin oluşturulması için gerekli olabilecek kriterler belirtilmiştir. Buna göre yazılım tabanlı oluşturulan bir dağıtık veri depolama sisteminde, farklı kullanım alanlarına göre aynı anda farklı silinti kodlama yöntemlerinin kullanılabileceği sonucu elde edilerek, dağıtık veri depolama çözümleri için hibrid sistem tasarımı önerilmiştir.

Şubat 2018, 95 sayfa

Anahtar Kelimeler: Dağıtık veri depolama sistemi, silinti kodlama, kodlamanın verimliliği, hata düzeltme, verinin taşınabilme kapasitesi, verinin güvenirliği

ABSTRACT

Master Thesis

COMPARISON OF NETWORK CODING ALGORITHMS
FOR DISTRIBUTED STORAGE SYSTEM

Burak BEZİRCİ

Ankara University
Graduate School of Natural and Applied Sciences
Department of Electrical and Electronics Engineering

Supervisor: Prof. Dr. Asım Egemen YILMAZ

In this thesis, a distrubited storage system model has been established. The generated system model is provided that creating data, encoding, sperating into nodes via network, correcting the erroneous data and recovering the original data with decoding. Reed-Solomon, Hamming and Parity-Array coding methods are compared practically according to specified criteria (coding/decoding time, efficiency, error correction rate, expandability, data migration capacity) with different data sizes to determine the optimal coding methods to use in the system model. As a result of the findings obtained, the use case of the examined coding methods are determined and criterias that may be required are specified for the optimum coding method. Accordingly, the result is obtained that different erasure coding methods can be used for different use cases in software defined distrubeted storage systems. Consequently, hybrid system design is proposed for distributed data storage solutions.

February 2018, 95 pages

Key Words: Distrubeted storage systems, erasure coding, efficiency for erasure coding, correcting the erroneous, data migration capacity, reliability of the data

TEŞEKKÜR

Yüksek lisans tezim ile ilgili bütün aşamalarda beni yönlendiren, her anlamda destek olan danışman hocam Sayın Prof. Dr. Asım Egemen YILMAZ'a (Ankara Üniversitesi Elektrik Elektronik Mühendisliği Anabilim Dalı Öğretim Üyesi), varlıklarıyla bana güç veren sevgili ailem Yener BEZİRCİ, Fatma BEZİRCİ ve Emre BEZİRCİ'ye, tüm özverileriyle bana her türlü desteği veren sevgili eşim Emine BEZİRCİ'ye, çalışmalarımın tamamlanmasında en büyük motivasyonum olan oğlum Uygar BEZİRCİ'ye, tez yazım sürecinde yardımlarını ve emeklerini esirgemeyen dostlarım Ercan BEŞER ve Emrah ÇİÇEK'e, sunum günü beni yalnız bırakmayan Pınar YILMAZ'a ve fikirleriyle beni yönlendiren, çalışmamın tamamlanmasında itici güç olan dostum Tuğrul AYDEMİR'e sonsuz teşekkürlerimi sunarım.

Burak BEZİRCİ
Ankara, Şubat 2018

İÇİNDEKİLER

SİMGELER DİZİNİ

c	Kod kelimesi
C	Kodlanan kelimeler kümesi
D	Kod kelimesi bloğu
$d(C)$	C kodunun Hamming uzaklığı
$d(u,v)$	u ve v vektörlerinin uzaklığı
e	Hata vektörü
$E(z)$	Reed-Solomon kodlamasında hata polinomu
G	Üreteç matrisi
F	Cisim
GF	Galois cismi (Galois Field)
G^{-1}	Üreteç matrisinin tersi
$G(z)$	Reed-Solomon kodlamasında üreteç polinomu
H	Parite kontrol matrisi
H^T	Parite kontrol matrisinin transpozu
Hx^T	x kod sözünün sendromu
I_k	k uzunluklu birim matris
k	Lineer C kodunun boyutu
$M(z)$	Reed-Solomon kodlamasında mesaj polinomu
n	Kod sözcüğü blok uzunluğu
N	Blok sayısı
p	Her bir bloğun parite düğüm uzunluğu
P	Her bir bloğun parite vektörü
p_u	parite uzunluğu
$P(z)$	Reed-Solomon kodlamasında parite polinomu
$p_{0..n}$	Parite bitleri
r	Alınan kelime
R	Reel vektör uzayı
s	Kod kelimesinin sendromu
$s(y)$	y'nin sendrom fonksiyonu
u, v	R^n kümesinin elemanları
t	Düzeltilebilen hata sayısı
$w(C)$	C kodunun minimum ağırlığı
$[n,k]$	Lineer C kodu
$[n,k,d]$	Lineer C kodu uzaklık ile ifadesi
$\Lambda(z)$	Reed-Solomon kodlamasında Chien hata yerleri polinomu
$\Omega(z)$	Reed-Solomon kodlamasında Forney hata değerleri polinomu

Kısaltmalar

BCH	Binary Coded Hexadecimal
DVDS	Dağıtık Veri Depolama Sistemi
G/Ç	Giriş/Çıkış
HDFS	Hadoop File System
HDO	Hata Düzeltme Oranı
IDC	International Data Corporation
MADS	Maksimum Düğüm Sayısı
MDR	Maximum Distance Rank
MDS	Maksimum Uzaklıkla Ayrılabilen: Maximum Distance Separable
MIDS	Minimum Düğüm Sayısı
MKS	Merkezi Kontrol Sunucusu
RAID	Redundant Array of Independent Disks
RS	Reed-Solomon
QFS	Quancast File System
UVS	Uzak Veri Sunucusu
TEDO	Tolere Edilebilir Düğüm Oranı
VAO	Veri Artış Oranı
XOR	Exclusive or

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

1. GİRİŞ

Bilişim teknolojileri, günlük yaşam da dahil olmak üzere bir çok sektörde etkili olmaya başlamıştır. Bununla birlikte dijital veri üretimi de artmaktadır. Elde edilen dijital verilerin işlenmesi, kişiye veya nesneye özel hizmetlerin oluşmasına imkan vermektedir. Bu gelişmeler nesnelerin interneti, dijital dönüşüm gibi kavramlarının gelişmesine neden olmaktadır.

Günümüzde akıllı telefon, akıllı saat gibi ürünlerin gelişmesine ve yaygınlaşmasına bağlı olarak dijital veri üretimi artmaktadır. Artan bu verinin anlamlandırılarak kullanılması ve ticari olarak bir kazanım elde edilmesi için birçok teknoloji firması bu doğrultuda yatırımlarını yapmaktadır. En büyük teknoloji şirketlerinin geliştirdikleri hizmetler de bu anlamda bilgi toplamanın en değerli kazanım olduğunu göstermektedir. Bu sayede kişiye özel reklam ve içerik geliştirmek mümkün olmaktadır. Örnek olarak; Amazon, Facebook, Google gibi firmalar verilebilir. Bu da dijital veri boyutunun her geçen gün daha çok artacağı anlamına gelmektedir. International Data Corporation'nın (IDC) yayınladığı bir rapora göre dijital veri boyutunun her iki yılda bir iki kat artması ve 2020 yılına kadar ise 44 Zetabayt olması beklenmektedir (Turner vd. 2014). Veri artışının bu ölçülerde artmasıyla birlikte veri merkezlerinde kullanılan veri depolama sunucularının kapasitelerinde ve sayılarında hızlı bir artış söz konusu olacaktır. Bu artış veri depolama maliyetlerinin de ciddi oranda yükselmesine neden olacaktır.

Veri merkezlerindeki atıl kapasitenin başka bir kaynak tarafından kullanılabilmesi fikri bugünün bulut bilişiminin temellerini oluşturmaktadır. Günümüzde bulut sistemlerindeki kaynakların yüksek kapasiteleri ve ağ yapılarının yüksek bant genişlikleri bulunmaktadır. Fakat veri depolama yapıları yüksek bant genişliğinde olsa bile Giriş/Çıkış (G/Ç) kapasiteleri, fiziksel bir depolama sunucusu için sınırlıdır. Bu doğrultuda, bulut sistemlerinde geliştirilen uygulamaların performans gereksinimleri standart mimarilere göre değişmektedir. Örneğin, verinin tek bir veri merkezindeki depolama sunucularında tutulduğunu ve buna göre verinin okuma/yazma işlemlerinin yapıldığı göz önüne alınırsa, uygulama performansındaki darboğazın, bant genişliğinden değil de G/Ç kullanımından olduğu tespit edilmiştir (Haytaoğlu 2015).

Artan veri depolama maliyetleri, esneklik ve karşılaşılan performans darboğazlarının aşılabilmesi için verinin tek bir kaynakta değil de birden çok kaynakta tutulabilmesi ve tek bir kaynakmış gibi işlem yapılabilmesi fikrini ortaya çıkartmıştır. Bu durum, dağıtık veri depolama sistemlerinin (DVDS) de temelini oluşturmaktadır.

Veri depolama yapılarındaki depolama sunucuları çeşitli nedenlerden dolayı arızalanabilmektedir. Bu sunucuların arızalanması bilgi teknolojilerinde, olağan bir durum olarak kabul edilmekle birlikte depolama birimlerinin tasarım ve maliyetleri buna göre ele alınmaktadır (Ghemawat vd. 2003). Depolama sunucularının arızalanması, veri kaybına, dolayısıyla da hizmet kesintisine neden olmaktadır. Eğer veri, yedekli bir yapıda ise yeniden verinin kullanılabilir olması için geçen süre de hizmet kesintisi olarak düşünülebilir. Bu kesinti, iş sürekliliğini etkilemekte ve depolama sunucu maliyetinden çok daha önemli olmakta; hatta bu kesintinin çok daha yüksek maliyetleri olabilmektedir. Arıza anında iş sürekliliğinin sağlanabilmesi için arızanın tolere edilebildiği yöntemlerin kullanılmasına gereksinim duyulmaktadır. Bilinen yöntemler arasında veri replikasyonu ve veri dağıtımı yöntemleri yer almaktadır.

Veri replikasyonuna en iyi örnek olarak, RAID'in (Redundant Array of Independent Disks), RAID1 ve RAID10 mimarileri gösterilebilir (Patterson vd. 1988). RAID yapılarında eş zamanlı iki hata tolere edilemediği için Cleversafe'e (2015) göre, büyük boyutlu verilerde verimli olmamaktadır (Haytaoğlu 2015). Ayrıca veri boyutu belli bir büyüklüğü aştığı zaman depolama sunucusunun da bu oranda kapasitesinin arttırılması gerekmektedir. Bu durumda da verinin çoklu kopyalama yöntemi ile esnekliği sağlanabilmektedir. Bu işlem depolama maliyetini çok fazla arttırmaktadır.

Veri dağıtımı yöntemi yaklaşımı Rabin (1989) tarafından tanımlanmıştır. Bu yöntemde veriler kodlanarak, parçalarına ayrılmakta, her bir parça ağ üzerinden depolama sunucularına dağıtılmaktadır. Uygulamanın, veriyi tek bir bütün olarak yeniden kullanabilmesi için tüm depolama sunucuları yerine belli bir orandaki sunuculara erişmesi yeterli olmaktadır. Erişilen depolama sunucuları üzerindeki veri parçalarının yapılandırılmasıyla da orijinal veri elde edilmektedir. Bu yaklaşım ile eş zamanlı hatalar

tolere edilebilmekte, veri boyutuna bağlı olarak esnek depolama birimleri eklenebilmektedir. Verinin birebir replikasyonuna göre kodlanması genellikle daha az veri depolama sunucusuna ihtiyaç olacağını, dolayısıyla da maliyetlerin düşürülebileceğini göstermektedir. Veri dağıtımı yaklaşımı aslında DVDS'nin kuramsal temelini oluşturmaktadır. Bunun uygulaması ise silinti (eresure) kodlama yöntemleri kullanılarak yapılabilmektedir.

Bu tez çalışması kapsamında, DVDS için yapılan çalışmaların literatür özetleri ikinci bölümde yer almaktadır. Ayrıca silinti kodlama yöntemi, bu tez kapsamında ele alınmış olup, temelini oluşturan kodlama teorileri ile birlikte farklı uygulamaları olan farklı kodlama yöntemlerinin kuramsal temelleri de ikinci bölümde anlatılmaktadır.

Üçüncü bölümde, oluşturulan sistem modelinin tanımları ve uygulama adımları açıklanmıştır. Sistem modeli oluşturulurken DVDS'lerinde kullanılan bazı modeller incelenmiş olup, kodlama yöntemlerine göre en uygun model belirlenmiştir.

Her bir silinti kodlama yönteminin, oluşturulan sistem modelinde belirlenen kriterlere göre çalıştırılması sağlanmış, elde edilen bulgular, kıyaslamalı açıklamaları ile dördüncü bölümde ifade edilmiştir.

Son olarak elde edilen bulgulara göre kodlama yöntemlerinin avantaj ve dezavantajlarına yer verilerek, optimal bir DVDS tasarlanmasında önemli olabilecek kriterler ve kullanım senaryoları açıklamıştır.

Bu tez çalışması kapsamında, bir DVDS modeli oluşturularak kullanılabilecek kodlama yöntemlerinin çeşitli kriterlere göre kıyaslanması, elde edilen sonuçlara göre de optimal bir DVDS'nin tasarlanmasına yardımcı olunması hedeflenmiştir.

2. KURAMSAL TEMELLER

2.1 Dağıtık Veri Depolama Sistemleri

Dağıtık bir sistem, kullanıcılara tek bir sistem olarak görünen, bağımsız işlemler bütünüdür (Tanenbaum ve Steen 2007).

DVDS, dağıtık veri işlemesinin yapılabildiği sistemlerin bütünüdür. Bu sistemler, dört ana başlıkta incelenebilir: dağıtık dosya sistemleri, dağıtık disk dizileri, silinti kodlama ile oluşturulan veri depolama sistemleri, hiyerarşik veri depolama sistemleri.

2.1.1 Dağıtık dosya sistemleri

DVDS'nin alt kırılımlarından biri dağıtık dosya sistemleridir. DDS, ağ üzerinden eş zamanlı birçok istemcinin erişimine ve kullanımına izin veren, birden çok veri depolama kaynağı kullanan herhangi bir dosya sistemidir.

DDS'ne geleneksel yapılarıyla en güzel örnekler; Cedar (Hagmann 1987), Sprite (Nelson vd. 1988), Coda (Satyanarayanan vd. 1990), Harp (Liskov vd. 1991) olarak gösterilebilir. Ayrıca büyük veri (big data) kullanımları için tablosuz (notable) yapıları ortaya çıkmaktadır. Tablosuz çalışmalara örnek olarak; Quancast File System (QFS) (http://quantcast.github.io 2012), Ceph (Weil vd. 2006), Hadoop File System (HDFS) (Dean ve Ghemawat 2004), Google File System (Ghemawat vd. 2003) verilebilir.

2.1.2 Dağıtık disk dizileri

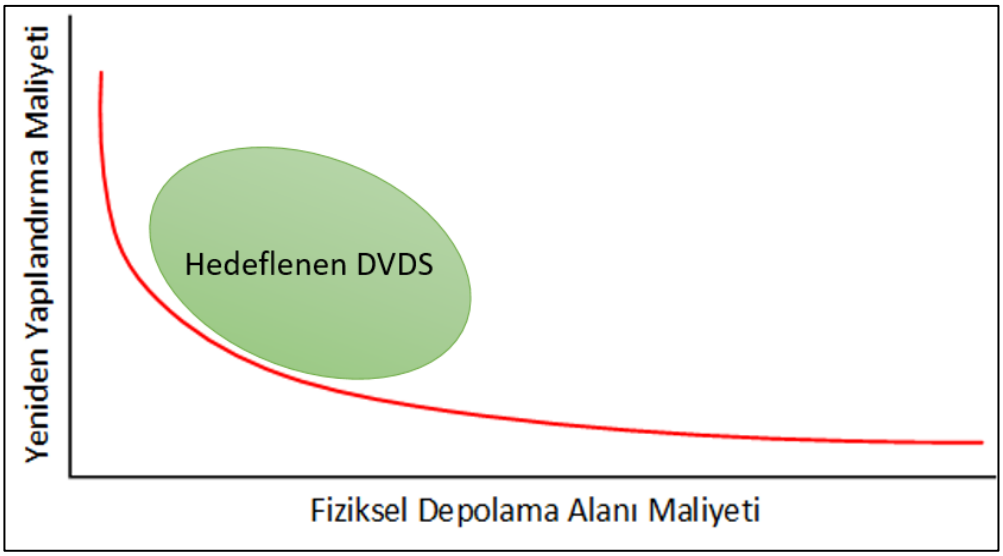
Geleneksel veri depolama çözümü olarak kullanılan ve günümüzde üzerine en fazla çalışma yapılan bir konudur. Özellikle RAID yapıları en iyi bilinen örneğidir (Patterson vd. 1988). RAID yapılarında şerit, replikasyon ve silinti kodlama yöntemlerinin de kullanılabilmesi ile farklı RAID yapıları ortaya çıkmıştır. Şeritleme (striping) kullanılarak oluşturulan RAID0, replikasyon kullanılarak oluşturulan RAID1, silinti

kodlama yöntemi kullanılarak oluşturulan RAID2, RAID3, RAID4, RAID5, RAID6 yapıları örnek olarak verilebilir.

RAID yapılarından farklı olarak Zebra (Hartman ve Ousterhout 1995), TickerTAIP (Venkataraman ve Wilkes 1994), NASD (Gibson vd. 1998) çalışmaları da DDS'ye örnek gösterilebilir.

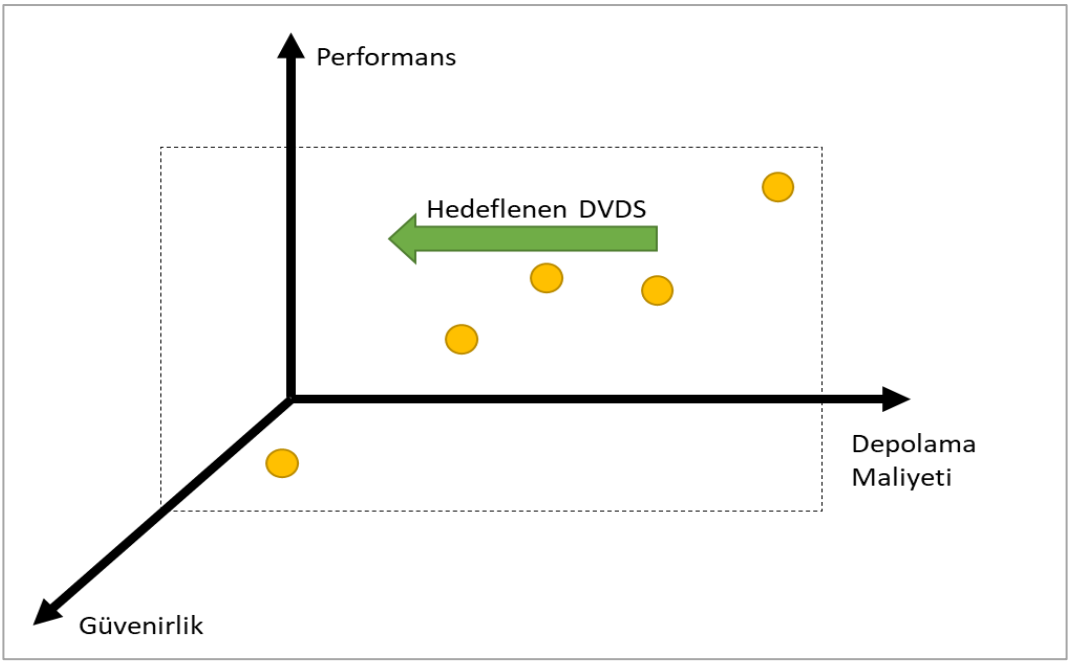
2.1.3 Silinti kodlama ile oluşturulan veri depolama sistemleri

Silinti kodlamaları uzun zamandır veri iletişiminde hata düzeltme yeteneklerinden dolayı kullanılmaktadır. DVDS için de diğer kodlama yöntemlerine göre silinti kodlamaların kullanılmasındaki temel neden, hata durumunun yanı sıra verinin hiç olmaması veya ağ bazlı ulaşılamaması durumunda orijinal verinin yeniden elde edilebilmesini mümkün kılmaktadır. Bu durumda silinti kodlama ile oluşturulan DVDS sadece hatalı veriyi tespit etmekle kalmaz erişilemeyen verinin yerini tespit edebilmektedir. Hamming kodlamaları ile başlayan silinti kodlama yöntemleri, dağıtık veri depolama çözümü olarak da kullanılmak üzere geliştirilmektedir (Plank vd. 1997, Plank 2005, 2013, Khan vd. 2012). Özellikle Reed-Solomon (RS) (Reed ve Solomon 1960), XOR (Sobe ve Peter 2008), Hover (Hafner 2006), STAIR (Li ve Lee 2014), kodlamaları geleneksel silinti kodlamalarına örnek olarak verilebilir. Bunun yanı sıra ağ ve fiziksel veri depolama alanlarının kullanımının düşürülmesi için veri merkezleri özelinde daha hızlı ve etkili silinti kodlama yöntemleri üzerine çalışılmaktadır (Rashmi vd. 2014). Bu doğrultuda elde edilmesi hedeflenen silinti kodlama yöntemlerinde yeniden yapılandırma maliyetlerinin düşürülmesi oldukça önemlidir. Fiziksel depolama alanı maliyeti ile yeniden yapılandırma maliyeti arasındaki ilişki şekil 2.1'deki gibi Huang vd. (2012) çalışmasında tanımlanmıştır. Bu ilişkiye göre her iki kriterin de sağlanmasının mümkün olmadığı sonucu çıkartılmıştır. Optimal bir DVDS oluşturulması için baz alınan başarı değerine göre her iki kriterin uç noktalarından ziyade şekil 2.1 gösterildiği gibi hedeflenen bölgede bir DVDS mimarisinin oluşturulması gerektiği sonucu çıkarılmıştır.



Şekil 2.1 Silinti kodlamada yeniden yapılandırma, depolama alanı maliyeti

Silinti kodlama yöntemleriyle elde edilecek bir DVDS'inde maliyetlerle birlikte dikkat edilmesi gereken diğer hususlar ise, performans ve verinin güvenirlik değerleri olmalıdır. Bu kriterler arasındaki ilişki Huang vd. (2012) çalışmasında belirttiği gibi şekil 2.2'de gösterilmiştir.



Şekil 2.2 DVDS - maliyet, performans, güvenirlik

Uçtan uca klasik silinti kodlama yöntemleri kullanılarak oluşturulan DVDS'ye örnek olarak; Oceanstore (Kubiatowicz vd. 2000), Window Azure Storage (Huang vd. 2012), Glacier (Haeberlen vd. 2005), Ursa Minor (Abd-El-Malek vd. 2005) sistemleri verilebilir. Bu tez çalışması kapsamında da silinti kodlama yöntemleri ile oluşturulan bir sistem modeli üzerine çalışılmıştır.

2.1.4 Hiyerarşik veri depolama sistemleri

Farklı gereksinimleri karşılamak üzere, hiyerarşik bir yapı içerisinde, DVDS'nin farklı alt sistemleri kullanılarak oluşturulan sistemlerdir. Bu sistemlerde veri merkezlerinin tüm verisinden en uçta kullanılan verinin türüne göre farklı alt sistemler kullanılmaktadır. Bu yapı hiyerarşik bir şekilde kullanıldığı için hiyerarşik veri depolama sistemleri olarak adlandırılmaktadır. Örneğin işlenen verinin depolanmasında Reed-Solomon yöntemi kullanılırken veri merkezlerinin yedekliliği XOR kodlama ile yapılmaktadır. Ayrıca kritik verilerin depolanması için RAID yapıları da kulanılmaktadır. Günümüzde performans, maliyet ve esneklik kriterlerine göre üzerine en çok çalışılan sistemlerden biri olmakla birlikte en bilinen örnekleri; Facebook f4 (Muralidhar vd. 2014), HP AutoRAID (Wilkes vd. 1996), Google Cloud Storage (https://developers.google.com 2010) olarak verilebilir.

2.2 Kodlama Teorisindeki Temel Kavramlar

Silinti kodlamanın matematiksel olarak temellerini kodlama teorisindeki kavramlar oluşturmaktadır. Bu bölümde kavramların özellikleri belirtilmiş olup sonraki bölümlerde bu kavramlar kullanılarak işlem yapılmaktadır.

2.2.1 Cebirsel grup, halka, cisim ve vektör uzayları

Bir reel vektör uzayı $(V; R; \oplus; \odot; +; \cdot)$ şeklinde gösterilebilir.

$G \neq \emptyset$; $*: GxG \rightarrow G$ için

Eğer $*$ operatörü için aşağıdaki aksiyomlar sağlanıyorsa $(G;*)$ bir grup oluşturulur.

i. $a*(b*c)=(a*b)*c\ ,\forall a,b,c\in G$ için birleşme,

ii. $\exists e\in G\mid a*e=e*a=a\ ,\forall a\in G$ için etkisiz eleman,

iii. $\forall a\in G\ ,\exists b\in G\mid a*b=b*a=e$ için ters eleman

Bu özelliklere ek olarak aşağıdaki aksiyomu sağlıyorsa değişmeli grup oluşturulur.

iv. $a*b=b*a\ ,\forall a,b\in G$ için değişme

$R\neq\emptyset$; $(R;*;\lozenge)$ sistemi için;

$*: RxR\ \rightarrow R$ ve $\lozenge: RxR\ \rightarrow R$ aşağıdaki koşulları sağlıyorsa halka olarak adlandırılır.

i. $(R;*)$ bir değişmeli grup oluşturuyor ise,

ii. $\lozenge$ operatörünün birleşme özelliği varsa,
$a\lozenge(b\lozenge c)=(a\lozenge b)\lozenge c\ ,\forall a,b,c\in R$

iii. $\lozenge$ operatörünün $*$ operatörü üzerinde dağılma özelliği varsa
$a\lozenge(b*c)=(a\lozenge b)*(\mathrm{a}\lozenge c)\ ,\forall a,b,c\in R$

Eğer, $\lozenge$ operatörünün değişme özelliği varsa (Eşitlik 2.1); değişmeli halka, $\lozenge$ operatörüne göre etkisiz elemanı varsa (Eşitlik 2.2); etkisiz elemanlı halka olarak adlandırılır.

$$a\lozenge b=b\lozenge a\ ,\forall a,b\in R \tag{2.1}$$

$$\exists e_{\lozenge}\in R\mid a*e_{\lozenge}=e_{\lozenge}*a=a\ ,\forall a\in R \tag{2.2}$$

$(R;*;\lozenge)$ etkisiz elemanlı değişmeli halka oluşturuyorsa ve $\lozenge$ operatörüne göre ters elemanı varsa (Eşitlik 2.3), $(R;*;\lozenge)$ bir cisim oluşturur. Sonlu cisimler Galois cismi (GF) olarak da adlandırılmaktadır.

$$\forall a \in R\ ,\ \exists b \neq e_* \mid (a \lozenge b) = (\mathrm{b} \lozenge a) = e_\lozenge \tag{2.3}$$

Buna göre cisim, vektör uzayının bir ifadesi olarak $(V; F; \oplus; \odot; +; \cdot)$ şeklinde tanımlanabilir.

Bu çalışmada ele alınan küme $B = \{0,1\}$, operatörler ise XOR ($\oplus$) ve AND ($\cdot$) olarak belirlenmiştir. Buna göre B kümesi vektör uzayında, $(V; B; \oplus; \cdot)$ şeklinde gösterilebilir. Ayrıca $(B; \oplus; \cdot)$ cismi olarak ifade edilebilmesi için belirtilen tüm şartları sağlamaktadır. $GF(q)$ olarak gösteriminde ise 2 elemanlı bir sistemde kodlama yöntemlerinin derecelerine göre $GF(2^m)$ olarak ele alınmaktadır.

Kodlama yöntemlerinde kullanılan işlemlerde toplama ifadesinde XOR, çarpma ifadesinde ise AND operatörlerine göre işlem yapılmıştır. AND işlem tablosu çizelge 2.1'de XOR işlem tablosu ise çizelge 2.2'de gösterilmiştir.

Çizelge 2.1 AND işlem çizelgesi

AND ($\cdot$) işlemi	**0**	**1**
0	0	0
1	0	1

Çizelge 2.2 XOR işlem çizelgesi

XOR ($\oplus$) işlemi	**0**	**1**
0	0	1
1	1	0

2.2.2 Kod sözcüğü

R, m elemanlı bir halka olmak üzere R^n kümesi aşağıda bulunan eşitlik (2.4) ile tanımlanabilir.

$$R^n = \{u = (u_{1,}u_{2,} \dots u_n) | u_i \in R\} \quad (2.4)$$

R^n kümesinin M elemanlı C alt modülü, uzunluğu n olan M elemanlı bir lineer kod olarak adlandırılmaktadır ve (n, M) ile gösterilmektedir. C kodunun herhangi bir elemanına kod sözcüğü adı verilmektedir (Chapman 1996).

2.2.3 Uzaklık fonksiyonu

u ve v, R^n kümesinin iki elemanı olmak üzere u ve v arasındaki uzaklık aşağıda verilen eşitlik (2.5) ile tanımlanmaktadır.

$$d(u, v) = |\{i | u_i \neq v_i\}| \quad (2.5)$$

Eşitlik (2.5) için dönüşüm eşitlik (2.6) ile tanımlanmaktadır.

$$d : R^n \times R^n \to N \cup \{0\} \quad (2.6)$$

$$(u, v) \to d(u, v) \quad (2.7)$$

Eşitlik (2.7)'de verilen dönüşüm aşağıdaki özellikleri sağlamaktadır (Chapman 1996).

i. $\forall\, u, v \in R^n$ için $d(u, v) \geq 0$, $d(u, v) = 0 \leftrightarrow u = v$

ii. $\forall\, u, v \in R^n$ için $d(u, v) = d(v, u)$

iii. $\forall\, u, v, t \in R^n$ için $d(u, v) \leq d(u, t) + d(t, v)$

2.2.4 Minimum uzaklık

Uzunluğu n, eleman sayısı M, minimum uzaklığı d olan bir C lineer kodu kısaca (n, M, d) ile gösterilmektedir (Chapman 1996). C kodunun minimum (Hamming) uzaklığı için matematiksel ifade eşitlik (2.8)'de verilmektedir.

$$d = d(C) = \min_{\substack{u,v \in c \\ u \neq v}} d(u, v) \tag{2.8}$$

2.2.5 Ağırlık fonksiyonu

x, $GF(q)$ vektör uzayının herhangi bir elemanı olmak üzere x'in sıfırdan farklı bileşenlerinin sayısı x elemanının ağırlığı olarak ifade edilmekte ve $w(x)$ ile gösterilmektedir.

