ÇOK AMAÇLI GENETİK ALGORİTMALARLA ÇOKLU DİZİ HİZALAMA

Hülya HARK

Yüksek Lisans Tezi

Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Mehmet KAYA

AĞUSTOS-2012

T.C

FIRAT ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

ÇOK AMAÇLI GENETİK ALGORİTMALARLA ÇOKLU DİZİ HİZALAMA

YÜKSEK LİSANS TEZİ

Hülya HARK

(092129108)

Tezin Enstitüye Verildiği Tarih : 1 Ağustos 2012

Tezin Savunulduğu Tarih : 16 Ağustos 2012

Tez Danışmanı:	Doç. Dr. Mehmet KAYA
Diğer Jüri Üyeleri :	Doç. Dr. Arif GÜLTEN
	Yrd. Doç. Dr. Taner TUNCER

AĞUSTOS-2012

ÖNSÖZ

Yüksek Lisans eğitimimi yaptığım süre boyunca tez konumun belirlenmesi, şekillenmesi, yürütülmesi ve tez sürecinin düzenlenmesinde en derin bilgilerini ve bilim alanındaki tecrübelerini benden esirgemeyen, araştırma ve çalışma sevkini kazanmamı sağlayan danışman hocam Sayın Doç. Dr. Mehmet KAYA'ya teşekkürlerimi bir borç bilirim.

Hülya HARK

ELAZIĞ-2012

İÇİNDEKİLER

Sayfa No

ÖZET

Çoklu dizi hizalama moleküler biyoloji ve biyoinformatikte önemli bir problemdir. Bu tezde biyolojik dizilerin çoklu hizalanması genetik algoritma yardımıyla gerçekleştirilmiştir. Biyolojik diziler üzerinde işlem yapan birçok yöntem bulunmaktadır. Bu yöntemlerin birçoğu tek bir amaca ulaşmak için çalışmaktadırlar. Bazı problemlerde birden fazla amacı gerçekleştirmek gerekir. Bu gibi durumlarda tek amacı optimize eden yöntemler istenilen sonucu vermez.

Bu tezde önerilen yöntem çoklu dizi hizalamayı birden fazla amaç kullanarak gerçekleştirmektedir. Yöntem daha önce geliştirilmiş yöntemlerden bu açıdan farklılık göstermektedir. Geliştirilen bu algoritma iki amaç parametresine bağlı en iyi çözüm kümesini sunmaktadır. Bu amaçlar benzerlik ve hizalama uzunluğudur. BALIBASE veritabanı üzerinde yapılan deneysel sonuçlar, önerilen yöntemin yaygın olarak kullanılan diğer yaklaşımları ClustalW ve SAGA ile karşılaştırıldığında etkili çözümler verdiği gösterilmiştir.

Anahtar Kelimeler: Çoklu dizi hizalama, çok amaçlı genetik algoritma, biyoinformatik.

SUMMARY

Multiple Sequence Alignment by Multi-Objective Genetic Algorithms

Multiple sequence alignment is an important problem in molecular biology and bioinformatics. Multiple alignment of biological sequences using genetic algorithm was carried out in this thesis. There are many methods that operate on biological sequences. Many of these methods work to achieve a single purpose. Some problems need to perform more than one purpose. In such cases, methods are desired to optimize the sole purpose of this feature.

In this thesis, using the proposed method performs more than one purpose multiple sequence alignment. The method in this respect are different from the methods developed previously. This algorithm was developed with two objectives depend on the parameter set offers the best solution. These are similarity and alignment lenght. The experimental results conducted on BALIBASE demonstrated that with repect to proposed method gives the effective solutions two well known other methods, ClustalW and SAGA.

Keywords: Multiple sequence alignment**,** multi-objective genetic algorithm and bioinformatics.

ŞEKİLLER LİSTESİ

Sayfa No

TABLOLAR LİSTESİ

1. GİRİŞ

Biyolojik dizilerin çoklu hizalanması, moleküler biyoloji ve genom analizinde çok önemli bir araçtır. Bu ilişki DNA dizileri arasında soyağacı oluşturmaya yardımcı olmakla beraber işlevi ve yapısı önceden bilinen diğer diziler ile hizalayarak bilinmeyen protein dizilerinin işlevini ve yapısını tahmin etmede kullanılır. Çoklu dizi hizalamanın (ÇDH); yeni dizilerin ikincil ve üçüncül yapısını tahmin etmek, yeni diziler ve mevcut aileler arasındaki benzerliği göstermek, aileler için örüntü tanımlamanın bulunmasına yardımcı olmak, PCR primerleri ve soyağacının yeniden yapılandırılması gibi çalışmalarda kullanılması amaçlanmaktadır [1].

