



Bilgisayar Mühendisliği:

BLM101: DONEM PROJESİ

Ogrenci No:**22360859226**

Ad ve Soyad:**SANAA LAMMAT**

2025/2026

BİTLER VE BYTE'LAR:

Bilgisayar bilimlerinde "Veri", fiziksel gerçekliğin diskret (ayrık) bir modelidir. Bilgi Teorisi (Information Theory) çerçevesinde veri, bir mesajın belirsizliğini azaltan her şeydir. Mühendislik düzeyinde veri temsili, donanım katmanındaki transistörlerin durumundan (0/1), yazılım katmanındaki nesne yönelimli yapılara kadar uzanan bir soyutlama zinciridir.

- **1.1. İkililik Mantık (Binary Logic):** Boole Cebri üzerine kurulu olan bu sistemde, verinin her bir birimi 2^n tabanında saklanır. Bir mühendis için veri, sadece bir sayı değil, bellekteki bir adresdir.

- **1.2. Kelime Uzunluğu (Word Size):** İşlemci mimarisine göre (32-bit veya 64-bit), verinin işlenme hızı ve bellek adresleme kapasitesi değişir.

METİN VERİSİNİN BİT DÜZEYİNDE KODLANMASI:

Metinler, sembollerin sayısal karşılıklarıyla temsil edilir.

- **2.1. ASCII Standartı:** 7 bitlik bir sistemdir. $2^7 = 128$ farklı karakteri temsil eder. Kontrol karakterleri (Null, ESC, Del) ve yazdırılabilir karakterleri içerir.
- **2.2. Genişletilmiş ASCII:** 8 bit kullanarak 256 karaktere çıkar, ancak bölgесel dil sorunları (Türkçe karakterler gibi) yaratır.
- **2.3. Unicode ve UTF-8:** Günümüz standardıdır. UTF-8, değişken uzunluklu bir kodlamadır. İlk 128 karakteri ASCII ile aynıdır, ancak diğer semboller için 2, 3 veya 4 byte kullanabilir. Bu, "Verimlilik vs. Kapsayıcılık" dengesidir.

METİN VERİSİNİN BİT DÜZEYİNDE KODLANMASI

Metinler, sembollerin sayısal karşılıklarıyla temsil edilir.

2.1. ASCII Standartı: 7 bitlik bir sistemdir. $2^7 = 128$ farklı karakteri temsil eder. Kontrol karakterleri (Null, ESC, Del) ve yazdırılabilir karakterleri içerir.

2.2. Genişletilmiş ASCII: 8 bit kullanarak 256 karaktere çıkar, ancak bölgесel dil sorunları (Türkçe karakterler gibi) yaratır.

2.3. Unicode ve UTF-8: Günümüz standardıdır. UTF-8, değişken uzunluklu bir kodlamadır. İlk 128 karakteri ASCII ile

aynırıdır, ancak diğer semboller için 2, 3 veya 4 byte kullanabilir. Bu, "Verimlilik vs. Kapsayıcılık" dengesidir.

Sayısal Verilerin Temsili (Integer):

Tam sayılar bellekte doğrudan ikilik sistemde saklanır.

- **Unsigned:** Sadece pozitif sayılar.
- **Signed (Two's Complement):** En anlamlı bit (MSB) işaret biti olarak kullanılır. Negatif sayılar, sayının tümleyeninin alınmasıyla ifade edilir. Bu yöntem, toplama ve çıkarma işlemlerinin aynı donanım birimi (ALU) üzerinde yapılmasını sağlar.

GÖRÜNTÜ VERİSİNİN SAYISALLAŞTIRILMASI

Görüntü, bir matris verisidir. $f(x, y)$ fonksiyonu ile ifade edilir.

- **3.1. Raster Modelleme:** Görüntü, piksellerden (picture elements) oluşur.
- **3.2. Renk Derinliği (Bit Depth):** 1-bit (Siyah-Beyaz), 8-bit (256 gri tonu) veya 24-bit (16.7 milyon renk). 24-bit bir sistemde her kanal (R, G, B) 8 bitlik bir işarettsiz tam sayıdır (0-255).
- **3.3. Çözünürlük Hesaplama:** 1920x1080 bir ekranın tek bir karesinin ham boyutu: $1920 \times 1080 \text{ 3byte} = \text{approx } 6 \text{ MB}$ dir.
- Sıkıştırma olmadan saniyede 60 kare (60fps) video izlemek, saniyede 360 MB veri transferi gerektirir.

