Journal of the Faculty of Engineering and Architecture of Gazi University 38:2 (2023) 875-888

Mühendislik Mimarlık Fakültesi Dergisi
Journal of The Faculty of Engineering and Architecture of Gazi University

Elektronik / Online ISSN : 1304 - 4915
Basılı / Printed ISSN : 1300 - 1884

# q-gram hash comparison based multiple exact string matching algorithm for DNA sequences

Abdullah Ammar Karcıoğlu[1], Hasan Bulut[2]*

[1]Department of Software Engineering, Engineering Faculty, Ataturk University, 25240 Erzurum, Türkiye
[2]Department of Computer Engineering, Engineering Faculty, Ege University, 35000 Izmir, Türkiye

**Highlights:**

- Using the perfect hash function to multiple exact string matching in DNA sequences
- *q*-gram hash comparison in multiple exact string matching.
- The proposed approach is more efficient than other multiple exact string matching algorithms.

**Keywords:**

- Multiple exact string matching algorithms
- Pattern matching
- Sequence analysis
- Hash function
- Wu-manber algorithm

**Article Info:**

Research Article
Received: 11.06.2021
Accepted: 16.04.2022

**DOI:**

10.17341/gazimmfd.951157

**Correspondence:**

Author: Hasan Bulut
e-mail: hasan.bulut@ege.edu.tr
phone: +90 232 311 2596

**Graphical/Tabular Abstract**

In this study, if the hash value of the q-length suffix of the text matches, the hash value of the q-length prefix is checked. If the prefix also matches, the first character of the pattern and text after the prefix is compared. If the character matches, the characters between the prefix and the suffix are checked by q-gram hash comparison. When there is a prefix or character mismatch, the q unit patterns are shifted to the right. The process of the proposed approach is shown in Figure A.

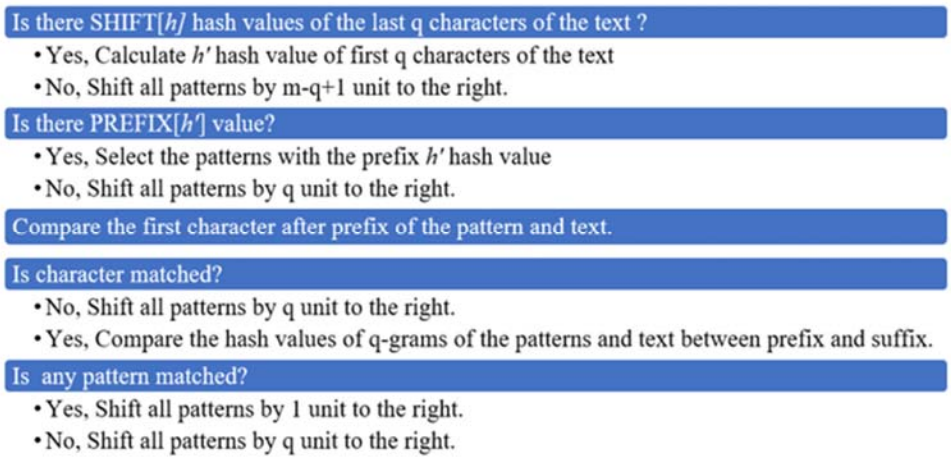



**Figure A.** The general proccess of the proposed approach

**Purpose:** The aim of this study is to solve the multiple exact string matching problem using the perfect hash function. The *q*-gram hash comparison based q-gram WM algorithm for DNA sequences is proposed and compared with the well-known algorithms in the literature.

**Theory and Methods:** In the proposed approach (q-gram WM), the hash collisions in suffixes and prefixes are eliminated and the fixed block length of the WM algorithm is increased. If the suffix and prefix hash values match, the first character of the pattern and text after prefix is checked. The reason for this is that if there is a character mismatch before proposed the q-gram hash comparison, it is possible to detect early that the pattern does not match and to gain the hash comparison cost. If this character matches, instead of comparing all characters of the patterns, q-gram hash comparisons of characters between q-length suffix and prefix characters are made thanks to the perfect hash function used in the proposed approach. In contrast to the traditional WM algorithm, the patterns are shifted more in the case of no pattern match in the q-gram WM algorithm. The used datasets are E. Coli contains 4.6 million and the Human Chromosome1 dataset contains 230 million DNA bases.

**Results:** Compared to the traditional WM algorithm, in the proposed approach, the better results have been achieved for the average runtime, the average number of character and hash comparisons. The proposed approach is more efficient than the well-known algorithms in the literature.

**Conclusion:** In the proposed approach, suffix and prefix hash collisions in the traditional WM algorithm are eliminated thanks to the perfect hash function. The fixed block length and shift size for patterns have been increased. The better results have been achieved in terms of the performance metrics.

Journal of the Faculty of Engineering and Architecture of Gazi University 38:2 (2023) 875-888

Mühendislik Mimarlık Fakültesi Dergisi
Journal of The Faculty of Engineering and Architecture of Gazi University

Elektronik / Online ISSN : 1304 - 4915
Basılı / Printed ISSN : 1300 - 1884

# DNA sekansları için q-gram hash karşılaştırmasına dayalı çoklu kesin dizi eşleştirme algoritması

Abdullah Ammar Karcıoğlu[1], Hasan Bulut[2]*

[1]Atatürk Üniversitesi, Mühendislik Fakültesi, Yazılım Mühendisliği Bölümü, 25240 Erzurum, Türkiye
[2]Ege Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, 35000 İzmir, Türkiye

## ÖNEÇIKANLAR

- DNA sekanslarında çoklu kesin dizi eşleştirmesi için mükemmel hash fonksiyonunu kullanma
- Çoklu kesin dizi eşleştirmede $q$-gram hash karşılaştırması
- Önerilen yaklaşım, diğer çoklu kesin dizi eşleştirme algoritmalarından daha verimlidir

Makale Bilgileri
Araştırma Makalesi
Geliş: 11.06.2021
Kabul: 16.04.2022

DOI:
10.17341/gazimmfd.951157

Anahtar Kelimeler:
Çoklu kesin dizi eşleştirme algoritmaları,
desen eşleştirme,
sekans analizi,
hash fonksiyonu,
wu-manber algoritması

ÖZ

Dizi eşleştirme algoritmaları tıp, biyoinformatik, biyoloji gibi birçok alandaki çeşitli uygulamaları nedeniyle bilgisayar bilimindeki önemli çalışma konularından olmuştur. Son yıllarda yeni algoritmalar geliştirilerek metin üzerinde dizi eşleştirme işlemleri hızlandırılmıştır. Dizi eşleştirme algoritmaları tekli ve çoklu olmak üzere iki kısma ayrılır. Çoklu kesin dizi eşleştirme algoritmaları verilen bir $T$ metni içinde $d$ adet $P$ desenlerinin bulunmasını içerir. Bu çalışmada, hash tabanlı çoklu kesin dizi eşleştirme algoritmalarından olan Wu-Manber algoritması ele alınmıştır. Wu-Manber algoritması etkili bir algoritma olmasına rağmen hash çakışmaları gibi bazı kısıtlamalara sahiptir. Çalışmamızda bu eksikliklere yönelik yeni yaklaşım önerilmiştir. Önerilen yaklaşımda, geleneksel Wu-Manber algoritmasının aksine, DNA sekanslarında hash çakışmasını kaldıran hash fonksiyonu kullanarak dizilerdeki arama işlemi $q$-gram hash karşılaştırması ile gerçekleştirilmiştir. Önerilen yaklaşım literatürde sıkça kullanılan çoklu kesin dizi eşleştirme algoritmalarıyla E. Coli ve Human Chromosome1 veri setinde karşılaştırmalar yapılmıştır. Yapılan deneysel çalışmalar sonucu önerilen yöntemin Wu-Manber algoritmasına kıyasla önerilen yaklaşımda ortalama çalışma zamanı, ortalama karakter ve hash karşılaştırma sayısı gibi performans metrikleri açısından daha iyi sonuçlar elde edilmiştir. Ayrıca, önerilen yaklaşımın Aho Corasick (AC) ve Commentz Walter (CW) gibi iyi bilinen algoritmalardan daha verimli olduğu gösterilmiştir.

# q-gram hash comparison based multiple exact string matching algorithm for DNA sequences

## HIGHLIGHTS

- Using the perfect hash function to multiple exact string matching in DNA sequences
- $q$-gram hash comparison in multiple exact string matching
- The proposed approach is more efficient than other multiple exact string matching algorithms

Article Info
Research Article
Received: 11.06.2021
Accepted: 16.04.2022

DOI:
10.17341/gazimmfd.951157

Keywords:
Multiple exact string matching,
pattern matching,
sequence analysis,
hash function,
wu-manber algorithm

ABSTRACT

The exact string matching algorithms are among the important study topics in computer science due to their various applications in many fields such as medicine, bioinformatics, and biology. New algorithms have been developed recently, and the string matching on the text has been accelerated. The string matching algorithms are divided into two parts, single and multiple. . The string matching algorithms are divided into two parts, single and multiple. The multiple exact string matching algorithms involve finding $d$ number patterns ($P$) in a given text T. In this study, the Wu-Manber algorithm, one of the hash-based multiple exact string matching algorithms, is discussed. Although the Wu-Manber algorithm is effective, it has some limitations, such as hash collisions. In our study, a new approach has is proposed for these limitations. In the proposed approach, unlike the traditional Wu-Manber algorithm, the searching in the sequences is performed by $q$-gram hash comparison, using the hash function that removes hash collisions in DNA sequences. The proposed approach has been compared with the multiple exact string matching algorithms with the well-known algorithms in the literature on E. Coli and Human Chromosome1 datasets. As a result of the experimental studies, better results have been achieved in terms of performance metrics such as the average runtime, the average number of character and hash comparisons in the proposed approach compared to the Wu-Manber algorithm. Also, the proposed approach is shown to be more efficient than well-known algorithms, such as Aho Corasick (AC) and Commentz Walter (CW).

*Sorumlu Yazar/Yazarlar / Corresponding Author/Authors : ammar.karcioglu@atauni.edu.tr, *hasan.bulut@ege.edu.tr / Tel: +90 232 311 2596

## 1. Giriş (Introduction)

Çoklu kesin dizi eşleştirme, verilen girdi metni içinde aranan desenlerin tüm oluşumlarını bulmayı hedefler. Metinlerde arama ve sahtecilik, bilgi alma, web filtreleme, izinsiz giriş algılama sistemleri, virüs tarayıcıları ve spam filtreleri, biyoinformatik sekanslarda arama gibi geniş çalışma alanına sahiptir.

Çoklu dizi eşleştirme algoritmalarını kullanarak literatürde dinamik desen eşleştirme işlemlerini yapmak için büyük boyuttaki verilerde çok sayıda sorgu ifadesinin aranması [1], çok sayıdaki desenlerin eşleştirilmesine yönelik yeni yöntemlerin geliştirilmesi [2], yüksek boyutlu girdi dizilerinde desenlerin eşleştirme işlemlerini hızlandırmak için verilerin sıkıştırılması [3], içerikten bağımsız dil kullanarak çoklu dizi eşleştirme [4], çoklu desenlerin eşleştirilmesi için ASCII tabanlı yaklaşımların geliştirilmesi [5], hacmi küçük desenlerin çoklu desen eşleştirilmesi [6] gibi birçok çalışma gerçekleştirilmiştir.