Optimizasyon, verilen amaç veya amaçlar için belirli kısıtlamaların sağlanarak en uygun çözümü bulma sürecidir. Optimizasyon problemlerinin çözümü için model geliştirilirken öncelikle doğrusal ve değişken sayısının az olduğu matematiksel modeller geliştirilmiş, ancak bunun her problemde kullanılamayacağı anlaşılmış ve doğrusal olmayan modellemelere gidilmiştir. Bu aşamada, doğal seçim stratejilerini kullanan evrimsel algoritmalar geliştirilmiştir [2]. Matematiksel modellemenin yapılamadığı veya kesin çözümün olmadığı problemlerde evrimsel algoritmalardan yararlanılır.

Evrimsel algoritmalardan olan genetik algoritmalar, doğal seçim ilkesine dayanan bir arama ve optimizasyon algoritmasıdır. Genetik algoritmanın temel ilkeleri John Holland [3] tarafından oluşturulduktan sonra, genetik algoritmalar üzerine birçok çalışma yapılmıştır. Geleneksel optimizasyon yöntemlerinden oldukça farklı olan genetik algoritmalar, parametre kümesinin kodlanmış biçimlerini kullanmaktadırlar. Genetik algoritmalar sadece amaç fonksiyonuna ihtiyaç duyar ve çözüm uzayının sadece bir bölümünde arama yaparlar. Bu sayede çözüme kısa sürede ulaşır ve popülasyonu eş zamanlı incelediği için yerel en iyi çözüme takılmazlar [4].

Gerçek dünya problemleri karmaşıktır ve en uygun çözümün elde edilmesi için birden fazla amacın sağlanması gerekir. Birçok proje tek bir amaca dayalı bir yaklaşımla çözüm getirirken, bu yaklaşım pek çok problem için mümkün olmayabilir. İlk olarak amaçlar birbiriyle çatışabilir, amaçlar aday çözüm kümesine uygun olmayabilir ve farklılıklar gösterebilir. Çok amaçlı genetik algoritmalarda bu iki durum minimize edilmektedir [5].

Çok amaçlı genetik algoritmalar temel genetik algoritmalardan geliştirilmiş olan bir optimizasyon tekniğidir.

Çoklu dizi hizalamanın çözümü için birçok algoritmalar ortaya atılmıştır, bu problem NP-sıkı () problem olduğu için daha iyi çözümün bulunmasına hala ihtiyaç vardır. Dinamik programlama ile uzunluğu kısa olan diziler için ÇDH problemini çözülebilir; ancak büyük uzunluktaki diziler için sistem kaynaklarını tüketmesinden dolayı iyi bir çözüm değildir [6]. Genetik algoritma tabanlı ÇDH probleminin çözümü mevcut aşamalı tekniklerden biridir. Aşamalı yöntemlerle karşılaştırıldığında dezavantajı aşamalı yöntemlere göre daha yavaş olmasıdır [7]. Daha önce yapılan çalışmalar göz önüne alınarak yapılacak olan bu çalışmada genetik algoritmada birden fazla amaç kullanılarak ÇDH probleminin çözümü yapılacaktır.

1.1. Bioinformatik, Dizi Hizalama ve Çoklu Dizi Hizalama

1.1.1. Bioinformatik:

DNA yapısının 1953 yılında çözümlenmesinden bugüne kadar moleküler biyoloji alanında büyük gelişmeler olmuştur. Biyomoleküler diziler üzerinde değişiklik yapılmasıyla büyük miktarlarda veri oluşturuldu ve halen de oluşturulmaya devam edilmektedir. Laboratuarlarda oluşturulan bu büyük çaplı verilerin incelenmesi zaman içerisinde yeni bir alanı ortaya çıkarmıştır. Biyolojik bilimlerle ilgilenen bilim adamları bu verileri bulan ve en fazla kullanan kişilerdir, fakat bu verilerin çok ve karmaşık olması matematik ve bilgisayar bilimleri gibi diğer alanların yardımına ihtiyaç duymuştur. Bu ihtiyaçlar sonucu oluşan yeni alan biyoloji ve enformatik kelimelerinin birleşiminden elde edilen bioinformatik kelimesiyle isimlendirilmiştir [8].

Bioinformatik genel olarak, moleküler biyolojideki problemlerin çözümünde matematik ve bilgisayar bilimlerinin kullanılması olarak tanımlanabilir. Bioinformatiğin amaçlarını DNA, RNA ve protein dizilerinin yapılarını ve fonksiyonlarını araştırmak, hastalıklara ve genetik bozukluklara çare üretebilmek, genetik hastalıkları tedavi edebilecek ilaçları üretebilmek şeklinde sıralanabilir. Bioinformatiğin gelişimi bilgisayarların moleküler biyolojide kullanımı üç boyutlu moleküler yapıların grafik temsili, moleküler dizilimler ve üç boyutlu moleküler yapı veri tabanları oluşturulması ile başlamıştır. Kısa sürede çok

yüksek miktarlarda veri üreten, endüstri düzeyinde gen ekspresyonu, protein-protein ilişkisi, biyolojik olarak aktif molekül araştırmaları, bakteri, maya, hayvan ve insan genom projeleri gibi biyolojik deneylerin doğurduğu talep sonucunda, bu alandaki bilişim uygulamaları büyük bir hızla gelişmiştir [9].