Bir C kodunun sıfırdan farklı tüm kod sözcüklerin ağırlıklarının en küçüğüne C kodunun minimum ağırlığı denilmekte ve $w(C)$ ile gösterilmektedir. Ayrıca Chapman'nın (1996) önermesine göre, bir C lineer kodunun minimum uzaklığı ile minimum ağırlığı birbirine eşittir (Eşitlik (2.9)).

$$d(C) = w(C) \tag{2.9}$$

2.2.6 Hata tanımı

Bir C kodlamasında c gönderilen kelime ve r alınan kelime olmak üzere eşitlik (2.10) ile tanımlanmakta; bu durumdaki hata ise eşitlik (2.11) ile bulunmaktadır. Buradaki e ifadesi R^n kümesinin keyfi bir elemanıdır. Bu şekilde tanımlanan e ifadesine hata vektörü adı verilmektedir (Chapman 1996).

$$r = c \oplus e \tag{2.10}$$

$$e = r \oplus (-c) \tag{2.11}$$

2.2.7 Tespit edilebilen hatalar

Bir (m, n) kodlamasında eğer $a \in C$ için $a \oplus e \notin C$ ise e hata vektörü tespit edilebilirdir. Bu durumda eğer $a \in C$ için $a \oplus e \in C$ ise e hata vektörü tespit edilemezdir (Chapman 1996).

Bir (m, n) kodlamasında ağırlığı en fazla t olan hata vektörlerinin tespit edilebilmesi için gerek ve yeter şart herhangi iki kod söz arasındaki minimum uzaklığın en az $(t + 1)$ olmasıdır (Chapman 1996).

2.2.8 Düzeltilebilen hatalar

$\forall \mathrm{c} \in C$ için $D(\mathrm{c} \oplus \mathrm{e}) = c$ ise e hata vektörü düzeltilebilirdir (Chapman 1996).

Bir (m, n) kodlamasında ağırlığı en fazla t olan hatalar düzeltilebilir ise herhangi iki kod söz arasındaki minimum uzaklık en az $(2t + 1)$ olmaktadır (Chapman 1996).

Kod sözleri arasında minimum uzaklığın $(2t + 1)$ olduğu bir kodlamada ağırlığı en fazla t olan hatalar düzeltilebilirdir (Chapman 1996).

2.2.9 Lineer kodlar

n uzunluğunda ve q elemanlı sonlu bir cisim üzerinde tanımlanmış bir vektör uzayı aşağıdaki eşitlik (2.12) verildiği gibi tanımlanabilir. Bu durumda $V(n, q)$ vektör uzayının alt vektör uzayına lineer kod denilir. Eğer C kodunun boyutu k ise o zaman C

kodu bir $[n, k]$ kodu olarak tanımlanabilir. Aynı zamanda C kodunun minimum uzaklığı d ise C koduna $[n, k, d]$ - kodu denir (Roman 1992).

$$v(n, q) = GF(q) \tag{2.12}$$

Üreteç Matrisi: C, bir lineer kod olması durumunda C'nin satırları C kodunun baz vektörlerinden oluşan kxn boyutlu G matrisine, C kodunun üreteç matrisi denilmektedir. Eğer G matrisi C kodunun üreteç matrisi ise C kodunun kod sözleri, G matrisinin satırlarının lineer birleşimidir. Bu durumda eşitlik (2.13) elde edilmektedir.

$$C = \{xG | x \in V(k, q)\} \tag{2.13}$$

Kontrol Matrisi: C kodunun üreteç matrisi aşağıdaki eşitlik (2.14)'de tanımlanmıştır. Eşitlik (2.15)'te verilen şartı sağlayan matris eşitlik (2.16)'da verilmiştir. Bu matris C kodunun kontrol matrisi olarak adlandırılmaktadır (Roman 1992). Eşitlik (2.15) ve eşitlik (2.16) göz önüne alındığında eşitlik (2.17) ortaya çıkmaktadır. Bu eşitlik H matrisinin satırları ile G matrisinin satırlarının birbirine dik olduklarını göstermektedir.

$$G = (I_k | A) \tag{2.14}$$

$$GH^T = 0 \tag{2.15}$$

$$H = (-A^T | I_{n-k}) \tag{2.16}$$

$$GA^T = (I_k | A) \begin{pmatrix} -A \\ I_{n-k} \end{pmatrix} = -A + A = 0 \tag{2.17}$$

2.2.10 Lineer kodların kod çözmesi

C kodu, kontrol matrisi H olan bir lineer kod olarak ele alındığında; herhangi bir $x \in C$ için, Hx^T ifadesine x kod sözünün sendromu denilmekte ve bu sendrom s ile gösterilmektedir. Dolayısıyla, $x \in C$ olması için gerek ve yeter şart x kod sözünün sendromunun sıfır olmasıdır. Başka bir deyişle alınan x kod sözü orijinal kod söz (hatasız giden kod söz) ise $Hx^T = 0$ olmaktadır (Roman 1992).

C, IF_q^n (n boyutlu $GF(q)$) üzerinde bir $[n, k, d]$ kod ve H, C kodunun parite kontrol matrisi olmak üzere w ve w' sırasıyla eşitlik (2.18) ve eşitlik (2.20)'de tanımlanmıştır. $s = w \circ w'$ şeklinde IF_q^n ifadesinden IF_q^{n-k} ($(n-k)$ boyutlu $GF(q)$) ifadesine bir dönüşüm tanımlandığında her $y \in IF_q^n$ için $s(y)$ fonksiyonuna y'nin sendromu adı verilmektedir.

$$w: IF_q^n \to M_{1\times n-k}(IF_q) \tag{2.18}$$

$$\begin{aligned} y = (y_1, y_2, \dots, y_n) \longmapsto w(y) &= [y_1, y_2, \dots, y_n]_{1\times n}.{H^T}_{n\times(n-k)} \\ &= [b_1, b_2, \dots, b_{n-k}]_{1\times(n-k)} \end{aligned} \tag{2.19}$$

$$w': M_{1\times n-k}(IF_q^n) \to IF_q^{n-k} \tag{2.20}$$

$$[\,_{b_1}, b_2, \dots, b_{n-k}]_{1\times(n-k)} \longmapsto w'([b_1, b_2, \dots, b_{n-k}]) = (b_1, b_2, \dots, b_{n-k}) \tag{2.21}$$

Eşitlik (2.22)'in sağlanması durumunda H parite kontrol matrisi olmak üzere $s(y)$ eşitlik (2.24) ve (2.25) ile ifade edilir.

$$\begin{aligned} h = (h_{11}, h_{12}, \dots, h_{1n}) &= (h_{21}, h_{22}, \dots, h_{2n}), \dots, h_{n-k} \\ &= (h_{(n-k)1}, h_{(n-k)2}, \dots, h_{(n-k)n}) \end{aligned} \tag{2.22}$$

$$H = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1n} \\ h_{21} & h_{22} & \cdots & h_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ h_{(n-k)1} & h_{(n-k)2} & \cdots & h_{(n-k)n} \end{bmatrix}_{(n-k)\times n} \tag{2.23}$$

$$s(y) = (yh_1, yh_2, \dots, yh_{n-k}) \tag{2.24}$$

$$s(y) = 0 \Leftrightarrow y \in C \tag{2.25}$$

u ve v aynı denklik sınıfına ait iki vektör olmak üzere eşitlik (2.26) sağlanmaktadır. Denklik sınıfları ve sendromlar arasında bire bir karşılık bulunmaktadır.

$$u + C = v + C \tag{2.26}$$

Sendrom kod çözme işlemi aşağıdaki gibi tanımlanır (Trappe ve Washington 2002):

- Eğer sendrom değeri sıfır ise o zaman alınan kod söz, orijinal kod sözdür.
- Eğer sendrom değeri H kontrol matrisinin $i.$ sütunu ise alınan kod sözün $i.$ bileşeninde hata vardır.
- Eğer sendrom değeri ne sıfır ne de H kontrol matrisinin bir sütunu değilse alınan kod sözde en az iki hata vardır.

2.3 Silinti Kodlama

Maksimum Uzaklıkla Ayrılabilen (MDS) kodlar hata kontrol kodlarının önemli bir sınıfıdır. Bu kodlar birçok özelliğe sahiptir. En önemlileri arasında $GF(q)$ sonlu cismi üzerinde (n, k, d) MDS kodu için minimum Hamming mesafesi $d = n - k + 1$ olan Singleton sınırına sahip olmasıdır. Bu özelliğinden dolayı hata kontrol yeteneği açısından optimaldir. MDS kodları için hata tespit edebilen ve düzeltebilen hata

sembollerinin sayısı diğer kodlardan daha fazladır. Bu sebepten ötürü veri iletimi için depolama sistemlerinde ve iletişim sistemlerinde oldukça sık kullanılmaktadır.

2.3.1 Reed Solomon

RS kodları ikili olmayan Binary Coded Hexadecimal (BCH) kodlarının bir alt kümesidir. RS kodları çevrimsel kodlardır. RS kodları sabit k ve n değerleri için en yüksek minimum Hamming uzaklığa sahiptir.

İkili BCH kodları, ikili olmayan kodlara genelleştirilebilmektedir. p bir asal sayı, q, p'nin bir kuvveti, s ve t herhangi iki tamsayı olmak üzere; blok uzunluğu $n = q^s - 1$, parite kontrol sembolü sayısı $n - k = 2st$ olan bir ikili olmayan BCH kodu vardır. Bu kod, $2st$ parite kontrol sembolü yardımıyla, t veya daha az sayıda hatalı sembolü düzeltme yeteneğine sahiptir. İkili BCH kodlarının üreteç polinomunun katsayıları $GF(2)$'nin elemanı iken, ikili olmayan BCH kodunun üreteç polinomunun katsayıları $GF(q^s)$ 'in elemanıdır. Üreteç polinomunun kökleri ise $\alpha, \alpha^2, \dots, \alpha^{2t}$ olarak tanımlanmaktadır. $\phi_i(x), \alpha^i$'nin minimal polinomu olmak üzere üreteç polinomu aşağıda verilen eşitlikteki gibi ele alınmaktadır.

$$g(x) = OKEK\{\phi_1(x), \phi_2(x), \dots, \phi_{2t}(x)\} \tag{2.27}$$

Her bir minimal polinomun derecesi s veya daha azdır. Bu nedenle $g(x)$'in derecesi $2st$ olabilir. Dolayısıyla $g(x)$ tarafından üretilen kod $2st$ parite kontrol sembolü içerir.
RS kodları, ikili olmayan BCH kodları içinde $s = l$ alınarak elde edilmektedir. Eşitlik t hata düzelten RS kodunun parametreleri olmak üzere; eşitlik (2.28)'de blok uzunluğu; eşitlik (2.29)'da parite kontrol bit sayısı; eşitlik (2.30)'da minimum uzaklık ifadeleri verilmiştir.

$$n = q - 1 \tag{2.28}$$

$$n - k = 2t \tag{2.29}$$

$$d_{min} = 2t + 1 \tag{2.30}$$

RS kodlarının önemli bir özelliği minimum Hamming uzaklığının her zaman parite kontrol sembolü sayısından bir fazla olmasıdır. $\alpha^i, GF(q = p^m)$ ifadesinin sıfırdan farklı bir ilkel elemanı ve $\phi(x)$ polinomunun bir kökü olmak üzere $2st$ hata düzelten doğrusal, dairesel ve blok kod olan RS kodunun üreteç polinomu aşağıdaki hali almaktadır. Eşitlikte g_i ifadeleri $GF(q = p^m)$ 'nin elemanıdır ve, $\alpha, \alpha^2, \dots, \alpha^{2t}$, $g(x)$'in kökleridir. $g(x)$'in derecesi en fazla $2t$'dir. $g(x)$ tarafından üretilen kod $(n, n - 2t)$ kodudur (Moreira ve Farrell 2006).

$$\begin{aligned} g(x) &= (x + \alpha)(x + \alpha^2) \dots (x + \alpha^{2t}) \\ &= g_0 + g_1 x + g_2 x^2 + \cdots + g_{2t-1} x^{2t-1} + x^{2t} \end{aligned} \tag{2.31}$$

$GF(2^m)$ üzerinde (n, k) parametreleriyle tanımlanan bir RS kod sisteminde, m-bit uzunluğundaki k sembol veri ve n sembol kod kelimesi, eşitlik (2.32)'te verilen polinomla ifade edilir.

$$C(z) = c_0 + c_1 z + \cdots + c_{n-1} z^{n-1} \tag{2.32}$$

Tüm kod kelimeleri, eşitlik (2.33)'da tanımlanan üreteç polinomunun tam katlarıdır. Verinin, blok olarak yer aldığı sistematik bir kod kelimesi ise eşitlik (2.34) ile ifade edilmektedir. Burada $Q(z)$ herhangi bir polinom olup $P(z)$ ise verinin sonuna eklenen parite sembollerinin polinom gösterimidir.

$$G(z) = \prod_{i=0}^{n-k-1} z - \alpha^{m_0 + i} \tag{2.33}$$

$$C(z) = Q(z) \times G(z) + P(z) \tag{2.34}$$

RS kodlama ile $2t = n - k$ olacak şekilde t adet hatalı sembol düzeltilebilmektedir. Hata oluşan kod kelimesinin tanımı ise eşitlik (2.35)'de verilen şekilde bir hata polinomu ile toplanmış olarak modellenmektedir.

$$R(z) = C(z) + E(z) \tag{2.35}$$

Reed Solomon Kodlama

$k = n - 2t$ olmak üzere, kodlanacak mesaj polinomu aşağıda verilen eşitlikteki gibi ele alınmaktadır.

$$M(z) = m_0 + m_1 z + m_2 z^2 + \cdots + m_{k-1} z^{k-1} \tag{2.36}$$

RS kod sözcükleri k sembollük mesaj polinomu $M(z)$'in $G(z)$ üreteç polinomu ile çarpılması sonucu elde edilebilir. Fakat elde edilen kod sözcüğü $C(z)$ sistematik halde olmayacaktır. Sadece $GF(q)$ içinde $G(z)$'e tam olarak bölünen sözcüklerin RS kod sözcükleri olabileceği dikkate alınırsa RS kodlama (encoding), bir bölme işlemi ile gerçekleştirilebilmektedir. Eşitlik (2.36)'da verilen $M(z)$ polinomu, z^{n-k} ile çarpılıp $G(z)$'e bölündüğü durumda bölümden kalan $P(z)$, $z^{n-k}M(z)$ ifadesine eklenirse elde edilecek polinom $G(z)$ polinomuna tam olarak bölünebilir. Bu durumda $P(z)$ polinomu üretilecek olan kod sözcüğünün (C) parite kontrol polinomu olmaktadır. $P(z)$'e göre RS kodlamasında kod sözcüğü eşitlik (2.40)'daki gibi belirtilmiştir.

$$M(z) \neq 0 \; mod \; G(z) \tag{2.37}$$

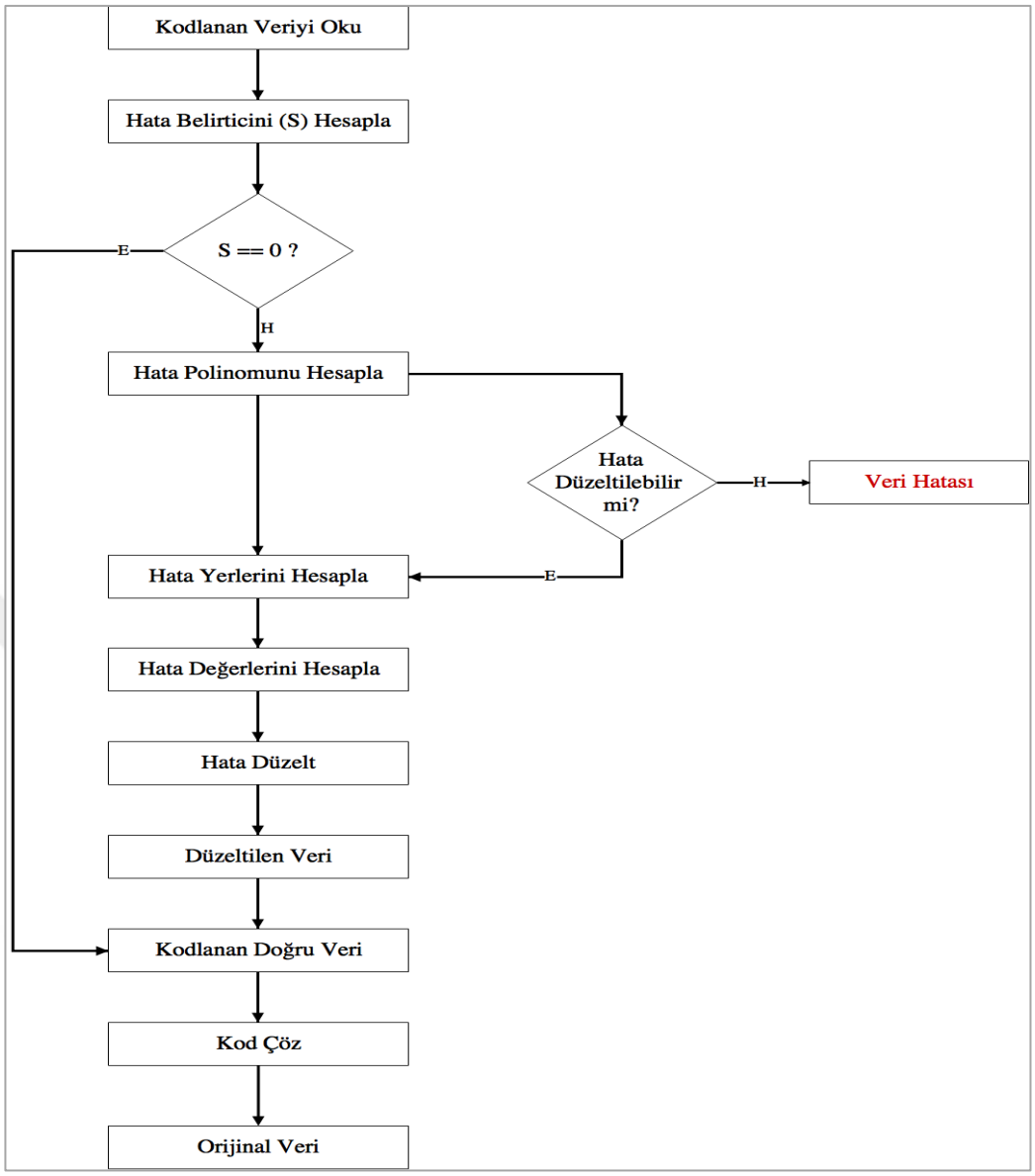
$$z^{n-k}M(z) = Q(z)G(z) + P(z) \tag{2.38}$$

$$z^{n-k}M(z) \; = P(z) \; mod \; G(z) \tag{2.39}$$

$$C = (p \mid m) \tag{2.40}$$

Reed Solomon Kod Çözme

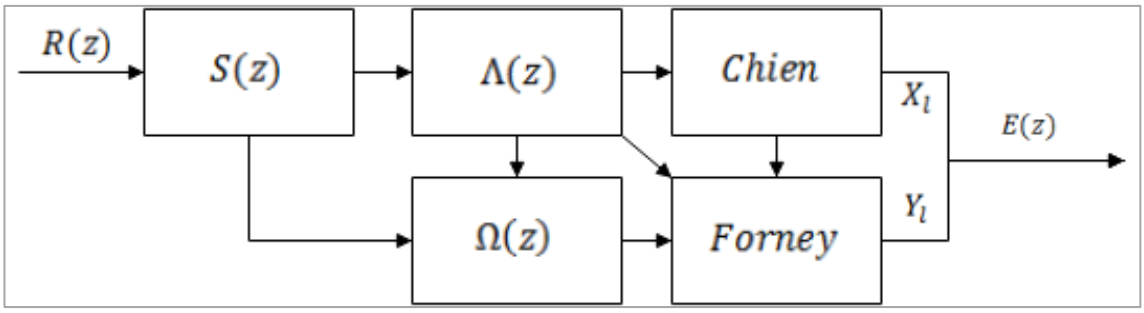
Doğrusal blok kodlarında ve ikili BCH kodlarında kod çözme işlemine hata belirtecinin hesabı ile başlanmaktadır. Hata belirtecinin hesabının ardından hata yerleri bulunmakta ve hatalı bitler düzeltilmektedir. İkili olmayan BCH kodlarında dolayısıyla RS kodlarında da kod çözme işleminin ilk adımı hata belirteci hesabıdır. Fakat hata yerleri bulunduktan sonra ikili kodlardan farklı olarak hata değerlerinin de bulunması gerekmektedir. Bunun nedeni RS kodlarının bitler üzerinde değil, bir kaç bitten oluşan sözcükler üzerinde çalışmasıdır. RS kodlarının çözümünde izlenecek adımlar için akış şeması şekil 2.3'te verilmiştir.



Şekil 2.3 RS kod çözme akış şeması

Sendrom tabanlı RS kod çözme işlemlerinde aşağıdaki adımlar takip edilir.

- Alınan kod kelimesinden hata elemanlarının hesaplanması
- Hata yeri polinomu ($S(z)$) 'in bulunması
- Chien (1964) algoritmasıyla ($S(z)$) 'in kökleri olan hata yerlerinin ($\Lambda(z)$) bulunması,
- Forney (1965) algoritmasıyla hata değerlerinin ($\Omega(z)$) hesaplanması
- Hatalı sözcüklerin düzeltilmesi ve alınan kod kelimesinin çözülmesi



Şekil 2.4 RS kod çözme blok diyagramı (Şen vd. 2016)

Chien bir arama algoritmasıdır. Bütün elemanlar tek tek iterasyonlarla denenerek $\Lambda(z)$ polinomunun kökleri bulunur. X_j olarak gösterilen bu köklerin $GF(2^m)$'deki tersleri hata yerlerini vermektedir. Hata yerleri bulunan kod sözcüğünde, Forney algoritması yardımıyla hataların değeri (Y_j) hesaplanmaktadır.

$$S(z) = s_0 + s_1 z + s_2 z^2 + \cdots + m_{n-k-1} z^{n-k-1} \tag{2.41}$$

$$X_i = \alpha^{ji} \; ; 0 \leq j_i \leq n-1 \tag{2.42}$$

$$Y_i = E_{j_i} \tag{2.43}$$

$$s_l = \sum_{i=1}^{s} Y_i X_i^l \tag{2.44}$$

Eşitlik (2.35)'de gibi modellenen hatalı alınan kod sözcüğünün sendrom polinomu eşitlik (2.41)'deki gibi belirtilmektedir. $S(z)$'nin katsayıları, $G(z)$'nin köklerinin, $R(z)$ polinomundaki değerlerin hesaplanmasıyla eşitlik (2.45) elde edilir. Buna göre hata yer polinomu ($\Lambda(z)$) hesaplanabilir (Eşitlik (2.46)).

$$s_i = R\left(\alpha^{m_0+i}\right) = C\left(\alpha^{m_0+i}\right) + E\left(\alpha^{m_0+i}\right) = R\left(\alpha^{m_0+i}\right) \tag{2.45}$$

$$\Lambda(z) = \prod_{i=1}^{s}(1 - X_i z) = 1 + \Lambda_1 z + \Lambda_2 z^2 + \cdots + \Lambda_e z^e \tag{2.46}$$

Hata değerinin bulunması için gerekli olan hata değeri polinomu ($\Omega(z)$) eşitlik (2.47)'de belirtilmiştir. Bu da göre hata değeri Y_i Forney algoritması yardımıyla bulunabilir.

$$\Lambda(z)S(z) = \Omega(z)\ mod(z^{n-k}) \tag{2.47}$$

$$Y_i = E_i = -\frac{z^{m_0}\Omega(z)}{z\Lambda'(z)} \tag{2.48}$$

2.3.2 Hamming

Hamming (1950) tarafından doğrusal hata düzelten bir kod olarak tanımlanmıştır. Hamming kodu tek bitlik hataları saptayıp düzeltebilen bir koddur. Bir veya iki bitlik hataları saptamak üzere de kullanılabilmektedir. Buna karşın, basit eşlik kodu iki bitin transpoze olduğu yerde hata bulamaz; bulsa da düzeltememektedir.

Her bir tam sayı $r \geq 2$ için, blok uzunluğu $n = 2r - 1$ ve mesaj uzunluğu $k = 2r - 1 - r$ olan bir kod vardır. Dolayısıyla Hamming kodlarının oranı $R = \frac{k}{n} = \frac{r-1}{2r-1}$, bu değer herhangi bir kod sözcüğünden diğer herhangi bir kod sözcüğüne gitmek için gereken minimum bit değişiklik sayısını göstermektedir ve blok uzunluğu $2r - 1$'dir. Parite kontrol matrisi bir Hamming kodunun, sıfır olmayan tüm uzunluk r sütunlarının listelenmesiyle oluşturulmaktadır.

Buna göre bir lineer kod olan Hamming kodlamasında üreteç matrisi ile parite kontrol matrislerinin matematiksel ifadeleri eşitlik (2.13) – eşitlik (2.17) gibi tanımlanmaktadır.

Hamming kodlama

Kodlama işleminde, $[1 \times k]$ boyutlu, kodlanacak mesaj sözcüğünün (m), tanımlanan üreteç matrisi ile çarpılması ile kod sözcüğü (c) oluşmaktadır.

$$c = mG \tag{2.49}$$

Üreteç matrisi $[k \times k]$ boyutlu birim matris (I_k) ile parite matrisinden (P) elde edilmektedir.

$$G = [I_k|P] \tag{2.50}$$

Buna göre P matrisinin boyutu $[k \times (n-k)]$ olmaktadır.

Kodlama sırasında, orijinal mesaj bitleri ile parite matrisindeki (P) değerler arasında XOR işlemi yapılarak kod sözcüğü elde edilir.

$$m = [m_0\ m_1\ m_2\ m_3] \tag{2.51}$$

$$c = [m_0\ m_1\ m_2\ m_3\ P_0\ P_1\ P_2] \tag{2.52}$$

Buna göre mesaj sözcüklerine ardına eklenen parite bitleri sayesinde hata belirleme ve hata düzeltme işlemleri gerçekleştirilebilir.

$$p_0 = m_0 \oplus m_1 \oplus m_2 \tag{2.53}$$

$$p_1 = m_1 \oplus m_2 \oplus m_3 \tag{2.54}$$

$$p_2 = m_0 \oplus m_1 \oplus m_3 \tag{2.55}$$

Hamming kod çözme

Üreteç matrisine göre elde edilebilecek tüm kod sözcüklerin listesi belli olmaktadır. Örnek bir gösterim çizelge 2.3'te verilmiştir.

Çizelge 2.3 Örnek kod sözcük tablosu

Sl No	m0	m1	m2	m3	p0	p1	p2
0	0	0	0	0	0	0	0
1	0	0	0	1	0	1	1
2	0	0	1	0	1	1	0
3	0	0	1	1	1	0	1
4	0	1	0	0	1	1	1
5	0	1	0	1	1	0	0
6	0	1	1	0	0	0	1
7	0	1	1	1	0	1	0
8	1	0	0	0	1	0	1
9	1	0	0	1	1	1	0
10	1	0	1	0	0	1	1
11	1	0	1	1	0	0	0
12	1	1	0	0	0	1	0
13	1	1	0	1	0	0	1
14	1	1	1	0	1	0	0
15	1	1	1	1	1	1	1

Buna göre kod çözmek için verinin bozulup, bozulmadığına bakılabilmektedir. Bununla birlikte hata sayısına bağlı olarak düzeltilebilen hatalar ile tespit edilen hata sayıları bulunabilmektedir.

Bunun yanı sıra kod çözme için hesaplanabilen parite kontrol matrisi (H) kullanılabilir (Eşitlik (2.16))

$[1 \times n]$ boyutlu r alınan kod kelimesini, $[1 \times k]$ boyutlu m orijinal mesaj bitini, $[k \times n]$ boyutlu G üreteç matrisini, $[1 \times n]$ boyutlu e ise belirlenebilen lokasyonda olan hatayı temsil etmek üzere aşağıdaki eşitlikler tanımlanabilmektedir.

$$r = mG + e \tag{2.56}$$

$$rH^T = (mG + e)H^T \tag{2.57}$$

$$rH^T = mGH^T + eH^T \tag{2.58}$$

$$mGH^T = [0\ 0\ 0] \tag{2.59}$$

$$rH^T = eH^T \tag{2.60}$$

Eşitlik (2.60)'de verilen eH^T ifadesi $[1 \times (n-k)]$ boyutludur ve hata belirlemede kullanılmak üzere sendrom matrisi olarak tanımlanmaktadır.

Elde edilen sendroma göre aşağıda sıralanan 3 sonuç çıkartılabilmektedir:

- n tane olası hata olabilir
- Eğer sendrom sıfır ise hata yok demektir
- Sendromun sıfır olmayan değerlerine göre hangi bit'in hatalı olduğu ve düzeltilebileceği anlaşılır.

2.3.3 Parite dizi kodlama

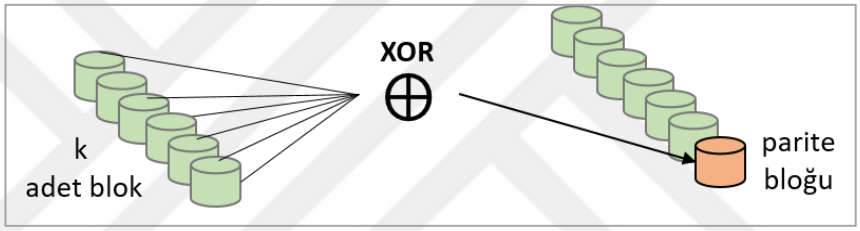
Parite dizisi kodlamasında, her bir düğümde bulunan veriler bir şerit olarak adlandırılan eşit uzunluktaki bloklara gruplanır. Parite bloklarını oluşturmak için bu düğümdeki verilere XOR işlemi uygulanır.

$k = n - 1$ olmak üzere düğüm verileri, eşitlik (2.61)'deki gibi tanımlanmaktadır. Buna göre parite dizi kodlaması sonrasında elde edilecek parite değeri (P) olarak gösterilmektedir (Eşitlik (2.62)).

$$D = D_1, D_2, \dots, D_k \tag{2.61}$$

$$P = \bigoplus D_i = D_1 \oplus D_2 \oplus \dots \oplus D_k \tag{2.62}$$

Parite değeri ile birlikte parite dizi kodlaması şekil 2.5'teki gibidir.