Çoklu desen eşleştirmede arama süresini kısaltmak için literatürde cuckoo filtrelemeye dayalı çoklu desen eşleştirme algoritmalarının paralelleştirilmesi [7], Rabin Karp algoritmasının paralelleştirilmesi [8], iki aşamalı paralel çoklu desen eşleştirme algoritması [9], Boyer-Moore-Horspool algoritmasına dayalı paralel çoklu dizi eşleştirme algoritmasının geliştirilmesi [10] ve mükemmel hash fonksiyonunu kullanarak GPU üzerinde paralel çoklu dizi eşleştirme algoritmalarının geliştirilmesi [11] gibi paralel programlama destekli birçok çalışma gerçekleştirilmiştir.

Yapılan biyolojik deneysel çalışmalar sonucu sekans veri setlerinin boyutları artmıştır. Bu yüksek hacimli veri setlerinde en ekili zamanda aranan desenlerin doğru tespit edilmesine ilgi zaman içinde artmıştır. Hash çakışmalarını ortadan kaldıran fonksiyonlar kullanarak DNA sekanslarında desen arama [12], oligonucleotidlerin doğru şekilde konumlarının tespit edilip en kısa zamanda çözüm üretecek yöntemlerin geliştirilmesi [13], sözlük eşleştirme işlemlerinde hastalık ismi tespiti için etkili dizi eşleştirme algoritmalarının geliştirilmesi [14], biyoinformatik sekanslarda online çoklu palindromların eşleştirilmesi [15], hastalığa yakalananlar için ilişkili desen modellerini kullanarak kişiye özel tedavi yöntemlerinin geliştirilmesi [16], çoklu dizi eşleştirme algoritmalarının performansları artırılarak benzerlik oranı yüksek DNA'ların tespitinin sağlanması [17] gibi çalışmalar gerçekleştirilmiştir.

DNA sekanslarına yönelik önek tabanlı dizi eşleştirme algoritmaları kullanarak, DNA sekanslarının ikili karşılaştırmasında ortaya çıkan çeşitli problemler için önek hesaplamalarını kullanan pratik paralel algoritmalar geliştirilmesi [18], çoklu dizi eşleştirmesi için paralel bir Rabin-Karp uygulaması olan eşleştirme doğrulamasını hızlandırmak için bir hash tablo ile birlikte algoritmanın paralelleştirilmesi en üst düzeye çıkarmak için temel olarak önek toplamları algoritmasının geliştirilmesi [19] gibi çalışmalar gerçekleştirilmiştir. DNA sekanslarına yönelik sonek tabanlı dizi eşleştirme algoritmaları kullanarak, verilen diziler üzerinde ardışık yinelemeler kullanarak transkripsiyon faktörleri için DNA dizileri setindeki bağlanma bölgelerini çıkaran yetkin bir algoritma önerilmesi [20], sonek ağaçları oluşturmak için pratik yöntemlerin geliştirilmesi [21] gibi çalışmalar geliştirilmiştir. DNA sekanslarında faktör tabanlı BOM algoritması geliştirilmiştir ancak kısa gen sekanslarında çalışma zamanı açısından verimli değildir [22]. Hash tabanlı dizi eşleştirme algoritmaları kullanılarak DNA sekanslarının eşleştirilmesinde Hash-q [23] ve Skip-q [24] gibi algoritmalar geliştirilmiştir, ancak DNA sekanslarında hash çakışması gibi problemlere sahiptirler.

Çoklu kesin dizi eşleştirme algoritmaları $T = t_0 t_1 t_2 \dots t_{n-1}$ olan $n$ uzunluklu girdi dizisi içinde $d$ adet $P = p^0 p^1 p^2 \dots p^{d-1}$ ve $m$ uzunluklu $p^r = p_0^r p_1^r \dots p_{m-1}^r$ her bir desenin tüm oluşumlarının bulunmasına dayanmaktadır. DNA sekanslarında Adenin (A), Sitozin (C), Timin (T) ve Guanin (G) olmak üzere 4 farklı baz kullanılmaktadır. Yani, toplam alfabe sayısı $|\sum| = 4$'tür. Desenlerin toplam uzunluğu ise $|P|$'dir.

Literatürde iyi bilinen çoklu kesin dizi eşleştirme algoritmaları AC, CW ve WM algoritmaları detaylı bir şekilde Bölüm 2.1'de açıklanmıştır.

## 2. Teorik Altyapı (Theoretical Background)

Verilen $n$ uzunluklu veri setinde $d$ adet desenin kaba kuvvet algoritması kullanarak aranmasının zaman karmaşıklığı $O(n|P|)$'dir. Girdi dizisinin ve aranan desenlerin uzunluğu arttıkça işlem süresi artacağından bu yöntem verimli değildir. Bu yüzden literatürde çoklu desenlerin etkili bir şekilde aranılması için bazı algoritmalar geliştirilmiştir. Bu algoritmalar genellikle, birden çok deseni eşzamanlı olarak arama yaparak çalışma zamanını etkili kullanma mantığına dayanmaktadır. Çoklu kesin dizi eşleştirme algoritmaları sonek, önek, faktör ve hash tabanlı algoritmalar olmak üzere 4 farklı kategoriye ayrılır [25].

- *Önek tabanlı algoritmalar:* Aranılan desenler düğümlerde temsil edilir. Burada kök düğüm boş bir diziyi temsil ederken, diğer düğümler ise her bir desenin önekini ifade eder. Aho Corasick (AC) [26] algoritması Knuth-Morris-Pratt (KMP) [27] algoritmasına dayalı olup literatürde önek tabanlı çoklu kesin dizi eşleştirme algoritması olarak sıkça kullanılmaktadır. AC algoritmasının zaman karmaşıklığı lineer zamandır, yani $O(n)$.
- *Sonek tabanlı algoritmalar:* Önek tabanlı algoritmaları ters yönlü otomata olarak temsil eder. Literatürde sıkça kullanılan sonek tabanı çoklu dizi eşleştirme algoritması Commentz Walter (CM) algoritmasıdır [28]. AC algoritmasının Boyer-Moore [29] algoritmasındaki kötü karakter tablosu eklenerek geliştirilmiş versiyonu CM algoritmasıdır. CM algoritmasının AC algoritmasından farklı olarak ön işleme aşamasında tersine durum makinesi oluşturur. CM algoritmasının zaman karmaşıklığı alt-lineer'dir.
- *Faktör tabanlı algoritmalar:* Desenlerin herhangi bir alt dizisini tanıyabilen ek geçişlere sahip bir trie oluştururlar [26]. Faktör tabanlı çoklu desen eşleştirme algoritmaları olarak Set Backward Oracle & Dawg Matching algoritmaları geliştirilmiştir [30].
- *Hash tabanlı algoritmalar:* Literatürde geliştirilen algoritmalar kullanılan bellek açısından verimli ve hızlı değillerdir. Bu yüzden çoklu desen eşleştirme işlemlerinde belleği ve zamanı daha verimli kullanmak için diziler verilen bir matematiksel hash fonksiyonundan geçirilerek elde edilen sonuca göre temsil edilirler. Wu-Manber (WM) [31] algoritması hash tabanlı çoklu dizi eşleştirme algoritmalarına öncülük etmektedir ve dizileri belirlenen blok uzunluklarında okuyarak eşleştirme yapar. Yüksek hacimli desenleri daha etkin bir şekilde eşleştirmek için WM algoritmasının bir versiyonu olan MDH algoritması da önerilmiştir [32].

### *2.1. WM Algoritması (WM Algorithm)*

WM algoritması çoklu dizi eşleştirme problemini çözmek için kötü karakter atlama tablosunu kullanan BM algoritmasının bir versiyonudur. WM algoritmasının amacı tüm karakterleri tek tek okumak yerine $B$ uzunluklu bloklar halinde okumaktır. Wu ve Manber [31], $B$ blok uzunluğunu yapmış oldukları deneysel çalışmalar sonucu 2 olarak seçmiştir. WM algoritması ön işleme ve arama olmak üzere iki aşamadan oluşmaktadır.

WM algoritmasının ön işleme aşamasında üç farklı tablo oluşturulur. Bunlar ATLAMA, ÖNEK ve HASH tablolarıdır. Tüm desenlerin $B$ uzunluklu atlama değerleri ATLAMA tablosunda temsil edilir. Eğer $B$ uzunluklu farklı karakter dizisi aranıyorsa $m - B + 1$ birim kadar atlama değeri atanır. HASH tablosu son $B$ uzunluklu karakter olarak temsil edilen soneklerin hash değerlerini tutar. Hash değerleri karakterlerin ASCII değerleri kullanılarak belli bir hash fonksiyonundan geçirildikten sonra elde edilen hash değeri ($h$) ile temsil edilir. WM algoritmasında kullanılan hash fonksiyonu $h = ((T[m-2] << 8 + T[m-1])\%65536)$. ÖNEK tablosu ise aynı soneke sahip olan desenlerin listesini tutar. $B'$ uzunluklu öneklerin hash değerleri ($h'$) ile ÖNEK tablosunda temsil edilirler. WM algoritması İngilizcede 'ion, ing' gibi sonu aynı biten çok sayıda kelimenin tek tek karşılaştırma maliyetini en aza indirgemek için ÖNEK tablosu oluşturulmuştur. Sonekleri aynı kelimeler çakışma olarak ele alınmıştır [31] ve karakter karşılaştırma maliyetini en aza indirgemek için $B'$ uzunluklu önekleri metnin önekiyle uyumlu kelimelerin tutulduğu ÖNEK tablosu geliştirilmiştir. Böylelikle soneki ve öneki girdi metniyle aynı olan kelimelerin kontrol edilmesi sağlanmıştır.

WM algoritmasının arama aşamasında ilk önce tüm desenler metnin başlangıcına hizalanır ve en kısa uzunluklu desen ($m$) belirlenir. Tüm desenlerin ilk m karakterleri ele alınır. Tüm desenlerin ATLAMA tablosu oluşturulduktan sonra metnin sonekinin hash değeri ($h$) hesaplanır ve bu sonekin atlama tablosunda olup olmadığı kontrol edilir. Eğer bu sonek yoksa $m - B + 1$ birim kadar desenler sola doğru kaydırılır. Eğer bu sonek varsa atlama değeri kontrol edilir. Atlama değeri 0'dan farklı ise bu değer kadar tüm desenler sola doğru kaydırılır. Sonekin atlama değeri 0'a eşit ise girdi metninin $B'$ uzunluklu hash değeri ($h'$) hesaplanır ve bu değere sahip desenler döndürülür. Önek uyuşmazlığı veya karakterlerden herhangi birinin eşleşmemesi durumunda tüm desenler $B$ birim kadar sağa doğru kaydırılır. Son olarak, sonek ve öneki uyan tüm desenler sırasıyla soldan sağa doğru tek tek karakter karşılaştırması yapılarak kontrol edilir. Herhangi bir desen eşleşirse tüm desenler bir (1) birim sağa doğru kaydırılır. Tüm bu işlemler girdi metninin sonuna kadar devam eder.