1.1.2. Dizi Hizalama

Biyoinformatikte dizi hizalaması, DNA, RNA veya protein dizilerini düzenlenerek benzer bölgelerin tespit edilmesi işlemidir. Bu bölgeler arasındaki benzerlik diziler arasında işlevsel, yapısal ve evrimsel bir benzerlik olduğunu gösterir. Hizalanmış olan nükleotid veya aminoasit kalıntı dizileri matrisin satırları olarak gösterilir. Dizideki kalıntıları temsil eden sembollerin arasına boşluk eklenerek, ardışık sütunlarda yer alan aynı veya benzer sembollerin bir hizada olması amaçlanmaktadır [9].

Hizalama yöntemlerinden global hizalamalarda, her dizideki her harfin hizalanması amaçlanır. Sorgu kümesindeki diziler birbirine benzer ve yaklaşık aynı uzunlukta olursa global hizalamalar en iyi sonuç verir. Bu yönteme dinamik programlamaya dayalı olan Needleman-Wunsch [10] algoritmasını örnek verebiliriz. Birbirine benzemeyen ama benzer bölgeler içerdiği tahmin edilen diziler için yerel hizalamalar daha yararlıdır. Benzer kısa dizi motiflerinin tespitinde yerel hizalamalar kullanılır, buna örnek olarak Smith-Waterman [11] algoritması da dinamik programlamaya dayalı bir yerel hizalama yöntemi verilebilir. Eğer diziler yeterince birbirine benziyorsa yerel ve genel hizalama sonuçları arasında bir fark olmaz. Bunların dışında birde hibrit yöntemler mevcuttur. Hibrit yöntemler dizilerin başı ve sonunu da kapsayan en iyi hizalamayı bulmaya çalışır. Dizilerden birinin sonu, diğerinin başı ile örtüşüyorsa bu özellikle yararlı olabilir. Bu durum için ne genel ne de yerel hizalama tamamen uygundur: Genel yöntem hizalamayı örtüşme bölgesinin dışına uzatmaya çalışırken, yerel yöntem ise örtüşme bölgesini yeterince kapsamayabilir [9].

1.1.3. Çoklu Dizi Hizalama

Üç ya da daha fazla dizinin hizalanmasını içerir. Önemli fonksiyonel kalıntıların tespiti, filogenetik analizi, yapısal tahmin gibi pek çok biyoinformatik uygulamaların önemli ilk adımını oluşturmaktadır. Birden fazla diziyi geleneksel çoklu dizi hizalama yöntemiyle çözmek mümkün olsa da bu problem NP-zor tipinde bir problemdir. Çok fazla sayıdaki dizilerin hizalanmasında çok farklı yöntemler geliştirilmiştir. Örneğin SAGA [12], bir amaç fonksiyonu verildiğinde çoklu hizalamayı optimize etmek için genetik algoritma kullanır [9]. POA [13] kısmi sıralı grafikler kullanarak çoklu hizalama geliştirir, MSA [14] branch and bound (dal ve sınır) tekniği kullanarak çoklu dizi hizalamayı oluşturur [15]. Çoklu dizi hizalaması ikili dizi hizalamanın bir uzantısı olarak kabul edilebilir. Çoklu hizalama yöntemi genellikle birbiriyle ilişkili olan bir grup dizideki korunmuş bölgeleri tespit etmede ve filogenetik ağaç oluşturulurken evrimsel bir ilişki ortaya koymak için kullanılmaktadır.

1.2.Tezde Geliştirilenler

Bu tezde geliştirilenler çok boyutlu ve çok fazla sayıdaki biyolojik dizilerin çok amaçlı genetik algoritmalar kullanılarak çoklu dizi hizalama işlemi için yeni bir yöntem geliştirmektir. Burada önemli olan, dizileri birbirine benzerliği en fazla olacak şekilde hizalamak, boşluk sayısını en aza indirmek ve buna bağlı olarak dizi uzunluğunu da minimum seviyede tutmaktır. Yani dizileri birlikte hizalarken boşluklar eklenecek ve bu boşluklar hizalanmış dizi uzunluğunu artıracaktır. Önerilen çok amaçlı genetik algoritma ile benzerlik ve hizalama uzunluğu amaçları birlikte ele alınacaktır. Bu sayede daha etkin bir hizalama yöntemi ile daha kaliteli çözümler bulunabilecektir.

1.3. Tezin Yapısı

Bu tezin ilerleyen bölümlerinden Bölüm 2 ve Bölüm 3'te Çoklu dizi hizalama, Çok amaçlı genetik algoritmalar hakkında genel bilgilere yer verilmektedir. Bölüm 4'te çoklu dizi hizalama için geliştirilen çok amaçlı genetik algoritma anlatılmaktadır. Geliştirilen algoritmanın uygulamada kullanımı ve elde edilen sonuçların değerlendirilmesi işlemlerine

Bölüm 5'te değinilirken, sonuç bölümünde Algoritmanın genel değerlendirmesi yapılarak tezin tamamlanması düşünülmektedir.