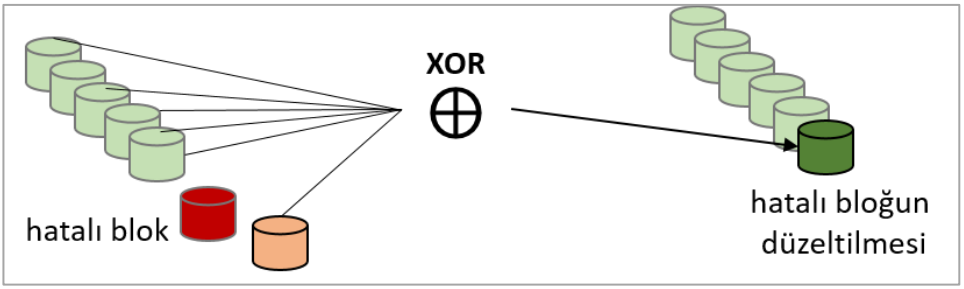


Şekil 2.5 Parite dizi kodlama

Hata bloğu D_e olarak tanımlanırsa, parite dizi kodlamasında, hata düzeltme ve kod çözme işlemleri eşitlik (2.63) kullanılarak hesaplanabilir.

$$D_e = \bigoplus P \oplus D_i = P \oplus D_1 \oplus D_2 \oplus \dots \oplus D_{k-1} \tag{2.63}$$

Şekil 2.5-2.6'da görüldüğü gibi parite dizi kodlamalarında kodlama ve kod çözme işlemleri XOR tabanlı işlemlerdir. Bu anlamda kod çözme işlemlerinde tersine bir işlem gerçekleştirilmediği için Parite dizi kodlamaları pratik bir kodlama yöntemidir.



Şekil 2.6 Parite dizi hata düzeltme ve kod çözme

Düğüm sayılarının sabit olduğu durumlarda hata düzeltme sayılarına göre iki farklı parite bloğu ($P\ ve\ Q$) elde edilebilir. Birinci parite (P) ile XOR tabanlı RAID5, birinci pariteye ek olarak ikinci parite (Q) ile XOR tabanlı RAID6 veri depolama sistemleri oluşturulabilir (Plank 2008).

XOR tabanlı RAID5, tek bir düğüm hata düzeltebilen kodlama yöntemidir. Buna göre kodlama ve kod çözme için sadece P değerinin elde edilmesi yeterli olacaktır. Fakat XOR tabanlı RAID6 yapılarında iki düğüm hata düzeltilebilmektedir. Eğer tek bir düğüm hata varsa P'nin elde edilmesi yeterli olurken, iki düğüm hatada hem P hem de Q parite değerleri kullanılmaktadır. Buna bağlı olarak Q'nun oluşturulması için D'nin her bir elemanının üreteç matrisleriyle kodlanması gerekmektedir.

$$D = D_1, D_2, \dots, D_{n-1} \in GF(q) \tag{2.64}$$

$$Q = \bigoplus g^i D_i = g^1 D_1 \oplus g^2 D_2 \oplus \dots \oplus g^{n-1} D_{n-1} \tag{2.65}$$

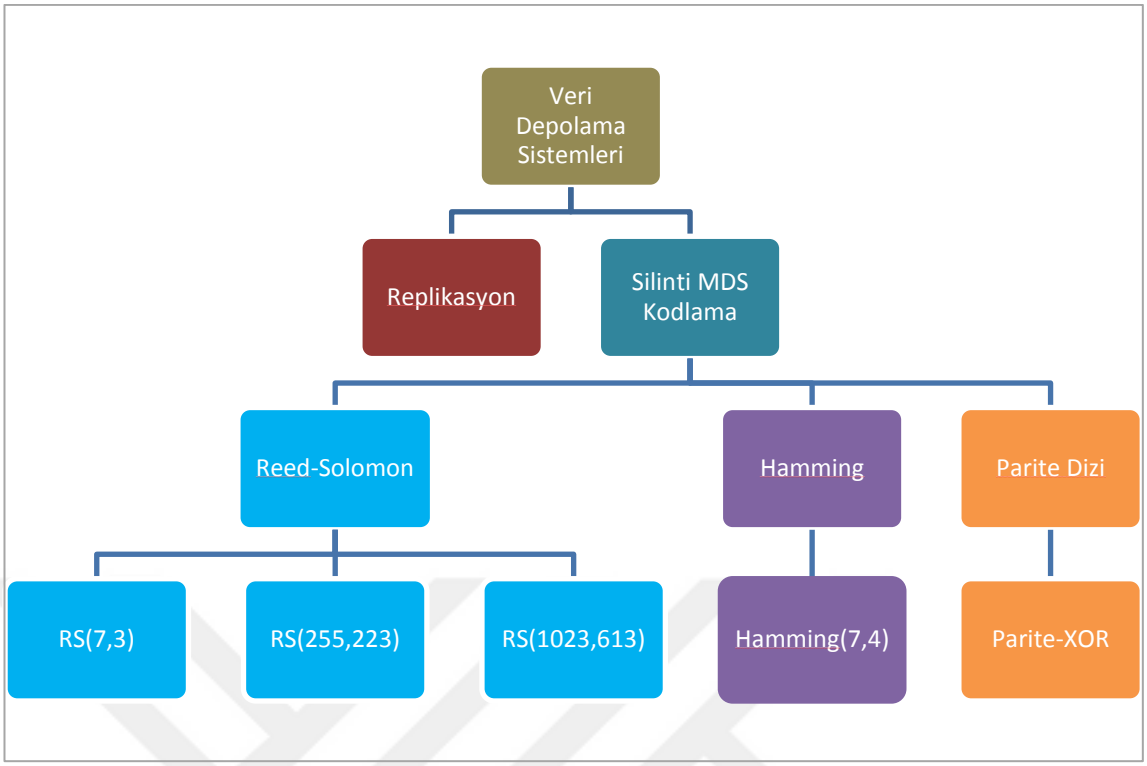
Eşitlik (2.65)'te belirtilen D ifadesi, $GF(q)$'nun bir elemanı olacak şekilde bir MDR (Maximum Distance Rank) kodlaması ile kullanılabilir (Wang vd. 2014). RS ve diğer kodlama yöntemlerinin de Parite dizi kodlamalarında sıklıkla kullanıldığı çalışmalar mevcuttur (Blaum vd. 1995, Wang vd. 2010, Wang vd. 2014).

3. MATERYAL VE YÖNTEM

Silinti kodlama ile uçtan uca oluşturulan bir DVDS modelinde, ele alınacak kodlama yöntemlerinin farklarını ortaya koyabilmek için RS, XOR ve Hamming kodlama yöntemleri seçilmiştir. Bu kodlama yöntemleri, DVDS ile ilgili literatürde de en çok kullanılan kodlama yöntemleridir. RS kodlamaları, esneklikleri ve hata düzeltme oranlarından (HDO) dolayı, XOR kodlamaları ise verinin yeniden yapılandırılmasının kolay ve etkili olmasından dolayı kullanılmaktadırlar. Bunun yanı sıra MDS kodlamalarının çıkış noktası olan ve veri iletiminin başarılı bir uygulaması olan Hamming kodlama yöntemi de, DVDS yapılarındaki kullanımının analizi için seçilmiştir.

3.1 Tanımlar ve Sistem Modeli

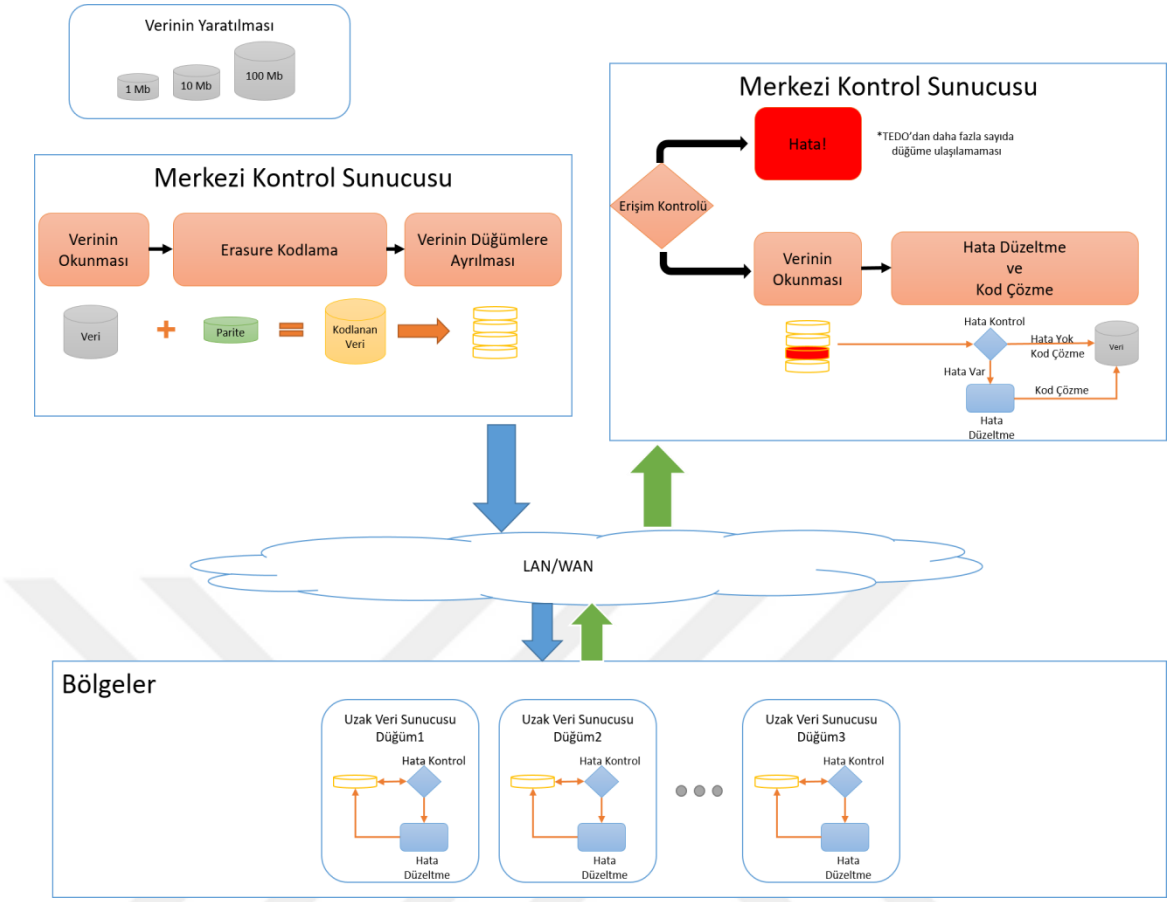
Bu bölümde, yüksek lisans tezi kapsamında, seçilen kodlama yöntemlerinin yanısıra, kodlama yöntemlerinin kendi alt kırılımları da incelenmiştir. Bu kırılımlar şekil 3.1'de görülmektedir. Ayrıca kodlama yöntemleri ile kıyaslamak üzere zaman zaman replikasyon yöntemine de değinilmiştir.



Şekil 3.1 Ele alınan kodlama yöntemleri ve kırılımları

Yöntemlerin incelenebilmesi için şekil 3.2'teki sistem modeli oluşturulmuştur. Oluşturulan bu sistem modelinde, kodlama yöntemlerinin kıyaslanabilmeleri için aşağıdaki adımların uygulanması gerekmektedir.

- Kodlama
- Kodlanan verinin dağıtılması
- Düğüm üzerindeki veri parçalarının doğrulanması
- Erişim kontrolü
- Hata kontrolü
- Hatalı verinin düzeltilmesi ve kod çözme



Şekil 3.2 Oluşturulan sistem modeli[1]

Kodlama yöntemlerinin farklı boyuttaki davranışlarını inceleyebilmek için 1 MB, 10 MB ve 100 MB olmak üzere üç farklı boyutta verinin oluşturulması sağlanmıştır. Bu veriler MATLAB ortamında yaratılmış olup, fiziksel bir donanımın kısıtlarına takılmamak için kodlama yöntemlerinin metriklerine uygun olacak şekilde sözde (pseudo) olarak oluşturulmuştur.

Merkezi Kontrol Sunucusu (MKS); yaratılan verinin okunması, kodlanması, düğümlere ayrılması, Uzak Veri Sunucularından (UVS) gelen verinin okunup hata kontrolünün yapılması, kodlama yöntemine göre hatalı verinin düzeltilerek orijinal verinin yeniden elde edilmesini sağlayan merkezi bir sunucudur.

[1] TEDO: Tolere edilebilir Düğüm Sayısı

MKS üzerinde kodlama türüne göre orijinal verinin üzerine parite verileri ilave edilerek, orijinal veriden daha büyük boyutta kodlanmış yeni bir veri elde edilmektedir. Her bir kodlama yönteminde minimum ve maksimum düğüm sayıları belirlenerek, bu sayılara göre kodlanmış verinin veri parçalarına ayrılması sağlanmaktadır. Her bir veri parçası kendi içinde anlamsız içeriklere sahip olmakla birlikte, tek başına anlamlandırılması veya orijinal verinin elde edilmesi için yeterli değildir.

Kodlama yöntemlerinin belirtilen metriklere göre kıyaslamalarında ağ bağlantılarının şekli ve türü göz ardı edilmiştir. MKS'nin düğümlerine ayırdığı veri parçaları kodlama yönteminde ele alınan düğüm sayısına bağlı olarak elle farklı UVS'lerine dağıtılmış ve UVS'lerinden verinin okunması sağlanmıştır.

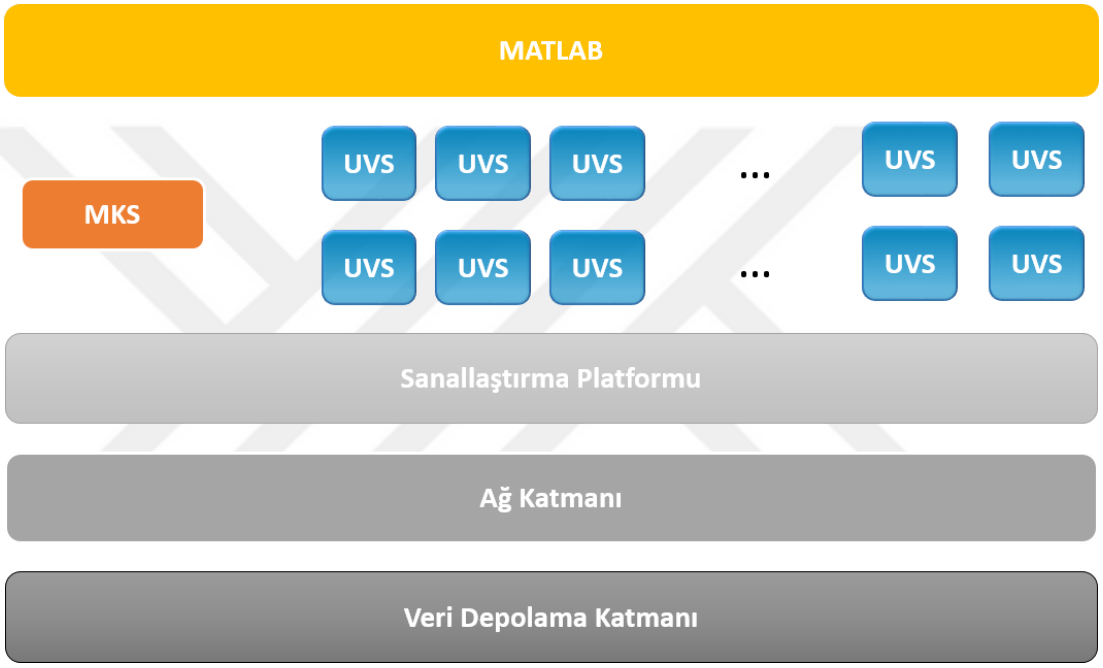
Sistem modeli, DVDS'nin bir uyarlaması olup, herhangi bir düğüm ile ağ iletişiminin kopmasından veya verinin iletilmesi sırasında kodlanmış verinin içeriğinin bozulmasından kaynaklanan durumlarda, orijinal verinin yeniden elde edilebilmesini sağlamayı amaçlamaktadır. Bu doğrultuda MKS üzerinde UVS'lerinden iletilen verinin okunması sağlanarak, kodlama yöntemine göre minimum düğüm sayısından (MIDS) sağlıklı veri alınıp alınamadığı kontrol edilmektedir. Eğer minimum gereksinimdeki düğüm sayısına erişilebiliniyorsa, orijinal verinin elde edilebilmesi için öncelikle hata kontrolü, varsa hatanın düzeltilmesi, sonra da kod çözme ile birlikte orijinal verinin elde edilmesi sağlanmaktadır.

Ayrıca orijinal veri ile kod çözülerek elde edilen veri arasında, fark analizi yapılmaktadır. Bu anlamda fark analizi, verinin birebir elde edildiğini veya elde edilen verinin hatalı bir veri olduğunu söyleyebilmektedir.

3.2 Sistem Modeli Uygulama Ortamı

Sistem modelindeki uygulama adımları şekil 3.3'te gösterildiği gibi bir ortamda uygulanmıştır. MKS, 64 bit Windows 10 tabanlı işletim sistemi üstündeki MATLAB v8.4 ortamında; 3 çekirdekli 2,8 GHz Intel Core i7 işlemcili, 8 GB RAM ve flash disk

kaynakları kullanılarak oluşturulmuştur. UVS ise aynı donanım üstünde VMware'in sanallaştırma platformu kullanılarak yaratılan 1vCPU işlemcili, 2 GB RAM ve flash disk kaynakları kullanılarak elde edilmiştir. UVS sayısı kodlama yöntemlerinde belirlenen düğüm sayılarına göre arttırılabilmektedir. Her bir sunucunun, sanallaştırma platformu sayesinde farklı ağ yapılarında ayarlamaları yapılmıştır. Aynı şekilde sunucuların fiziksel depolama alanları olan diskleri, ağ üzerinden okuma/yazma işlemlerini yapmaktadır. Bu sayede DVDS'nin bir modeli oluşturulabilmiştir.



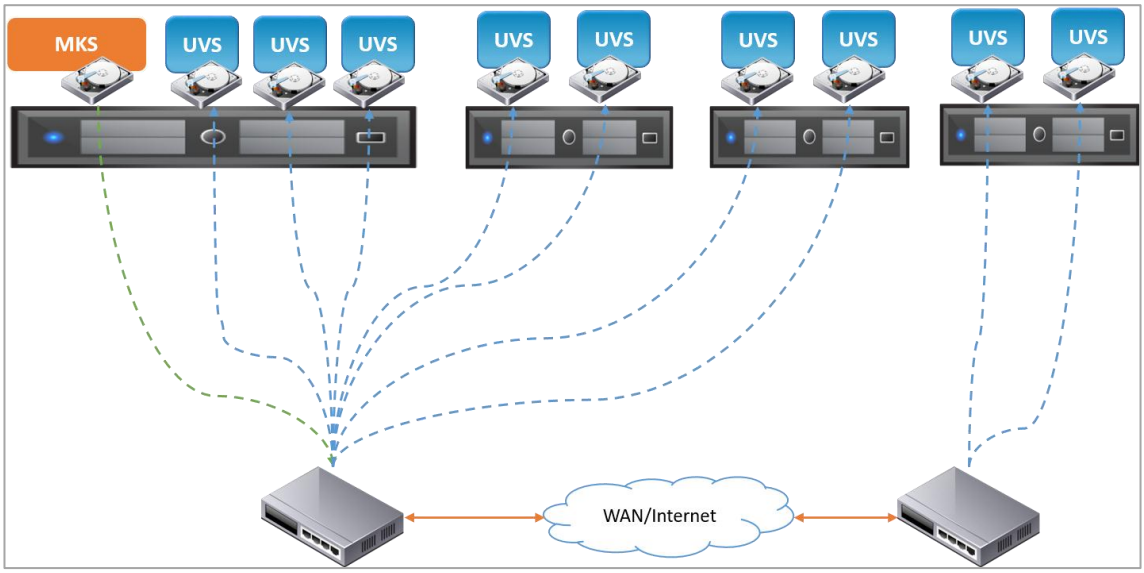
Şekil 3.3 Sistem modeli uygulama ortamı

3.2.1 Çalışma prensibi

Oluşturulan sistem modeline göre DVDS esnek bir yapıda olabilmelidir. Bu doğrultuda öncelikle tek bir fiziksel donanım üstünde MKS oluşturulmuştur. Aynı fiziksel donanım üstünde ihtiyaca göre birden fazla UVS oluşturulabilmektedir. Tek bir donanım üstünde çalıştırılan MKS ve UVS ler en fazla fiziksel donanımın işlemci ve bellek kapasitesi kadar eklenebilmektedir. İhtiyaç halinde, farklı bir donanım üstünde aynı uygulama katmanı baz alınarak yeni UVS'ler eklenebilmektedir.

Veri depolama alanı aynı şekilde fiziksel donanım üstündeki yerel diskler kullanılarak elde edilmiştir. Her bir sunucunun kendine ait diskleri çalıştığı fiziksel donanım üstünde bulunmaktadır. İşletim sistemi ve Matlab uygulamasına ait veriler, Sistem Disk'inde bulunurken asıl işlenecek olan veri, Veri Disk'inde bulunmaktadır.

Verinin ilk yaratılması MKS üzerinde olduğu için MKS, kodladığı veriyi belirlenen düğüm sayısına göre ağ üzerinden UVS'lere iletmektedir. Dağıtılan verinin yeniden MKS'ye depolanmasına gerek olmamaktadır. MKS, ağ üzerinden erişebildiği UVS'ler üzerindeki veriyi okuyabilmektedir dolayısı ile de orijinal verinin kullanılabilmesini sağlamaktadır. Bu anlamda veriyi işleyecek olan herhangi bir uygulamanın MKS'ye erişmesi yeterli olacaktır. Ancak UVS'ye erişilememesi veya herhangi bir fiziksel bir hata anında ilgili UVS'nin verisini MKS yeniden kendi üstünde yaratmaktadır. Orijinal verinin doğruluğu kontrol edildikten sonra MKS yeniden elde ettiği veriyi sağlıklı bir UVS üzerine iletebilmektedir. Bu şekilde hata anında verinin işlenmesindeki kesinti en fazla kod çözme süresi kadar olacaktır. Bu da fiziksel ve ağ üzerindeki bir hatanın çözülmesi süreciyle kıyaslanamayacak kadar azdır.



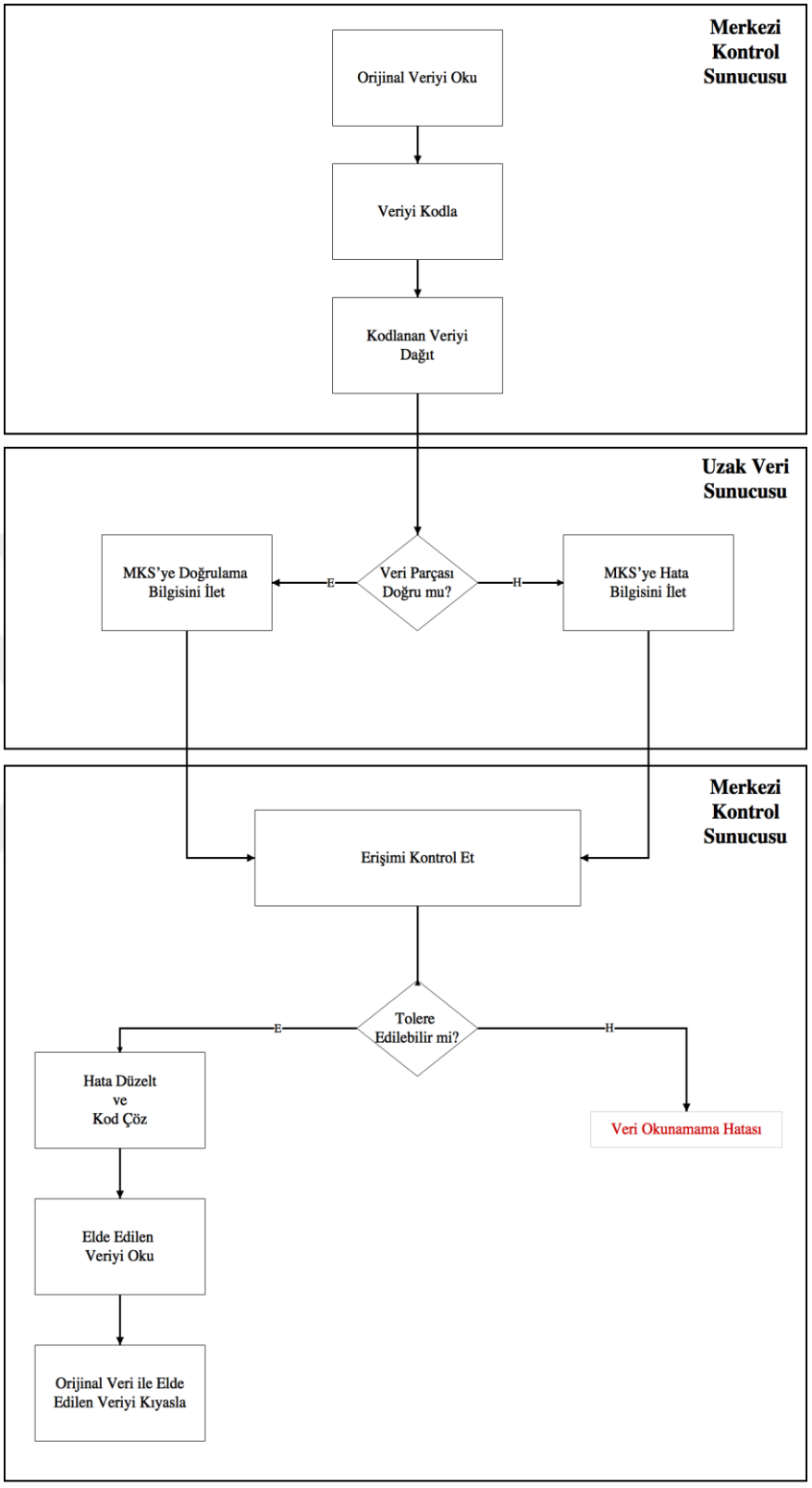
Şekil 3.4 Çalışma prensibi

UVS'ler aynı donanımlar üstünde çalışmasına rağmen her biri aslında aynı veya farklı bir ağda bulunan birer düğüm noktasını belirtmektedir. Düğüm noktaları, Yerel Alan Ağı (Local Area Network) üzerindeki aynı ağda çalışabildikleri gibi farklı yönlendirilebilir bir ağa da dahil edilebilir. Aynı zamanda Geniş Alan Ağı (Wide Area Network) veya internet üzerinden erişilebilen UVS ler de oluşturulan bu DVDS'ne dahil edilebilir.

Fiziki bir donanım üstünde farklı rolleri ve farklı verileri depolayacak bir sistem modeli oluşturabilmek için sanallaştıma teknolojisinden yararlanılmıştır. Sunucu sanallaştırması sayesinde her bir sunucunun (MKS ve UVS'ler) işlemci, bellek ve disk alanları birbirlerinden izole edilerek çalıştırılması sağlanmıştır. Ağ sanallaştırması sayesinde ise farklı ağ topolojilerinde MKS ve UVS'lerin iletişimleri sağlanmıştır. Bu sayede farklı ağlar üzerinde verinin işlenebileceği bir DVDS modeli oluşturulmuştur.

3.3 Uygulama Adımları

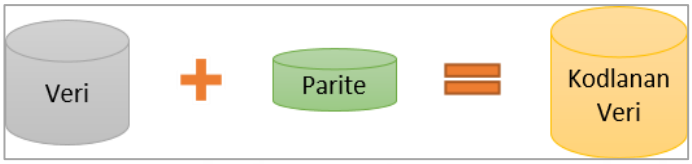
Tanımlanan sistem modelinde MKS'nin ve UVS'nin görevleri farklı olmakla birlikte ağ üzerinden sürekli birbirleriyle iletişim halindedirler. Bu iletişim kalp atış (heartbeat) mesajlarıyla sağlanmaktadır. Görevlerin sunuculardа doğru bir şekilde uygulanması için her bir kodlama yöntemine göre uygulama adımları belirlenmiştir. Bu uygulama adımları, şekil 3.5'teki akış şemasında gösterildiği gibi DVDS'ye uygun olacak şekilde verinin okunmasından tekrar elde edilmesine kadarki adımları göstermektedir.



Şekil 3.5 Uygulama adımları akış şeması

3.3.1 Silinti kodlama

Geleneksel bir (n, k) silinti kodlamasında orijinal veri önce k sayıda eşit parçaya bölünmektedir. Sonra her bir parçanın, kodlama yöntemlerinde belirtilen kodlama işlemleri sonrasında, parite verileri oluşturulmaktadır. Orijinal veriye eklenen parite verileriyle birlikte elde edilen yeni veriye kod verisi denilmektedir.

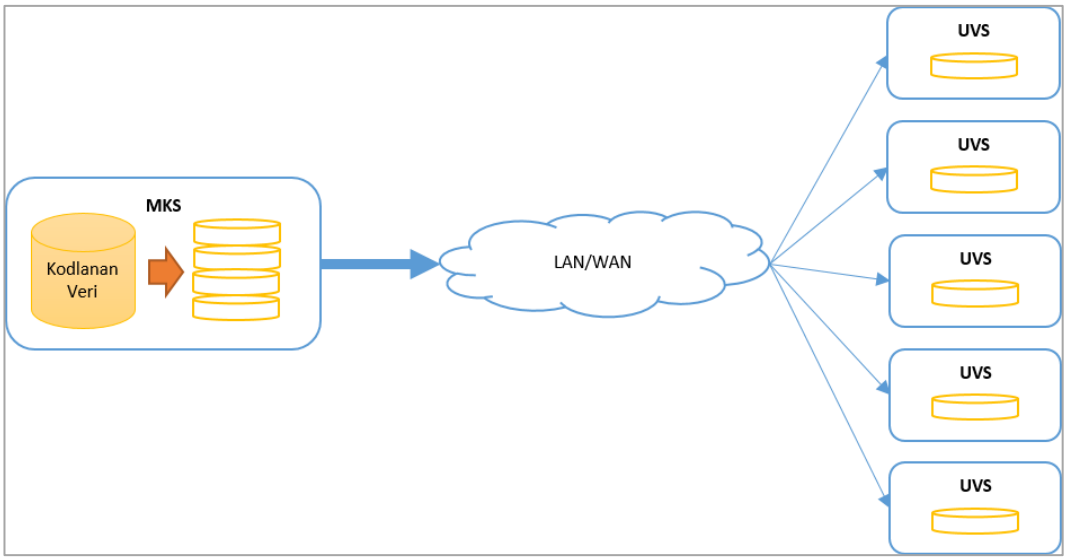


Şekil 3.6 Silinti kodlama

Kod verisinin tümü MKS üzerindeki fiziksel depolama alanında durum koşullarına göre tutulmaktadır. Eğer veri ilk kez oluşturuluyorsa belirlenen düğüm sayısına göre sırasıyla UVS'lerine dağıtılma işlemi yapılmaktadır. Eğer veri, ağ üzerinden UVS'lerden okunuyorsa yine ağ üzerinden UVS'lerine yazılması sağlanır.