DNA sekans verilerinde WM çoklu kesin dizi eşleştirme algoritması uygulandığında yapılan deneysel çalışmalar sonucu bu algoritmanın bazı kısıtlamalara sahip olduğu gözlemlenmiştir. Bunlar;

*Kısıtlama 1:* Alfabe sayısı ($|\sum|$) belli olan farklı uzunluktaki olası üretilebilecek farklı hash değeri sayısı $|\sum|^{blok\_uzunluğu}$'dur. DNA sekanslarında alfabe sayısı (4) az olduğundan üretilebilecek farklı hash değeri sayısı azalacağı için DNA sekanslarındaki hash çakışma olasılığının İngilizce alfabelerindeki hash çakışma olasılığından çok yüksek olduğu görülmektedir. DNA sekanslarının alfabe sayısı $|\sum| = 4$ ve WM algoritmasının blok uzunluğu $B = 2$ olduğundan üretilebilecek farklı hash değeri sayısı $\sum^B = 4^2 = 16$'dır.

Soneki ve öneki 16 farklı alt dizi $\{AA, AC, AG, AT, CA, CC, CG, CT, GA, GC, GG, GT\}$ olan çok sayıda desen olması hash çakışmasına neden olur. Bu nedenle sonek değerlerinde çakışma olduğu için önek kontrolü yapılır ve hash hesaplaması çalışma süresi artar.

*Kısıtlama 2:* Soneki eşleşen metnin önekinin hash değerlerinin hesaplanmasında da çakışmaya neden olur. Çok sayıda desende $B$ uzunluklu öneke sahip desenlerin karakter kontrolü yapılacağından çalışma süresi artar.

*Kısıtlama 3:* Önek eşleşmezse veya önekten sonra gelen herhangi bir karakter eşleşmezse desenler $B = 2$ kadar metin üzerinde kaydırılır. Bu yüzden desenler daha az kaydırıldığından arama süresi artmaktadır.

## 3. Önerilen Yaklaşım (Proposed Approach)

DNA sekanslarında çoklu dizi eşleştirme problemini çözmek için kullanılan WM algoritmasındaki çözüm üreten mükemmel hash fonksiyonunu kullanarak yeni bir yöntem önerilmiştir. Kullanılan mükemmel hash fonksiyonuna ek olarak $q$-gram hash karşılaştırmasına dayalı desen eşleştirme yaklaşımı önerilmiştir. Bu bölümde önerilen yaklaşımdan ve çalışma örneklerinden bahsedilerek WM algoritmasındaki eksiklere yönelik yapılan katkılardan da bahsedilmiştir.

### *3.1. Önerilen Yaklaşım: q-Gram WM Algoritması (Proposed Approach: q-Gram WM Algorithm)*

Çalışmamızda kullanılan ve başka bir çalışmamızda [12] geliştirilen hash fonksiyonu Eş. 1'de verilmiştir. Bu hash fonksiyonu DNA sekansı karakterleri olan $\{A, C, G, T\}$ karakterlerinin ASCII değerlerini yeni değerlere verimli bir şekilde eşler ve tüm $q$ değerleri için benzersiz hash değerleri üretir.

$$FNG = (\sum_{i=0}^{q-2}(x[i]\&0b110) \ll 2(q-i-1)-1) + (x[q-1]\&0b110) \gg 1 \quad (1)$$

DNA sekanslarında hash çakışmasını ortadan kaldıran hash fonksiyonunu kullanarak sonek ve önek dizilerinin hash değerlerindeki çakışma ortadan kaldırılmıştır. Geleneksel WM algoritmasındaki sabit blok uzunluğu $B = 2$ yerine mükemmel hash fonksiyonu kullanarak blok uzunluğu artırılmıştır. Bunu nedeni, bilgisayar mimarilerinde 8-bit kayıtçı kullanıldığından $4^8$ farklı değer üretildiği için blok uzunluğu maksimum 8'dir. Önerilen yaklaşımda blok uzunluğu $B$ yerine $q$ ile ifade edilir. Başka bir çalışmamızdan [12] elde edilen deneysel çalışmalar sonucu en etkili blok uzunluğu olarak belirlenen değer bu çalışma için $q = 8$ olarak seçilmiştir.

Aranan desenlerin soneklerinde hash çakışması ortadan kaldırılmıştır ve böylece önek kontrolü yapmadan desenler kaydırılır. Soneklerde olduğu gibi öneklerdeki hash çakışması da ortadan kaldırılmıştır ve kontrol edilecek desen sayısı optimize edilmiştir. Ayrıca, $q$ uzunluklu önek veya önekten sonra gelen herhangi bir karakter eşleşmezse, WM algoritmasının aksine, $B = 2$ birim kadar değil $q = 8$ birim kadar desenler metin üzerinde kaydırılarak önerilen yaklaşımda daha fazla atlama miktarı elde edilmiştir.

Tablo 1'de gösterildiği gibi önerilen $q$-Gram WM algoritmasının ön işleme aşaması, WM algoritmasının temeline dayanmaktadır ancak ATLAMA ve ÖNEK tablolarındaki değerleri $q = 8$ göz önünde bulundurularak hesaplanır. Tablo 2'de gösterildiği gibi, önerilen $q$-Gram WM algoritmasının arama aşaması da WM algoritmasına benzer işlemektedir. Yani, $q$ uzunluklu sonekinin hash değeri ATLAMA tablosunda varsa metnin $q$ uzunluklu öneki kontrol edilir, aksi halde desenler $m - q + 1$ birim kadar sağa kaydırılır. Metnin $q$ uzunluklu önekinin hash değeri ÖNEK tablosunda yoksa desenler $q$ kadar sağa kaydırılır. Eğer öneki varsa bu öneke sahip desenlerin ilk olarak önek karakterinden sonra gelen ilk karakteri karşılaştırılır. Buradaki amaç, önerdiğimiz $q$-gram hash karşılaştırmasını yapmadan önce karakter uyuşmazlığı varsa desenin eşleşmediğini erkenden tespit edip hash karşılaştırma maliyetinden kazanç elde etmektir. Eğer bu karakter eşleşirse, desenlerin tüm karakterlerini karşılaştırmak yerine önerilen yaklaşımda kullanılan mükemmel hash fonksiyonu sayesinde $q$ uzunluklu sonek ve önek karakterleri arasındaki karakterlerin $q$-gram hash karşılaştırması yapılır.

Verilen $m$ uzunluklu desende toplam $(\left\lfloor \frac{m}{q} \right\rfloor - 2)$ $q$-gram hash karşılaştırması yapılır. Sonek ve önek mükemmel hash fonksiyonu kullanılarak eşleştiği için tekrardan kontrol etmemize gerek yoktur ve

**Tablo 1.** $q$-Gram WM Algoritmasının Ön İşleme Aşamasının Sözde Kodu (Pseudocode of Preprocessing Phase of $q$-Gram WM Algorithm)

```
1.  q ← 8,  d ← desen_sayisi
2.  ATLAMA tablosunun tüm elemanlarına m − q + 1 ata
3.  Algoritma ATLAMA_tablosu(P, d)
4.   minUzunluk ← min _desen_uzunlugu
5.   for i ← 0 to d − 1 do
6.      for j ← 0 to minUzunluk − q do
7.          atlama ← minUzunluk − j + q
8.          önek[q] ← {P[i][j], P[i][j + 1], P[i][j + 2], P[i][j + 3], P[i][j + 4], P[i][j + 5], P[i][j + 6], P[i][j + 7]}
9.          ekle(ATLAMA, sonek, atlama)
10. Algoritma ÖNEK_tablosu(P, d)
11.  for i ← 0 to d − 1 do
12.     önek[q] ← {P[i][0], P[i][1], P[i][2], P[i][3], P[i][4]}
13.     ekle(ÖNEK, önek, P[i])
```

**Tablo 2.** $q$-Gram WM Algoritmasının Arama Aşamasının Sözde Kodu (Pseudocode of Searching Phase of $q$-Gram WM Algorithm)

```
1.  Algoritma ATLAMA_tablosu(T, P, d, q, minUzunluk)
2.   kaydır ← minUzunluk − q + 1
3.   basla ← minUzunluk − 1
4.   metin_uzunluğu ← uzunluk (T)
5.   while start < metin_uzunluğu do
6.      altDizi[q] ← {T[basla − 7], T[basla − 6], T[basla − 5], T[basla − 4], T[basla − 3], T[basla − 2], T[basla −
        1], T[basla]}
7.      atlama ← get(ATLAMA, altDizi)
8.      if atlama = −1 then
9.         basla ← basla + kaydır
10.     else if kaydır = 0 then
11.        preS ← basla − minUzunluk − 1
12.        önek[q] ← {T[preS], T[preS + 1], T[preS + 2], T[preS + 3], T[preS + 4], T[preS + 5], T[preS +
        6], T[preS +
        7]}
13.        dügümler ← dügüm_döndür(ATLAMA, önek)
14.        while dügümler do
15.           gecici ← 0, eslesen ← 1, max _eslesen ← yuvarla(m/q), i ← q
16.           if (m[i + 1] = T[pres + 1]) then
17.              while i < m − (eslesen ∗ q) AND (FNG(önek, preS + (q ∗ (eslesen − 1))) = FNG(desen, preS + (q ∗
        (eslesen − 1)))) do
18.                 eslesen ← eslesen + 1
19.           if(i >= m − (eslesen ∗ q)) then
20.              gecici ← 1
21.           else gecici ← 0
22.        basla ← basla + 1
23.     else then basla ← basla + kaydır
```

toplam $q$-gram uzunluklu karakter sayısından iki (2) adet $q$-gram çıkarmış oluruz. Yapılan sonek ve önek hash karşılaştırma sayılarına ek olarak bu $q$-gram hash karşılaştırma sayıları da dahil edilir. DNA sekansları için ATLAMA ($h$) ve ÖNEK ($h'$) tablolarının hash değerleri ($q = 8$ için) mükemmel hash fonksiyonu kullanılarak Eş. 2 ve Eş. 3'te gösterildiği gibi hesaplanır. Aranan veri setine hizalanan desenlerin $q$ uzunluklu sonek ve önek dizilerinin hash değerleri hesaplanıp ATLAMA ve ÖNEK tablolarında bu değerlerin olup olmadığı kontrol edilmiş olunur. Burada $h$ değeri $q = 8$ uzunluklu sonek dizisinin hash değerinin hesaplanmasını, $h'$ ise $q = 8$ uzunluklu önek dizisinin hash değerinin hesaplanmasını temsil etmektedir.