2. ÇOKLU DİZİ HİZALAMA

Çoklu dizi hizalama (ÇDH); dizideki semboller arasına boşlukların eklenmesiyle iki veya daha fazla dizinin hizalanmasına değinir. Amaç, diziler arasında eşleşen sembollerin sayısını eğer boşluk eklenmesine izin verildiyse minimum sayıda boşluk ekleyerek arttırmaktır. Bu problem birkaç alanda uygulanır; bunlar moleküler biyoloji, jeoloji ve bilgisayar bilimleridir. Özellikle biyolojide, yeni proteinlerin tasarımına yardımcı protein yapılarını analiz etmek için ve DNA dizilerine dayalı evrimsel ağaç oluşturmak için kullanılır [16].

Çoklu dizi hizalama moleküler dizi analizinin önemli bir parçasıdır. DNA, RNA veya protein örnekleri arasındaki benzerlikleri belirlemek ve ölçmek için kullanılmaktadır. Amaç nükleotid veya amino asit arasındaki benzerliği en üst düzeye çıkarmaktır. Bu bölgelerin benzer olması, diziler arasında işlevsel, yapısal veya evrimsel bir ilişki olduğu anlamına gelir. Filogenetik ağaçların yeniden yapılandırılması veya bilinmeyen bir proteinin yapısının tahmini noktasında önemli bir araçtır. ÇDH zor optimizasyon problemleri sınıfına girer [17].

DNA alanı, tekrarlı dizilerde çoklu dizi hizalama çalışmalarında ortaya çıkan bir etkendir. Bu DNA dizileri, genom boyunca pek çok tekrarlanan biyolojik fonksiyonu olmaksızın genellikle anlaşılmamıştır. Tekrarlar genellikle tüm dizi boyunca değildir, ancak eklemeler, silmeler, eşleştirmeler az sayıda birbirinden farklıdır. Örneğin, ALU tekrarı yaklaşık olarak 300 bp uzunluğunda ve insan genomunda 600.000'den fazla görülmektedir. Bir insan genomunun %60 kadarının bilinen bir biyolojik fonksiyonu olmadan tekrarlanan dizilerden oluştuğuna inanılmaktadır [17].

Bir çoklu hizalama protein dizilerini, verilen bir sütunda artıkların homojen olması amacıyla matris şeklinde düzenleyerek özdeş veya ortak bir işlevsel rol oynamaktadır.

Yakından ilişkili proteinlerin aslında eşdeğer olmalarına karşın dizi, yapı ve işlev evrimsel zamanla farklı kriterlerde farklı hizalamalara neden olabilmektedir. Manuel geliştirilen hizalamalar otomatik yöntemlerden üstün olmaya devam etmektedir. ÇDH araçları biyolojik doğruluğunu iyileştirmek için sürekli bir çaba içindedirler. Buna ek

olarak, en basit yüksek maliyet hesaplamalı algoritmalar, mevcut verilerin sırasını iyileştirmek için hız iyileştirmeleri ve bellek kullanımını motive etmektedirler [18].

Çoklu dizi hizalama, üssel zaman karmaşıklığı ile optimizasyon probleminin bir sınıfına ait, kombinasyonel problemler ile işlem yapmaktadır. N dizi sayısı ve L hizalanmış dizinin ortalama uzunluğu olsun bu durumda zaman karmaşıklığı $O(L^N)$ olarak gösterilir. Biyolojide diziler yüzlerce (protein), binlerce (RNA) ya da milyonlarca (DNA) birim uzunluğunda olabilir. Buradan, birkaç diziyi hizalamak için kabul edilemez büyüklükte bir zamana ihtiyaç olduğu sonucu çıkarılır. Farklı hizalamaları karşılaştırırken, bir fitness fonksiyon eşleşen sembollerin sayısına ve boşlukların sayısına ve büyüklüğüne göre tanımlanmaktadır [16].

ÇDH biyoinformatikte önemli görevlerden biridir. Moleküler biyoloji ya da biyoinformatikte birçok nükleotit dizisinin veya amino asit dizisinin eş zamanlı olarak hizalanmasında önemli bir araçtır, bu nükleik asit veya protein primer dizilerinin birleşiminden önemli benzerlik bölgelerinin tespitinde önemli bir rol oynamaktadır. Dizi hizalama ve ÇDH'nın temeli evrim teorisidir. Bu teoriye göre, evrim sırasında, mutasyon meydana gelir ve aileler arasında farklılıklar oluşturulur [19]. ÇDH giderek artan sayıdaki biyolojik model yöntemlerinin doğru olarak birleştirilmesine bağlıdır. Bu modeller filogenetik ağaçlar, yapı tahmini ve profilleri içermektedir. Mevcut çoklu dizi hizalamalardan hiçbiri optimum değildir. Çoklu dizi hizalamanın amacı birkaç dizi arasında benzer ilişkide olanların bir araya getirilmesidir [20].