3.3.2 Kodlanan verinin dağıtılması

Kodlama yöntemlerine göre kod verisi farklı sayıda parçalara bölünür. Her bir veri parçası, belirlenen düğüm noktasına gönderilmektedir. Bu şekilde kodlanan verinin dağıtılması sistematik olmaktadır. Düğüm sayısı elde edilen hücre sayısı kadardır. Toplamda en fazla n tane aktif düğüm olabilmektedir. Ağ üzerinde bir hata anında erişilemeyen düğümdeki verilerin, yeniden MKS üzerinde elde edildikten sonra erişilebilen yeni düğüm noktalarına iletilmeleri sağlanmaktadır.

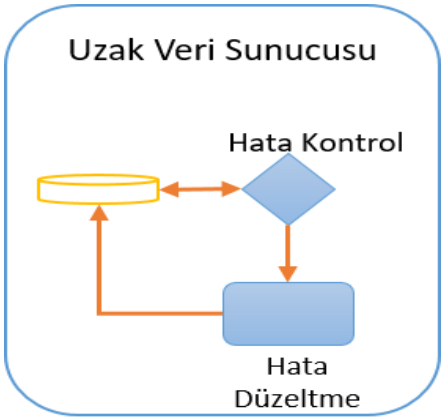


Şekil 3.7 Verinin düğümlere ayrılması

Oluşturulan sistem modelinde sadece verinin ilk oluşturulma anında, tüm veri bir arada geçici olarak MKS üzerinde tutulmaktadır. Sonrasında DVDS'nin bir uygulaması olarak veriler belirlenen düğüm sayısı kadar parçalarına ayrılmakta ve UVS'lerine iletilmektedirler.

3.3.3 Düğüm üzerindeki veri parçasının doğrulanması

Her bir UVS üzerinde kodlama yöntemine göre üreteç matrisi bulunmaktadır. Bu sayede veri parçaları üzerinde hata tespiti yapılabilmektedir (Şekil 3.8). Tespit edilen hatalar için hata düzeltme oranı (HDO) kadar düzeltme de sağlanabilmektedir. Tez kapsamında önerilen bu sistem modeli sayesinde sadece tüm verinin doğrulanması değil, veri parçalarının da bulundukları UVS'lerde doğrulamaları yapılabilmektedir. Bu nedenle doğru verinin daha verimli bir şekilde elde edilebilmesi amaçlanmıştır.



Şekil 3.8 Düğümlerde veri parçasının doğrulanması

3.3.4 Erişim kontrolü

UVS üzerinde veri parçasının doğrulaması yapıldıktan sonra verinin sağlık durumu MKS'ye iletilir. Eğer veride hata düzeltme oranından daha fazla hata varsa hata kodu, aksi durumda doğrulama kodu bildirilir.

Erişimin kontrolü sadece düğümlerdeki verinin doğruluğunun kontrolüyle değil, ağ üzerinden herhangi bir şekilde düğüme erişilememe durumunun kontrolünü de içermektedir. MKS ile düğümlerdeki UVS'ler arasında karşılıklı kalp atış mesajlarını gönderecek şekilde kontrol sistemi geliştirilmiş olup, verinin okunmasından önce düğümle arasındaki iletişim kontrol edilmektedir.

Hatalı bir veri varsa veya düğüme erişilemiyorsa, MKS bu düğümdeki verinin okunması için ağı meşgul etmemekte, hatalı düğümün verisini kendi üzerinde düzelterek orijinal verinin elde edilmesini ve okunmasını sağlamaktadır. Bu sayede silinti kodlamaların ağ bandındaki yüksek kullanım senaryosunda iyileştirme sağlanması hedeflenmiştir.

3.3.5 Hata kontrolü

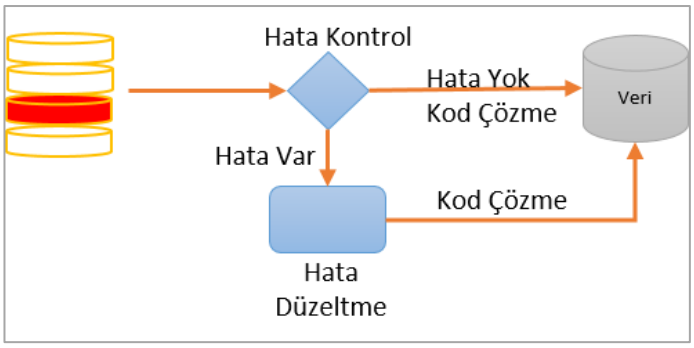
Hata kontrolü, veri iletişimi sırasında MKS ile UVS arasındaki gürültü, veri ağındaki arıza veya diğer aksaklıklardan dolayı oluşan hataların saptanmasıdır.

MKS, hata kontrolü yaparken, tüm kodlama yöntemlerinde belirlenen HDO'dan daha fazla düğüme erişemiyorsa, kod çözme adımına gerek kalmadan, veriye sağlıklı bir şekilde erişemediği için hata vermektedir. Eğer düzeltilebilir sayıda hata varsa okunan hatalı veriden doğru verinin elde edilebilmesi için hatalar, kodlama yönteminin özellikleri doğrultusunda düzeltilmektedir.

3.3.6 Hatalı verinin düzeltilmesi

Hata düzeltme, erişilemeyen düğümlerin belirlenmesi veya hataların bulunması sonrasında kodlama verisinin yeniden elde edilmesidir.

Hata bulma ve düzeltme, tüm iletişim hatlarında güvenilir veri iletişimi sağlayan tekniklerdir. Veri iletişiminde birbirinden bağımlı veya bağımsız olarak birden çok faktör devrede olduğu için hatanın tespit edilip düzeltilmesi veri iletişiminde oldukça önemlidir. Hata düzeltme orijinal verilerin yeniden oluşturulmasını sağlarken, hata algılama teknikleri hataları tespit etme işlemini yürütmektedir. Şekil 3.9'da gösterildiği gibi, hata düzeltme işlemi sistem modelinde gerçekleştirilmektedir.



Şekil 3.9 Hatalı verinin düzeltilmesi

Ele alınan sistem modelindeki her bir kodlama yönteminde, kodlama algoritmalarındaki farklılıklardan dolayı tespit ve düzeltme oranları birbirinden farklılıklar göstermektedir. Bu farklılıkların kodlama yöntemlerinin avantaj ve dezavantajlarının yanı sıra uygulama adımlarına göre de kıyaslamaları ele alınacaktır.

3.3.7 Kod çözme

Kodlanan bir veriden yeniden orijinal verinin elde edilebilmesi için kod çözme işlemi gerçekleştirilmektedir. Kod çözme işlemi için hata yerleri ve hatalı verinin değerinin bulunması gerekmektedir. Sistem modelinde, genel olarak kod çözme sırasında kod verisinden türetilen sendrom tablosu oluşturulmuştur. Bu sayede orijinal veri yeniden elde edilirken sınıf liderleri ve bunlara ait sendromların bulunması yeterli olmaktadır. Bu tabloya sendrom arama tablosu adı verilmektedir. Ayrıca kod çözme sürelerinin düşürülmesi için veriye ait hangi parçanın hangi düğümde bulunduğuyla alakalı olarak bir veritabanı oluşturulmuştur. Bu veri tabanı sayesinde eğer tüm düğümler sorunsuz bir şekilde okunuyorsa parite verilerinin kullanılmasına gerek kalmadan, dolayısı ile de kod çözme işlemine gerek kalmadan orijinal veri elde edilebilir. Eğer hatalı bir düğüm varsa kod çözme işlemi yerine sadece hatalı verinin düzeltilmesi sağlanmaktadır. Bu sayede ağ üzerindeki trafiğin ve kod çözme sürelerinin düşürülmesi hedeflenmiştir.

3.4 Reed Solomon Kodlama Yöntemleri

Ele alınan sistem modelinde RS kodlamasının kod uzunluğu (n) ve sembol sayısı (k) nicelikleri değiştirilerek üç farklı RS kodlama yöntemi oluşturulmuştur.

- n = 7 ve k = 3 için RS(7,3) kodlaması
- n = 255 ve k = 223 için RS(255,223) kodlaması
- n = 1023 ve k = 613 için RS(1023,613) kodlaması

Bu kodlama yöntemleri oluşturulurken işlenecek olan verinin farklı blok uzunluklarına göre işlem yapılarak, DVDS uygulama adımlarındaki farklılıkların ortaya çıkarılması hedeflenmiştir.

3.4.1 Kodlama

Her bir RS kodlama yöntemine göre üreteç matrisleri elde edilmektedir. Üreteç matrislerin boyutları, oluşturulan RS kodlamalarına göre değişkenlik göstermektedir.

RS(7,3) kodlamasının üreteç polinomu $\mathrm{t} = \frac{7-3}{2} = 2$ alınarak eşitlik (2.31)'e göre ele alındığında aşağıdaki eşitlikler elde edilir.

$$g(X) = (x + \alpha)(x + \alpha^2)(x + \alpha^3)(x + \alpha^4) \tag{3.1}$$

$$g(X) = g_0 + g_1 x + g_2 x^2 + g_3 x^3 + x^4 \tag{3.2}$$

RS(255,223) kodlamasının üreteç polinomu $\mathrm{t} = \frac{255-223}{2} = 16$ alınarak eşitlik (2.31)'e göre ele alındığında aşağıdaki eşitlikler elde edilir.

$$g(X) = (x + \alpha)(x + \alpha^2) \dots (x + \alpha^{31})(x + \alpha^{32}) \tag{3.3}$$

$$g(X) = g_0 + g_1 x + g_2 x^2 + \cdots + g_{31} x^{31} + x^{32} \tag{3.4}$$

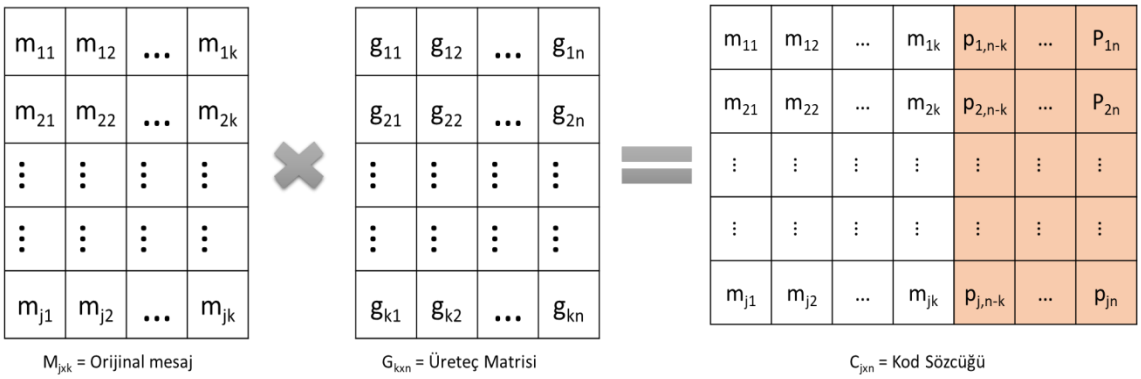
RS(1023,613) kodlamasının üreteç polinomu $\mathrm{t} = \frac{1023-613}{2} = 205$ alınarak eşitlik (2.31)'e göre ele alındığında aşağıdaki eşitlikler elde edilir.

$$g(X) = (x + \alpha)(x + \alpha^2) \dots (x + \alpha^{409})(x + \alpha^{410}) \tag{3.5}$$

$$g(X) = g_0 + g_1 x + g_2 x^2 + \cdots + g_{409} x^{409} + x^{410} \tag{3.6}$$

Elde edilen eşitliklere göre şekil 3.10'daki gibi üreteç matrisleri üretilebilir.

RS yönteminin kodlaması sırasında, temel olarak orijinal verinin (mesaj sözcüğünün) üreteç matrisi ile çarpılması sonucu parite sembollerinin oluşması sağlanmaktadır.



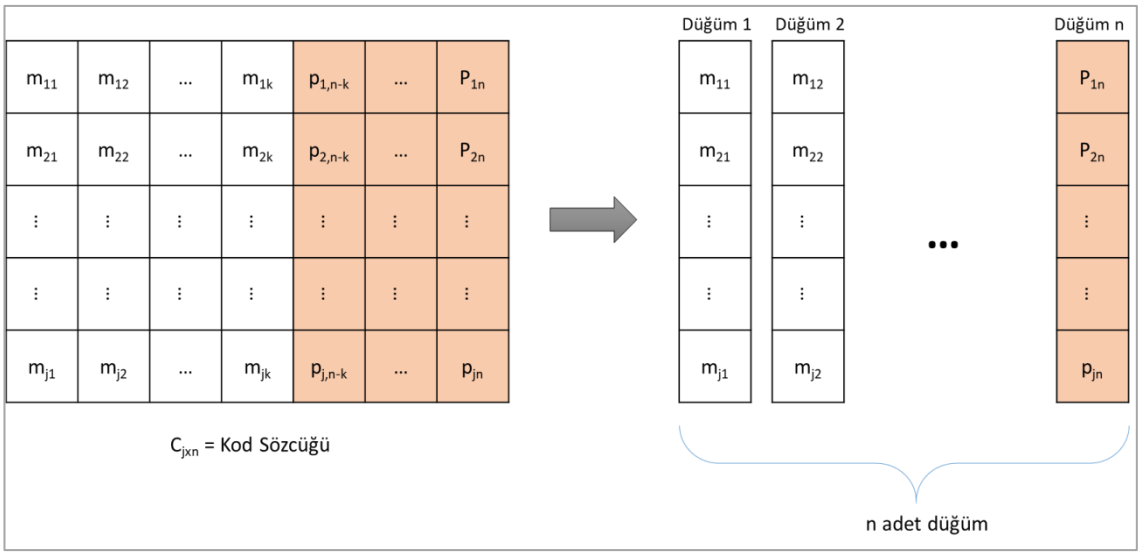
Şekil 3.10 Reed-Solomon kodlaması

3.4.2 Kodlanan verinin dağıtılması

Kodlanan verinin dağıtılması sayesinde DVDS'nin ilk adımı tamamlanmış olmaktadır. Oluşturulan RS kodlama yöntemlerine göre elde edilen üreteç matrisleri sabittir, farklı veriler veya veri boyutlarına göre değişkenlik göstermemektedir.

Bu doğrultuda, MKS'de oluşturulacak düğüm sayısına karar verilmelidir. Düğüm sayısına göre orijinal verinin hücrelerine ayrılması sağlanır. Her bir hücre ayrı ayrı kodlanarak, hücredeki veri MKS üzerinde tutulmadan ağ üzerinden daha önce tanımlanan düğümlere iletilir. Bu noktada MKS üzerinde sadece elde edilen parite verileri geçici olarak tutulmaktadır. Tüm parite verileri elde edildikten sonra düğüm sayısına göre, ya parite verileri mevcut düğümlere dağıtılır ya da yeni bir düğüm noktası olarak ağ üzerinden yeni bir düğüme tüm hücre bilgisi iletilir.

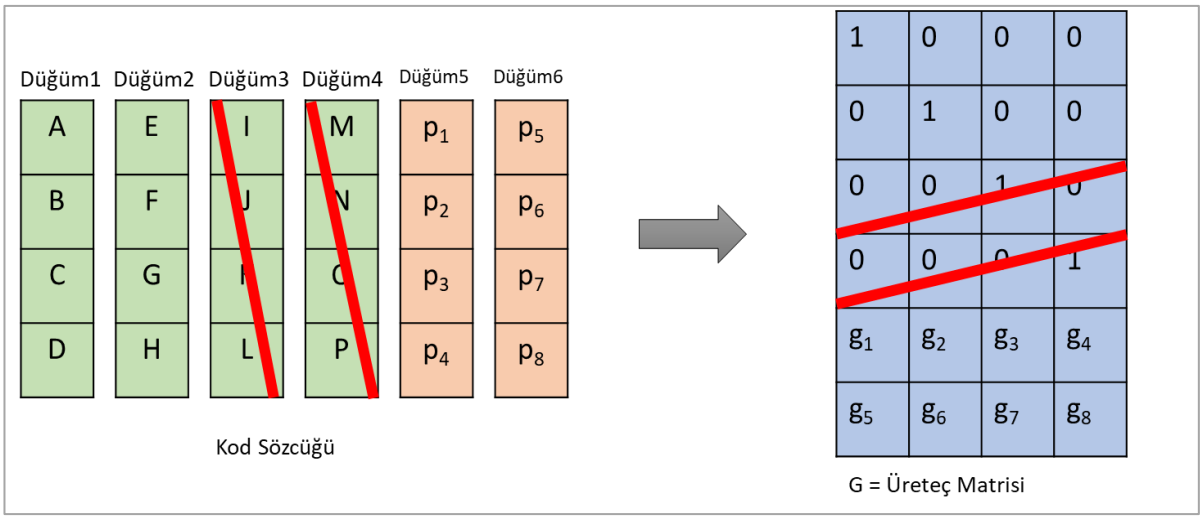
Belirlenebilecek ve dağıtılabilecek maksimum düğüm sayısı (MADS), veri blok sayısı (n) kadardır. Bu durum şekil 3.11'de görülmektedir.



Şekil 3.11 Belirlenebilecek ve dağıtılabilecek maksimum düğüm sayısı veri blok sayısı

3.4.3 Hata veya veriye erişilememe durumu

Ağ üzerinde veya fiziksel bir donanıma bağlı olarak oluşan herhangi bir hata durumunda RS kodlama yöntemleri hata düzeltebilme kapasitelerine (t) göre işlem yapabilmektedirler. Sistem modelinde hem MKS üzerinde hem de UVS'lerde hatalı veri veya düğüm sayısı kontrol edilmektedir. Bu sayede eğer t değerinden daha büyük bir veriye erişilememe durumu varsa MKS, verinin kullanımını durdurarak, veri düğümlerini kilitlemektedir. Böylelikle daha fazla verinin bozulmasına engel olunması amaçlanmıştır.

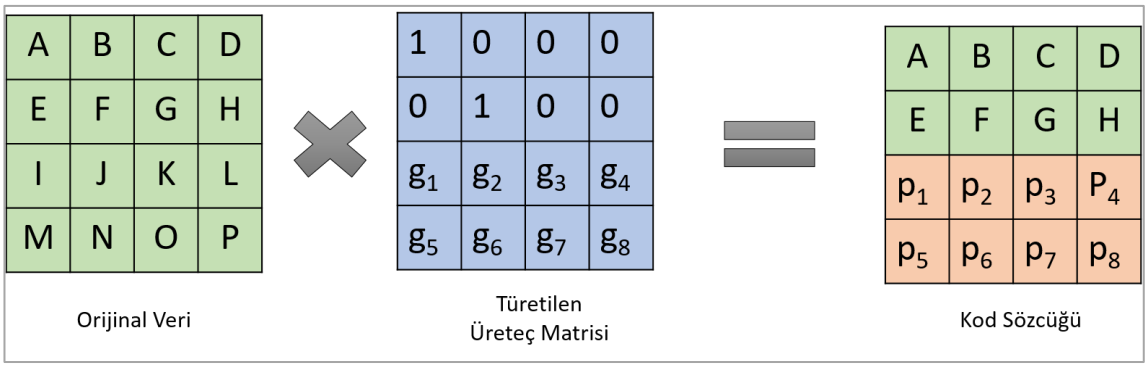


Şekil 3.12 Hata durumu

Üreteç matrisleri hem MKS hem UVS üzerinde, kodlama yöntemine bağlı olarak tutulmaktadır. Hata durumunda eğer t değeri, erişilemeyen düğüm veya veri sayısından düşükse, hangi düğüme erişilemiyorsa onun karşılığı olan üreteç matrisindeki hücreler matristen kaldırılır (Şekil 3.12). Böylelikle yeni boyutlu geçici bir üreteç matrisi elde edilmiş olunur. Bu geçici üreteç matrisi ile erişilemeyen veya hatalı olan verinin düzeltilmesi sağlanır.

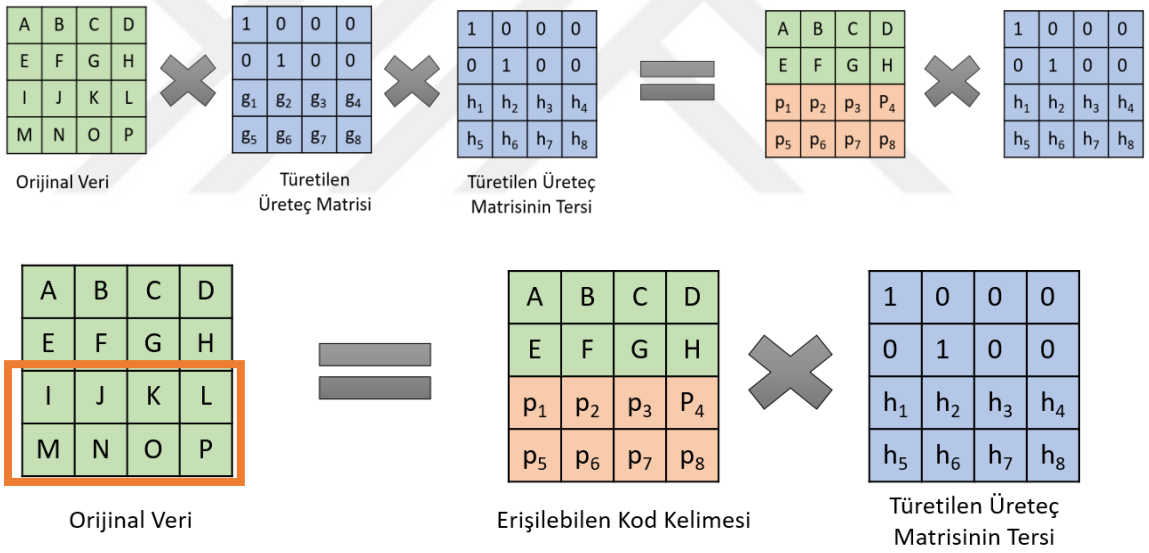
3.4.4 Kod çözme hatalı verinin düzeltilmesi

Hatalı veya erişilemeyen düğüme göre geçici olarak oluşturulan üreteç matrisi ile hem hatalı verinin düzeltilmesi hem de kod çözme işlemi yapılarak, orijinal verinin elde edilmesi sağlanmıştır. Şekil 3.13'te belirtilen kod kelimesi erişilebilen düğümleri simgelemektedir. Buna göre geçici üreteç matrisinin kullanılmasının, orijinal veriye ulaşmak için yeterli olacağı kanıtlanmaktadır.



Şekil 3.13 Hatalı veriye göre erişilebilen kod kelimesi

Orijinal veriye ulaşabilmek için geçici üreteç matrisinin tersi alınarak, erişilebilen kod kelimesi ile çarpılmasının sonucunda, hem hatalı verilerin düzeltilmesi hem de orijinal verinin yeniden elde edilmesi sağlanmıştır (Şekil 3.14).



Şekil 3.14 Orijinal verinin yeniden elde edilmesi

Kod çözme algoritmaları uygulama örneğindeki gibi kolay olmamaktadır. Bir matrisin tersinin alınması, tüm fiziksel işlem gücünün buna harcanmasına neden olmaktadır. Buna göre birçok kod çözme algoritması geliştirilmekte olup, ele alınan sistem modelinde sendrom tablosu oluşturularak bu tabloya göre hatalı verinin yeri tespit edilip

düzeltilmesi sağlanmaktadır. Bu tablo hem MKS hem de UVS üzerinde üreteç matrisi ile birlikte tutulmaktadır.

3.5 Hamming Kodlama Yöntemi

Lineer bir kodlama olan Hamming kodlama yönteminde kod uzunluğu (n) 7 ve sembol sayısı (k) 4 olan bir Hamming(7,4) kodlaması oluşturulmuştur. Bu nicelikler özellikle, kod uzunluğu aynı olan RS(7,3) kodlaması ile kıyaslanabilmesi için seçilmiş ve ele alınan sistem modelinde uygulanmıştır.

3.5.1 Kodlama

Hamming(7,4) kodlama yöntemi, kod sözcüklerinin minimum uzaklığı 2 (d©=2) olduğu için tek hata düzeltebilen bir kodlama yöntemidir. Buna göre elde edilebilecek üreteç matrisi eşitlik (3.7)'deki gibi tanımlanabilir.

$$G = (I_k|P_{n-k}) \tag{3.7}$$

Parite matrisinin de her bir elemanının minimum uzaklığı $(d(P))$ 1 olmalıdır. Bu şartlar altında ele alınan Hamming(7,4) kodlamasında aşağıdaki eşitlikler dikkate alınarak bir üreteç matrisi oluşturulmuştur.

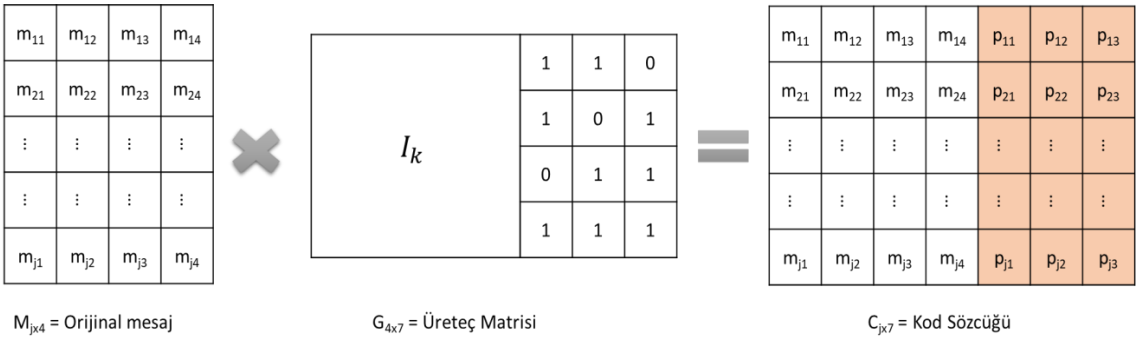
$$P_1 = m_1 + m_2 + m_4 \tag{3.8}$$

$$P_2 = m_1 + m_3 + m_4 \tag{3.9}$$

$$P_3 = m_2 + m_3 + m_4 \tag{3.10}$$

Şekil 3.15'te belirtilen M orijinal mesaj verisi, sütun uzunluğu yani k sembol sayısı sabit olacak şekilde j adet satırdan oluşmaktadır. Hamming(7,4) kodlama yönteminde

sembol sayısı (k) 4 olmakla birlikte verinin büyüklüğüne bağlı olarak satır sayısı (j) değişmektedir.



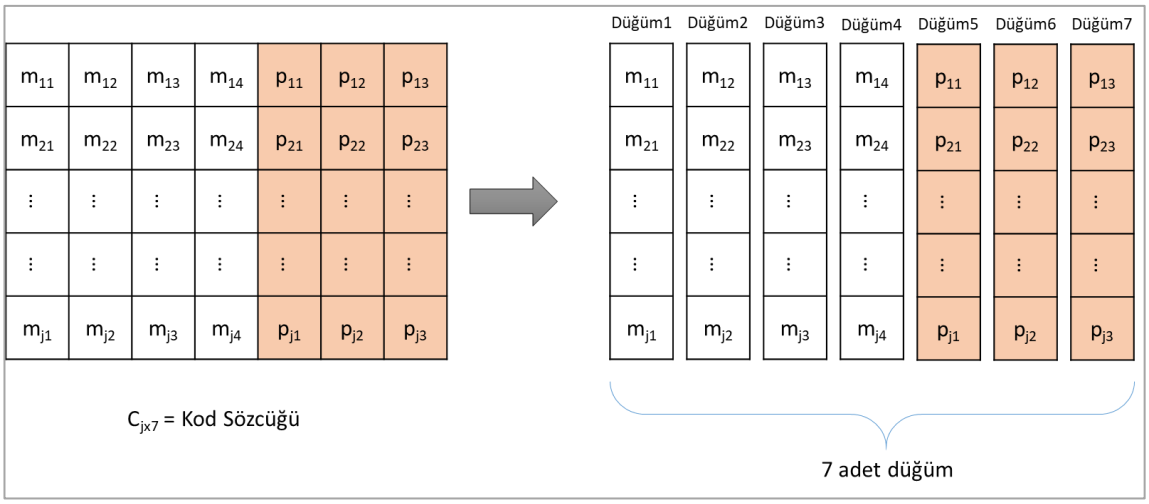
Şekil 3.15 Hamming(7,4) kodlama

Belirlenen üreteç matrisi ile orijinal verinin çarpılması sonucu Hamming(7,4) kodlama yöntemine göre orijinal verinin kodlanması sağlanmış olur. Üreteç matrisinde k boyutlu birim matrisin yer almasından dolayı elde edilen kod sözcüğünün k elemanı orijinal verinin aynısıdır. Parite verileri ise $n-k$ uzunluğunda orijinal veriye eklenmiş sütunlar olarak düşünülebilir.

3.5.2 Kodlanan verinin dağıtılması

Hamming(7,4) kodlama yönteminde düğüm sayısı sabit olmakla birlikte, bu sayı veri blok uzunluğuna (n) eşittir. Bu durumda MKS dahil, ancak yedi adet düğüm noktası oluşturulabilir. Hamming(7,4) kodlama yöntemine göre düğüm sayısının sabit olmasına bağlı olarak herhangi bir düğüme erişilememesi durumunda MKS, erişilemeyen veriyi kendi üzerinde yeniden elde edip başka bir erişilebilen düğüm noktasına göndererek düğüm sayısını sabitlemiş olur.

Okunan veya ilk defa oluşturulan orijinal verinin her bir sütunu, düğüm noktalarındaki verinin tümünü oluşturmaktadır. Bu doğrultuda MKS'de elde edilen $n-k$ uzunluğundaki parite verilerinin her biri de ayrı düğüm noktalarına dağıtılmaktadır. Bu veriler MKS üzerinde tutulmadan UVS'ye iletilmesi sağlanmaktadır.