SES VERİSİ VE ÖRNEKLEME TEOREMİ:

Ses analogu bir dalgadır. Bilgisayar bu dalgayı anlayabilmek için "örnek" almak zorundadır.

- **4.1. Nyquist-Shannon Teoremi:** Bir sinyali hatasız dijitalleştirmek için, en yüksek frekansın en az iki katı hızda örnekleme yapılmalıdır ($f_s \geq 2f_{\text{max}}$). İnsan kulağı 20kHz duyabildiği için CD kalitesi 44.1kHz'dır.
- **4.2. Kuantizasyon Hatası:** Analog değerin en yakın dijital değere yuvarlanması sırasında oluşan gürültüdür. Bit derinliği arttıkça (16-bit'ten 24-bit'e) bu hata azalır.

VERİ SIKISTIRMA NEDEN ZORUNLUDUR?

Sıkıştırma, verinin entropisini (bilgi yoğunluğunu) artırma işlemidir.

1. **I/O Darboğazı (Bottleneck):** Hard disk veya SSD okuma hızları, CPU'nun veri işleme hızından çok yavaştır. Sıkıştırılmış veriyi okuyup RAM'de açmak (decompress), ham veriyi diskten okumaktan daha hızlıdır.
2. **Bant Genişliği Kısıtı:** Fiber optik kabloların bile bir üst sınırı vardır.
3. **Bulut Bilişim Maliyetleri:** AWS, Azure gibi servislerde depolanan veri miktarı doğrudan faturalandırılır.

Veri Sıkıştırmanın Teorik Temelleri:

Sıkıştırma, verideki **fazlalığı (redundancy)** azaltma işlemidir.

- **Spatial Redundancy (Mekansal):** Bir resimdeki yan yana aynı renkli pikseller.
- **Temporal Redundancy (Zamansal):** Videodaki iki ardışık kare arasındaki benzerlik.
- **Spectral Redundancy (Spektral):** Kanallar arası (RGB) korelasyon.

Kayıpsız Sıkıştırma (Lossless Compression):

Girdi verisinin bit-perfect (bit'i bit'ine) geri döndürülebildiği algoritmalarıdır.

- **Kullanım:** Metin belgeleri, veritabanları, çalıştırılabilir dosyalar (.exe).
- **Algoritmalar:** RLE, Huffman, LZW, Arithmetic Coding.

Kayıplı Sıkıştırma (Lossy Compression):

Verinin bir kısmının kalıcı olarak silindiği ancak insan algısının bunu fark etmediği yöntemlerdir.

- **Kullanım:** Fotoğraflar (JPEG), Videolar (MP4), Sesler (MP3).
- **Neden?**
- Sıkıştırma oranı 100:1 gibi çok yüksek seviyelere çıkabilir.

Run-Length Encoding (RLE) Mekanizması:

RLE, ardışık sembollerin uzunluğunu kodlar.

- **Mantık:** Veri dizisindeki AAAAA yapısını 5A olarak saklar.
- **Sembol Çifti:** {Değer, Tekrar Sayısı}. Düşük entropili ve yüksek tekrarlı verilerde (faks dökümanları, basit ikonlar) çok etkilidir.

RLE'nin Sınırları ve "Worst Case" Analizi:

RLE, karmaşık verilerde (gürültülü fotoğraflar gibi) veriyi sıkıştmak yerine büyütür.

- **Örnek:** ABCDE -> 1A1B1C1D1E (Boyut 5'ten 10'a çıkar). Mühendislikte bu durumu önlemek için **Escape Sequences** veya **Flag bits** kullanılır.

Run-Length Encoding (RLE) Uygulama Varyasyonları:

RLE sadece "karakter + sayı" demek değildir. Mühendislikte farklı tipleri vardır:

1. **Bit-level RLE:** Sadece 0 ve 1'lerden oluşan diziler için (Faks makineleri).
 2. **Byte-level RLE:** Standart karakter dizileri için.
 3. **Pixel-level RLE:** RGB değerlerinin (3 byte) bir bütün olarak tekrarını sayar.
- **Problem:** 1111... (256 tane 1) olduğunda, sayı (counter) 8-bit ise taşıma (overflow) olur. Bu durumda veri bölünerek kodlanır.