Aşağıda verilen ATCCGCTTAC, CTCCTAG, ATCCGCTA, ve TCTCCGTAC 4 DNA dizisinin hizalanması gösterilmektedir ve - sembolü boşluk olarak belirtilmektedir:

-ATCCGCTTAC

-CT-C-CT-AG

A-TCCGCT-A-

TCTCCG-T-AC

2.1. Çoklu Dizi Hizalama Algoritmaları

Çoklu hizalamada çalışmanın amacına bağlı olarak genel veya yerel hizalama yapılabilir. Global hizalamada dizilerin bütün uzunluğunda hizalama yapılır, bu nedenle de

dizilerin tamamında benzerliği yüksek olanlar ele alınır. Bu yönteme örnek olarak Needleman-Wunsch [10] algoritması dizilere boşluklar ekleyerek iki dizinin global hizalanmasında dinamik programlamayı kullanmaktadır. Dizinin bütün uzunluğunda hizalama çalışmaları yerel alanlarda uyumsuzluğa neden olabilir, örneğin iki protein dizisinin global hizalanmasında, iki proteinin ortak paylaşılan alanı sınırlıdır; bunların N-terminal bölgesinde proteinlerinin tamamının uzunluk bilgisi kullanılır ve burada ortak etki alanları doğru hizalanmayacaktır. Smith-Waterman [11] algoritması yerel hizalama sonuçlarına ekstra bir durum ile dinamik programlama kullanarak bu soruna bir çözüm sağlar. Dinamik programlama maksimize edilmesi için gerekli olan belirli bir skor fonksiyonu için optimal hizalamayı matematiksel olarak garantilemektedir. İkiden fazla dizinin hizalanması durumunda dinamik programlama algoritmaları çok boyutlu uzaya genişletilebilir; bu nedenle üçten fazla dizide kullanımı gerekli zaman ve bellek hesaplamalarından dolayı doğru değildir. Dinamik programlama iyi bir hizalama sağladığı halde pratik kullanımı sağlanan yeni teknoloji ve bilgisayarlar ile mümkün değildir. Yeni sezgisel yaklaşımların farklı stratejiler kullanılarak geliştirilme nedeni budur [21].

Çoklu dizi hizalama yöntemleri 4 kategoride sınıflandırabilir:

Dinamik programlama: Bu yöntem optimale çok yakın çoklu hizalamada yüksek kalitede heuristik yöntemlerdir. Bu şekilde çoklu hizalamada hesaplanma zamanı ve bellek kullanımı uygun değildir [22]. Herhangi bir sayıdaki diziler için kullanılabilir. Bu yöntem iki dizinin hizalanmasında yaratılan matrisin n-boyutlu karşılığının inşasını gerektirir, burada n sorgu kümesindeki dizi sayısıdır. Bu yaklaşımda arama uzayı artan n ile üssel şekilde büyür ve dizi uzunluğuna da kuvvetle bağımlıdır. Bir ÇDH'da n dizi için işlem *O(ln)* sürede tamamlanır. Bu n dizi için genel optimumu bulmanın NP-sıkı problem olduğunu gösterir [23]. Bu yaklaşım ile sorgu kümesindeki her bir dizi çifti için dinamik programlama hizalamaları yapılır, sonra bu hizalamaların n-boyutlu kesişimi civarında n-li hizalama için arama yapılır. Bu ÇDH algoritması hizalamadaki her pozisyon için, karakter çiftlerinin toplamlarını optimize eder. Bu algoritma sayesinde yüksek sonuçlu hizalamalar yapılabilir. Sonuca odaklı çözümler için ideal değerler verebilir, fakat dizi sayısı arttıkça sonuca ulaşmak zorlaşır. İşlem basamakları ağırlaşır. Bu yüzden küçük veri kümelerinde kullanılır. Daha karmaşık yapıları çözmek için aşağıda açıklanan sezgisel yöntemler kullanılmalıdır.

Örnek: GAATTCAGTTA(dizi #1) GGATCGA (dizi #2) dizilerinin dinamik programlama ile hizalanması aşağıdaki gibidir:

Dinamik programlama 3 adımda gerçekleşir:

1. Başlangıç
2. Matris Doldurma (puanlama)
3. Geri izleme (hizalama)

Başlangıçta matrisin tüm değerleri 0 ile doldurulur.

		G	A	A	T	T	C	A	G	T	T	A
	0	0	0	0	0	0	0	0	0	0	0	0
G	0											
G	0											
A	0											
T	0											
C	0											
G	0											
A	0											

Şekil 2.**1**. Dinamik programlama 1

Matris aşağıdaki formülle doldurulur:

$\mathbf{M_{i,j}}$ **= MAXIMUM[**

$\mathbf{M_{i-1,\,j-1}}$ **+** $\mathbf{S_{i,j}}$ (köşegende eşleşen/eşleşmeyen),