Şekil 3.16 Hamming(7,4) kodlaması – verilerin dağıtılması

Düğüm noktalarına iletilen verinin UVS üzerinde yeniden hata tespiti veya düzeltmesi Hamming(7,4) kodlamaları için mümkün olmamaktadır. Bunun nedeni ise hata tespiti ve düzelmesinde kullanılan parite verileri ile orijinal verilerin aynı düğüm noktalarında bulunmamalarından kaynaklanmaktadır.

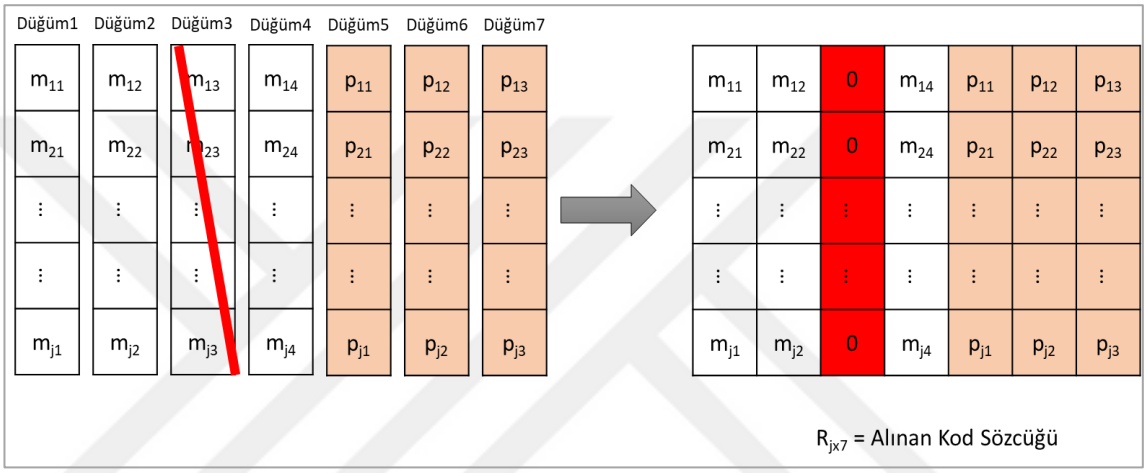
MKS üzerinde, oluşturulan bir veri tabanı ile düğüm numarasına göre bir içerik belirleme yapılmıştır. Buna göre hangi düğümde orijinal veya parite verisinin olduğu anlaşılmakta, sütun numarasına göre de verinin doğru bir şekilde içeriklendirilmesi sağlanmaktadır (Çizelge 3.1).

Çizelge 3.1 Örnek Hamming (7,4) veri tabanı tablosu

ID	Düğüm No	Verinin Türü	Verinin Sütun No	Erişilebilirlik Durumu
1	1	Orijinal	2	Doğru
2	3	Parite	5	Doğru
3	5	Parite	7	Doğru
4	6	Orijinal	1	Doğru
5	4	Orijinal	4	Yanlış
6	7	Parite	6	Doğru
7	2	Orijinal	3	Doğru
8	8	Orijinal	4	Doğru

3.5.3 Hata veya veriye erişilememe durumu

Ağ üzerinden, düğümlerdeki verilerin MKS'de okunması, buna göre kod çözme işleminden önce hatanın tespit edilmesi gerekmektedir. Hamming(7,4) kodlaması, tek hata düzeltebilen bir kodlama yöntemi olduğu için yedi düğüme dağıtılmış bir ortamda sadece tek bir düğüm noktasına erişilememesi durumunu veya tek bir düğüm noktasından gelen hatalı verileri düzeltebilir.



Şekil 3.17 Hamming(7,4) hata durumu

MKS üzerinde hata kontrolü yapıldıktan sonra, veritabanındaki tabloya göre sıralı bir şekilde düğümlerdeki veriler okunur. Okunan veriler ile $(j \times k)$ boyutunda bir matris elde edilir. Eğer herhangi bir düğüme erişilemiyorsa bu düğümün bulunduğu sütun sıfırlanır. Bu şekilde yeniden oluşturulan matrisin boyutu değişmemiş olur ve Hamming(7,4) kodlamasına göre hata ve kod çözme işlemleri yapılabilir.

$$m_e = [m_{1\times k}] = 0 \tag{3.11}$$

3.5.4 Kod çözme ve hatalı verinin düzeltilmesi

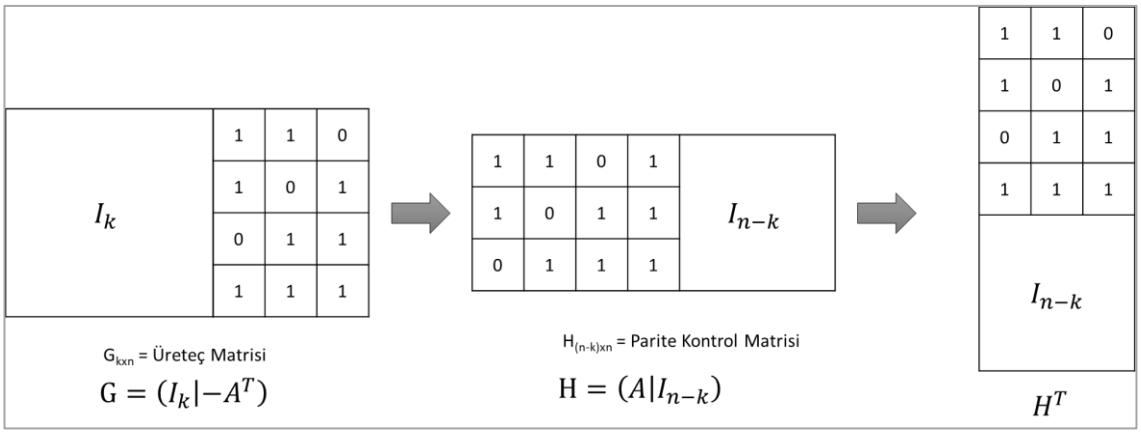
Hamming(7,4) kodlaması tek bir hata düzeltebilen bir kodlama olmasına rağmen MKS üzerinde oluşturulan veri tabanı sayesinde aşağıda belirtilen koşullara göre düğüm hata düzeltmesi yapılmaktadır (Çizelge 3.1).

Koşul-1: Eğer, veri türü "Orijinal" ve tüm düğüm numaralarının erişilebilirlik değeri "Doğru" ise, orijinal veri, kod çözme işlemi gerçekleştirilmeden elde edilir. Orijinal veri elde edilirken sadece veri sütun numarasına göre orijinal veriyi içeren düğümlerin sırasıyla okunması sağlanır. Bu durumda parite verilerinin okunmasına, dolayısıyla da kod çözme işlemine gerek olmamaktadır.

Koşul-2: Eğer, en az bir düğüm noktasının veri türü "Orijinal" ve birden çok erişilebilirlik değeri "Yanlış" ise, orijinal veri yeniden elde edilemez.

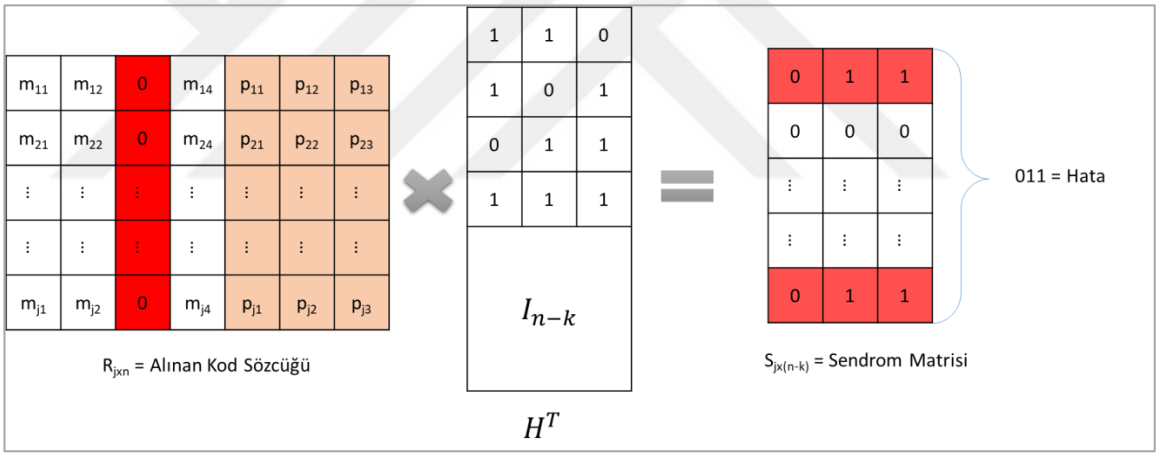
Koşul-3: Eğer, sadece bir düğüm noktasının veri türü "Orijinal" ve erişilebilirlik değeri "Yanlış" ise, orijinal veri, kod çözme ile elde edilir.

Hamming(7,4) kodlama yöntemine göre kod çözme işlemi için öncelikle hatanın tespit edilmesi gereklidir. Hata tespiti, parite kontrol matrisi kullanılarak yapılmaktadır. Eşitlik (2.14)'e göre, üreteç matrisi kullanılarak parite kontrol matrisi elde edilebilir (Şekil 3.18). Bu adımın, her bir verinin kod çözümü sırası tekrarlanmasına gerek yoktur. MKS üzerinde hem üreteç hem de parite kontrol matrisi verileri tutulmaktadır.



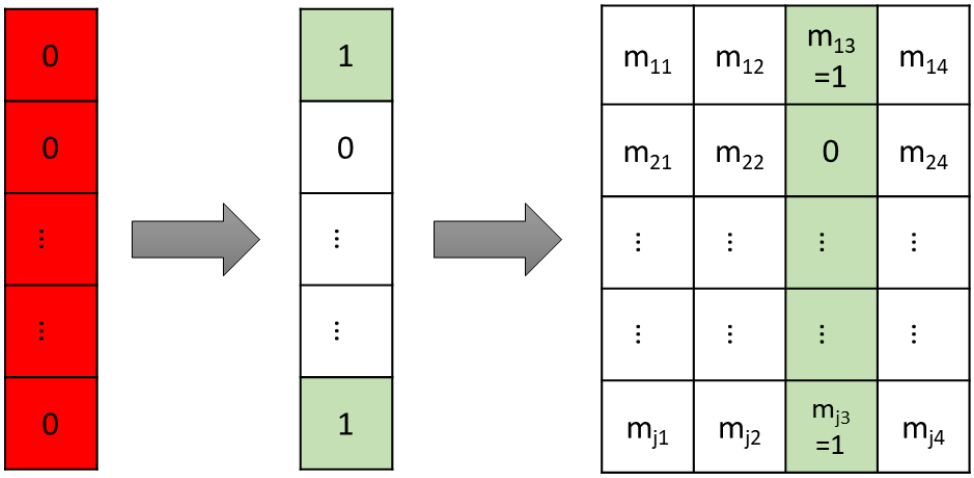
Şekil 3.18 Hamming(7,4) parite kontrol matrisinin üretilmesi

Eşitlik (2.56) – eşitlik (2.60) denklemleri kullanılarak hatalı verinin sıfırdan farklı hata bitlerinin yerleri tespit edilebilir. Bu da hatalı verinin belirlenmesini sağlamaktadır.



Şekil 3.19 Hamming(7,4) hata tespiti

Şekil 3.19'daki gibi $1 \times (n-k)$ boyutundaki verilerin sıfırdan farklı olanlarının değerleri kaçıncı sütunda hata olduğunu belirlemektedir. Bu durumda tüm düğümün verisinin düzeltilmesine gerek kalmadan sadece hatalı bitlerin düzeltilmesiyle tüm hatalı düğümün verisi elde edilebilir.



Şekil 3.20 Hamming(7,4) hatalı düğümün düzeltilmesi ve kod çözme

Hatalı düğümün düzeltilmesi sayesinde veritabanındaki düğüm numarası ile sütun numarasındaki ilişkiye göre de kodlanan veriden orijinal veri elde edilebilmektedir (Şekil 3.20).

3.6 Parite-XOR Kodlama Yöntemi

Parite-XOR kodlaması, tek hata düzeltebilen bir kodlama yöntemidir. Veri blok uzunluğu (n) dört, sembol sayısı (k) üç olarak belirlenmiştir. Ele alınan diğer tüm kodlama yöntemlerine göre en düşük blok uzunluğuna sahip bir kodlama yöntemidir. Tek hata düzeltebilen bir kodlama yöntemi olması nedeniyle, Hamming(7,4) kodlama yöntemiyle kıyaslanması, aynı zamanda kodlama ve kod çözme algoritmalarının basitliği, bu kodlama yönteminin seçilmesinde etkili olmuştur.

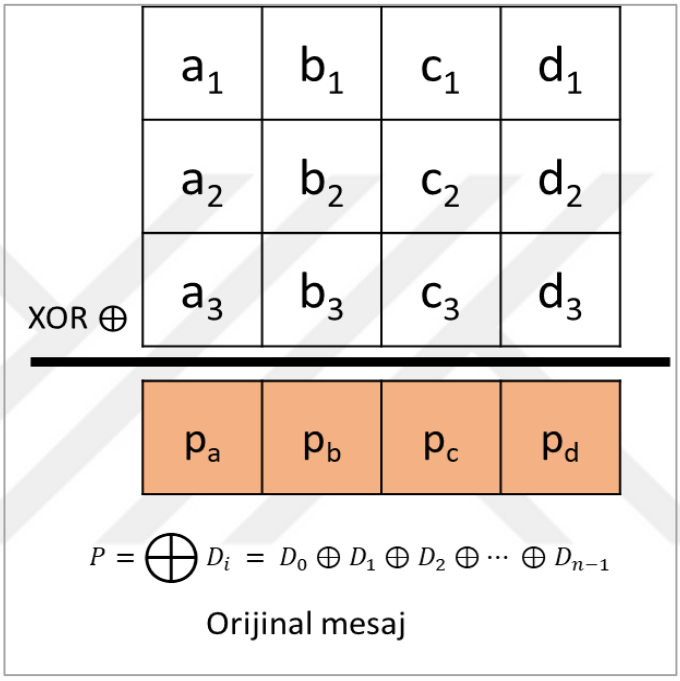
3.6.1 Kodlama

Diğer kodlama yöntemlerinde oluşturulan üreteç matrisinin, Parite-XOR kodlamasında oluşturulmasına gerek yoktur. Kodlama, XOR işlemiyle sağlanmaktadır. Orijinal verinin sütunundaki her bir elemanın XOR işlemi sonucunda elde edilen değer, işlem yapılan sütünün parite verisini oluşturmaktadır. Eşitlik (2.62)'deki denklem kullanılarak parite verileri elde edilmektedir. Buna göre;

$$P_a = a_1 + a_2 + a_3 \tag{3.12}$$

$$P_b = b_1 + b_2 + b_3 \tag{3.13}$$

$$P_c = c_1 + c_2 + c_3 \tag{3.14}$$



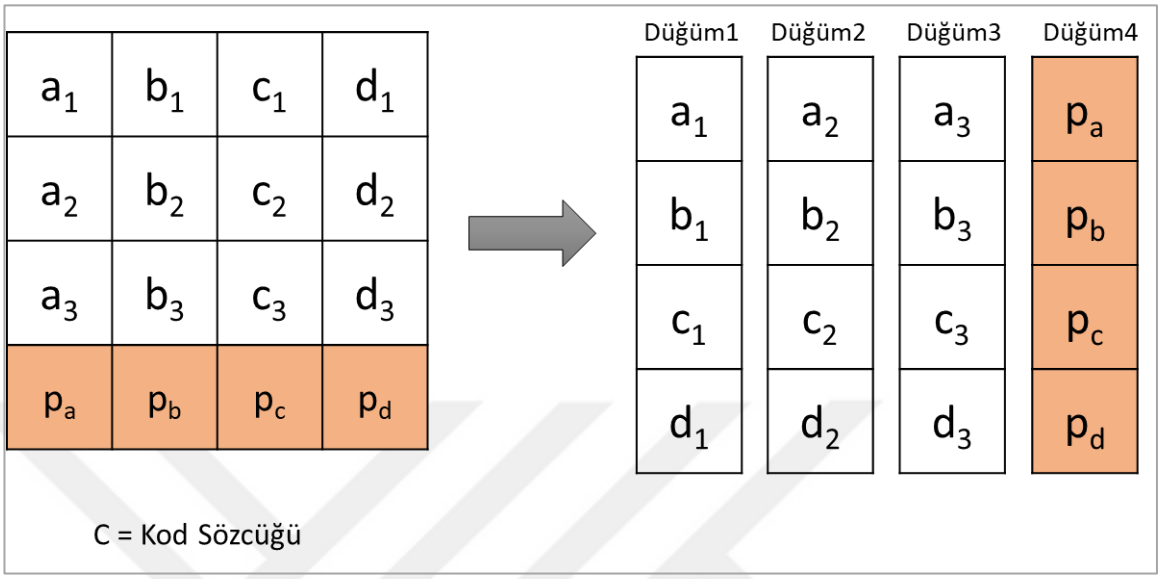
Şekil 3.21 Parite-XOR kodlama

3.6.2 Kodlanan verinin dağıtılması

Parite-XOR kodlamasında, veri blok uzunluğu (n) kadar yani dört adet düğüm noktasının erişilebilir olması gerekmektedir. Verinin büyüklüğüne bağlı olarak düğüm sayısı değişmemektedir.

Eğer veri ilk kez oluşturulmuşsa, MKS öncelikle orijinal veriyi belirlenen düğüm sayısı kadar UVS'na ağ üzerinden iletir. Sonrasında elde edilen parite verileri, sırasıyla dağıtılan düğümlere iletilir. Eğer verinin her bir parçası ağ üzerinden, MKS'de

okunuyorsa, elde edilen parite verileri, MKS üzerinde depolanmadan tüm düğüm noktalarına sırasıyla iletilir.

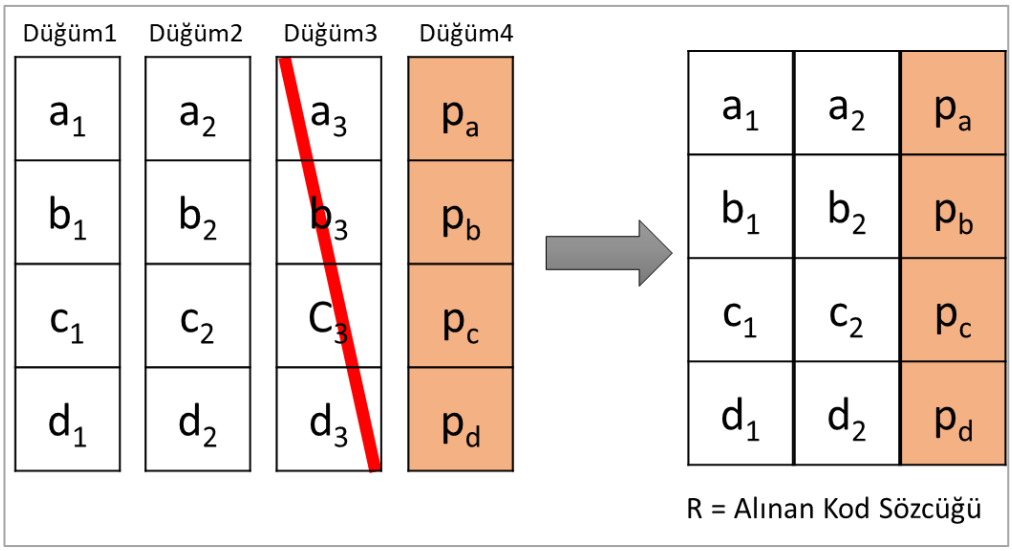


Şekil 3.22 Parite-XOR – verinin düğümlere dağıtılması

Hamming(7,4) kodlama yönteminde olduğu gibi işlemlerin daha hızlı yapılabilmesi için Parite-XOR kodlaması için de bir veritabanı oluşturulmuştur. Bu veritabanı sayesinde, dağıtılan verinin hangi düğümlerde olduğu bilinmekte ve takip edilebilir olması sağlanmaktadır (Çizelge 3.1).

3.6.3 Hata veya veriye erişilememe durumu

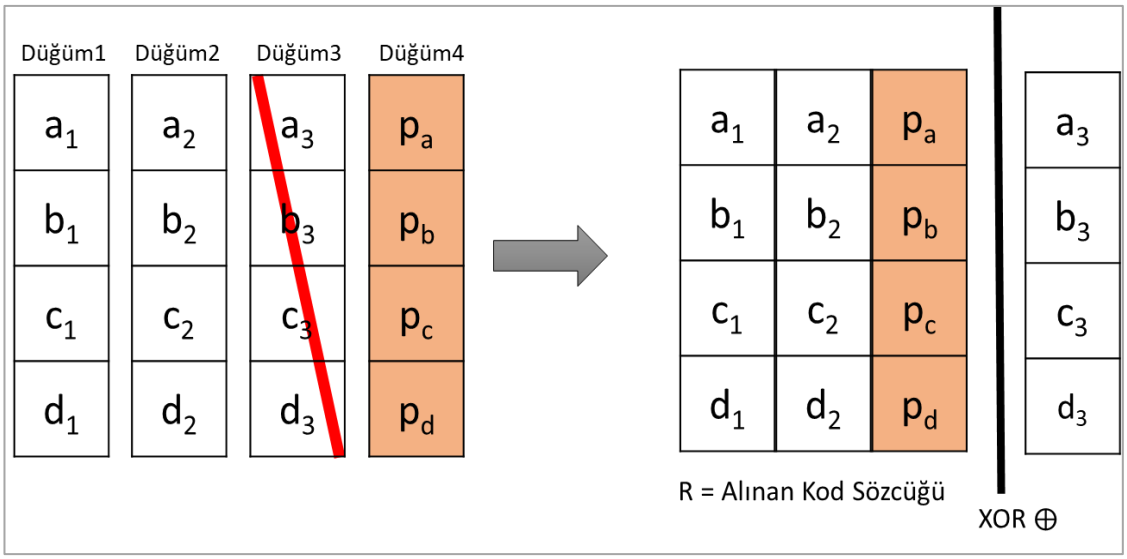
Parite-XOR kodlaması, tek hata düzeltebilen bir kodlama yöntemidir. Bu kodlama yöntemine göre, sadece tek bir düğüm noktasına erişilememesi bir hata olarak algılanmakta ve verinin yeniden elde edilmesi sağlanmaktadır. Buna göre herhangi bir düğüm noktasına erişilememesi durumunda, erişilebilen diğer üç düğüm şekil 3.23'teki gibi satır satır okunmaktadır. Her bir satır erişilemeyen düğümün yeniden oluşturulması ve kod çözme adımlarının uygulanması sırasında kullanılacaktır.



Şekil 3.23 Parite-XOR hata durumu

3.6.4 Kod çözme hatalı verinin düzeltilmesi

Parite-XOR kodlama yönteminde, hata tespiti yapıldıktan sonra hatanın düzeltilmesi ve orijinal verinin yeniden elde edilebilmesi için kod çözme işlemleri yapılabilir. Kodlama sırasında herhangi bir üreteç matrisi kullanılmamıştır. Hata düzeltme için de parite kontrol matrisine ihtiyaç duyulmamaktadır. Kodlama için kullanılan eşitlik (2.62) hata düzeltme, yani erişilemeyen düğümün verisinin MKS üzerinde yeniden yaratılması için de kullanılmaktadır. Yeniden elde edilen hatalı düğüm verisi, erişilebilir başka bir düğüm noktasına iletilir. Bu sayede Parite-XOR kodlaması için sabit olması gereken düğüm sayısı sağlanmış olur.



Şekil 3.24 Parite-XOR hatalı düğümün düzeltilmesi

Hatalı veri düzeltildikten sonra, kod çözme işlemi için oluşturulan sendrom tablosu yerine sistem modelinde oluşturulan veritabanı kullanılmaktadır (Çizelge 3.1). Bu sayede kod çözme süreleri ve ağ kullanım oranlarının düşürülmesi amaçlanmıştır. Buna göre aşağıdaki durumlara göre kod çözme işlemi gerçekleştirilerek orijinal veri elde edilir.

Koşul-1: Eğer, veri türü "Orijinal" ve tüm düğüm numaralarının erişilebilirlik değeri "Doğru" ise, orijinal veri, kod çözme işlemi gerçekleştirilmeden elde edilir. Bu durumda parite verilerinin okunmasına da gerek olmamaktadır.

Koşul-2: Eğer, sadece bir düğüm noktasının veri türü "Orijinal" ve erişilebilirlik değeri "Yanlış" ise, orijinal veri, hata düzeltme sonucunda elde edilmiş olur.

4. ARAŞTIRMA BULGULARI

DVDS'nin bir modeli oluşturularak, bu tez çalışması kapsamında aşağıdaki kodlama yöntemleri ele alınmıştır:

- RS (7,3)
- RS (255,223)
- RS (1023, 613)
- Hamming (7,4)
- Parite-XOR

Her bir kodlama yöntemi önerilen sistem mimarisine göre ele alınmış olup DVDS'lerde önemli olabilecek farklı metriklere göre kıyaslanmıştır. Bu metrikler aşağıda sıralanmıştır.

- Kodlama ve Kod Çözme Süreleri
- Veri Artış Oranı ve Verimlilik
- Hata Düzeltme Oranı
- Genişletilebilirlik
- Verinin Taşınabilme Kapasitesi

4.1 Kodlama ve Kod Çözme Süreleri

Boyutları; 1MB, 10MB ve 100MB olan veriler yaratılarak, verilerin boyutlarına göre her bir kodlama yöntemi için kodlama ve kod çözme süreleri hesaplanmıştır. Verilerin disk(lere) yazılıp okunması ve tüm işlemlerin Matlab üzerinden hesaplanması gerçek zamanlı bir sisteme göre kararsızlıkların oluşmasına; ayrıca Matlab ile fiziksel katman arasında işletim sisteminin olması, kodlama ve kod çözme sürelerinin yüksek çıkmasına neden olmaktadır. Bunların yanında bu süreler kullanılan fiziksel donanıma ve farklı Matlab versiyonlarına göre de değişkenlik gösterebilmekte olup birbirlerine göre oranları donanım bağımsız sabit olacaktır.

Her bir kodlama yöntemi farklı blok boyutlarında kodlama ve kod çözme işlemlerini gerçekleştirmektedir. Kodlanacak olan verinin (kod sözcüğünün) boyutu çizelge 4.1'deki blok boyutlarına göre bloklara ayrılmaktadır. Blok sayısı ve blok boyutlarına bağlı olarak işlemci ve disk performans değerleri önem taşımaktadır. Blok sayısının artması işlemci üzerindeki işlem yükünü artırmaktadır. Blok boyutunun artması ise disk üzerindeki kuyruk uzunluğunun artmasına neden olmaktadır.

Çizelge 4.1 Kodlama yöntemlerinin blok boyutları

Silinti Kodlama	**Blok Boyutu**
RS(7,3)	7 Byte
RS(255,223)	255 Byte
RS(1023,613)	1023 Byte
Hamming(7,4)	7 Byte
Parite-XOR	4 Byte

Çizelge 4.2 Veri boyutlarına göre blok sayıları

Kodlama Yöntemi	**Blok Sayısı**		
	1 MB	**10 MB**	**100 MB**
RS(7,3)	146K	1463K	14629K
RS(255,223)	4K	40K	402K
RS(1023,613)	1K	10K	100K
Hamming(7,4)	146K	1463K	14629K
Parite-XOR	256K	2560K	25600K

4.1.1 Kodlama süreleri

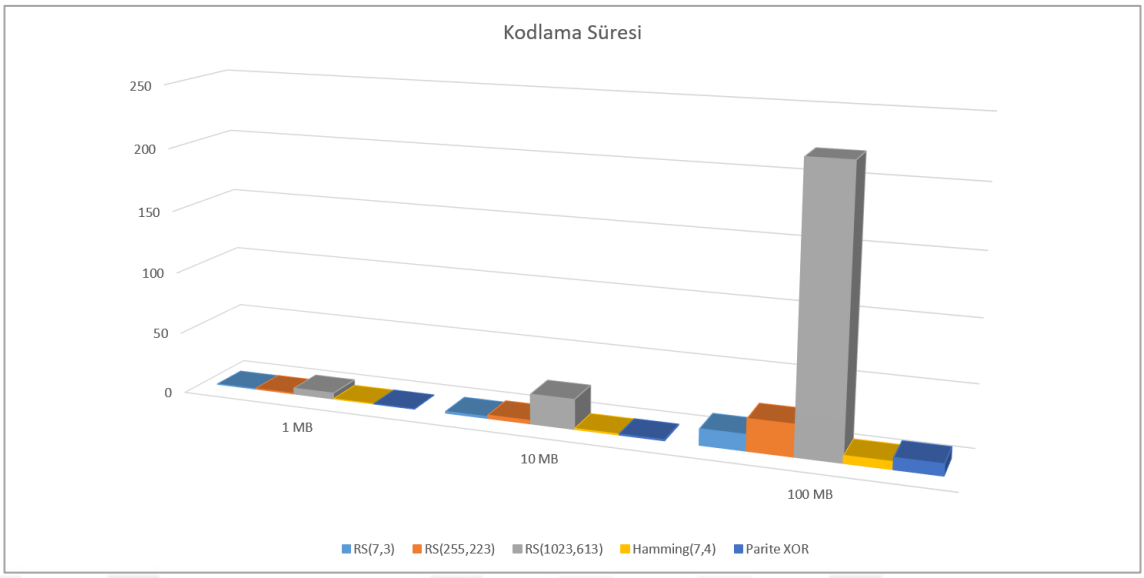
Kodlama süresi, işlemci kullanımına, verinin okuma/yazılma hızına ve fiziksel disk üzerindeki oluşan G/Ç performans değerine bağlıdır. Kodlama süresinin artması disk üzerindeki işlem kuyruğu sayısının artmasına bu da yapılan her bir kodlama için yazma süresinin artmasına neden olur.

Ele alınan kodlama yöntemleri arasında tüm veri boyutları için, Hamming(7,4) kodlamasının en hızlı, en yavaş kodlama yönteminin ise RS(1023,613) olduğu gözlemlenmiştir. Çizelge 4.3'te elde edilen kodlama süreleri, farklı zamanlarda elde edilen on beş ölçümün birer ortalaması olarak hesaplanmıştır. Bu kodlama sürelerinin grafik olarak üretilmesi sonucunda şekil 4.1 elde edilmiştir. Tüm veri boyutları için de hızlıdan yavaşa kodlama yöntemleri aşağıdaki gibidir. Ayrıca tüm ölçümlerde aşağıdaki sıralamanın değişmediği gözlemlenmiştir.