RLE ve Dosya Formatları: TGA ve BMP Analizi:

Targa (TGA) ve Windows Bitmap (BMP) formatları RLE'yi opsyonel olarak sunar.

- **BMP RLE8:** 8-bitlik indeksli renklerde kullanılır. İki byte'lık modlar içerir: "Encoded Mode" (tekrar eden veri) ve "Absolute Mode" (tekrar etmeyen veri). Absolute mode, RLE'nin dezavantajı olan "boyut büyümemesini" engellemek için kullanılan bir kaçış (escape) mekanizmasıdır.

Huffman Kodlaması :

Karakterlerin kullanım sıklığına göre değişken uzunluklu kodlar atar.

- Sık kullanılan harfler (e, a, t) kısa bit dizileriyle (Örn: 2 bit).
- Nadir harfler (z, q) uzun bit dizileriyle (Örn: 10 bit) temsil edilir. Bu bir "Prefix-free" kodlamadır (Hiçbir kod diğerinin ön eki değildir).

Huffman Ağacı İnşası (Adım Adım):

Bir Huffman ağacı oluşturmak için:

1. Her karakterin frekansını (olasılığını) say.
2. En düşük frekanslı iki düğümü seç ve birleştir.
3. Yeni bir düğüm oluştur (toplam frekans ile).
4. Tek bir kök düğüm kalana kadar devam et.
5. Sola gidişlere '0', sağa gidişlere '1' ata. Bu yöntem, Minimum Redundancy sağlar.

Sözlük Tabanlı Sıkıştırma (LZ77 ve LZW):

Daha önce görülen veri kalıplarını bir sözlükte tutar.

- **LZ77:** Kayan pencere (Sliding Window) kullanır. "Geriye doğru 10 karakter git, 5 karakter kopyala" şeklinde çalışır.
- **LZW:** Dinamik olarak bir kod tablosu oluşturur. GIF formatının temelidir.

LZ77 Mimarisi:

1977 yılında Lempel ve Ziv tarafından geliştirilen bu algoritma, veriyi bir "pencere" üzerinden okur.

- **Sliding Window:** Geçmişte okunan verileri bellekte tutar.
- **Match Finding:** Yeni gelen veri dizisi, geçmiş pencerede varsa, verinin kendisi yerine (mesafe, uzunluk) ikilisi yazılır.
- **Örnek:** "Portakal al" cümlesinde "al" kısmı daha önce geçtiği için tekrar yazılmaz.

LZW (Lempel-Ziv-Welch) ve Dinamik Sözlük:

LZW, önceden bir sözlük gerektirmez; veriyi okurken sözlüğü inşa eder.

- **Kod Tablosu:** Başlangıçta 256 temel ASCII karakteriyle doludur. Yeni diziler bulundukça 257, 258... olarak tabloya eklenir.
- **Uygulama:** GIF ve TIFF formatlarının kalbidir. Ancak patent süreçleri nedeniyle tarihte birçok tartışmaya yol açmıştır.

Video Sıkıştırma: Kareler (Inter-frame):

Video, zaman içindeki görüntü dizisidir. Çoğu kare, bir önceki kareye çok benzer.

- **I-Frame:** Tam kare.
- **P-Frame:** Sadece değişen pikselleri (farkı) tutar.
- **B-Frame:** Hem önceki hem sonraki kareden tahmin yapar.

Videolarda piksellerden ziyade "objelerin hareketi" saklanır.

- **Macroblocks:** Kare 16x16 bloklara bölünür.
- **Motion Vector:** Bir bloğun bir sonraki karede nereye kaydığını hesaplanır. Sadece bu vektör (yön ve mesafe) saklanır. Bu sayede 1 GB'lık video 10 MB'a inebilir.

Ses Sıkıştırma ve Psiko-akustik Modelleme:

MP3 gibi formatlar "İşitsel Maskeleme" (Auditory Masking) kullanır. Eğer çok yüksek bir ses varsa, o andaki çok kısık bir ses insan kulağı tarafından duyulmaz. Sıkıştırma algoritması bu duyulmayan sesi veriden tamamen atar.

MP3 ve AAC formatları **Modified Discrete Cosine Transform (MDCT)** kullanır.