$$h = \big((T[m-1]\&110) \ll 13\big) + \big((T[m-2]\&110) \ll 11\big) + ((T[m-3]\&110) \ll 9) + ((T[m-4]\&110) \ll 7) + \big((T[m-5]\&110) \ll 5\big) + \big((T[m-6]\&110) \ll 3\big) + ((T[m-7]\&110) \ll 1) + ((T[m-8]\&110) \gg 1) \quad (2)$$

$$h' = ((T[0]\&110) \ll 13) + ((T[1]\&110) \ll 11) + ((T[2]\&110) \ll 9) + ((T[3]\&110) \ll 7) + ((T[4]\&110) \ll 5) + ((T[5]\&110) \ll 3) + ((T[6]\&110) \ll 1) + ((T[7]\&110) \gg 1) \quad (3)$$

### *3.2. WM ve q-Gram WM Algoritmalarının Karşılaştırma Sayısı Örnekleri*
*(Examples of Number of Comparisons of The WM and q-Gram WM Algorithms)*

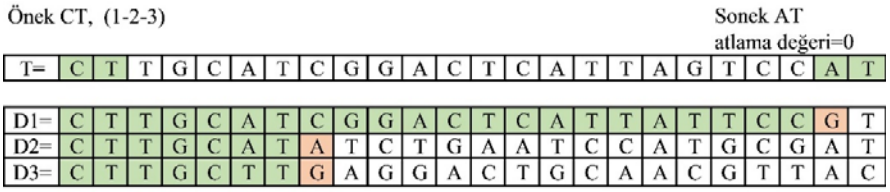
WM ve $q$-gram WM algoritması için örnek bir girdi ve desen kümesi vererek bu algoritmalardaki başarılı ve başarısız dizi eşleştirme durumlarının örneklerle gösterilmiştir. Girdi metni ve desen uzunlukları 24, önerilen yaklaşım için $q = 8$ seçilmiştir. Başarılı karşılaştırmalar yeşil ile, başarısız karşılaştırmalar ise açık kırmızı ile gösterilmiştir. Şekil 1 verilen $T$ metninde aranan $D1, D2, D3$ desenlerinin başarısız desen eşleştirme durumunu göstermektedir. Desenler T metninin başına hizalandıktan sonra metnin $B = 2$ uzunluklu $AT$ sonek dizisinin $h$ hash değeri hesaplanır. ATLAMA[$h$] değeri 0'a eşit olduğu için metnin $B = 2$ uzunluklu $CT$ önek dizisinin $h'$ hash değeri hesaplanır. ÖNEK[$h'$] değeri 1,2,3 nolu desenleri döndürür ve bu desenlerde baştan sona doğru karakterler tek tek karşılaştırılır. Sonuç olarak, 2 hash karşılaştırması ve 39 karakter karşılaştırması yapılan bu örnekte başarısız desen eşleştirmesi olduğu için desenler $B = 2$ kadar metin üzerinde sağa kaydırılır. Atlama

tablosunda sonekler ve değerleri sırasıyla {AA=6, AC=0, AG=14, AT=0, CA=6, CC=2, CG=1, CT=11, GA=1, GC=3, GG=13, GT=0, TA=1, TC=3, TG=4, TT=2} ve önek tablosu {CT→1,2,3} numaralı desenlerdir.

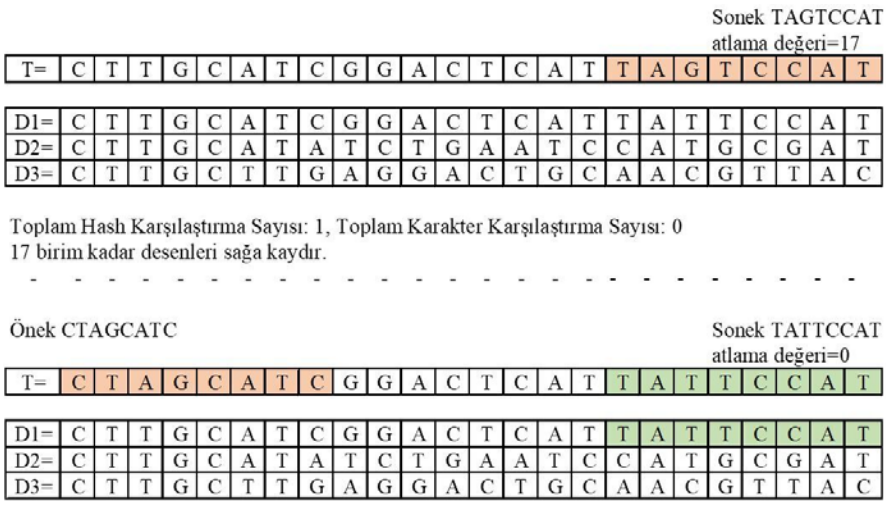
Şekil 2 önerilen $q$-gram WM algoritmasındaki sonek ve önek eşleşmeme durumunu göstermektedir. $T$ dizisinin soneki ATLAMA tablosunda olmadığından $m-q+1=17$ birim desenler sağa kaydırılır. Soneki eşleşmesine rağmen önek eşleşmeme durumunda ise WM algoritmasındaki $B=2$ aksine $q=8$ birim kadar desenler kaydırılır. Herhangi bir karakter karşılaştırması yapmadan önerilen yaklaşımda sonek ve önek eşleşmeme durumlarında performans sağlanır. Atlama tablosunda sonekler ve değerleri sırasıyla { TATTCCAT=0, TTATTCCA=1, ATTATTCC=2, CATTATTC=3, TCATTATT=4, CTCATTAT=5, ACTCATTA=6, GACTCATT=7, GGACTCAT=8, CGGACTCA=9, TCGGACTC=10, ATCGGACT=11, CATCGGAC=12, GCATCGGA=13, TGCATCGG=14, TTGCATCG=15, CTTGCATC=16, Diğerleri=17} ve önek tablosu {CTTGCATC=1, CTTGCATA=2, CTTGCATG=3} numaralı desenlerdir.

Şekil 3 WM algoritmasındaki başarılı desen eşleştirme durumunu göstermektedir. Sonek ve önek eşleştikten sonra önek tablosunun döndürdüğü (Desen 1-2-3) desenlerin karakterleri baştan sona kontrol edilir. Desen 1 eşleştiği için desenler sağa 1 birim kaydırılır ve 2 hash karşılaştırması ve 40 karakter karşılaştırması yapılır. Atlama tablosunda sonekler ve değerleri sırasıyla {AA=6, AC=0, AG=14, AT=0, CA=6, CC=2, CG=1, CT=11, GA=1, GC=3, GG=13, GT=0, TA=1, TC=3, TG=4, TT=2} ve önek tablosu {CT→1,2,3} numaralı desenlerdir.

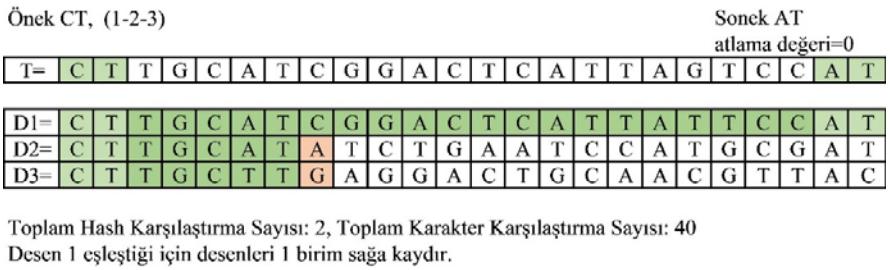
Şekil 4 önerilen $q$-gram WM algoritmasındaki başarılı desen eşleşme durumunu göstermektedir. Sonek eşleştikten sonra önek eşleşmiştir ve ÖNEK tablosundan Desen 1'i döndürmüştür. Sonek ve önek karakterlerinde çakışma olmadığından önekden sonra gelen ilk karakter ile metne denk gelen karakter kontrol edilir. Karakter uyumu



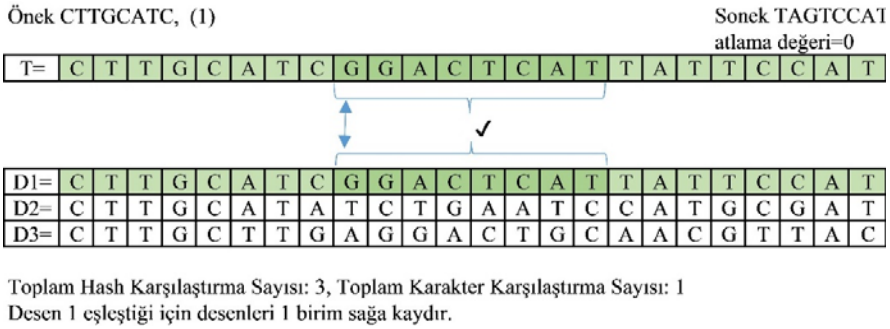

**Şekil 1.** WM Algoritmasının Başarısız Desen Eşleştirme Durumu
(Unsuccessful pattern matching case of the WM algorithm)




**Şekil 2.** $q$-Gram WM Algoritmasının Başarısız Desen Eşleştirme Durumu
(Unsuccessful pattern matching case of the $q$-gram WM algorithm)




**Şekil 3.** WM Algoritmasının Başarılı Desen Eşleştirme Durumu (Successful pattern matching case of the WM Algorithm)




**Şekil 4.** q-Gram WM Algoritmasının Başarılı Desen Eşleştirme Durumu
(Successful pattern matching case of the $q$-gram WM Algorithm)

sağlandığı için önekden sonra gelen $q$-gram'ların hash kontrolü yapılır. Sonuç olarak toplam 3 hash karşılaştırması ve 1 karakter karşılaştırması yapılarak karakter karşılaştırma sayısı minimuma indirgenmiştir. Atlama tablosunda sonekler ve değerleri sırasıyla { TATTCCAT=0, TTATTCCA=1, ATTATTCC=2, CATTATTC=3, TCATTATT=4, CTCATTAT=5, ACTCATTA=6, GACTCATT=7, GGACTCAT=8, CGGACTCA=9, TCGGACTC=10, ATCGGACT=11, CATCGGAC=12, GCATCGGA=13, TGCATCGG=14, TTGCATCG=15, CTTGCATC=16, Diğerleri=17} ve önek tablosu {CTTGCATC=1, CTTGCATA=2, CTTGCATG=3} numaralı desenlerdir.