Hamming(7,4) > Parite-XOR > RS(7,3) > RS(255,223) > RS(1023,613)

Çizelge 4.3 Kodlama süreleri

Silinti Kodlama Süreleri (Saniye)	**1 MB**	**10 MB**	**100 MB**
RS(7,3)	0,194	1,282	12,763
RS(255,223)	0,374	2,569	25,157
RS(1023,613)	4,758	23,850	221,030
Hamming(7,4)	0,0505	0,630	5,928
Parite-XOR	0,0917	0,729	9,092



Şekil 4.1 Kodlama süreleri

Reed-Solomon kodlama süreleri

RS kodlama yöntemleri arasında tüm veri boyutlarında hızlıdan yavaşa sıralama aşağıda verilmiştir. Alınan bu sonuca dair grafiksel gösterim ise şekil 4.1'de görülmektedir.

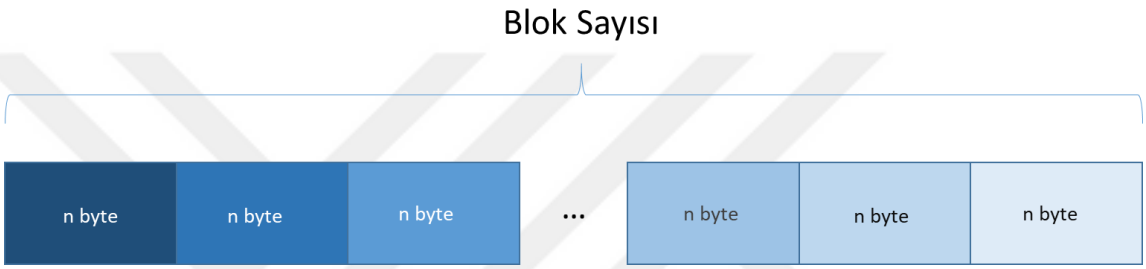
RS(7,3) > RS(255,223) > RS(1023,613)

Üç farklı RS kodlamasının kodlama sürelerinin şekil 4.1'deki gibi birbirleri arasında oldukça farklı olduğu görülmektedir. Bu durumda RS kodlamaları arasındaki farklı blok boyutlarının aynı fiziksel donanım üzerinde kodlama sürelerine direkt etki ettiği ve kodlama sürelerindeki bu farklılıkların blok boyutlarından kaynaklandığı söylenebilir. RS(7,3) kodlamasında blok boyutu 7 Byte'dır, blok sayısı ise 1 MB boyutlu bir veride 146K'dır. RS(1023,613) kodlamasında ise blok boyutu 1023 Byte'dır, blok sayısı ise 1 MB boyutlu bir veride 1K'dir. Her bir blok üzerinde kodlama işlemi yapılmaktadır. RS(7,3) kodlaması, RS(1023,613) kodlamasına göre blok boyutu az olduğu için disk üzerindeki kuyruk işlem yükünü düşürecektir buna karşın blok sayısı fazla olduğu için de işlemci üzerindeki yükü arttıracaktır. İşlemciler disklere göre mimarilerinden kaynaklı olarak her bir işlemi çok daha hızlı bir şekilde tamamladıkları için düşük blok

boyutundaki RS kodlamalarının kodlama süreleri daha düşük olacaktır. Bu durumda blok boyutu azaldıkça kodlama süresinin de azalacağı söylenebilir.

Bir RS(n,k) kodlamasında veri boyutu M Byte, her bir blok boyutu n Byte olarak ele alındığında blok sayısı (N) eşitlik (4.1)'te gibi elde edilmektedir. Bu durum şekil 4.2'deki gibi gösterilebilir.

$$N = \frac{M}{n} \tag{4.1}$$



Şekil 4.2 Silinti kodlamalarda blok yapısı

4.1.2 Hamming(7,4) - Parite-XOR - RS(7,3)

Blok boyutu ve blok sayısı aynı olan Hamming(7,4) kodlaması ile RS(7,3) kodlaması arasında, Hamming(7,4) kodlama süresinin daha kısa olduğu görülmektedir.

Şekil 4.1 incelendiğinde blok boyutlarına göre blok boyutu 4 Byte olan Parite-XOR kodlamasının kodlama süresinin en hızlı olması beklenirken, Hamming(7,4)'e göre daha uzun, RS(7,3)'e göre de daha kısa olduğu sonucuna ulaşılmıştır. Kodlama süresindeki bu farklılıklar kodlama algoritmalarının karmaşıklığı ile açıklanabilmektedir. Hamming kodlaması, RS ve Parite-XOR kodlamalarına göre, teorik olarak da ifade edildiği gibi, çok daha basit bir algoritma kullandığını göstermektedir. Aynı şekilde Parite-XOR kodlaması hem blok boyutu olarak hem de RS(7,4) kodlamasına göre teorik olarak daha az karmaşık olduğu için kodlama süresi daha düşüktür.

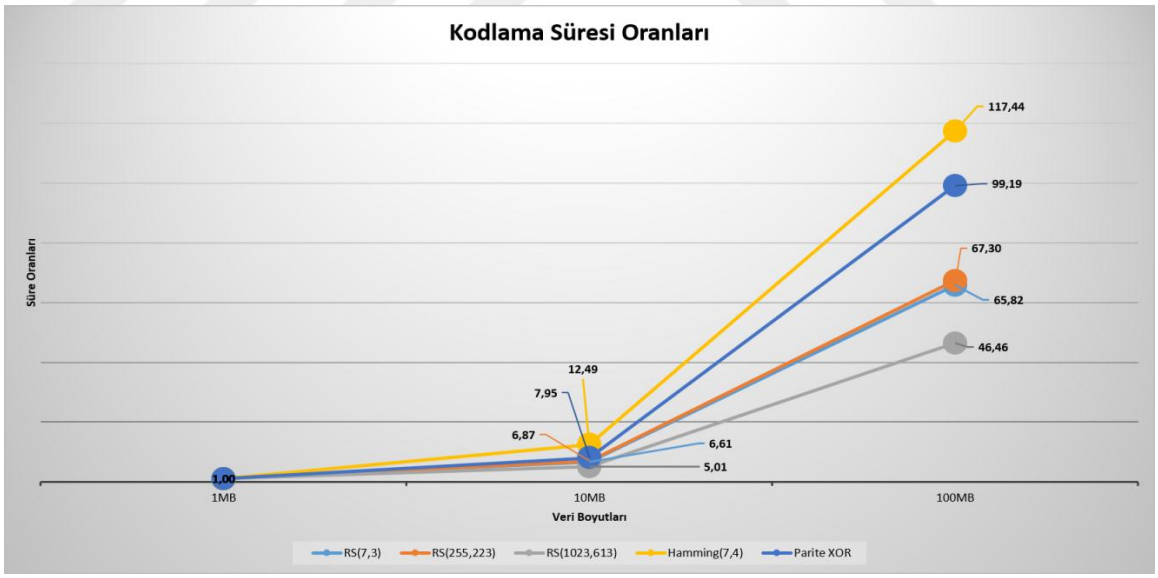
4.1.3 Kodlama süresi artış oranı

Her bir kodlama yönteminin kendi içinde veri boyutu arttıkça kodlama süresindeki değişim oranı sabit değildir. Kodlama süresi artış oranı, veri boyutuna bağlı olarak kodlama yöntemlerinin davranışlarının yorumlanmasına yardımcı olmaktadır. Bu durumda her kodlama yöntemi için 1 MB boyutlu verinin kodlama süresi 1 birim süre olarak ele alınırsa 10 MB ve 100 MB boyutlu verilerin kodlama sürelerindeki artış oranı hesaplanabilmektedir. Bu sonuç şekil 4.3'te görülmektedir.

Tüm veri boyutlarına göre kodlama sürelerindeki artış oranları aşağıdaki gibidir.

RS(1023,613) > RS(7,3) > RS(255,223) > Parite-XOR > Hamming(7,4)

Şekil 4.3 incelendiğinde kodlama süreleri ile kodlama sürelerindeki artış oranları arasında tam ters bir ilişki gözlenmektedir.



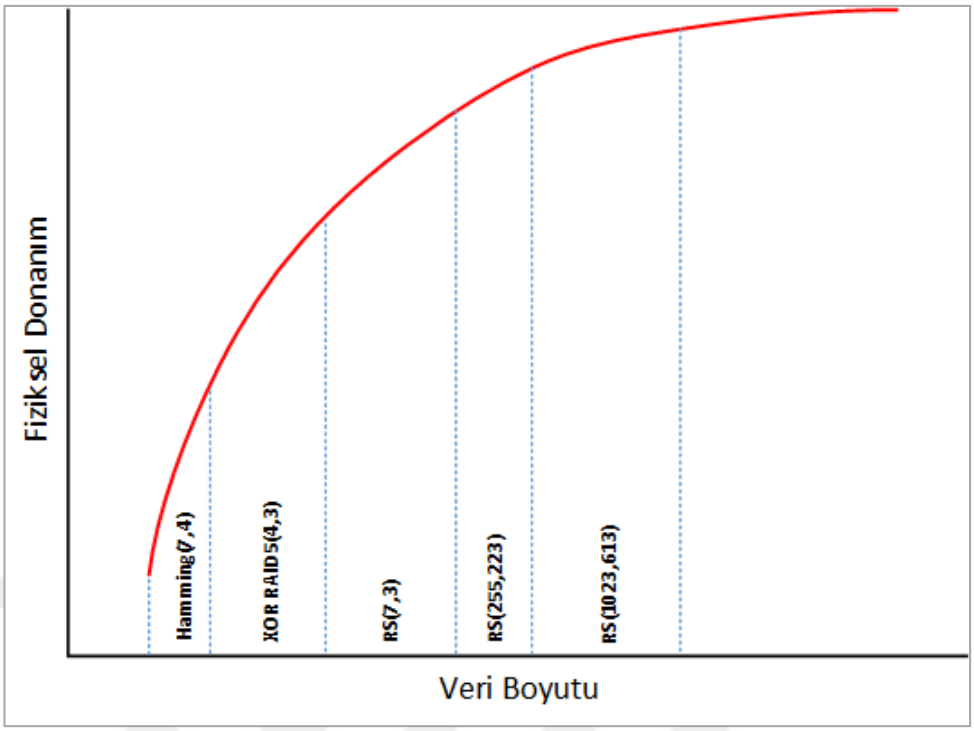
Şekil 4.3 Kodlama süresi artış oranları

Hamming(7,4) kodlaması, en hızlı kodlamayı yapabilmesine rağmen veri boyutu arttıkça kodlama süresindeki verimi düşmektedir. 1 MB boyutunda bir verinin 1 birim

sürede kodlamasının yapıldığı durumda 10 MB boyutlu verinin kodlama süresindeki artış oranı 12,49:1; 100 MB boyutlu veride ise 117,44:1 olacaktır. Tam tersi olarak RS(1023,613) kodlamasında, en yavaş kodlama olmasına rağmen, veri boyutu arttıkça kodlama süresindeki verimi artmaktadır. 1 MB boyutunda verinin 1 birim sürede kodlamasının yapıldığı durumda 10 MB boyutlu verinin kodlama süresindeki artış oranı 5,01:1, 100 MB boyutlu veride ise 46,46:1 olarak ortaya çıkmaktadır.

Kodlama süreleri fiziksel donanıma bağlı olarak değiştirilebilmektedir. Kodlama sürelerinin yüksek performanslı disklerin kullanımı ile düşürülmesi mümkün olmaktadır. Okuma/yazma kapasitesi yüksek olan diskler ile okuma/yazma performans değerleri yükseltilebilmektedir. Bu sayede disk kuyruk uzunluğu (Disk Queue Lenght) düşürülebilir. Disk kuyruk uzunluğundaki performans iyileştirmesi daha büyük blok boyutlarında işlem yapılabilmesini mümkün kılacaktır.

Fiziksel donanımların gelişmesine bağlı olarak kodlama süresi iyileştirilebilmektedir. Fakat kodlama yöntemlerine göre artış oranları aynı kalacaktır. Bu doğrultuda her bir kodlama yönteminin avantaj ve dezavantajları ortaya çıkmaktadır. Örneğin küçük boyuttaki verilerin, düşük kapasiteli (düşük seviyeli donanıma sahip) ortamlarda, kodlaması ile ilgili Hamming(7,3) kodlaması diğer kodlama yöntemlerine göre daha avantajlıyken yüksek kapasiteli ortamlarda veri boyutu arttıkça bu avantajını kaybetmektedir. Bunun yerine RS(1023,613) kodlaması daha avantajlı bir hale gelmektedir.



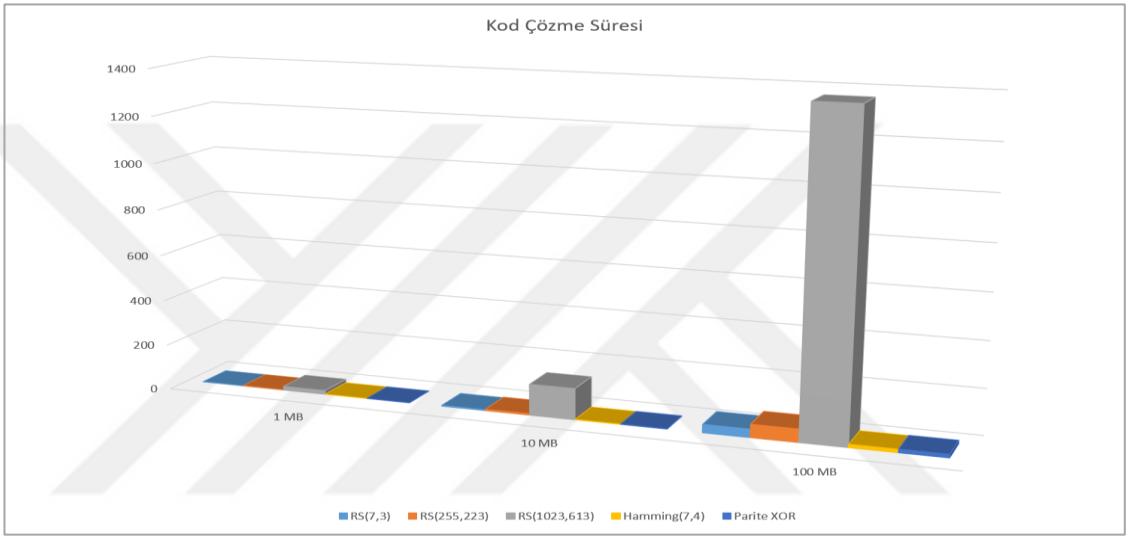
Şekil 4.4 Optimal kodlama yöntemi: veri boyutu, fiziksel donanım kapasitesi ilişkisi

Optimal kodlama yönteminin belirlenmesinde kodlama süresi niceliği, seçilecek olan kodlama yöntemi için çok önemlidir. Fakat sadece kodlama süresine bakılması da yeterli olmamaktadır. Kodlama yönteminin kendi içinde veri boyutuna göre fiziksel donanım kapasitesi ile veri boyutu arasında bir ilişki şekil 4.4'teki gibi belirlenmiştir. Buna göre kodlama süreleri iyileştirilirken kodlanacak olan verinin boyutuna göre de seçim yapılması gerekmektedir. Bu durumda RS(1023,613) kodlaması performans gerektirmeyecek, blok olarak kullanılan büyük boyutlu verilerin kodlanmasında kullanılabilirken, performans gerektiren küçük boyutlardaki verilerin kodlanması için de Hamming(7,4) kodlama yöntemi kullanılabilir.

4.1.4 Kod çözme süreleri

Silinti kodlamada temel olarak; orijinal veri (kod sözcüğü) ile üretici matrisinin çarpma işleminden çıkan sonuç bize kodlanan veriyi vermektedir. Fakat kod çözmede ise temel anlamda; üreteç matrisinin tersinin alınması ile kodlanan verinin çarpımı orijinal veriyi vermekte; aynı zamanda kod çözme sırasında hata düzeltebilme kapasitesi kadar hatanın

da belirlenip düzeltilmesi sağlanmaktadır. Kod çözme işlemleri, iterasyon sayısı oldukça fazla işlemler grubudur. Bu anlamda kodlama ile kod çözme süreleri arasında sayısal olarak ciddi bir fark vardır. Ama her bir kodlama yönteminin kendi arasında kodlama/kod çözme süreleri ve artış oranındaki sıralamasının aynı olduğu görülmüştür. Çizelge 4.4'teki değerler, kodlama sürelerinde kullanılan on beş ölçümün kod çözme ortalamalarını vermektedir. Şekil 4.5'te kod çözme süreleri ortalamaları ve şekil 4.6'da kod çözme süresi artış oranı grafikleri görülmektedir.



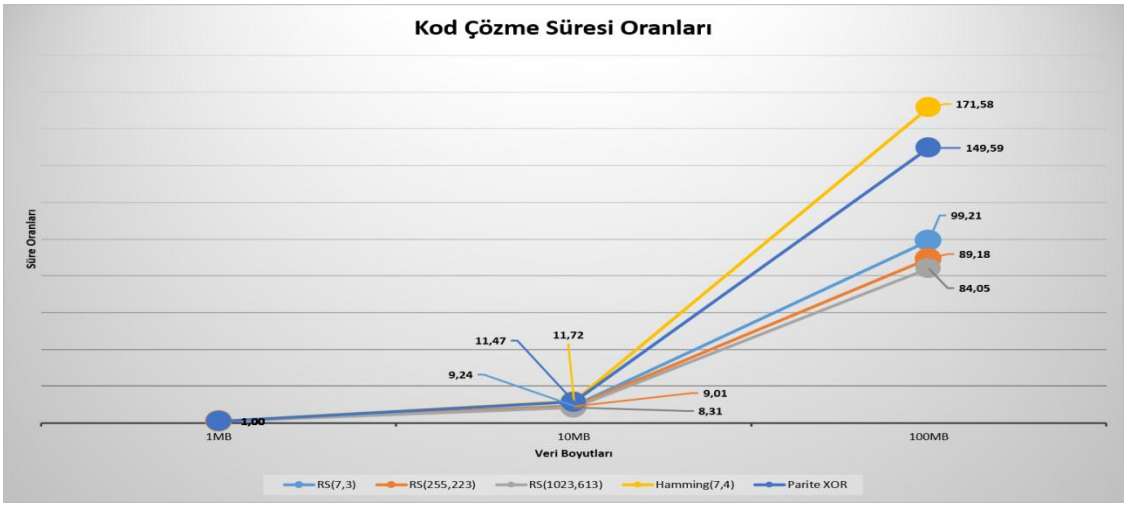
Şekil 4.5 Kod çözme süreleri

Çizelge 4.4'te göre kod çözme süreleri ortalamaları aşağıdaki gibidir.

Hamming(7,4) > Parite-XOR > RS(7,3) > RS(255,223) > RS(1023,613)

Çizelge 4.4 Kod çözme süreleri

Silinti Kod Çözme Süreleri (Saniye)	1 MB	10 MB	100 MB
RS(7,3)	0,359	3,317	35,591
RS(255,223)	0,634	5,714	56,568
RS(1023,613)	16,288	135,393	1368,952
Hamming(7,4)	0,0788	0,923	13,514
Parite-XOR	0,096	1,103	14,397



Şekil 4.6 Kod çözme süresi artış oranları

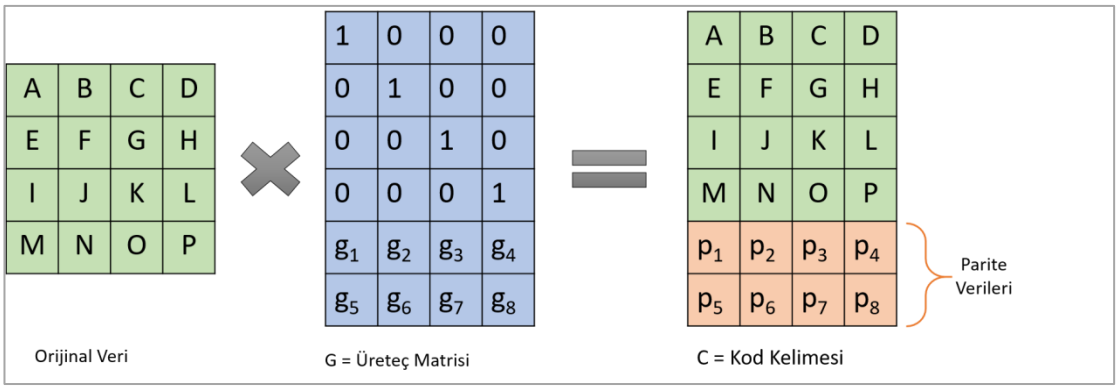
Tüm veri boyutlarına göre kod çözme sürelerindeki artış oranları her bir ölçümde sıralamaları aynı çıkmakla birlikte aşağıdaki gibidir.

RS(1023,613) > RS(7,3) > RS(255,223) > Parite-XOR > Hamming(7,4)

4.2 Veri Artış Oranı ve Verimlilik

Silinti kodlaması ile kodlanacak olan orijinal verinin boyutu artmaktadır. Bir silinti kodlamada temel olarak orijinal verinin kodlama yöntemine göre oluşturulan üreteç matrisi hesaplanabilir. Üreteç matrisi ile orijinal verinin çarpılması ile elde edilen kod sözcüğü, kodlanan veri olarak adlandırılmıştır. Kodlanan veride orijinal verinin tamamına ek olarak da parite verileri de bulunmaktadır.

Parite verilerinin boyutlarının fazla olması fiziksel depolama alanlarında fazladan yer işgal etmesine, ağ üzerinde fazladan trafik oluşturmasına ve verinin işlenmesinde fazladan işlem gücü harcanmasına neden olmaktadır. Fiziksel depolama alanlarının günümüzde pahalı çözümler ve sürekli büyüyen yapılar olduğu göz önüne alınırsa depolama alanlarının en verimli şekilde kullanılması gerekliliği ortaya çıkacaktır. Bu nedenden dolayı kodlama yöntemlerinde veri artış oranı kodlama yönteminin verimliliğini ortaya koymaktadır.



Şekil 4.7 Silinti kodlama - parite verileri

4.2.1 Veri artış oranı

Veri artış oranı (VAO) verinin boyutuna göre değil, kodlama yönteminin blok ve parite uzunluğuna (p_u), göre değişmektedir. Her bir bloğun parite uzunluğu, kod sözcüğünün blok uzunluğunun (n), orijinal verinin sembol uzunluğunun (k) farkına eşittir. Buna dair ifade eşitlik (4.2)'de verilmiştir.

$$p_u = n - k \tag{4.2}$$

VAO ise parite uzunluğunun (p_u), orijinal verinin sembol uzunluğuna (k) oranını ifade eder. Yüzdesel olarak VAO, eşitlik (4.3)'te belirtilmiştir.

$$VAO = \frac{p_u}{k} \times 100 \tag{4.3}$$

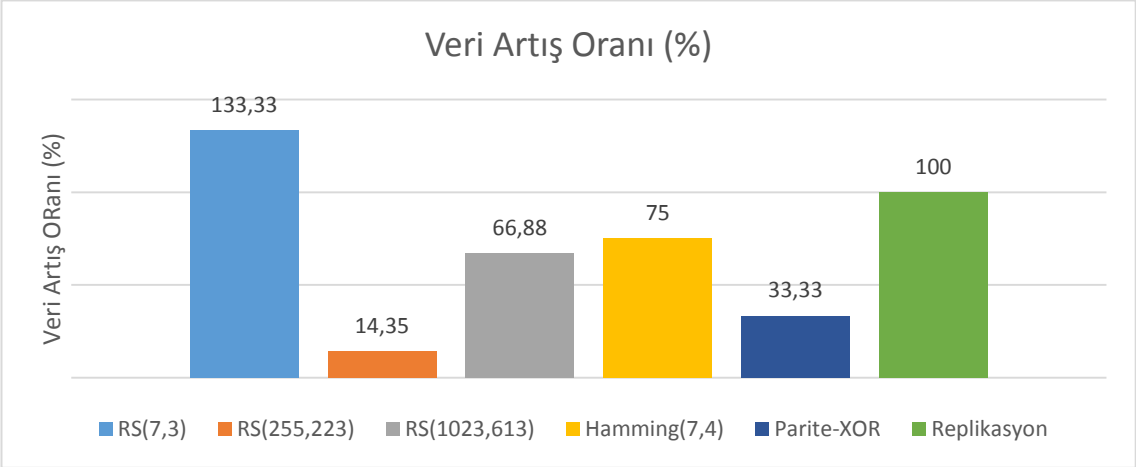
Veriye erişilememe veya herhangi bir nedenden dolayı verinin bozulması fiziksel depolama alanlarının maliyetinden çok daha önemli olmaktadır. Bu nedenden dolayı verinin yedeklenmesi gerekliliği ortaya çıkmaktadır. Veriyi yedekleyebilmek için geleneksel olarak replikasyon yöntemi kullanılmaktadır. Replikasyon yönteminde verinin birebir kopyası aynı ağda farklı bir depolama biriminde tutulmaktadır. Merkezi verinin bozulması veya veriye erişilememesi durumunda kopyalanan verinin

kullanılması sağlanmaktadır. Böylece iş sürekliliği minimum kesintiyle sağlanmış olmaktadır.

Silinti kodlamanın temel amaçlarından biri, kullanılan verinin hata düzeltebilme özelliğidir. Hata düzeltebilme özelliği sayesinde verinin birebir kopyasının fiziksel depolama alanında tutulmasına gerek kalmamaktadır. Kodlama yöntemlerinin mimari yapılarına göre bozulan verinin kurtarılması ve yeniden kullanılabilmesi mümkündür.

Silinti kodlama yöntemlerinde ortaya çıkan parite veriler, orijinal verinin boyutunun artmasına, fiziksel depolama alanında daha fazla yer kaplamasına neden olmaktadır. Buna rağmen silinti kodlamaların geleneksel yöntemlerde kullanılan replikasyon yönteminin belli bir oranda yerini tuttuğu söylenebilmektedir. Optimal bir DVDS'nin oluşturulması için geleneksel sistemlerin karşılaştırılması bu noktada önemli bir konu olmaktadır. Şekil 4.8'deki gibi ele alınan kodlama yöntemleri ve replikasyon yönteminin VAO değerleri hesaplanmıştır. Buna göre büyükten küçüğe sırasıyla VAO değerleri aşağıdaki gibidir:

RS(7,3) > Replikasyon > Hamming (7,4) > RS(1023,613) > Parite-XOR > RS(255,35)



Şekil 4.8 Veri artış oranı–kodlama yöntemleri ve replikasyon

Replikasyon yönteminde performansa etki edecek verinin okuma/yazma süreleri dışında herhangi bir etken yoktur. Kodlama yöntemlerinde ise okuma/yazma sürelerine ek olarak kodlama/kod çözme süreleri de performansa etki etmektedir. Replikasyonun veri artış oranı bir eşik değer olarak kabul edilirse DVDS'lerde kullanılması gereken optimal kodlama yönteminin bu eşik değerden daha fazla olmaması beklenebilmektedir. VAO'su en yüksek yöntem %133,33 ile RS(7,3) kodlamasıdır. Bu değer replikasyon yönteminden bile daha fazla verinin fiziksel depolama alanında yer işgal etmesine neden olmaktadır. Diğer kodlama yöntemleri ise belirlenen eşik değerin altında VAO'ya sahiptir.

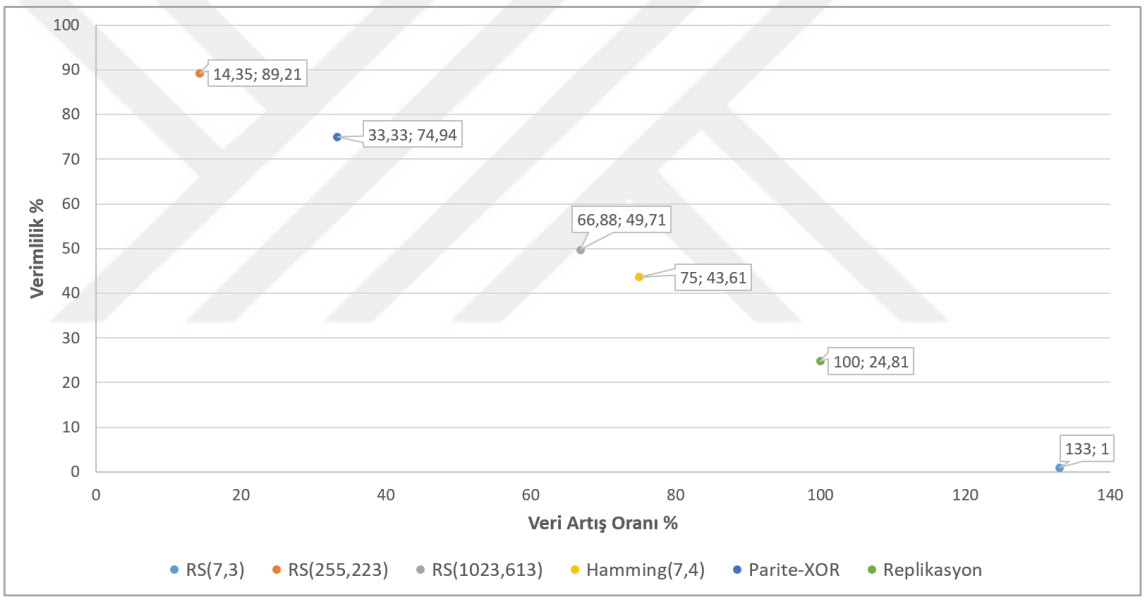
VAO ile blok uzunluğu arasında doğrusal bir orantı olmasına rağmen aynı blok uzunluğuna sahip RS(7,3) ile Hamming(7,4) kodlamaları arasında yaklaşık %68'lik bir fark vardır. Bu fark kodlama yöntemlerinin algoritmik yapılarından kaynaklı oluşmaktadır. RS(7,3) kodlaması 2 hata düzeltebilen bir kodlama yöntemi iken Hamming(7,4) tek hata düzeltebilen bir kodlama yöntemidir.

Reed-Solomon kodlamaları arasında ise en düşük VAO, %14,35 ile RS(255,223) kodlamasıdır. Blok uzunlukları ile arasında doğrusal bir ilişki bulunmasına rağmen elde edilen sonuçların düzensiz olduğu gözlenmiştir. Aynı algoritmaya sahip Reed-Solomon kodlamaları arasındaki bu farklılıklar lineer kodlamalarda elde edilen üreteç matrislerinin (G) boyutları ile ilgilidir. Üreteç matrislerinin boyutları hata düzeltebilme kapasitelerine göre üretilmiş olup VAO değerlerine etki etmektedir.