- **Zaman Etki Alanı -> Frekans Etki Alanı:** Ses dalgası frekanslarına ayrılır.
- **Bit Allocation:** Önemli seslere daha çok bit, arka plan gürültüsüne daha az bit atanır.

Veri Güvenliği ve Sıkıştırma İlişkisi:

Sıkıştırma bazen bir güvenlik açığıdır.

- **CRIME Saldırısı:** Sıkıştırılmış verinin boyutuna bakarak içindeki gizli veriyi (Cookie vb.) tahmin etme yöntemidir.
- **Şifreleme sırası:** Veri önce sıkıştırılmalı, sonra şifrelenmelidir. Şifrelenmiş veri (rastgele göründüğü için) sıkıştırılamaz.

Donanım Katmanında Sıkıştırma:

Modern CPU ve GPU'lar, sıkıştırma işlemlerini hızlandırmak için özel komut setlerine (SSE, AVX) sahiptir. SSD kontrolcüleri, veriyi diske yazmadan önce donanım düzeyinde sıkıştırarak diskin ömrünü uzatır ve performansı artırır.

Ağ İletişimi ve Bant Genişliği Tasarrufu:

HTTP/2 ve HTTP/3 protokolleri, başlık (header) sıkıştırması (HPACK) kullanır. Bir web sayfası istenirken gönderilen tekrarlı metin verileri sıkıştırılarak sayfanın yüklenme hızı (FCP - First Contentful Paint) optimize edilir.

Big Data ve Kolon Bazlı Saklama:

Veritabanı mühendisliğinde (Örn: Apache Parquet), veri kolon bazlı saklanır. Bir kolondaki veriler genellikle aynı tipte ve benzer değerlerde olduğu için RLE ve sözlük sıkıştırması burada devasa verimlilik sağlar.

Sıkıştırma Algoritmalarının Karşılaştırılması:

Algoritma	Tür	Avantaj	Dezavantaj
RLE	Lossless	Çok Hızlı, Basit	Düşük Sıkıştırma Oranı
Huffman	Lossless	Optimal Sembol Kodlama	Tablo Oluşturma Maliyeti
LZW	Lossless	Kalıp Bulmada İyi	Bellek Kullanımı Yüksek
JPEG	Lossy	Devasa Alan Tasarrufu	Veri Kaybı (Artifacts)

Sıkıştırma Performans Metrikleri ve Kalite Ölçümü:

Sadece boyut yetmez, kalite de ölçülmelidir:

- PSNR (Peak Signal-to-Noise Ratio):** Sinyal gürültü oranı. Yüksek olması iyidir.
- SSIM (Structural Similarity):** İnsan gözünün algıladığı yapısal benzerlik.
- Sıkıştırma Zamanı:** Gerçek zamanlı sistemlerde (Zoom, Discord) gecikme (latency) en kritik metriktir.

Büyük Veri ve Dosya Sistemleri (ZFS, Btrfs):

Modern dosya sistemleri "Transparent Compression" (Şeffaf Sıkıştırma) yapar.

- İşletim sistemi veriyi diske yazarken arka planda LZ4 veya ZSTD algoritmasıyla sıkıştırır. Kullanıcı bunu fark etmez ama disk kapasitesi %30-40 artmış olur.

Özet:

veri sıkıştırma, modern bilgisayar bilimlerinde verinin entropisini yöneterek donanım kısıtlarını aşma sanatıdır. Bit düzeyinde temsil edilen metin, resim ve ses verileri, içerdikleri gereksiz tekrarlardan (redundancy) arındırılarak depolama maliyetlerini düşürür ve ağ transferlerini hızlandırır. Bu süreçte **RLE** gibi algoritmalarla veriyi kayıpsız bir şekilde "sayarak paketlemek" temel bir yaklaşımken; görsel ve işitsel medyada insan algısının sınırlarını kullanan kayıplı yöntemler, %90'a varan devasa alan tasarrufu sağlar. Günümüzde bulut bilişimde düşük maliyet, IoT cihazlarında uzun pil ömrü ve platformlar arası hız için kritik olan bu teknolojiler, gelecekte yapay zeka destekli nöral sıkıştırma modelleriyle veriyi sadece küçültmeyecek, aynı zamanda daha akıllı ve anlam odaklı bir yapıya kavuşturacaktır.