### *3.3. Önerilen Yaklaşımın Akış Diyagramı*
*(Flowchart of the proposed approach)*

Önerilen $q$-gram WM algoritmasının akış diyagramı Şekil 5'te gösterilmiştir. Ön işleme aşamasında öncelikle ATLAMA ve ÖNEK tabloları oluşturulur. Arama aşamasında, desenler metnin başlangıcına hizalandıktan sonra metnin $q$ uzunluklu sonekinin hash değeri kontrol edilir. Hash değerleri varsa, metnin $q$ uzunluklu önekinin hash değeri kontrol edilir. Eğer bu hash değeri de varsa bu öneke sahip desenler döndürülür. Sonra, önekten sonra gelen ilk karakter karşılaştırılır. Karakter eşleşirse sonek ve önek karakterleri arasında kalan karakterler soldan sağa $q$-gram hash karşılaştırmasıyla kontrol edilir. Herhangi bir eşleşme sağlanırsa, desenler metin üzerinde bir birim sağa kaydırılır; eşleşme sağlanmazsa, $q$ birim sağa kaydırılır. Tüm adımlar metnin sonuna kadar devam eder.

### *3.4. Önerilen Yaklaşımın Çalışan Örneği*
*(Worked examples of the proposed approach)*

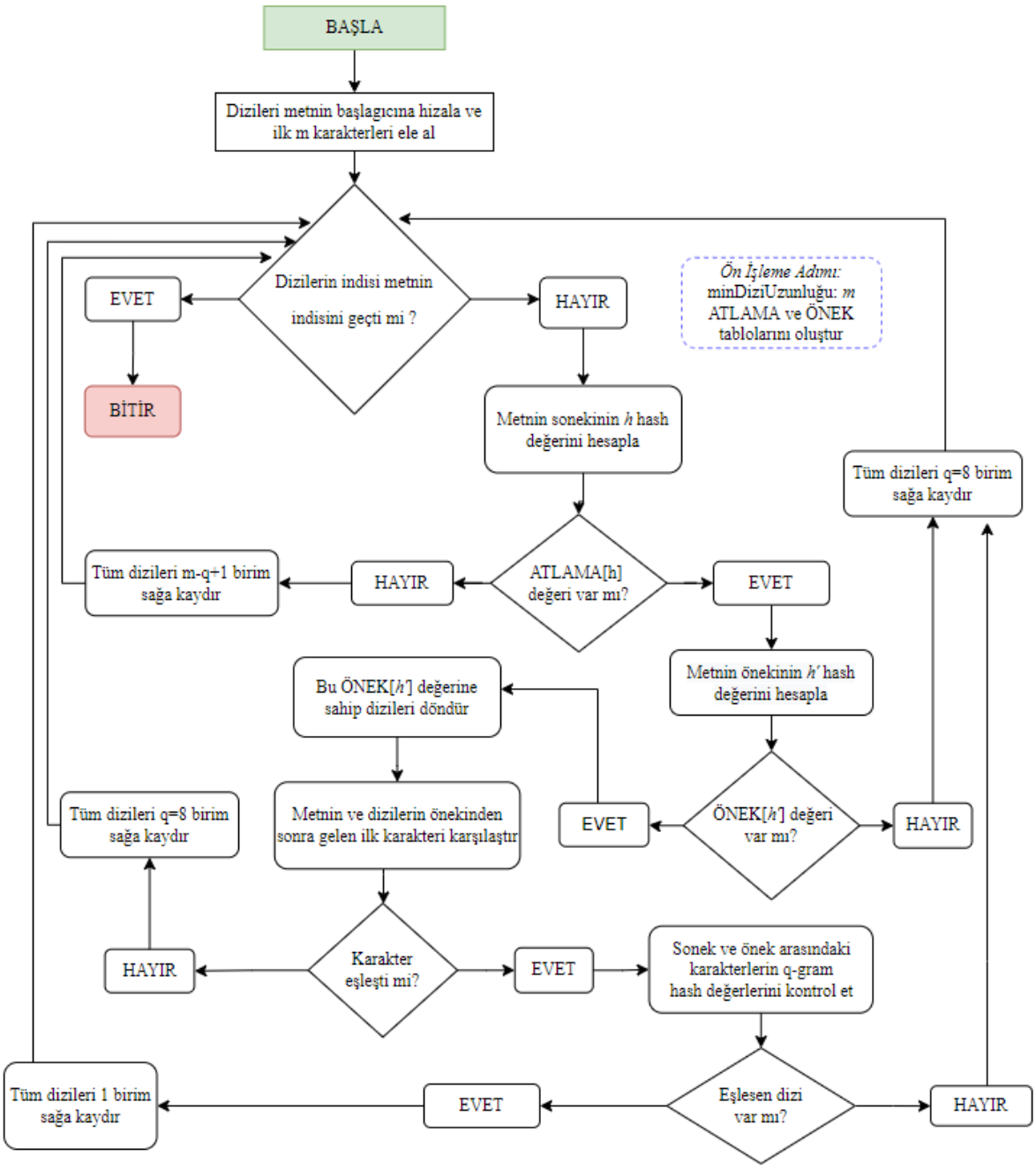
$T$ girdi dizisi ve $P$ desenler kümesi olarak sırasıyla $T = TACTGGAATCGATATCTGCACGCGATTCGACTAACGGTCTTCACCATT$ uzunlukları birbirine eşit 2 farklı desen $P = \{D1 = \text{TACTGGAATCGATATCTGCACGCG}, D2 = \text{ATTCGACTAACGGTCTTCACCATT}\}$ seçilmiştir. Her adımda hash karşılaştırma sayısı, karakter karşılaştırma sayısı ve eşleşen desen gösterilmiştir. Başarısız hash ve karakter karşılaştırmaları açık kırmızı ile, başarılı hash karşılaştırmaları açık yeşil ile, başarılı karakter karşılaştırmaları ise koyu yeşil ile gösterilmiştir.

Tablo 3 verilen desenler için WM algoritmasındaki, Tablo 4 ise önerilen $q$-gram WM algoritmasındaki ön işleme adımı sonucu elde edilen ATLAMA ve ÖNEK tablolarını göstermektedir. Şekil 6 ve 7 ön işleme adımı sonucu oluşturulan tablolardaki bilgileri kullanarak geleneksel WM algoritmasında ve önerilen $q$-gram WM algoritmasında girdi dizisinde aranacak desenler kümesinin işleyişi gösterilmiştir. Şekil 6 WM algoritmasının işleyiş örneğini göstermektedir. Tüm desenlerin eşleştiği örnekte hash karşılaştırma sayısı 6, karakter karşılaştırma sayısı 48'dir. Şekil 7 önerilen $q$-gram WM algoritmasının işleyen örneğini göstermektedir. Tüm desenlerin eşleştiği örnekte hash karşılaştırma sayısı 8, karakter karşılaştırma sayısı2'dir. Önerilen yaklaşımda, karakter karşılaştırması yapmadan desen eşleştirme yapılmıştır. İşleyen örneklerde önerilen $q$-gram WM algoritmasında WM algoritmasından daha iyi performans sonuçları elde edildiği gösterilmiştir.

AC algoritmasının en iyi ve en kötü durum zaman karmaşıklığı $O(n)$'dir. Ancak AC algoritmanın geliştirilmiş bir versiyonu olan CW algoritmasının en iyi durum karmaşıklığı alt lineerdir. Hash tabanlı WM algoritmasının en iyi durum zaman karmaşıklığı $O((n\log_{|\Sigma|}|P|)/m)$ ve en kötü durum zaman karmaşıklığı $O(n|P|\log_{|\Sigma|}|P|)$'dir [19]. Önerilen $q$-gram WM algoritmasında $q$ uzunluklu sonek ve önek karakterlerinin karşılaştırması yapılmamaktadır. Sonuç olarak, $d$ adet desenin toplam desen uzunluğu $|P|$ olduğundan $d$ adet desenin sonek ve önek karakterleri karşılaştırılmadığı için toplam karşılaştırılacak karakter sayısı $|P - 2dq|$'dur. Bu nedenle önerilen $q$-gram WM algoritmasının en iyi durum zaman karmaşıklığı $O((n\log_{|\Sigma|}|P - 2dq|)/(m - 2q))$ ve en kötü durum zaman karmaşıklığı $O(n|P - 2dq|\log_{|\Sigma|}|P - 2dq|)$'dır. Bu çalışmada zaman karmaşıklığı WM algoritmasından daha verimli $q$-gram WM algoritması önerilmiştir.

Literatürde tam metin indeksleme problemi için, tüm karakterler için geçerli, metni sıkıştırılmış biçimde depolamak için gerekli olan alanı kullanarak verimli alt dizi aramalarını destekleyen iki sıkıştırılmış veri yapısı tasarlanmıştır [32]. Başka bir çalışmada büyük boyutlu DNA veri tabanlarında kısa uzunlukta desen aramalarını hızlandıran hızlı bir bit düzeyinde kesin dizi eşleştirme algoritması geliştirilmiştir [33]. Ayrıca başka bir çalışmada kanser tedavisinde, önemli bir yere sahip olan P53 DNA-bağlayan konsensüs sekansının eşleşmesi için bit seviyesinde dizi eşleştirme algoritması geliştirilmiştir [34]. Gerçekleştirilen bu çalışmalara kıyasen önerdiğimiz yaklaşımda, veri setlerinde herhangi bir metin sıkıştırma işlemi gerekmeksizin DNA sekanslarında mükemmel hash fonksiyonuna kullanan $q$-gram hash karşılaştırmasına dayalı çoklu kesin dizi eşleştirme algoritması önerilmiştir. Bit seviyesinde yapılan çalışmalara kıyasen, önerilen yaklaşımda tekli bir desen değil 1000-3000-5000 gibi çok sayıda desenin verimli bir şekilde eşleştirmesi sağlanmıştır. Sadece 32-bitlik bilgisayarlarda değil, tüm bilgisayar mimarilerine uyumlu mükemmel hash fonksiyonu bu çalışmada kullanılmıştır.

Literatürde ayrıca sekans analizi problemleri için, bölütlenen beyin bölgelerinin tıbbi görüntü steganografi için değerlendirilmesinde hash fonksiyonları kullanılarak DNA kodlama ile şifrelenen ve sıkıştırılan yüksek kapasiteli mesaj, görüntülerin tümör olmayan piksellerinin en az anlamlı bitlerinde gizlenmesi [35], sayısal haritalama teknikleri ve fourier dönüşümü kullanılarak DNA sekanslarının dizilimlerinin sınıflandırılarak sekans analizi gerçekleştirilmesi [36], DNA genom dizilimi üzerinde dijital sinyal işleme teknikleri kullanılarak elde




**Şekil 5.** Önerilen $q$-Gram WM Algoritmasının Akış Şeması (The Flowchart of the proposed approach)

edilen ekson ve intron bölgelerinin EfficientNetB7 mimarisi ile sınıflandırılması [37] gibi birçok çalışma gerçekleştirilmiştir. Literatürde sıkça kullanılan çoklu dizi eşleştirme algoritmalarından AC algoritmasının en iyi ve en kötü durum zaman karmaşıklığı $O(n)$'dir. CW algoritması ise AC algoritmasının geliştirilmiş bir versiyonu olup zaman karmaşıklığı alt lineerdir. WM algoritmasının zaman karmaşıklığı $O(\mathrm{n}|\mathrm{P}|\log_{|\Sigma|}|P|)$'dir [19]. Formüldeki $|P|$, $d$ adet desenin toplam desen uzunluğudur. Bu çalışmada önerilen $q$-gram WM algoritması mükemmel hash fonksiyonunu kullanarak sonek ve önek kontrolü yapmadan eşleştirme işlemini gerçekleştirdiği için karşılaştırılacak karakter sayısı $|P-2dq|$'dur. Sonuç olarak önerilen yaklaşımın zaman karmaşıklığı $O(\mathrm{n}|\mathrm{P}-2\mathrm{dq}|\log_{|\Sigma|}|P-2dq|)$ olarak elde edilmiştir ve WM algoritmasından daha verimli olduğu gösterilmiştir.