Bu doğrultuda orijinal verinin uzunluğuna bağlı olarak farklı boyutlarda üreteç matrisleri elde edilmektedir. Bu üreteç matrislerinin boyutları ise parite verilerin boyutlarını belirlemektedir. Parite verilerin uzunluğu veri artış oranındaki farklılıklara neden olmaktadır.

4.2.2 Verimlilik

Bir kodlama yönteminin verimliliği, fiziksel depolama alanının birim maliyetine göre kullanılan alan olarak tanımlanabilmektedir. Veri artış oranına bağlı olarak kodlamanın fiziksel depolama alanındaki kullanım miktarına göre kodlama yönteminin verimliliğinden bahsedilebilmektedir. En az bir hata düzeltebilen bir silinti kodlama yönteminde veri artış oranının olmadığı kodlama yönteminin verimliliği %100 olarak düşünülebilir. Verimlilik değerinin en düşük değeri ise veri artış oranı en yüksek olan RS(7,3) kodlamasına göre hesaplanmıştır. Buna göre elde edilen verimlilik değerleri şekil 4.9'daki gibi elde edilmektedir.



Şekil 4.9 Verimlilik

RS(7,3) kodlamasının veri artış oranı %133,33 olduğu için verimlilik değeri sıfır olarak alınmaktadır. Replikasyon yönteminde tüm veri birebir çoğaltılırken veri artış oranı %100'dür. Replikasyon yönteminde fiziksel depolama alanı kullanımına göre verimlilik ifadesinden bahsedilememektedir. Buna rağmen replikasyon yöntemi RS(7,3) kodlama yöntemine göre daha verimli bir yöntem olduğu söylenebilir.

4.3 Hata Düzeltme Oranı

Silinti kodlamalarının en temel özelliği hata düzeltebilme yeteneğidir. Veri iletişiminde sıkça ve oldukça başarılı uygulamaları olmakla beraber DVDS'leri için de bir avantaj sağlamaktadır.

MDS bir kodlamada hata düzeltme kapasitesi aşağıdaki eşitlikler ile ifade edilmektedir.

$$n = 2^m - 1 \tag{4.4}$$

$$k = 2^m - 1 - 2t \tag{4.5}$$

$$t = \frac{n-k}{2} \tag{4.6}$$

Hata düzeltebilme kapasitesi temel olarak parite düğüm uzunluğunun yarısına eşittir. Düğüm boyutunun artması hata düzeltebilme kapasitesini arttıracaktır. Fakat bu durumda kodlama yönteminin fiziksel depolama alanı üstünde kapladığı yer, verimliliği düşürecektir. Hata düzeltme kapasitesinden ziyade, her bir kodlama yönteminin kendi içinde hata düzeltme oranının ifade edilmesi daha doğru bir yaklaşım olacaktır. Bu konuda da iki farklı yaklaşım söz konusudur. Bunlardan birincisi; parçalarına ayrılan ve düğümlere dağıtılan verinin bütünlüğünün, düğümlerin doğru ve hatasız çalışmasına bağlı olduğunu ifade eden yaklaşımdır. Diğer bir deyişle düğüm içerisindeki veri bütünlüğünün bozulmasına değil de ağ üzerindeki düğümün erişilemez olması veya düğümün bozulmasını dikkate alarak ölçümlenen hata düzeltme oranı ile ilgilidir. İkincisi ise; her bir düğüm üzerindeki verinin doğruluğunu kontrol ederek bir düğüm üzerindeki oluşabilecek hataların düzelme oranıdır.

Bu tez kapsamında oluşturulan sistem modelinde hata düzeltme oranının her iki yaklaşımı da kullanılmıştır. MKS, düğüm erişimini kontrol eder, eğer hata düzeltme oranından daha fazla sayıda düğüme erişilemiyorsa dağıtılmış olan veri elde edilememektedir. Fakat hata düzeltme oranından daha az sayıda düğüme erişilemiyorsa,

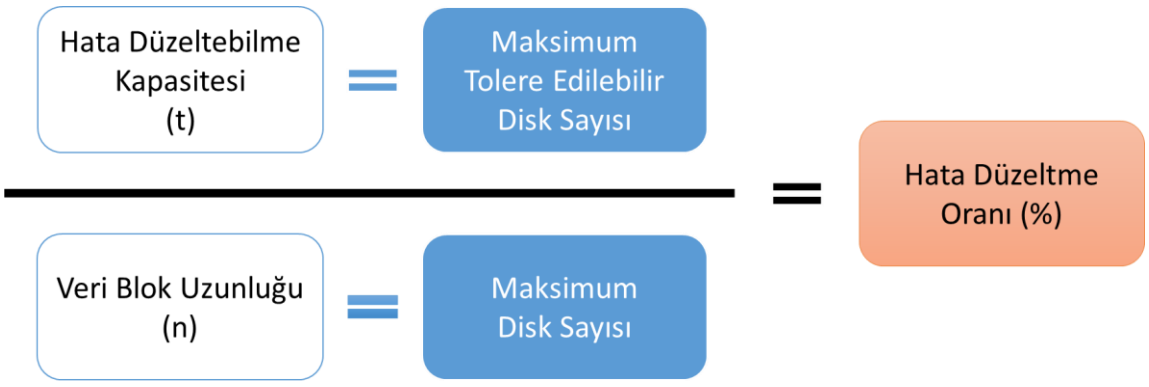
erişilemeyen düğümdeki verilerin yeniden elde edilmesi sağlanarak verinin kod çözülmesi gerçekleştirilir. Diğer yaklaşım için ise, UVS'de bulunan dağıtılmış olan verinin hata kontrolü ve hata düzeltmesi, hata düzeltme oranında gerçekleştirilmektedir. Bu sayede her bir düğüm üzerinde verinin sağlıklı bir şekilde saklanması ve kullanılması sağlanmaktadır.

Buna göre, hata düzeltme oranları her iki yaklaşım için de aşağıdaki gibi ifade edilebilmektedir.

$$\text{Tolere Edilebilir Düğüm Oranı (\%)} = \frac{\text{Tolere edilebilir Düğüm Sayısı}}{\text{Toplam Düğüm Sayısı}} \times 100 \tag{4.7}$$

$$\text{Verinin Hata Düzeltme Oranı (\%)} = \frac{\text{Hata Düzeltebilme Kapasitesi (t)}}{\text{Veri Blok Uzunluğu (n)}} \times 100 \tag{4.8}$$

İki farklı yaklaşım olsa bile ele alınan sistem modelinde maksimum düğüm sayısı veri blok uzunluğuna ve maksimum tolere edilebilir düğüm sayısı da hata düzeltebilme kapasitesine eşitlenmiştir. Verinin blok boyutuna ve fiziksel ortamın esnekliğine göre toplam disk sayısı değişkenlik gösterebilmekte olup tolere edilebilir düğüm sayısı da buna göre etkilenmektedir.



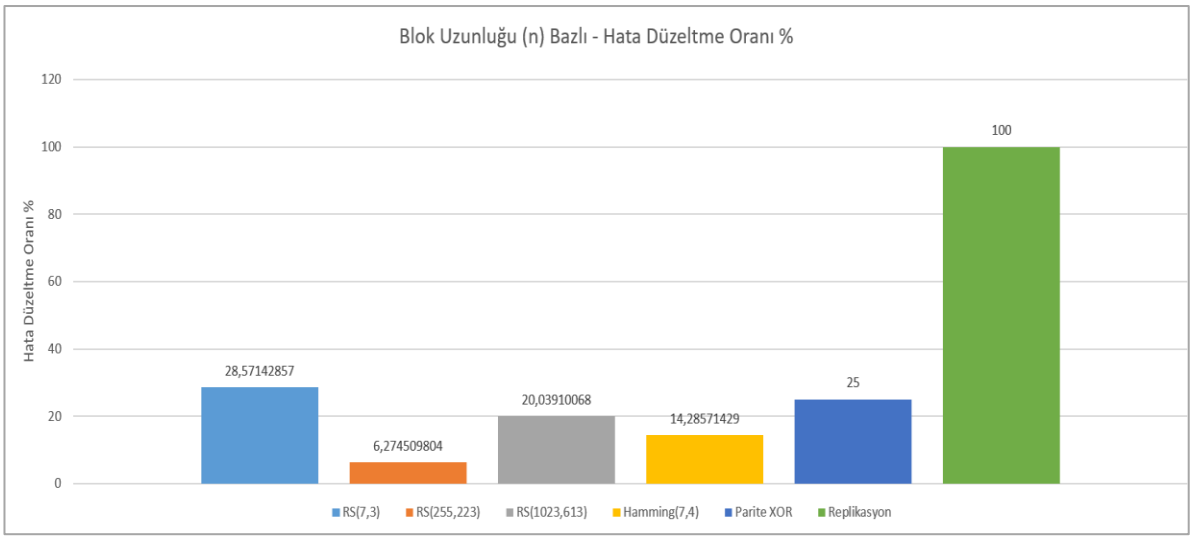
Şekil 4.10 Hata düzeltme oranı

Kodlanacak olan veri, düğüm sayısına göre parçalarına ayrılarak kodlanmaktadır. Hata düzeltme oranı; okuma, yazma veya ağ üzerinde taşınma sırasında oluşan hataların düzeltilebilme oranını ifade etmektedir.

Replikasyon yönteminde hata düzeltme yönteminden bahsedemeyiz fakat verinin bozulma veya erişilememe anında, verinin tümünü yeniden sağlayabildiği için hata düzeltme oranı kıyaslamasında bulunulabilmektedir. Bu doğrultuda replikasyon yönteminde hata düzeltme oranı, düğüm sayısının 1 olduğu ortamda %100 olarak ifade edilebilmektedir. Kodlama yöntemlerinde ise durum biraz daha farklı olmaktadır. Çünkü düğüm sayısı, kodlama yöntemlerine ve hata düzeltme oranlarına göre farklılıklar gösterebilmektedir. Düğüm sayısındaki değişkenliklere rağmen ele alınan tüm kodlama yöntemlerinin hata düzeltme oranları sabittir.

Kodlama yöntemleri arasındaki farklılıkların incelenmesi sonucunda hata düzeltme oranları sırasıyla aşağıdaki gibidir:

RS(7,3) > Parite-XOR > RS(1023,613) > Hamming(7,4) > RS(255,223)



Şekil 4.11 Hata düzeltme oranı

Hata düzeltme oranının yüksek olması verinin yedekliliğini ve dağıtık veri sisteminin esnekliğini sağlamaktadır. Replikasyon yöntemiyle verinin birebir düzeltilmesi sağlanırken silinti kodlama yöntemlerinde hata düzeltme oranın %20'den düşük olmaması beklenmektedir. Ele alınan sistem modeline göre RS(255,213) ve Hamming(7,4) kodlamalarının hata düzeltme oranlarının %20'den düşük olduğu görülmüştür. Şekil 4.11 incelendiğinde diğer kodlama yöntemlerinde ise bu oranın %20 ile %29 aralığında olduğu görülmektedir.

4.4 Genişletilebilirlik

Genişletilebilirlik kriteri DVDS'lerinde aktif olarak düğüm sayısının ne kadar arttırılabileceği değerini ifade etmektedir. Veri üzerinde kodlama, kod çözme, veri iletimi gibi işlemler devam ederken kesintisiz bir şekilde, veri etkilenmeden düğüm sayısı arttırılabilmektedir. Bu sayede DVDS'lerinin dinamik bir şekilde büyüyebilmeleri kolaylıkla sağlanabilmektedir.

Genişletilebilirlik, MIDS değeri ile MADS değeri arasındaki kullanılabilir düğüm sayısı aralığını ifade eder. MIDS değeri her bir kodlama yönteminin blok uzunluğunun hata düzeltebilme kapasitesine oranına göre hesaplanmaktadır. Bu durum eşitlik (4.9) ile ifade edilmektedir. MADS değeri ise kodlama yönteminin blok uzunluğuna (n) eşittir.

$$\text{Minumun Düğüm Sayısı } = \frac{n}{t} \tag{4.9}$$

Bu çalışma kapsamında her bir kodlama yöntemine göre kullanılabilir düğüm sayısı aralığı yani genişletilebilirlik değerleri incelenmiştir. Buna göre çizelge 4.5'teki değerler elde edilmiştir.

Çizelge 4.5 Kullanılabilir düğüm aralığı

Silinti Kodlama Yöntemleri	MIDS	MADS
RS(7,3)	4	7
RS(255,223)	16	255
RS(1023,613)	5	1023
Hamming(7,4)	7	7
Parite-XOR	4	4

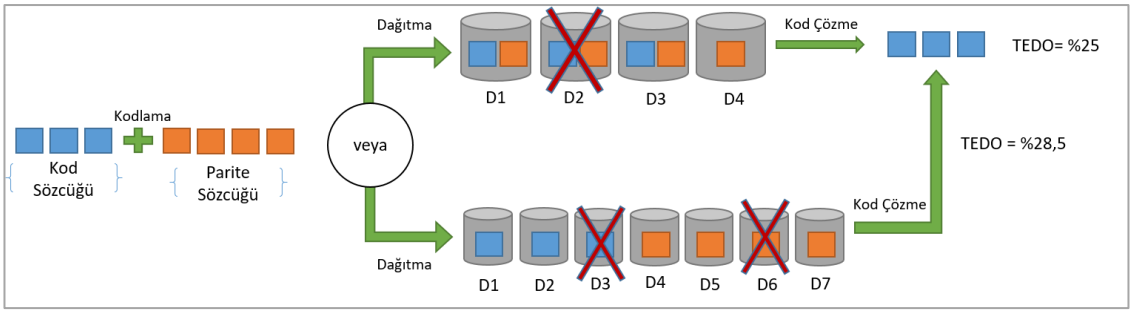
Çizelge 4.5'ten yararlanılarak MKS üzerinde düğüm sayısı belirlenir. Düğüm sayısı MIDS ve MADS aralığında olmak zorundadır. Buna göre kodlanan verinin parçalanarak dağıtılması işlemi yapılmaktadır.

RS kodlama yöntemlerinde kullanılabilir düğüm aralığı tanımlanabilirken, Hamming(7,4) ve Parite-XOR kodlama yöntemlerinde kullanılabilir düğüm sayısı sabittir. Bunun nedeni ise her iki kodlama yönteminde de hata düzeltme kapasitesinin (t) 1 olmasıdır. RS kodlama yöntemleri ise birden fazla hata düzeltebilen kodlama yöntemleridir. Buna göre genişletilebilirlik kavramı hata düzeltme kapasitesi değerine bağlı olarak açıklanabilir.

Kodlanan veri, blok uzunluğu (n) kadar parçalarına ayrılırsa her bir düğüme birer blok dağıtılabilir. Yani MADS değeri kadar düğüm sayısı olan bir ortamda her bir düğüm kod verisinin bir bloğuna sahiptir. Hata kapasitesi sabit olduğuna göre tolere edilebilir düğüm sayısı t kadardır. MIDS belirlenirken tek düğüm hatanın tolere edilebilir olması gerekmektedir. Bu doğrultuda bloklar, blok uzunluğunun (n) hata düzeltme kapasitesine (t) oranına göre dağıtılmaktadır. Düğüm sayısı arttıkça hata düzeltme oranına bağlı olarak tolere edilebilir düğüm sayısı sabit değildir. Buna göre tolere edilebilir düğüm oranı niceliği için her bir kodlama yönteminin kendi içinde farklılıkları söz konusudur.

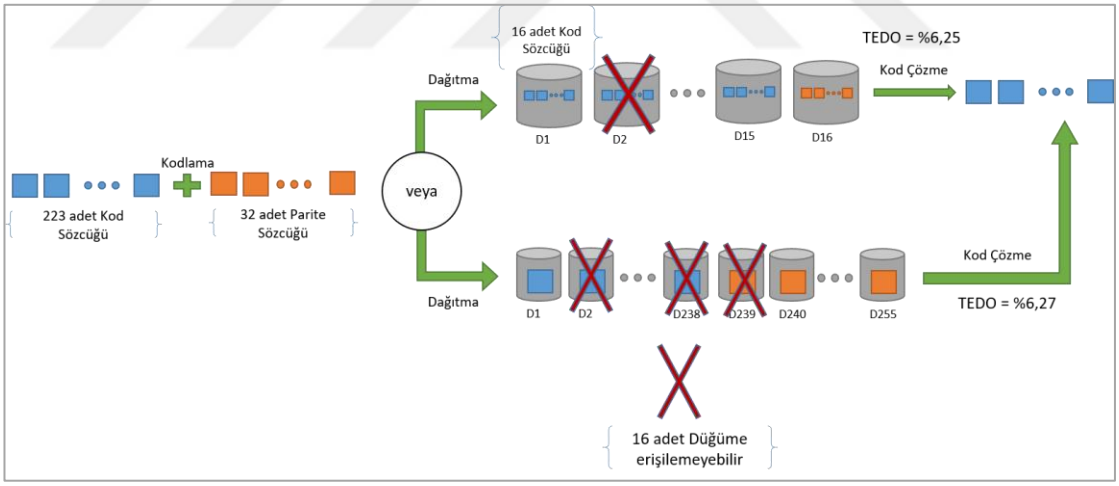
Ele alınan kodlama yöntemleri tek tek incelendiğinde aşağıdaki sonuçlar elde edilmiştir:

- RS(7,3) kodlamasında; genişletilebilirlik 4 ile 7 düğüm arasında gerçekleştirilebilir. Tolere edilebilir düğüm sayısı ise minimum 1, maksimum 2'dir (Şekil 4.12).



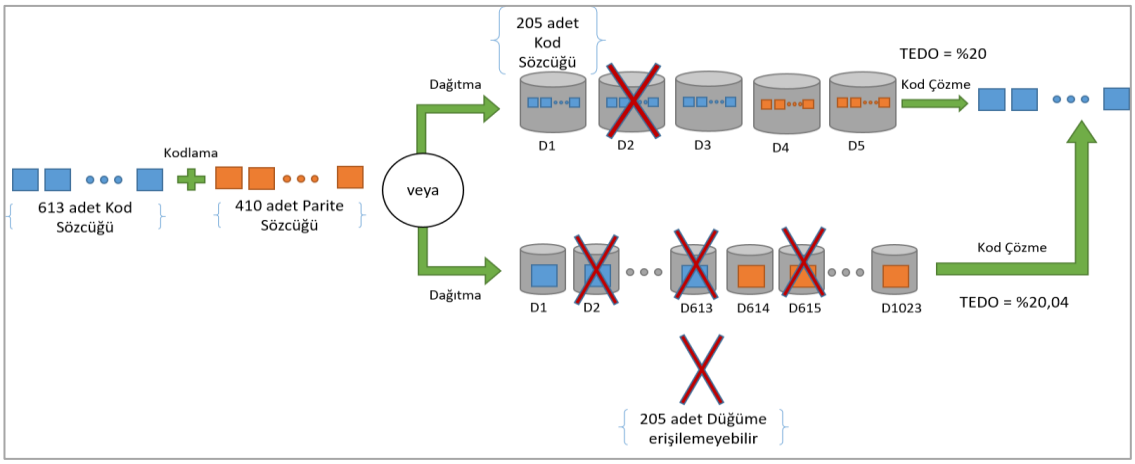
Şekil 4.12 RS (7,3) genişletilebilirlik

- RS(255,223) kodlamasında; genişletilebilirlik 16 ile 255 düğüm arasında gerçekleştirilebilir. Tolere edilebilir düğüm sayısı ise minimum 1, maksimum 16'dır (Şekil 4.13).



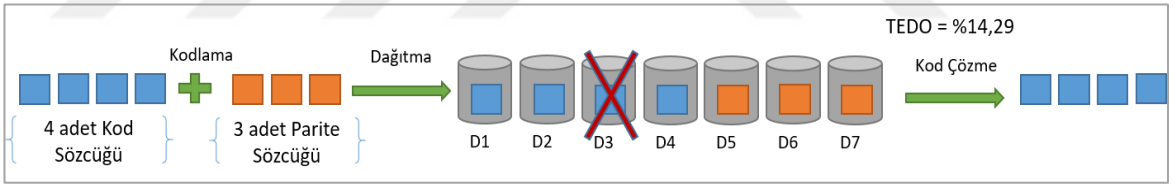
Şekil 4.13 RS(255,223) genişletilebilirlik

- RS(1023,613) kodlamasında; genişletilebilirlik 5 ile 1023 düğüm arasında gerçekleştirilebilir. Tolere edilebilir düğüm sayısı ise minimum 1, maksimum 255'tir (Şekil 4.14).



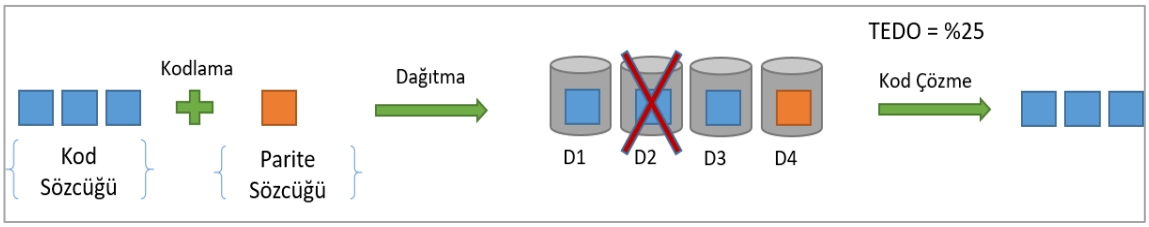
Şekil 4.14 RS(1023,613) genişletilebilirlik

- Hamming(7,4) kodlamasında; genişletilebilirlik durumundan bahsedilememektedir. Hata düzeltme oranı 1 olduğu için ancak 7 düğüm oluşturulabilir. Buna göre de tolere edilebilir düğüm sayısı 1 olarak elde edilir (Şekil 4.15).



Şekil 4.15 Hamming(7,4) genişletilebilirlik

- Parite-XOR kodlamasında da genişletilebilirlik durumundan bahsedilememektedir. Hata düzeltme oranı 1 olduğu için ancak 4 düğüm oluşturulabilir. Buna göre de tolere edilebilir düğüm sayısı yine 1 olarak elde edilir (Şekil 4.16).



Şekil 4.16 Parite-XOR genişletilebilirlik

Genişletilebilirlik özelliği sayesinde, sistemde herhangi bir mimari değişikliğe veya verinin yeniden yapılandırılmasına ihtiyaç duyulmamaktadır. Böylece, fiziksel depolama alanı ihtiyacına bağlı olarak kesintisiz bir şekilde düğüm sayısı arttırılabilmektedir.

4.5 Verinin Taşınabilme Kapasitesi

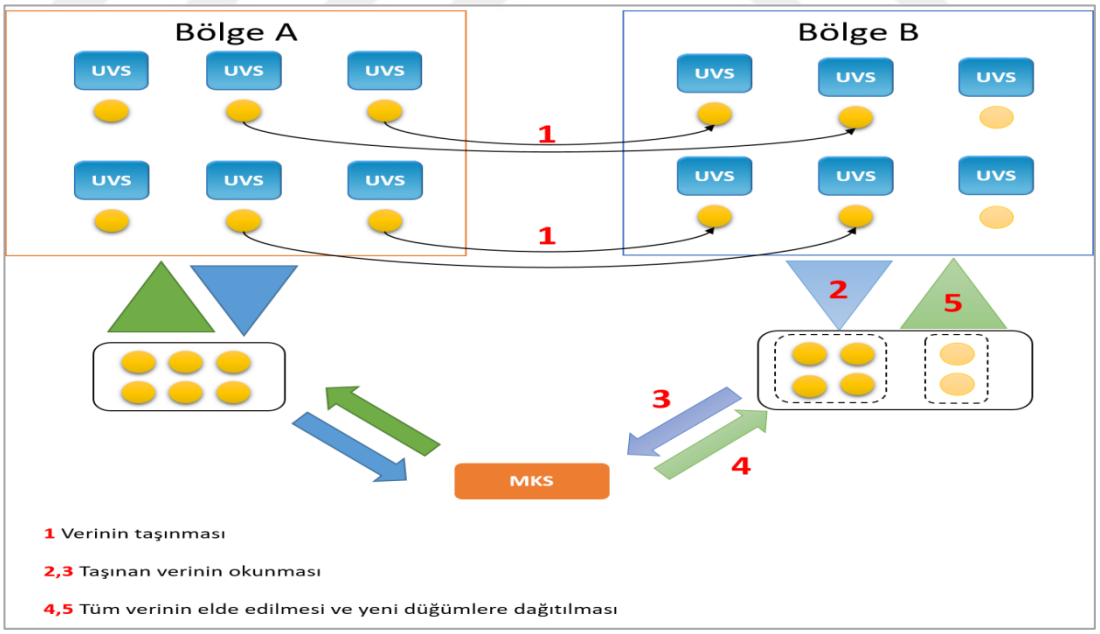
Veri boyutunun kullanıma bağlı olarak artmasıyla birlikte, ortamdaki fiziksel depolama alanlarının kapasitelerinin arttırılması söz konusu olmaktadır. Bununla birlikte ana verinin yedeklenmesi, felaket anında başka bir coğrafi bölgede hizmet vermeye devam edebilmesi için replikasyonu ve fiziksel olarak başka bir donanım veya lokasyona taşınabilmesi gibi operasyonel aktiviteler gerekli durumlarda kullanılmaktadır. Bu aktiviteler, bu tarz kullanımlar için tasarımı yapılmamış ağ topolojilerinde çok fazla darboğaz oluşmasına, verinin kesintiye uğramasına, harcanan fiziksel ve teknolojik eforun oldukça yüksek olmasına neden olmaktadır. Bunun yerine ele alınan DVDS'lerde silinti kodlama yöntemlerinin veri taşınabilme kapasiteleri mevcuttur. DVDS'ler ağ üzerinde birden çok düğüm noktası kullanarak fiziksel depolama alanı çözümü sunduğu için ağ üzerinde verinin taşınması sırasında herhangi bir darboğaz oluşması söz konusu olmamaktadır.

Silinti kodlama yöntemlerinin veri taşınabilme kapasitesi, tolere edilebilir düğüm oranına (TEDO) göre hesaplanmaktadır. Veri taşınabilmesindeki temel amaç; kesintisiz, minimum efor ve yüksek performans ile verinin bir düğümden başka bir düğüme taşınabilmesidir. Veri taşınabilme kapasitesi, ana veriyi yeniden elde edebilmek için ne kadar düğüm sayısına ihtiyaç olduğunu belirtmektedir. Buna göre hatalı veya ağ

üzerinde erişilemeyen düğüm sayısına göre tüm verinin yeniden elde edilebilmesi oranını ifade etmektedir.

$$\text{Veri Taşınabilme Kapasitesi (\%)} = \frac{\text{Düğüm Sayısı} - \text{Tolere Edilebilir Düğüm Sayısı}}{\text{Toplam Düğüm Sayısı}} \times 100 \tag{4.10}$$

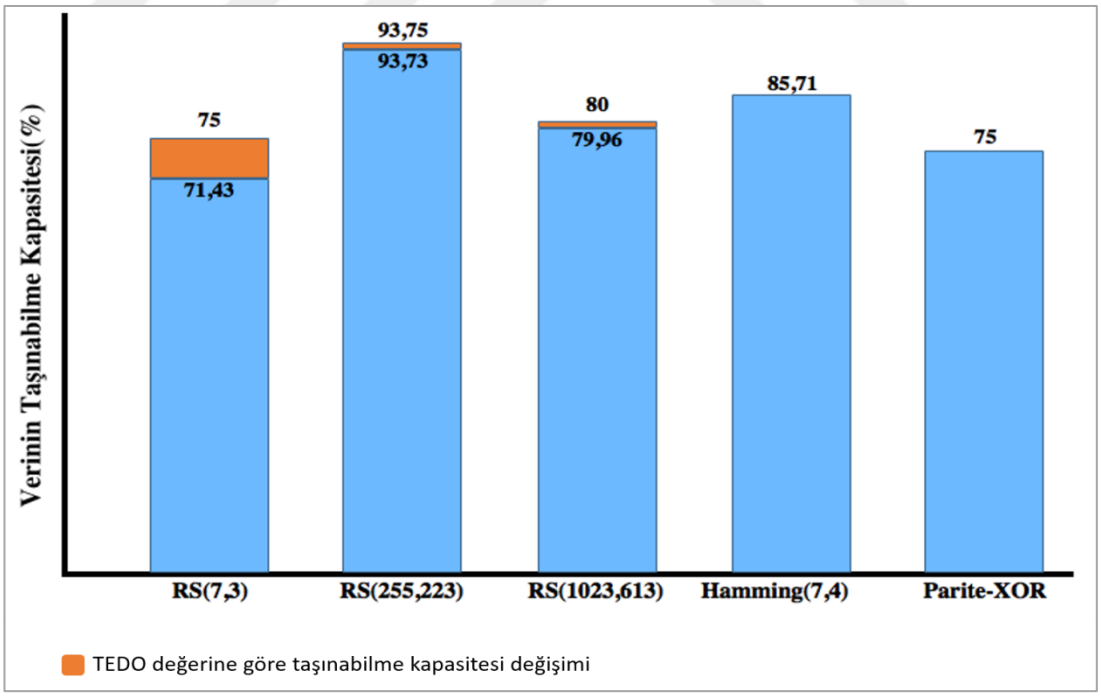
Şekil 4.17'deki örnekte gösterildiği gibi, hatalı düğüm düzeltme oranı %33,33 olan bir kodlama yönteminde taşınabilme kapasitesi %66,33'tür. Başka bir deyişle, tüm veri 6 düğüme dağıtıldığı zaman herhangi 4 düğümün taşınması, tüm verinin yeniden elde edilebilmesi için yeterli olacaktır. Şekil 4.18'deki gibi verinin tamamının Bölge A'dan Bölge B'ye taşınması durumunda sadece 4 düğüm noktasının Bölge B'ye taşınması yeterli olacaktır. MKS üzerinden Bölge B'den okunan 4 düğüm noktasındaki veriler ile verinin tamamı oluşturulabilir. Böylelikle ağ üzerindeki kullanım ve performans açısından kazanım elde edilebilmektedir.