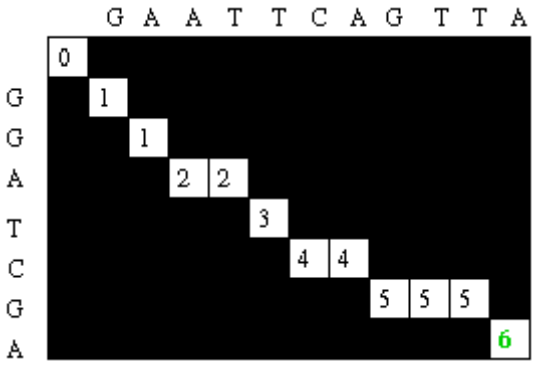
$\mathbf{M_{i,j-1}}$ **+ w** (dizi #1'deki boşluklar),

$\mathbf{M_{i-1,j}}$ **+ w** (dizi #2'deki boşluklar)**]**

		G	A	A	T	T	C	A	G	T	T	A
	0	0	0	0	0	0	0	0	0	0	0	0
G	0	1	1	1	1	1	1	1	1	1	1	1
G	0	1	1	1	1	1	1	1	2	2	2	2
A	0	1	1	2	2	2	2	2	2	2	2	3
T	0	1	2	2	3	3	3	3	3	3	3	3
C	0	1	2	2	3	3	4	4	4	4	4	4
G	0	1	2	2	3	3	4	4	5	5	5	5
A	0	1	2	3	3	3	4	5	5	5	5	**6**

Şekil 2.2. Dinamik programlama 2

Geri izleme adımının sonunda optimum hizalama aşağıdaki gibi belirlenir:



Şekil 2.3. Dinamik Programlama 3

G A A T T C A G T T A
| | | | | |
G G A _ T C _ G _ _ A olarak hizalanır.

Aşamalı Hizalama: Çoklu dizi hizalamasında en yaygın kullanılan yöntem, aşamalı yöntem olarak bilinen bir sezgisel aramadır [24]. Bu yöntemde, hizalamaya ilk önce en benzer dizi çiftlerinden başlanır, daha sonra gittikçe daha az benzeyen dizi çiftleri eklenir. Bu yöntem iki aşamadan oluşur: Birinci aşamada diziler arasındaki ilişkinin kılavuz ağaç denen bir filogenetik ağaç olarak oluşturulmasıdır. İkinci aşamada büyüyen ÇDH'ye dizilerin sırayla eklenerek ÇDH'nin inşa edilmesidir. İlk filogenetik ağacı oluşturmak için, verimli bir kümeleme yöntemi kullanılır. Bundaki amaç, herhangi bir adımda oluşan hataları sonuca dağıtmaktır. Aşamalı hizalamalar global optimal olamaz. Temel sorun, ÇDH oluşturulurken yapılan hataların nihai sonuca kadar taşınmasıdır. Bu yüzden ilk yapılan kıyaslama önemlidir. Diğer kıyaslamalar bunun üzerinden devam ettirilir ve sonuca direk etki eder. Aşamalı yöntemlerin yüksek kaliteli ilk hizalamalara muhtaç olmalarının nedeni, bu hizalamaların hep nihai sonuçta yer almasıdır. Yani bir dizi bir ÇDH içinde yerini aldıktan sonra onun hizalaması tekrar gözden geçirilmez. Bu yaklaşım doğruluktan kaybetme pahasına hızı artırır. Kümedeki diziler birbirlerine uzaktan ilişkiliyse algoritmanın performansı özellikle kötüdür. Çoğu modern aşamalı yöntemler, sorgu kümesinin her bir üyesi için skor fonksiyonlarını değiştirir. Bu değişken skor, dizilerin en yakın komşularına olan genetik uzaklığına bağlı olarak nonlineer değişen bir

ağırlık fonksiyonuyla hesaplanır. Böylece, hizalama programının dizileri rastgele olmayan bir şekilde seçmesinin etkisi düzeltilmiş olur [9].

Bu algoritmalar daha az bellek ve zaman kullanımıyla çoklu dizi hizalamada en yaygın olarak kullanılan algoritmalardır. Bu hizalamada ikili olarak diziler dinamik programlamayla eş zamanlı hizalanır. Tüm dizi çiftleri ikili olarak hizalanarak hesaplanır, hizalanmış bu diziler önceden hesaplanmış ağaca eklenir. Kısaca aşamalı algoritmalar:

- Bütün dizileri ikili olarak hizala
- İkili hizalama puanlarını kullanarak uzaklık matrisinin oluştur.
- Mesafe ağacını oluştur.
- En yakın iki diziyi hizala. Hizalamak için sonraki en yakın diziyi ekle.