## 4. Deneysel Çalışmalar (Experimental Studies)

Deneysel çalışmalarda, Tablo 5'te gösterildiği gibi veri setlerinin referans numaraları kullanarak NCBI sitesinden açık erişimi mümkün olan E. Coli [38] ve Human Chromosome1 (H-Chr1) [39] veri setleri metin dosya formatında indirilmiştir. E. Coli veri seti 4,6 milyon ve H-Chr1 veri seti 230 milyon DNA karakteri içermektedir. Sırasıyla küçük ve büyük hacimli olan bu veri setleri algoritmaların karşılaştırılması için seçilmiştir.

**Tablo 3.** Verilen Desenler İçin WM Algoritmasındaki Sonek ve Önek Tabloları
(Suffix and prefix tables of the WM algorithm for given patterns)

| Atlama Tablosu | | | | Önek Tablosu | |
|---|---|---|---|---|---|
| Sonek | Değer | Sonek | Değer | Önek | Desen No |
| AA | 14 | GA | 12 | AT | 1 |
| AC | 3 | GC | 1 | TA | 2 |
| AG | 17 | GG | 11 | | |
| AT | 1 | GT | 10 | | |
| CA | 2 | TA | 10 | | |
| CC | 3 | TC | 6 | | |
| CG | 0 | TG | 6 | | |
| CT | 7 | TT | 0 | | |

**Tablo 4.** Verilen Desenler İçin *q*-Gram WM Algoritmasındaki Sonek ve Önek Tabloları
(Suffix and prefix tables of the q-Gram WM algorithm for given patterns)

| Atlama Tablosu | | | | Önek Tablosu | |
|---|---|---|---|---|---|
| Sonek | Değer | Sonek | 1 | Önek | Desen No |
| TCACCATT | 0 | TGCACGCG | 2 | ATTCGACT | |
| TTCACCAT | 1 | CTGCACGC | 1 | TACTGGAA | |
| CTTCACCA | 2 | TCTGCACG | 2 | | |
| TCTTCACC | 3 | ATCTGCAC | 3 | | |
| GTCTTCAC | 4 | TATCTGCA | 4 | | |
| GGTCTTCA | 5 | ATATCTGC | 5 | | |
| CGGTCTTC | 6 | GATATCTG | 6 | | |
| ACGGTCTT | 7 | CGATATCT | 7 | | |
| AACGGTCT | 8 | TCGATATC | 8 | | |
| TAACGGTC | 9 | ATCGATAT | 9 | | |
| CTAACGGT | 10 | AATCGATA | 10 | | |
| ACTAACGG | 11 | GAATCGAT | 11 | | |
| GACTAACG | 12 | GGAATCGA | 12 | | |
| CGACTAAC | 13 | TGGAATCG | 13 | | |
| TCGACTAA | 14 | CTGGAATC | 14 | | |
| TTCGACTA | 15 | ACTGGAAT | 15 | | |
| ATTCGACT | 16 | TACTGGAA | 16 | | |
| Diğerleri | 17 | | | | |


**Şekil 6.** WM Algoritmasının Çalışan Örneği (The Worked Example of WM algorithm)


**Şekil 7.** $q$-Gram WM Algoritmasının Çalışan Örneği (The Worked Example of $q$-Gram WM algorithm)

**Tablo 5.** Veri Setleri (Datasets)

| Veri Seti | NCBI Referans No | Uzunluğu |
|---|---|---|
| E. Coli | NC_010473.1 | 4,686,137 |
| H-Chr1 | NC_000001.11 | 230,480,853 |

Her bir veri seti içinden rastgele $m = 16, 32, 64, 128, 256, 512, 1024$ uzunluklu desenler seçilmiştir. $P$ desenler kümesi $m$ uzunluklu $d = 1000, 3000, 5000$ adet desenden oluşmaktadır. Her bir algoritma 10 kere çalıştırılarak ortalama çalışma zamanları (sn.) elde edilmiştir. Ayrıca her bir algoritma için ortalama karakter karşılaştırma sayısı ve hash karşılaştırma sayısı elde edilmiştir. Önerilen $q$-gram WM algoritmasındaki $q = 8$ olarak seçilmiş olup, tüm veri setlerine uygulanarak sonuçları WM algoritmasıyla ve literatürde iyi bilinen algoritmalarla karşılaştırılmıştır. Tüm $d$ adet desen kümelerindeki her bir desen uzunluğu için ortalama çalışma zamanındaki hızlanma yüzdeleri, ortalama karakter ve hash karşılaştırma sayılarındaki artış yüzdeleri gösterilmiştir. Elde edilen en yüksek yüzde değerleri ise koyu renkle gösterilmiştir. Tüm algoritmalar C programlama dilinde gerçekleştirilmiştir. Kullanılan makinenin teknik özellikleri ise Intel(R) Core (TM) 64-bit, i7-9750H CPU, 2.60 GHz, 32 GB RAM, Nvidia GeForce RTX 2060.

### *4.1. WM ve q-Gram WM Algoritmasının Karşılaştırılması*
*(Comparison of WM and q-Gram WM Algorithm)*

#### *4.1.1. Çalışma zamanı sonuçları (The results of runtimes)*

Tablo 6 veri setlerindeki $d$ adet $m$ uzunluklu her bir desendeki ortalama çalışma zamanı sonuçlarını göstermektedirler. E. Coli veri setinde WM algoritmasına kıyasla önerilen $q$-gram WM algoritmasında tüm $d = 1000, 3000, 5000$ adet $m$ uzunluklu desenlerde ortalama çalışma zamanında sırasıyla 98,73%, 99,48%, 99,71% ($m = 64, 32, 16$ uzunluklarında) hızlanmalar elde edilmiştir. Ayrıca önerilen yaklaşımda tüm desen uzunluklarında hızlanmalar elde edilmiştir. H-Chr1 veri setinde WM algoritmasına kıyasla $q$-gram WM algoritmasında tüm $d = 1000, 3000, 5000$ adet $m$ uzunluklu desenlerde ortalama çalışma zamanında sırasıyla 99,76%, 99,88%, 99,89% ($m = 256, 32, 256$ uzunluklarında) hızlanmalar elde edilmiştir. Ayrıca önerilen yaklaşımda tüm desen uzunluklarında hızlanmalar elde edilmiştir.

#### *4.1.2. Karakter karşılaştırma sayısı sonuçları*
*(The results of number of character comparisons)*

Tablo 7 veri setlerindeki $d$ adet $m$ uzunluklu her bir desendeki ortalama karakter karşılaştırma sayısı sonuçlarını göstermektedirler. Tüm veri setlerinde WM algoritmasına kıyasla önerilen $q$-gram WM algoritmasında tüm $d = 1000, 3000, 5000$ adet $m$ uzunluklu desenlerde ortalama karakter karşılaştırma sayılarında 99,99% daha az karşılaştırma elde edilmiştir. Böylelikle, önerilen yaklaşımda karakter karşılaştırma sayısı en aza indirgenmiştir.

#### *4.1.3. Hash karşılaştırma sayısı sonuçları*
*(The results of number of hash comparisons)*

Tablo 8 veri setlerindeki $d$ adet $m$ uzunluklu her bir desendeki ortalama hash karşılaştırma sayısı sonuçlarını göstermektedirler. Tüm veri setlerinde WM algoritmasına kıyasla önerilen $q$-gram WM algoritmasında tüm $d = 1000, 3000, 5000$ adet $m$ uzunluklu desenlerde ortalama hash karşılaştırma sayılarında 99,99% daha az karşılaştırma elde edilmiştir. Böylelikle, önerilen yaklaşımda mükemmel hash fonksiyonu kullanarak hash karşılaştırma sayısı en aza indirgenmiştir.

### *4.2. İyi Bilinen Algoritmalarla Karşılaştırma*
*(Comparisons With The Well-Known Algorithms)*

Literatürde iyi bilinen çoklu dizi eşleştirme algoritmalarından bazılarıyla önerilen $q$-gram WM algoritmasının ortalama çalışma zamanı açısından karşılaştırması gerçekleştirilmiştir. Önerilen yaklaşımla karşılaştırma yapmak için önek tabanlı AC algoritması, sonek tabanlı CW, hash tabanlı WM algoritması seçilmiştir ve veri setlerine uygulanmışlardır.

**Tablo 6.** Ortalama Çalışma Zamanı (The Average Runtime)

| | | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
|---|---|---|---|---|---|---|---|---|---|
| d = 1000 | E. Coli | WM | 2,411 | 2.467 | 2.454 | 2.481 | 2.584 | 2.636 | 2.710 |
| | | q-gram WM | 0,034 | 0,032 | 0,031 | 0,039 | 0,046 | 0,066 | 0,105 |
| | | Hızlanma | 98,58% | 98,70% | *98,73%* | 98,42% | 98,21% | 97,49% | 96,12% |
| | | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | H-Chr1 | WM | 126,82 | 129.85 | 132,76 | 134,87 | 132,55 | 142,34 | 141,11 |
| | | q-gram WM | 0,663 | 0,465 | 0,377 | 0,335 | 0,314 | 0,367 | 0,385 |
| | | Hızlanma | 99,47% | 99,63% | 99,71% | 99,75% | *99,76%* | 99,74% | 99,72% |
| d = 3000 | E. Coli | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | | WM | 7,201 | 8,447 | 7,635 | 7,758 | 8,105 | 7,845 | 9,035 |
| | | q-gram WM | 0,042 | 0,044 | 0,049 | 0,061 | 0,089 | 0,146 | 0,265 |
| | | Hızlanma | 99,42% | *99,48%* | 99,36% | 99,21% | 98,90% | 98,14% | 97,07% |
| | | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | H-Chr1 | WM | 422,13 | 425,13 | 454,66 | 474,49 | 480,45 | 483,31 | 492,79 |
| | | q-gram WM | 1,155 | 0,725 | 0,632 | 0,592 | 0,576 | 0,668 | 0,812 |
| | | Hızlanma | 99,73% | 99,83% | 99,86% | *99,88%* | 99,88% | 99,86% | 99,84% |
| d = 5000 | E. Coli | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | | WM | 16,052 | 13,395 | 13,669 | 14,937 | 14,314 | 14,444 | 16,038 |
| | | q-gram WM | 0,047 | 0,052 | 0,108 | 0,145 | 0,178 | 0,243 | 0,412 |
| | | Hızlanma | *99,71%* | 99,61% | 99,21% | 99,03% | 98,76% | 98,32% | 97,43% |
| | | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | H-Chr1 | WM | 746,18 | 769,17 | 817,11 | 840,45 | 852,54 | 856,64 | 858,23 |
| | | q-gram WM | 1,562 | 1,018 | 0,907 | 0,957 | 0,937 | 1,055 | 1,342 |
| | | Hızlanma | 99,79% | 99,87% | *99,89%* | *99,89%* | *99,89%* | 99,88% | 99,84% |