Şekil 4.17 Kodlanan verinin taşınabilme kapasitesi

Şekil 4.18'deki grafik incelendiğinde, ele alınan kodlama yöntemlerinde en etkili taşımanın yapılabildiği kodlama yönteminin RS(7,3) kodlama yöntemi olduğu; bunun yanı sıra RS(255,223) kodlama yönteminin ise verinin taşınması sırasında en fazla efor ve kaynak kullanımını gerçekleştirecek olan kodlama yöntemi olduğu görülmektedir.

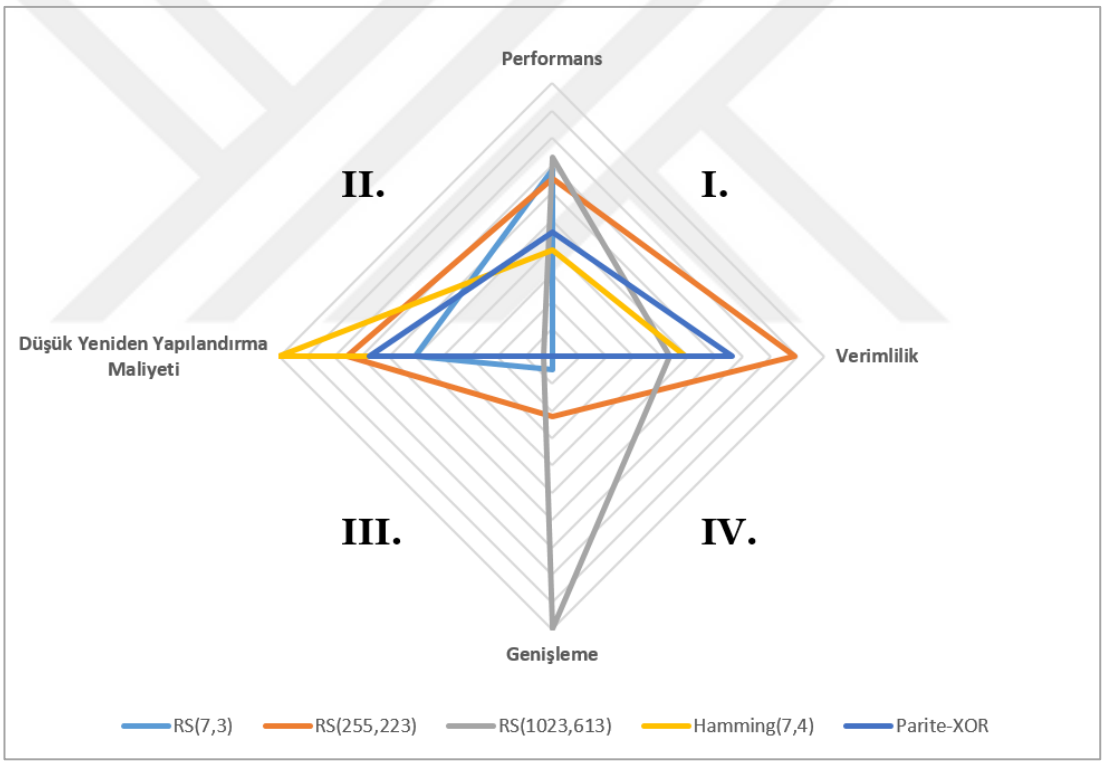
Ortamdaki düğüm sayısının arttırılabilmesi, genişletilebilirlik kavramı ile açıklanmıştır. Buna göre DVDS'lerde düğüm sayısına bağlı olarak verinin taşınabilirlik değerlerinde farklılıklar oluşabilmektedir. RS(7,3) kodlamasında %3,57, RS(255,223) kodlamasında %0,02, RS(1023,613) kodlamasında ise %0,04 lük bir değişim söz konusudur. Bu değişim oranları sadece Reed-Solomon kodlamalarında ortaya çıkmış olup, Hamming(7,4) ve Parite-XOR kodlamalarında bu durumdan söz edilememektedir. Bu tez kapsamında ele alınan kodlama yöntemleri arasındaki değişim oranları, verinin taşınabilirliğine ciddi oranda etki etmemektedir. Buna rağmen düğüm sayısına bağlı olarak elde edilen farklar kodlama yöntemlerindeki iyileştirmeler ile taşınabilirlik değerinin geliştirilebileceğini ortaya koymaktadır.



Şekil 4.18 Verinin taşınabilme kapasitesi

4.6 Kodlama Yöntemleri Spektrumu

Silinti kodlama yöntemi belirlenirken; yeniden yapılandırma maliyeti, verimlilik, performans, genişleme spektrumuna göre ele alınması gerekmektedir. Silinti kodlama yöntemleri ile oluşturulan DVDS'lerin tüm kriterleri aynı ölçüde karşılaması beklenememektedir. Bu anlamda ele alınan kodlama yöntemleri arasında optimal kodlama yöntemini belirleyebilmek için incelenen beş bulguya göre şekil 4.19'da gösterildiği gibi bir spektrum oluşturulmuştur. Bu kriterler; kodlama/kod çözme performansı, veri artış oranı, hata düzeltme oranı, genişletilebilirlik ve verinin taşınabilme kapasitesi olarak belirlenmiştir.



Şekil 4.19 Kodlama yöntemleri spektrumu

Her bir kodlama yönteminde, kıyaslanan kriterlere göre farklı sonuçlar ortaya çıkmıştır. Kriterleri etkileyen değişkenler, kodlama türlerinin algoritmasına, veri boyutlarına ve blok uzunluklarındaki farklılıklara göre etkilemektedir.

Performans açısından bakıldığında önerilen sistem modelinde kodlama türüne göre; verinin kodlanması, düğümlere ayrılması, verinin dağıtılması, dağıtılan verinin merkezi sunucuda okunması, hata kontrolünün yapılıp kod çözülmesi ile orijinal veriye ulaşılması işlemlerinin hepsi fiziksel donanım üzerinde bir yük getirmektedir. Bu yükün herhangi bir darboğaz oluşturmadan belirli bir süre içerisinde işlemleri yapabilme kapasitesi performans olarak adlandırılmaktadır. Performans değeri, süre ve yapılan işlemlerin boyutuna bağlı olarak değişmektedir.

Genişleme değeri, kodlama yöntemleri için hesaplanan genişletilebilirlik değer aralığına bağlı olarak belirlenmektedir. Kodlama yöntemlerinde aktif çalışılabilen düğüm sayısı arttıkça genişleme değeri artmaktadır.

Yeniden yapılandırma maliyeti, kullanılan kodlama yöntemine göre gerekli olan fiziksel donanım maliyetini belirtmektedir. Verinin kodlandığı ilk adımdan, tekrar elde edildiği son adıma kadar, gerekli olan işlem gücüne sahip fiziksel donanım kapasitesi göz önüne alınarak bir yaklaşımda bulunulmuştur.

Verimlilik, fiziksel depolama alanında birim depolama maliyetine göre kullanılan alanın değerini belirlemektedir. Buna göre fiziksel depolama alanındaki her 100 birimlik kullanım, 1 birim maliyet olarak ifade edilirse, kodlama yöntemleri arasında verimlilik kıyaslaması yapılabilir.

Şekil 4.19'da gösterilen spektrumdaki bölgelere göre kodlama yöntemleri incelenirse;

4.6.1 RS(7,3) bulguları

RS(7,3) kodlaması ikinci ve üçüncü bölgelerde bulunmaktadır. Bu kodlama yönteminin değerleri verimlilik kriterlerini sağlamamaktadır. Veri artış değerinden dolayı verimli bir kodlama yöntemi olmadığı sonucu ortaya çıkmaktadır.

Reed-Solomon kodlama yöntemleri arasında en hızlı kodlama/kod çözme sürelerine sahiptir; bu da performans değerini arttırmaktadır. Fakat veri boyutuna bağlı olarak değişen kodlama süresi artış oranı, performans süresini negatif olarak etkilemektedir. Aynı zamanda veri boyutunun artmasına bağlı olarak daha yüksek kapasiteli fiziksel donanıma ihtiyaç olduğunu göstermektedir.

RS(7,3)'ün genişleme değeri, üçüncü bölgede bulunan diğer kodlama yöntemlerine göre en düşük değeri vermektedir. Genişleme değerine bağlı olarak kullanılabilecek fiziksel depolama alanının maliyeti daha az olacaktır.

4.6.2 RS(255,223) bulguları

RS(255,223) kodlaması tüm bölgelerde bulunan bir kodlama yöntemidir.

Veri artış oranının düşük, dolayısı ile de verimlilik değerinin en yüksek olduğu kodlama yöntemidir. Kullanılan verinin fiziksel depolama alanında en ucuz maliyetle saklanmasına olanak sağlamaktadır.

Blok uzunluğuna bağlı olarak kodlama/kod çözme süreleri yavaştır fakat veri boyutunun artması ile birlikte kodlama süresindeki artış oranı düşmektedir. Bu da daha az maliyetle performansın arttırılmasına imkan sağlamaktadır.

RS(255,223)'te işlenen veri boyutuna bağlı olarak düğüm sayısının arttırılabilmesi mümkündür. Bu anlamda genişleme değeri en iyi ikinci kodlama yöntemidir.

4.6.3 RS(1023,613) bulguları

RS(1023,613) kodlaması tüm bölgelerde bulunan diğer bir Reed-Solomon kodlama yöntemidir. RS(255,223) kodlama yönteminin özelliklerine sahip olmakla birlikte bazı dezavantajlara sahiptir.

Kodlama/kod çözme süreleri en yavaş olan yöntemdir. Buna rağmen veri boyutunun artmasına bağlı olarak büyük boyutlu verilerde en iyi performansa sahip olabilecek bir kodlama yöntemidir. Fakat büyük boyutlu veriler 1023 blok uzunluğuna göre kodlanacağı için, yüksek işlemci ve yüksek kapasiteli fiziksel veri depolama donanımlara ihtiyaç olacaktır. Bu da performansın artmasına bağlı olarak maliyetlerin de artmasına neden olacaktır.

RS(1023,613), genişleme değerinin en yüksek olduğu kodlama yöntemidir. Düğüm sayısının çalışabileceği aralığın geniş olması, bu kodlama yönteminin yüksek boyutlu verilerle çalışabilmesine imkan tanımaktadır.

4.6.4 Hamming(7,4) bulguları

Hamming(7,4) kodlaması birinci ve ikinci bölgede bulunmaktadır. Bu kodlama yönteminin değerleri genişleme kriterini sağlamamaktadır.

Reed-Solomon kodlamalarına göre daha hızlı kodlama/kod çözme sürelerine sahip olması performans değerini arttırmaktadır. Fakat veri boyutuna bağlı olarak değişen kodlama süresi artış oranı, performans süresini negatif olarak etkilemektedir. Aynı zamanda veri boyutunun artmasına bağlı olarak daha yüksek kapasiteli fiziksel donanıma ihtiyacı olduğunu göstermektedir.

Hamming(7,4), verimlilik değeri birinci bölgede bulunan diğer kodlama yöntemlerine göre en düşük değeri vermektedir. Verilerin işlenmesinde düşük verimle kodlanan veriler, fiziksel depolama alanındaki kullanımlarına bağlı olarak yüksek maliyet oluşturmaktadır.

4.6.5 Parite-XOR bulguları

Parite-XOR kodlaması birinci ve ikinci bölgede bulunmaktadır. Bu kodlama yönteminin değerleri genişleme kriterini sağlamamaktadır.

Tüm kodlama yöntemlerine göre daha hızlı kodlama/kod çözme sürelerine sahip olması performans değerini arttırmaktadır. Fakat veri boyutuna bağlı olarak değişen kodlama süresi artış oranı performans süresini negatif olarak etkilemektedir. Aynı zamanda veri boyutunun artmasına bağlı olarak daha yüksek kapasiteli fiziksel donanıma ihtiyacı olduğunu göstermektedir.

Parite-XOR'un verimlilik değeri birinci bölgede bulunan RS(255,223) kodlamasından sonra en yüksek değeri vermektedir. Bu da kullanılan verinin fiziksel depolama alanında ucuz maliyetle saklanmasına olanak sağlamaktadır.

5. TARTIŞMA VE SONUÇ

Bu tez çalışması kapsamında DVDS'lerde kullanılabilecek silinti kodlama yöntemleri için bazı kriterler belirlenerek bu kriterler doğrultusunda optimal bir DVDS'nin oluşturulması için sonuçlar incelenmiştir. Bulguların elde edilebilmesi ve kriterlerin değerlendirilebilmesi için bir sistem modeli önerilmiş olup bu sistem modelinde tüm kodlama yöntemlerinin farklı veri boyutlarında çalıştırılması sağlanmıştır.

Kodlama yöntemlerinin kırılımlarına göre kıyaslanması üç aşamada yapılmıştır. Birinci aşamada; Reed-Solomon kodlamaları kendi aralarında ele alınmıştır. İkinci aşamada ise, Reed-Solomon kodlamaları, MDS kodlamanın başka bir türü olan Hamming(7,4) kodlaması ile kıyaslanmıştır. Özellikle RS(7,3) kodlaması ile Hamming(7,4) kodlamalarındaki blok uzunluklarının aynı olması, Reed-Solomon ve Hamming kodlamalarında kullanılan algoritmaların farklılıklarını ortaya koymuştur. Üçüncü aşamada ise, her bir kriter için Parite dizi kodlamasının bir çeşidi olan ve sıkça kullanılan Parite-XOR kodlaması diğer kodlama yöntemleri ile kıyaslanmıştır.

Elde edilen bulgular sonucunda, her bir kodlama yöntemi için en etkili kullanım senaryolarında tasarlanabilecek bir DVDS oluşturulabileceği sonucu çıkarılmıştır.

RS(7,3):

- Küçük veri boyutlarının orta ölçekli performans/maliyet kriterlerine göre işlenmesinde,
- Düğüm sayısının çok fazla artmasına gerek olmayan uygulama veya sistemlerde,
- Hata düzeltme oranı en yüksek kodlama olduğu için kritik verilerin işlenmesinde de kullanılabilir.

RS(255,223):

- Büyük veri boyutlarının orta ölçekli performans/maliyet kriterlerine göre işlenmesinde kullanılabilir.
- Düğüm sayısının dinamik bir şekilde artmasına ihtiyaç olan uygulama veya sistemlerde uygulanabilir.
- Hata düzeltme oranına bağlı olarak kritik seviyeli verilerin işlenmesinden daha çok iş kaybına neden olmayacak uygulamalarda kullanılabilir.
- Bu kodlama yönteminin kullanımına en uygun olarak, yedekleme sistemlerindeki veriler örnek olarak gösterilebilir.

RS(1023,613):

- Büyük veri boyutlarında, yüksek maliyetlerle en iyi performans değerleri elde edilebilir.
- Düğüm sayısının dinamik bir şekilde artmasına ihtiyaç olan uygulama veya sistemlerde kullanılabilir.
- Hata düzeltme oranına bağlı olarak kritik seviyeli verilerin işlenmesinden daha çok iş kaybına neden olmayacak uygulamalarda kullanılabilir.
- Bu kodlama yönteminin kullanımına en uygun olarak dosya sunucuları, yapılandırılmamış veri (unstructured data) yapıları örnek olarak gösterilebilir.

Hamming(7,4):

- Küçük veri boyutlarının düşük performans/maliyet kriterlerine göre işlenmesinde kullanılabilir.
- Düğüm sayısının sabit olarak tanımlandığı uygulama veya sistemlerde kullanılabilir.
- Hata düzeltme oranından dolayı kritik olmayan uygulamalarda kullanılabilir.

Parite-XOR:

- Küçük veri boyutlarının düşük performans/maliyet kriterlerine göre işlenmesinde kullanılabilir.
- Düğüm sayısının sabit olarak tanımlandığı uygulama veya sistemlerde kullanılabilir.
- Hata düzeltme oranından dolayı kritik verilerin kullanıldığı uygulama veya sistemlerde kullanılabilir.

Kodlama yöntemlerinin yanı sıra replikasyon yöntemine de yer verilmiştir. Replikasyon bir kodlama yöntemi değildir. Fakat kodlama yöntemleri ile verinin sağlıklı erişimi, verinin hata anında tekrar erişilebilir olması ve maliyet açılarından kıyaslanabilmesi için ele alınmıştır. Replikasyon yönteminde, veri üzerinde herhangi bir kodlama/kod çözme işlemi yapılmadan birebir çoğaltma ve birleştirme işlemleri yapılır. Bu yöntemde verinin büyüklüğüne ve fiziksel depolama alanının kapasite değerlerine göre verinin yeniden erişilebilirliği değerlendirilmektedir. Ayrıca bu yöntemde verinin dağıtık bir şekilde çalışması mümkün olmamaktadır.

Buna rağmen replikasyon yönteminde verinin birebir bir kopyası bulundurulduğu için verinin yeniden elde edilmesinde herhangi bir limit yoktur. Kodlama yöntemlerinde, bu limitler hata düzeltme oranı ve tolere edilebilir düğüm kapasitesi değerleri ile verilmektedir.

Replikasyon yönteminin, kodlama yöntemlerine göre dezavantajları aşağıdaki gibidir:

- Anlık kesintilere karşı verinin korunmasını sağlamamaktadır.
- Veri, replikasyon sırasındaki bir hatadan kalıcı olarak etkilenebilir, bu da saklanan verinin bozulmasına, bir daha kullanılamamasına neden olmaktadır.
- İhtiyaca bağlı olarak kesintisiz bir şekilde düğüm sayısın arttırılması mümkün olmamaktadır.
- Fiziksel depolama alanının kullanım maliyetinin yüksek olması bir dezavantaj olarak ortaya çıkmaktadır.

Sonuç olarak her bir kodlama yönteminin farklı kriterlere bağlı olarak birbirleri arasında üstünlükleri olduğu sonucu çıkarılmıştır. Bu doğrultuda optimal DVDS için Huang vd. (2012) çalışmasında belirtilen bölgelerde olacak bir silinti kodlama yönteminin geliştirilmesi hedeflenebilir (Şekil 2.1- 2.2). Bunun yanı sıra kullanım senaryolarına veya ihtiyaçlara bağlı olarak esnek bir DVDS oluşturulabilir.

DVDS'lerin günümüz teknolojilerine göre esneklik sağlayan ve hızla büyümeyi kolaylaştıran özelliği ise hibrid olarak çalıştırılabilmeleridir. Hibrid sistemler aynı ortamda işlenecek olan verinin türüne ve boyutuna göre farklı silinti kodlama yöntemleri ile kodlanarak hedeflenen kriterlerde başarı sağlanmasını, dolayısı ile kaynakların en etkili şekilde kullanılmasını sağlamaktadır. Silinti kodlama ile geliştirilen DVDS'ler son kullanıcı ve uygulama özelinde farklı politikalar üretilmesine, böylelikle de yazılım tabanlı depolama sistemlerinin oluşturulmasına olanak sağlamaktadır.

Hedeflenen değerlerde bir DVDS oluşturabilmek için bu tez çalışmasının sonucuna göre hiyerarşik veri depolama sistemlerinin yanısıra hibrid veri depolama sistemleri üzerine çalışmaların yapılabileceği ortaya çıkmıştır.

KAYNAKLAR

Abd-El-Malek, M., Courtright, W.V.II, Cranor, C., Ganger, G.R., Hendricks, J., Klosterman, A. J., Mesnier, M., Prasad, M., Salmon, B., Sambasivan, R. R., Sinnamohideen, S., Strunk, J. D., Thereska, E., Wachs, M. and Wylie, J. J. 2005. Ursa minor: Versatile cluster-based storage. Proceedings of the USENIX Conference on "File and Storage Technologies" (FAST). 13-16 December 2005; San Francisco, California, USA

Anonymous. 2010. Web Sitesi: https://developers.google.com/storage/docs/durable-reduced-availability, Erişim Tarihi: 29.11.2017.

Anonymous. 2012. Web Sitesi: http://quantcast.github.io/qfs, Erişim Tarihi: 12.11.2017.

Blaum, M., Brady, J., Bruck, J. and Menon, J. 1995. EVENODD: An efficient scheme for tolerating double disk failures in RAID architectures. IEEE Transactions on Computers, 44(2); 192–202.

Cao, P., Lin, S.B., Venkataraman S. and Wilkes, J. 1994. The tickertaip parallel raid architecture. ACM Transactions on Computer Systems (TOCS), 12(3); 236-269.

Chapman, H. 1996. Coding Theory. St Edmundsbury Pres, 12-105, Great Britian.

Chien, R.T. 1964. Cyclic decoding procedure for the Bose-Chaudhuri Hocquenghem codes. IEEE Transactions on Information Theory, 10(4); 357–363

Dean, J. and Ghemawat, S. 2008. MapReduce: Simplified Data Processing on Large Clusters. Communications of the ACM, 51(1), 107-113.

Forney, G.Jr. 1965. On Decoding BCH Codes. IEEE Transactions on Information Theory, 11(4); 549–557.

Ghemawat, S., Gobioff, H. and Leung, S. T. 2003. The Google file system. Proceedings of the nineteenth ACM Symposium on Operating Systems Principles, 37(5); 29-43

Gibson, G.A., Nagle, D.F., Amiri, K., Butler, J., Chang, F. W., Gobioff, H., Hardin, C., Riedel, E., Rochberg, D. and Zelenka, J. 1998. A cost-effective, high-bandwidth storage architecture. Special Interest Group on Programming Languages (SIGPLAN) of ACM, 33(11); 92-103.

Haeberlen, A., Mislove, A. and Druschel, P. 2005. Glacier: Highly durable, decentralized storage despite massive correlated failures. Proceedings of the Conference on "Symposium on Networked Systems Design & Implementation" (NSDI). 02-04 May 2005. ACM-DL No: 2, (143-158); Berkeley, CA, USA.

Hafner, J.L. 2006. Hover erasure codes for disk arrays. International Conference on "Dependable Systems and Networks" (DSN), 25-28 June 2006; Philadelphia, PA, USA.

Hagmann, R. 1987. Reimplementing the cedar file system using logging and group commit. Proceedings of the ACM Symposium on Operating Systems Principles (SOSP). 08 - 11 November 1987; Austin, TX, US.

Hamming, R.W. 1950. Error detecting and error correcting codes. Bell System technical journal, 29(2); 147-160.

Hartman, J.H. and Ousterhout, J.K. 1995. The zebra striped network file system. ACM Transactions on Computer Systems (TOCS), 13(3); 274-310.

Haytaoğlu, E. ve Dalkılıç, M.E. 2015. Dağıtık depolama sistemleri için tamir ve yapılandırım üzerine bir çalışma. Doktora Tezi. Ege Üniversitesi, Fen Bilimleri Enstitüsü, İzmir.

Huang C., Simitci H., Xu Y., Ogus A., Calder B., Gopalan P., Li J. and Yekhanin S. 2012. Erasure coding in windows azure storage. Proceedings of the Usenix Annual Technical Conference (ATC). 13-15 June 2012; Boston, MA, USA.

Khan, O., Burns, R., Plank, J., Pierce, W. and Huang, C. 2012. Rethinking Erasure Codes for Cloud File Systems: Minimizing I / O for Recovery and Degraded Reads. 10th USENIX Conference on "File and Storage. Technologies" (FAST '12). 14-17 February 2012; San Jose, CA, USA.

Kubiatowicz, J., Bindel, D., Chen, Y., Czerwinski, S., Eaton, P., Geels, D., Gummadi, R., Rhea, S., Weatherspoon, H. and Weimer, W. 2000. Oceanstore: An architecture for global-scale persistent storage. ACM Special Interest Group on Programming Languages (SIGPLAN) Notices, 35(11); 190-201.

Liskov, B., Ghemawat, S., Gruber, R., Johnson, P., Shrira, L. and Williams, M. 1991. Replication in the harp file system. Proceedings of the ACM Symposium on "Operating Systems Principles (SOSP)", 13 - 16 October 1991; Pacific Grove, CA, USA.

Li, M. and Lee, P.P.C. 2014. Stair codes: A general family of erasure codes for tolerating device and sector failures in practical storage systems. Proceedings of the USENIX Conference on "File and Storage Technologies" (FAST), 17-20 February 2014; Santa Clara, CA, USA.

Moreira, J.C. and Farrell, P.G. 2006. Essentials of Error-Control Coding, John Wiley&Sons, Ltd., 357, West Sussex.

Muralidhar, S., Lloyd, W., California, S., Roy, S., Hill, C., Lin, E., Liu, W., Pan, S., Shankar, S., Sivakumar, V., Tang, L. and Kumar, S. 2014. f4: Facebook's Warm BLOB Storage System. 11th USENIX Symposium on "Operating Systems Design and Implementation", 06-08 October 2014; Broomfield, CO, USA.

Nelson, M.N., Welch, B.B. and Ousterhout, J.K. 1988. Caching in the sprite network file system. ACM Transaction on Computer Systems (TOCS), 6(1); 134-154.

Patterson, D.A., Gibson, G.A. and Katz, R.H. 1988. The Case for Redundant Arrays of

Inexpensive Disks (RAID). Proceedings ACM Special Interest Group on Management of Data (SIGMOD) Conference, 01-03 June 1988; Chicago, IL, USA.

Plank J.S. 1997. A tutorial on reed-solomon coding for fault-tolerance in raid-like systems. Software: Practice and Experience, 27(9); 995-1012.

Plank, J.S. 2005. Erasure Codes for Storage Applications. 4th USENIX Conference on "File and Storage Technologies", 13-16 December 2005; San Francisco, CA, USA.

Plank, J.S. 2008. The RAID-6 liberation codes. Proceedings of the 6th USENIX Conference on "File and Storage Technologies" (FAST), 26-29 February 2008; San Jose, CA, USA

Plank, J.S. 2013. Tutorial on Erasure Coding for Storage Applications. 11th USENIX Conference on "File and Storage Technologies", 12-15 February 2013; San Jose, CA, USA.

Rabin, M.O. 1989. Efficient Dispersal of Information for Security, Load Balancing, and Fault Tolerance. Journal of ACM, 36(2); 335–348.

Rashmi, K., Shah, N. B., Gu, D., Kuang, H., Borthakur, D. and Ramchandran, K. 2014. A hitchhikers guide to fast and efficient data reconstruction in erasure-coded data centers. Proceedings of the ACM conference on "Special Interest Group on Data Communications (SIGCOMM)", 17-22 August 2014; Chicago, IL, USA.

Reed, S. and Solomon, G. 1960. Polynomial codes over certain finite fields. Journal of the Society for Industrial and Applied Mathematics, 8(2); 300-304.

Roman S. 1992. Coding and Information Theory, Graduate Text in Mathematics, Springer Verlag, No: 134, 488, New York.

Satyanarayanan, M., Kistler, J.J., Kumar P., Okasaki, M.E., Siegel, E.H. and Steere, D. C. 1990. Coda: A highly available file system for a distributed workstation environment. IEEE Transactions on Computers, 39(4); 447-459.

Sobe, P. and Peter, K. 2008. Flexible parameterization of XOR based codes for distributed storage. Proceedings of the 7th IEEE International Symposium on "Networking Computing and Applications (NCA)", 10-12 July; Cambridge, Massachusetts, USA.

Şen, C., Yeşil, S. ve Kolağasıoğlu, E. 2016. Karma-OTİ Sistemler için Silinti Düzelten Reed Solomon Kodların FPGA Gerçeklemesi. The Institute of Electrical and Electronics Engineers, 2(4); 16-31.

Tanenbaum, A.S. and Steen, M.V. 2007. Distrubeted Systems Principles and Paragdigms.Pearson Prentice Hall, 669, Amsterdam.

Trappe, W. and Washington, L.C. 2002. Introduction to Cryptography with Coding

Theory, Prentice Hall, 571, Dallas.

Turner, V., Gantz, J.F., Reinsel, D. and Minton, S. 2014. The Digital Universe of Opportunities: Rich Data and the Increasing Value of the Internet of Things. IDC IDC Analyze the Future; 9.

Wang, Y. 2014. MDR codes: A new class of RAID-6 codes with optimal rebuilding and encoding. IEEE Journal on Selected Areas in Communications, 32(5); 1008-1018.

Wang, Z., Dimakis, A. and Bruck, J. 2010. Rebuilding for array codes in distributed storage systems. Proceedings of Global Telecommunications (GLOBECOM) Workshops, 06-10 December 2010; Miami, Florida, USA.

Weil, S.A., Brandt, S.A., Miller, E.L., Long, D.D.E. and Maltzahn, C. 2006. Ceph: A scalable, high-performance distributed file system. Proceedings of Symposium on "Operating Systems Design and Implementation (OSDI)", 06 – 08 November 2006; Seattle, WA, USA.

Wilkes, J., Golding, R., Staelin, C. and Sullivan, T. 1996. The hp autoraid hierarchical storage system. ACM Transactions on "Computer Systems (TOCS)", 14(1); 108-136.

ÖZGEÇMİŞ

Adı Soyadı : Burak BEZİRCİ

Doğum Yeri : Elazığ

Doğum Tarihi : 02.01.1988

Medeni Hali : Evli

Yabancı Dili : İngilizce

Eğitim Durumu

Lise **:** Bursa Cumhuriyet Lisesi (2005)

Lisans **:** Ankara Üniversitesi Mühendislik Fakültesi Elektronik Mühendisliği Bölümü (2010)

Yüksek Lisans **:** Ankara Üniversitesi Fen Bilimleri Enstitüsü Elektrik-Elektronik Mühendisliği Anabilim Dalı (2018)

Çalıştığı Kurum/Kurumlar ve Yıl

VMware Turkey (2014---)

Koç Sistem (2009)

<u>Ulusal Kongre Sunum</u>

Bezirci B., Yılmaz A. E., Metinlerin Okunabilirliğinin Ölçülmesi Üzerine Bir Yazılım Kütüphanesi ve Türkçe için Yeni Bir Okunabilirlik Ölçütü, DEÜ Mühendislik Fakültesi Fen Bilimleri Dergisi (SİU 2010 Seçilmiş Bildiriler Özel Sayısı), cilt 12, sayı 3, s. 49-62, Ekim 2010. (ULAKBİM)

Bezirci B., Yılmaz A. E., Türkçe İçin Yeni Bir Okunabilirlik Ölçütü Önerisi, IEEE 18. Sinyal İşleme ve İletişim Uygulamaları Kurultayı (SİU 2010), s. 368-371, 22-24 Nisan 2010, Diyarbakır, Türkiye.

Bezirci B., Yılmaz A. E., İngilizce İçin Geliştirilmiş Okunabilirlik Ölçütlerinin Türkçeye Uyarlanabilirliği Üzerine Bir Çalışma, 3. Mühendislik ve Teknoloji Sempozyumu (MTS), s. 34-42, 29-30 Nisan 2010, Ankara, Türkiye.