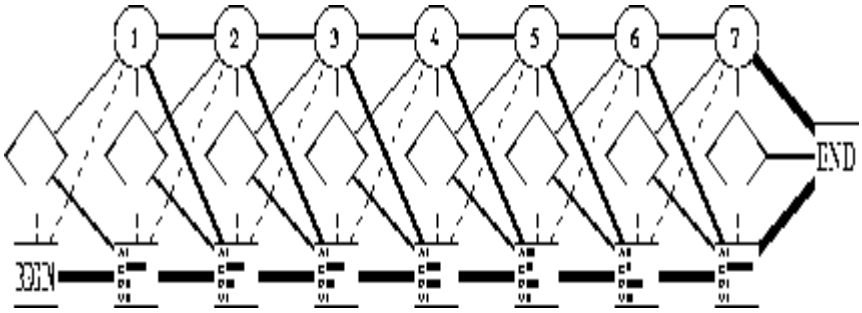
Ağaç tarafından belirtilen sıraya göre birden fazla hizalama eklenir ve aşamalı hizalama ile devam edilir. Aşamalı hizalama yöntemleri, çok sayıda diziye uygulanabilecek derecede verimli çalışırlar. Bu yöntem ClustalW ve T-Coffee programlarında kullanılmıştır. En popüler aşamalı hizalama yöntemi Clustal ailesi olmuştur. Ağırlıklı versiyonu olarak da özellikle ClustalW çok yaygındır [24].

İteratif Hizalama: Aşamalı yöntemlere özgü hataları azaltan yöntemler grubuna iteratif yöntemler denmiştir. Çünkü bu yöntem aşamalı yöntemlere benzer şekilde çalışmaktadır. Farkı ise büyüyen ÇDH'ya yeni diziler eklerken ilk dizileri tekrar tekrar hizalamaya çalışmasıdır. Aşamalı yöntemlerin zayıf bir yönü, ilk ikili hizalamanın doğru olmasına olan büyük bağımlılıktır. Yani bir dizi bir ÇDH içinde yerini aldıktan sonra onun hizalaması tekrar gözden geçirilmez. Buna karşın, iteratif yöntemler, daha önce hesaplanmış ikili hizalamalara veya sorgu dizisini içeren alt-ÇDH'ler geri gelebilirler. Bu sayede, yüksek kaliteli bir hizalama skoru elde etmeyi sağlayacak bir genel objektif fonksiyon optimize edilebilir [9].

İteratif yöntemlerde hizalamada daha iyi iyileştirmeler yapılıncaya kadar yapılan yinelemeler üretilen algoritmaya bağlıdır. İteratif yöntemler hizalama, geliştirmek için kullanılan bir stratejiye bağlı olarak ya da olasılıksal olabilir. Basit iteratif stratejiler belirleyicidir. Daha fazla geliştirme yapılamadığında prosedür sonlandırılır. Olasılıksal iteratif yöntemler genetik algoritmaları ve benzetilmiş tavlamaları içerir. İteratif

yöntemlerde en çok bilinen teknikler olarak benzetimsel tavlama, genetik algoritmalar, evrimsel programlama, Saklı Markov Modeli örnek verilebilmektedir.

Saklı Markov Model (Hidden Markov Model (SMM)): Bu model hizalanmış diziyi oluşturmak için eşleşen, eşleşmeyen ve boşlukların tüm kombinasyonlarını dikkate alan olasılıksal bir modeldir. SMM temelli yöntemler yakın zamanda geliştirilmiş olmasına rağmen önemli iyileştirmeler göstermiştir. SMM, protein modellerinde çalışmalar için geliştirilmiş istatiksel bir araçtır. Protein analizi için moleküler biyoloji, istatistik ve bilgisayar bilimleri ile aralarındaki ilişkilerin incelendiği biyoinformatik tabanlı yeni bilimsel alandır. Doğrusal Saklı Markov Modeli, çoklu dizi hizalamada her bir düğüme karşılık gelen bir dizidir. SMM'de her bir düğümün farklı anlamı vardır; eşleşme durumu kare şeklinde, ekleme durumu elmas şeklinde ve silme durumu ise daire şeklindedir. Bu modelin avantajı dizilerin bütün bir aile olarak karakterize etme özelliğine sahiptir [25].



Şekil 2.4. Saklı Markov Modelinin gösterimi [25]

2.1.1. ClustalW

Dizilerin tüm uzunlukları boyunca yada sadece belirli bölgelerinde hizalama yapılabilir. Çoklu dizi hizalama için en yaygın olarak Clustal serisi programlar kullanılmaktadır. İlk Clustal programı Des Higgins [24] tarafından geliştirilmiş ve bugünün standartlarına göre zayıf işlem gücü olan kişisel bilgisayarlarda verimli çalışması için tasarlanmıştır. Feng ve Doolittle [26] ve Taylor [27] tarafından aşamalı hizalama stratejileri ile dinamik programlamanın birleşimidir. Çoklu hizalama bir rehber ağacı dallarındaki sırayı izleyerek, ikili hizalamaların serisi tarafından oluşturulur. 1992'de ClustalV olarak adlandırılan Neighbour-Joining (NJ) yöntemini kullanarak çoklu hizalama sonucu üretilen ağaçlar ve hizalama profillerinin birleştirilmesiyle geliştirilen yeni bir sürümdür. Serinin üçüncü jenerasyonu, Thompson ve arkadaşları [28] tarafından gerçekleştirilen ClustalW (1994),

ilerici bir yaklaşımla global hizalama gerçekleştirir [29]. Varsayılan prosedür şu şekildedir.