**Tablo 7.** Ortalama Karakter Karşılaştırma Sayısı (The Average Number of Character Comparisons)

| | | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
|---|---|---|---|---|---|---|---|---|---|
| d=1000 | E. Coli | WM | 407045960 | 410623080 | 412068230 | 404530510 | 418058000 | 400118770 | 415956560 |
| | | q-gram WM | 1360 | 1150 | 1670 | 1230 | 1000 | 1020 | 1130 |
| | | Artış | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* |
| | | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | H-Chr1 | WM | 21741678975 | 21072127896 | 22491502264 | 22029836324 | 21264368589 | 22144952500 | 21078145425 |
| | | q-gram WM | 245523 | 62736 | 43166 | 35173 | 15869 | 22753 | 27106 |
| | | Artış | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* |
| d = 3000 | E. Coli | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | | WM | 1221137880 | 1231869240 | 1236204690 | 1213591530 | 1254174000 | 1200356310 | 1247869680 |
| | | q-gram WM | 3848 | 3451 | 4923 | 3690 | 3000 | 3060 | 3385 |
| | | Artış | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* |
| | | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | H-Chr1 | WM | 65252283507 | 64756522381 | 66523632624 | 66729987205 | 63794694394 | 64925185209 | 64705630754 |
| | | q-gram WM | 1582368 | 468407 | 436603 | 419725 | 263990 | 335862 | 354549 |
| | | Artış | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* |
| d = 5000 | E. Coli | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | | WM | 2035229800 | 2053115400 | 2060341150 | 2022652550 | 2090290000 | 2000593850 | 2079782800 |
| | | q-gram WM | 6800 | 5750 | 8350 | 6150 | 5000 | 5100 | 5650 |
| | | Artış | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* |
| | | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | H-Chr1 | WM | 109058704220 | 107636088848 | 109417623815 | 110881363112 | 107764301694 | 107952998214 | 107261739728 |
| | | q-gram WM | 3341788 | 1209957 | 1143475 | 1318083 | 1023647 | 1083551 | 1239708 |
| | | Artış | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* | *99,99%* |

**Tablo 8.** Ortalama Hash Karşılaştırma Sayısı (The Average Number of Hash Comparisons)

| | | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
|---|---|---|---|---|---|---|---|---|---|
| d = 1000 | E. Coli | WM | 302854910 | 303186354 | 302057612 | 304696578 | 313148710 | 296888284 | 309030544 |
| | | q-gram WM | 516842 | 184911 | 87347 | 51392 | 23843 | 69030 | 144446 |
| | | Artış | 99,83% | 99,94% | 99,97% | 99,98% | *99,99%* | 99,98% | 99,95% |
| | | | | | | | | | |
| | | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | H-Chr1 | WM | 16032814087 | 15569879429 | 16540313554 | 16270721871 | 15705473323 | 16234129970 | 15541185642 |
| | | q-gram WM | 24246401 | 7720371 | 2743940 | 1346660 | 717470 | 1005180 | 2156399 |
| | | Artış | 99,84% | 99,95% | 99,98% | *99,99%* | *99,99%* | *99,99%* | *99,99%* |
| d = 3000 | | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | E. Coli | WM | 899192490 | 900186854 | 896800692 | 904717718 | 930074370 | 881293604 | 917721404 |
| | | q-gram WM | 516842 | 189451 | 107027 | 85832 | 104543 | 195510 | 134366 |
| | | Artış | 99,94% | 99,98% | *99,99%* | *99,99%* | *99,99%* | 99,98% | *99,99%* |
| | | | | | | | | | |
| | | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | H-Chr1 | WM | 47633740413 | 47314050896 | 48417121945 | 48736261072 | 46668453551 | 47389678746 | 47153217797 |
| | | q-gram WM | 23842049 | 7879008 | 5093809 | 5709032 | 5268299 | 9565394 | 20490907 |
| | | Artış | 99,95% | 99,98% | *99,99%* | *99,99%* | *99,99%* | 99,98% | 99,96% |
| d = 5000 | | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | E. Coli | WM | 1495530070 | 1497187354 | 1491543772 | 1504738858 | 1547000030 | 1465698924 | 1526412264 |
| | | q-gram WM | 516842 | 185811 | 92807 | 46772 | 47543 | 101809 | 203666 |
| | | Artış | 99,97% | 99,99% | 99,99% | *99,99%* | *99,99%* | 99,99% | 99,99% |
| | | | | | | | | | |
| | | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | H-Chr1 | WM | 79444520557 | 78571618521 | 79666891178 | 80732838579 | 78561742975 | 78649123023 | 78064299616 |
| | | q-gram WM | 25169961 | 10281457 | 9236167 | 13751304 | 1023647 | 27963012 | 65447603 |
| | | Artış | 99,97% | *99,99%* | *99,99%* | 99,98% | *99,99%* | 99,96% | 99,92% |

Deneysel çalışmalar sonucu ortalama çalışma zamanı açısından elde edilen hızlanma yüzdeleri gösterilmiştir ve en iyi hızlanma yüzdeleri koyu renkle gösterilmiştir. Tablo 9 önerilen yöntem ile iyi bilinen algoritmaların her bir veri seti için $d$ adet $m$ uzunluklu her bir desendeki ortalama çalışma zamanı sonuçlarını göstermektedirler.

E. Coli veri setinde, literatürdeki iyi bilinen algoritmalara kıyasla $q$-gram WM algoritmasında tüm $d = 1000,3000,5000$ adet $m$ uzunluklu ($m = 64,64,32$ uzunluklarında) desenlerde ortalama çalışma zamanında sırasıyla 98,01%, 98,95%, 99,22% hızlanmalar elde edilmiştir. Önerilen $q$-gram WM algoritmasının literatürdeki algoritmalardan daha verimli olduğu gösterilmiştir. H-Chr1 veri setinde, literatürdeki iyi bilinen algoritmalara kıyasla $q$-gram WM algoritmasında tüm $d = 1000,3000,5000$ adet $m$ uzunluklu desenlerde ortalama çalışma zamanında sırasıyla 99,61%, 99,76%, 99,77% hızlanmalar elde edilmiştir. Önerilen $q$-gram WM algoritmasının literatürdeki algoritmalardan daha verimli olduğu gösterilmiştir. AC algoritmasının $P$ desen kümesindeki herhangi bir desenin tüm oluşumlarını $T$ girdi dizisinde arama süresi, çoğu veri türü için alfabenin boyutu olarak ($|\Sigma|$) hesaplabilir [19]. Bu yüzden desen uzunluğu artmasına rağmen AC algoritmasında çalışma zamanı artsa da lineer bir artış göstermektedir. CW algoritması ise AC algoritmasının gelişmiş bir versiyonu olduğu için daha iyi sonuçlar elde edilmiştir. WM algoritması DNA sekanslarındaki hash çakışmalarından dolayı çalışma zamanı olarak en yavaş algoritmadır. Önerilen $q$-gram WM algoritması ise DNA sekanslarında eşsiz hash fonksiyonunu ve önerilen yaklaşımı kullandığından en verimli sonuçlar elde edilmiştir.

**Tablo 9.** Ortalama Çalışma Zamanı (The Average Runtime)

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| d = 1000 | E. Coli | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | | AC | 1,255 | 1,462 | 1,712 | 1,765 | 1,845 | 1,905 | 1,953 |
| | | CW | 1,196 | 1,332 | 1,554 | 1,611 | 1,715 | 1,756 | 1,824 |
| | | WM | 2,411 | 2,467 | 2,454 | 2,481 | 2,584 | 2,636 | 2,710 |
| | | q-gram WM | 0,034 | 0,032 | 0,031 | 0,039 | 0,046 | 0,066 | 0,105 |
| | | Hızlanma | 97,16% | 97,60% | *98,01%* | 97,58% | 97,32% | 96,24% | 94,24% |
| | H-Chr1 | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | | AC | 58,75 | 66,23 | 72,15 | 80,56 | 89,63 | 97,81 | 105,72 |
| | | CW | 55,21 | 61,71 | 68,63 | 75,65 | 80,21 | 89,56 | 96,32 |
| | | WM | 126,82 | 129,85 | 132,76 | 134,87 | 130,55 | 142,34 | 141,11 |
| | | q-gram WM | 0,663 | 0,465 | 0,377 | 0,335 | 0,314 | 0,367 | 0,385 |
| | | Hızlanma | 98,80% | 99,25% | 99,45% | 99,56% | *99,61%* | 99,59% | 99,60% |
| d = 3000 | E. Coli | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | | AC | 3,765 | 4,386 | 5,136 | 5,295 | 5,535 | 5,715 | 5,859 |
| | | CW | 3,588 | 3,996 | 4,662 | 4,833 | 5,145 | 5,268 | 5,472 |
| | | WM | 7,201 | 8,447 | 7,635 | 7,758 | 8,105 | 7,845 | 9,035 |
| | | q-gram WM | 0,042 | 0,044 | 0,049 | 0,061 | 0,089 | 0,146 | 0,265 |
| | | Hızlanma | 98,83% | 98,90% | *98,95%* | 98,74% | 98,27% | 97,23% | 95,16% |
| | H-Chr1 | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | | AC | 176,25 | 198,69 | 216,45 | 241,68 | 268,89 | 293,43 | 317,16 |
| | | CW | 165,63 | 185,13 | 205,89 | 226,95 | 240,63 | 268,68 | 288,96 |
| | | WM | 422,13 | 425,13 | 454,66 | 474,49 | 480,45 | 483,31 | 492,79 |
| | | q-gram WM | 1,155 | 0,725 | 0,632 | 0,592 | 0,576 | 0,668 | 0,812 |
| | | Hızlanma | 99,30% | 99,61% | 99,69% | 99,74% | *99,76%* | 99,75% | 99,72% |
| d = 5000 | E. Coli | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | | AC | 6,275 | 7,31 | 8,56 | 8,825 | 9,225 | 9,525 | 9,765 |
| | | CW | 5,98 | 6,66 | 7,77 | 8,055 | 8,575 | 8,78 | 9,12 |
| | | WM | 16,052 | 13,395 | 13,669 | 14,937 | 14,314 | 14,444 | 16,038 |
| | | q-gram WM | 0,047 | 0,052 | 0,108 | 0,145 | 0,178 | 0,243 | 0,412 |
| | | Hızlanma | 99,21% | *99,22%* | 98,61% | 98,20% | 97,92% | 97,23% | 95,48% |
| | H-Chr1 | | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| | | AC | 293,75 | 331,15 | 360,75 | 402,8 | 448,15 | 489,05 | 528,6 |
| | | CW | 276,05 | 308,55 | 343,15 | 378,25 | 401,05 | 447,8 | 481,6 |
| | | WM | 746,18 | 769,17 | 817,11 | 840,45 | 852,54 | 856,64 | 858,23 |
| | | q-gram WM | 1,562 | 1,018 | 0,907 | 0,957 | 0,937 | 1,055 | 1,342 |
| | | Hızlanma | 99,43% | 99,67% | 99,74% | 99,75% | *99,77%* | 99,76% | 99,72% |

## 5. Sonuçlar ve Tartışmalar (Results and Discussions)

Bu çalışmada DNA sekanslarında hash çakışmasını kaldıran eşsiz hash fonksiyonunu ve $q$-gram hash karşılaştırma yöntemini kullanarak WM algoritmasına dayalı $q$-gram WM çoklu dizi eşleştirme algoritması önerilmiştir. Kullanılan hash fonksiyonu sayesinde geleneksel WM algoritmasındaki sabit blok uzunluğu artırılarak ortalama çalışma zamanında, ortalama karakter karşılaştırma sayısında ve ortalama hash karşılaştırma sayısında daha iyi sonuçlar elde edilmiştir.

