



# **Bölüm 5: Dizgi Algoritmaları**

## **Algoritmalar**



# Dizgi Algoritmaları

- Metinlerle dolu bir dünyada yaşıyoruz.
- E-postalar, mesajlar, sosyal medya paylaşımları, haber metinleri...
- Bilgisayarlarımızda her gün sayısız metinle karşılaşırız.
- Peki, bu metinler nasıl düzenlenir ve analiz edilir?
- Dizgi (String) algoritmaları,
  - metinlerde arama,
  - değiştirme,
  - karşılaştırma gibi işlemleri gerçekleştirir.



# Dizgi Eşleştirme Algoritmaları

- Brute Force (Kaba Kuvvet):
  - Metindeki her konum örüntü ile eşleştirmek için kontrol edilir.
  - Maksimum sayıda karşılaştırma gerektirebilir.
- Knuth-Morris-Pratt (KMP)
  - Başlangıçta tablo oluşturularak arama süresi azaltılır,
  - Karakter karşılaştırmalarını azaltarak hızlı çalışır.
- Boyer-Moore
  - Uzun aramalarda etkili. Kök bulma ve kaydırma stratejisi kullanır.
- Rabin-Karp Algoritması
  - Olasılıksal bir algoritma. Hashing kullanır.



# Dizgi Sıkıştırma Algoritmaları

- Sıralı Sıkıştırma Kodlaması (Run Length Encoding)
  - Aynı veri değerleri tek bir değer ve sayı olarak saklanır.
  - Tekrar eden değerler yerine tekrar eden veri sayısı saklanır.
- Lempel-Ziv-Welch (LZW)
  - GIF gibi formatlarda kullanılan sözlük tabanlı sıkıştırma algoritması.
  - Tekrar eden örüntüleri sözlük oluşturarak kısa sembollerle temsil eder.
  - Dinamik bir sözlük kullanarak sıkıştırma sağlar.



# Dizgi Sıralama Algoritmaları

- Sözlüksel Sıralama (Lexicographic Order)
  - Dizgiler, alfabetik sıraya benzer sıralanır.
  - Her karakterin ASCII değeri karşılaştırılarak sıralama yapılır.
- Radix Sıralama
  - Karşılaştırmalı olmayan bir tam sayı sıralama algoritmasıdır.
  - Veriler tamsayı anahtarlarına sahiptir.
  - Aynı konumda aynı değeri paylaşan verileri gruplandırarak sıralar.
  - Her basamak için ayrı ayrı işlem yapılır.



# Dizgi Ayırıştırma (Parsing) Algoritmaları

- Düzenli İfadeler (Regular Expressions)
  - Bir arama örüntüsünü tanımlayan karakter dizisi,
  - Belirli bir örüntüye uyan tüm dizgileri bulmak için kullanılır
- Sonlu Durum Makineleri (Finite State Machines - FSM)
  - Dizgi içindeki örüntüleri tanımak için kullanılan hesaplama modelleri,
  - Belirli bir girdi dizisindeki geçişlerin durumlarını izleyen bir otomat,
  - Karmaşık ayırıştırma ve analiz işlemlerinde kullanılır.



# Dizgi Düzenleme Mesafesi Algoritmaları

- Levenshtein Mesafesi
  - İki dizgi arasındaki benzerliği ölçen bir metrik,
  - Bir dizgiden diğerine dönüştürmek için gereken minimum tek karakterli düzenleme sayısı olarak tanımlanır.
- En Uzun Ortak Alt Dizi (Longest Common Subsequence - LCS)
  - İki dizginin ortak olan en uzun alt dizisi,
  - Karakterlerin sıralı olmasını gerektirmez, ancak sıra korunmalıdır.
  - Dizgiler arasındaki benzerlik veya farkı belirlemek için kullanılır.



# Dizgi Dönüşüm Algoritmaları

- Sonek Dizisi (*Suffix Array*)
  - Bir dizginin tüm son eklerinin bir dizisi.
  - Dizgi içindeki alt dizgilerin bir temsili olarak kullanılır.
- Burrows-Wheeler Dönüşümü (*BWT*)
  - Bir dizginin tersine dönüştürülmesiyle elde edilen yeni bir form,
  - Bzip2 gibi sıkıştırma algoritmaları için ön işlem adımı olarak kullanılır.





# Sonek Dizisi (Suffix Array)

- Bir dizginin tüm soneklerinin alfabetik olarak sıralanmış halidir.
- Her bir sonek, dizginin belirli bir konumundan başlayan bir alt dizisidir.
- Dizgi arama, sıralama, genetik dizilim analizi benzeri işlemlerde kullanılır.



# Oluşturma Yöntemleri

- Brute Force:
  - Tüm sonekleri oluşturup ardından sıralar.
  - $O(n^2 \log n)$  zaman karmaşıklığına sahiptir.
- Manber-Myers Algoritması:
  - Lineer zaman karmaşıklığına sahiptir. ( $O(n \log n)$ )
  - Soneklerin sıralanması sırasında tekrar tekrar harf karşılaştırması yapılmaz.
- Larsson-Sadakane Algoritması:
  - Sıralama işleminde art arda soneklerin karşılaştırılması esas alınır.





# Burrows-Wheeler Dönüşümü (BWT)

- Girdi dizisini yeniden düzenler.
- Aynı karakterlerin bir araya toplanmasını sağlar.
- Sıkıştırma algoritmalarının performansını artırır.
- Tersine çevrilebilir: Orijinal veri, dönüşümden geri elde edilebilir.



# Uygulama Adımları

- Girdi dizgisinin tüm döndürülmüş hallerini oluştur.
- Döndürülmüş dizgileri alfabetik olarak sırala.
- Sıralanmış dizgilerin son karakterlerinden yeni bir dizgi oluştur.



# Örnek

- Girdi:
  - BANANA
- Döndürmeler:
  - BANANA, ANANAB, NANABA, ANABAN, NABANA, ABANAN
- Sıralanmış Döndürmeler:
  - ABANAN, ANABAN, ANANAB, BANANA, NABANA, NANABA
- Son Karakterler:
  - NNBAAA
- BWT Sonucu:
  - NNBAAA



# Burrows-Wheeler

S = abraca

a	b	r	a	c	a
b	r	a	c	a	a
r	a	c	a	a	b
a	c	a	a	b	r
c	a	a	b	r	a
a	a	b	r	a	c

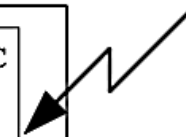
Initial Matrix

S' = caraab

a	a	b	r	a	c
a	b	r	a	c	a
a	c	a	a	b	r
b	r	a	c	a	a
c	a	a	b	r	a
r	a	c	a	a	b

Sorted Matrix

L





SON