Ortalama çalışma süreleri için, WM algoritmasına kıyasen önerilen $q$-gram WM algoritmasında E. Coli veri setinde 98,73% - 99,71% arasında hızlanma, H-Chr1 veri setinde 99,76% - 99,89% arasında hızlanma elde edilmiştir. WM algoritmasına kıyasen önerilen $q$-gram WM algoritmasında tüm veri setlerinde ortalama karakter ve hash karşılaştırma sayılarında 99,99% azalma elde edilmiştir.

Böylelikle, tüm performans metriklerinde daha verimi bir algoritma önerildiği gösterilmiştir. Bu çalışmadaki literatür katkılarımızı aşağıdaki gibi sıralayabiliriz;

- DNA sekansları için WM çoklu dizi eşleştirme algoritmasında meydana gelen hash çakışmalarını ortadan kaldıran mükemmel hash fonksiyonu kullanılmıştır.
- Mükemmel hash fonksiyonu sayesinde WM algoritmasındaki sabit $B = 2$ blok uzunluğunu kullanmak yerine $q = 8$ aralığında daha büyük blok uzunluğu kullanılmıştır.

- Önek dizisi veya karakter eşleşmediğinde WM algoritmasındaki $B = 2$ birim desenleri kaydırma yerine $q = 8$ birim desenlerin kaydırılması sağlanmıştır.
- DNA sekanslarında aranan desenlerin $q$ uzunluklu sonek ve önek dizilerinin karşılaştırılmasına gerek kalmamış ve bu ekler arasındaki karakterler $q$-gram hash karşılaştırılmasıyla kontrol edilmiştir.
- Önek dizisinden sonra ilk karakter kontrolü yapılarak olası eşleşmeme olduğunda fazladan $q$-gram hash karşılaştırma maliyeti en aza indirilmiştir.
- Gelecekteki çalışmalarda dizi eşleştirme algoritmaları ile önerilen yaklaşım birleştirilerek hibrit modeller oluşturulabilir. Mükemmel hash fonksiyonunu kullanarak oluşturulan yeni modelin başarısı deneysel çalışmalarla doğrulanabilir.

## Kaynaklar (References)


1. Sukhanov S., Wu R., Debes C., Zoubir A. M., Dynamic pattern matching with multiple queries on large scale data streams, Signal Processing, 171, 107402, 2020.
2. Song S., Gu G., Ryu C., Faro S., Lecroq T., Park K., Fast algorithms for single and multiple pattern Cartesian tree matching, Theoretical Computer Science, 849, 47-63, 2021.
3. Aldwairi M., Hamzah A. Y., Jarrah M., MultiPLZW: a novel multiple patterns matching search in LZW-compressed data, Computer Communications, 145, 126-136, 2019.
4. Kumar S., Singh S., Khatoon A., Agarwal S., A Multiple String and Pattern Matching Algorithm Using Context-Free Grammar, Emerging Trends in Expert Applications and Security, 841, 97-102, 2019.
5. Singh M., Sharma V., ASCII based Sequential Multiple Pattern Matching Algorithm for High Level Cloning, International Journal of Advanced Computer Science And Applications, 8 (6), 271-276, 2017.
6. Faro S., Külekci M. O., Towards a Very Fast Multiple String Matching Algorithm for Short Patterns, Stringology, 78-91, 2013.
7. Ho T., Cho S., Oh S., Parallel multiple patterns matching schemes based on Cuckoo filter for deep packet inspection on graphics processing units, IET Inf. Secur, 12 (4), 381–388, 2018.
8. Nunes L. S. N., Bordim J. L., Ito Y., Nakano K., A Rabin-Karp Implementation for Handling Multiple Pattern-Matching on the GPU, IEICE Transactions on Information and Systems, 103 (12), 2412-2420, 2020.
9. Lai W. S., Wu C. C., Lai L. F., Sie M. C., Two-Phase PFAC Algorithm for Multiple Patterns Matching on CUDA GPUs, Electronics, 8 (3), 270, 2019.
10. Rasool A., Ahmed G. F., Barskar R., Khare N., Efficient multiple patterns matching algorithm based on BMH: MP-BMH, Int. Arab J. Inf. Technol., 16 (6), 1121-1130, 2019.
11. Lin C., Li J., Liu C., Chang S., Perfect Hashing Based Parallel Algorithms for Multiple String Matching on Graphic Processing Units, IEEE Transactions on Parallel and Distributed Systems, 28 (9), 2639-2650, 2017.
12. Karcioglu A. A., Bulut H., Improving hash-q exact string matching algorithm with perfect hashing for DNA sequences, Computers in Biology and Medicine, 131, 104292, 2021.
13. Hyyrö H., Juhola M., Vihinen M., On exact string matching of unique oligonucleotides, Computers in Biology and Medicine, 35 (2), 173-181, 2005.
14. Xu K., Yang Z., Kang P., Wang Q., Liu W., Document-level attention-based BiLSTM-CRF incorporating disease dictionary for disease named entity recognition, Computers in biology and medicine, 108, 122-132, 2019.
15. Kim H., Han Y. S., OMPPM: online multiple palindrome pattern matching, Bioinformatics, 32 (8), 1151-1157, 2016.
16. Yang X., Han G., Chen J., Cai H., Finding Correlated Patterns via High-Order Matching for Multiple Sourced Biological Data, IEEE Transactions on Biomedical Engineering, 66 (4), 1017-1025, 2019.
17. Al-Qiari A. K., Al-Issa Y., A fast improved multiple patterns matching algorithm, IEEE 9th International Conference on Information and Communication Systems (ICICS), Irbid-Jordan, 55-60,2018.
18. Aluru S., Futamura N., Mehrotra K., Parallel biological sequence comparison using prefix computations, Journal of Parallel and Distributed Computing, 63 (3), 264-272, 2003.
19. Nunes L. S., Bordim J. L., Ito Y., Nakano K., A Prefix-Sum-Based Rabin-Karp implementation for multiple patterns matching on GPGPU, International Symposium on Computing and Networking (CANDAR), Takayama-Japan,139-145, 2018.
20. Prakash S., Agarwal H., Agarwal U., Biswas P., Jaypee S. D., Discovering motifs in DNA sequences: A suffix tree based approach, 8th International Advance Computing Conference (IACC), Greater Noida-India, 327-332, 2018.
21. Tian Y., Tata S., Hankins R. A., Patel J. M., Practical methods for constructing suffix trees, The VLDB Journal, 14 (3), 281-299, 2005.
22. Allauzen C., Crochemore M., Raffinot M., Efficient experimental string matching by weak factor recognition, Annual Symposium on Combinatorial Pattern Matching, 51-72, 2001.
23. Lecroq T., Fast exact string matching algorithms, Information Processing Letters, 102 (6), 229-235, 2007.
24. Faro S., A very fast string matching algorithm based on condensed alphabets, International Conference on Algorithmic Applications in Management, 65-76, 2016.
25. Kouzinopoulos C. S., Margaritis K. G., Multiple pattern matching: Survey and experimental results, Neural, Parallel and Scientific Computation, 22, 563-593, 2014.
26. Aho A.V., Corasick M.J., Efficient string matching: an aid to bibliographic search, Communications of the ACM, 18 (6), 333–340, 1975.
27. Knuth D.E., Morris J.H, Pratt V.R., Fast pattern matching in strings, SIAM Journal on Computing, 6 (2), 323–350, 1977.
28. Commentz-Walter B., A string matching algorithm fast on the average, Proceedings of the 6th Colloquium on Automata Languages and Programming, 118–132, 1979.
29. Boyer R. S., Moore J. S., A fast string searching algorithm, Communications of the ACM, 20 (10), 762-772, 1977.
30. Navarro G., Raffinot M., Flexible pattern matching in strings: practical on-line search algorithms for texts and biological sequences, Cambridge university press, Cambridge-United Kingdom, 2002.
31. Wu S., Manber U., A fast algorithm for multi-pattern searching, Univ. Arizona, Tech. Rep, TR-94-171994, 1994.
32. Ferragina P., Manzini G., Indexing compressed text, Journal of the ACM (JACM), 52 (4), 552-581, 2005.
33. Özcan G., Ünsal, O. S., Fast bitwise pattern-matching algorithm for DNA sequences on modern hardware, Turkish Journal of Electrical Engineering & Computer Sciences, 23 (5), 1405-1417, 2015.
34. Özcan,G. Detection of P53 Consensus Sequence: A Novel String Matching With Classes Algorithm, Uludağ University Journal of The Faculty of Engineering, 21 (2), 269-282, 2016.
35. Karakış R., Gurkahraman K., Çiğdem B., Oztoprak I., Topaktas A. S., Evaluation of segmented brain regions for medical image steganography, Journal of the Faculty of Engineering and Architecture of Gazi University, 36 (4), 2301-2314, 2021.
36. Daş B., Türkoğlu İ., Classification of DNA sequences using numerical mapping techniques and Fourier
37. transformation, Journal of the Faculty of Engineering and Architecture of Gazi University, 31(4), 921-932, 2016.
38. Akalın F., Yumuşak N., Classification of exon and intron regions obtained using digital signal processing
39. techniques on the DNA genome sequencing with EfficientNetB7 architecture, Journal of the Faculty of Engineering and Architecture of Gazi University, 37 (3), 1355-1372, 2022.
40. Plunkett G., Escherichia coli, https://www.ncbi.nlm.nih.gov/nuccore/NC_010473.1.Yayın tarihi Şubat 20, 2008. Erişim tarihi Ocak 11, 2021.
41. Lander E. S., Homo sapiens chromosome, https://www.ncbi.nlm.nih.gov/nuccore/NC_000001.11.Yayın tarihi Ekim 26, 2006. Erişim tarihi Ocak 11, 2021.