Başlangıçta program sezgisel yaklaşım kullanarak bütün dizileri ikili olarak hizalar. Her hizalama puanlarının belirttiği benzerlik uzaklık ölçüsüne dönüşür ve mesafe matrisi oluşturulur. Bir sonraki adım Neighbour-Joining yöntemiyle uzaklık matrisinden bir ağaç oluşturmaktır. Bu metotta köksüz ağaç oluşturulur, böylece kök en büyük dallarının ortasına yerleştirilir. Her sırada çok benzer dizilerin bir gruba doğru eğilimini önlemek için ağırlık hesaplanır. Ağırlıklar dizinin kökten uzaklığına göre değişir; fakat diğer diziler ile birlikte ortak bir dalı olan dizilerin paylaşılan dalı türetilen ağırlığı paylaşabilirler. Bir kılavuz ağacını kullanarak aşamalı dizileri dinamik programlama ile hizalanır. Benzer diziler önce hizalanır ve sonra daha az benzer olanlar hizalanır. İlerici hizalama boyunca boşluk cezaları için karmaşık bir fonksiyon kullanılır. Boşluk açıklığı ve boşluk cezaları uzantısı kullanılan ağırlık matrisine bağlıdır. Hizalanmış dizilerin benzerliği, dizi uzunluğu ve bu tür korunmuş kalıntılara bağlıdır [29]. ClustalW'nin en önemli özelliği çok büyük sistem kaynaklarına ihtiyaç duymaması ve diğerlerine kıyasla kısa sürede sonuca ulaşmasıdır. Fakat bunu yaparken doğruluktan tavizler vermektedir.

2.1.2. T-Coffee

T-coffee "Ağaç Bazlı Tutarlılık Uyum Amaç Fonksiyonunun Değerlendirilmesi" anlamına gelir. Çoklu dizi hizalama için geliştirilen yeni bir programdır. Daha uzun çalışma zamanı gerektirse de diğer programlara göre daha kesin hizalama gerçekleştirmektedir. ClustalW gibi aşamalı bir yapıda olmasına rağmen aynı zamanda hizalanmış bütün dizileri ifade eden bütün dizi setiyle karşılaştırmaktadır. T-coffee ve ClustalW arasındaki ana fark Tcoffee'nin dizileri hizalamak için matris yer değişimini doğrudan kullanmamasıdır. Başlıca dizi ve yapıların hizalanmasında EXPRESSO, bir hizalamanın kesinliğini değerlendirmede CORE, birçok alternatif çoklu dizi hizalamalarını tek olarak birleştirmek için de Mcoffee gibi çok farklı uygulama ve modüle sahiptir. Kısaca, T-coffee DNA, RNA, protein dizileri ve yapılarının çoklu dizi hizalamasını kullanma, hesaplama ve değerlendirme için bir araç topluluğuymuş gibi tanımlanabilir. İkili hizalamaları hesaplamak için T-Coffee iki farklı yöntemle elde edilen hizalamaları birleştirir, bunlar dizi çiftlerinin doğrudan hizalanması ve çiftteki her diziyi üçüncü bir dizi

ile hizalanması ile elde edilen diğer hizalamalar. Elde edilen hizalama ve filogenetik ağaç, yeni ve daha doğru ağırlık faktörleri üretmek için kullanılır [30].

Avantajları

- Diğer metotlara göre daha kesin karşılaştırmalar ortaya koymaktadır.
- Yapı hizalama, hesaplama ve hizalama birleştirme için birçok modülle donatılmıştır.
- Birçok giriş formatını desteklemektedir. (Fasta, Swiss-Prot ve PIR gibi)
- Çeşitli formatlarda dizi hizalaması yapabilir. Başka programlarda giriş verisi olarak kullanılabilir.
- DNA, RNA ve protein dizileri listesiyle çalışabilmektedir.
- Core modülü ile herhangi bir hizalamanın sonucunu değerlendirebilir.

Dezavantajları

- Çoklu dizileri karşılaştırmada diğer programlardan daha uzun zaman almaktadır.
- ClustalW' ye göre daha az sayıda derlenmiş ve uygulanmaktadır.

2.1.3. DALİGN-T

Dalign-T çoklu dizi hizalama için segment tabanlı bir programdır [31]. Dalign ayırt edici özelliği, bir köşegen yöntemi (bu nedenle adı DALIGN) kullanarak hem yerel hem de global olarak dizileri hizalamasıdır. Tek tek artıkları karşılaştırmak yerine, nokta matriste köşegenleri oluşturacak kalıntıları karşılaştırır; boşluklara ya da eşleşmeyenlere izin vermemektedir. Sonuç olarak boşluk cezası ve boşluk uzantıları yoktur ve ilişkisiz dizileri hizalama dışında bırakabilir [32].

2.1.4. MUSCLE

MUSCLE yüksek boyutlu uygulamalar için alternatif seçenekler sunar. Muscle, hız ve doğruluk avantajlarına sahip olması yönüyle çoklu dizi hizalamada yeni bir jenerasyon olarak kabul edilir. Araştırmacılar MUSCLE algoritmasını ClustalW ve Tcoffee ile birlikte

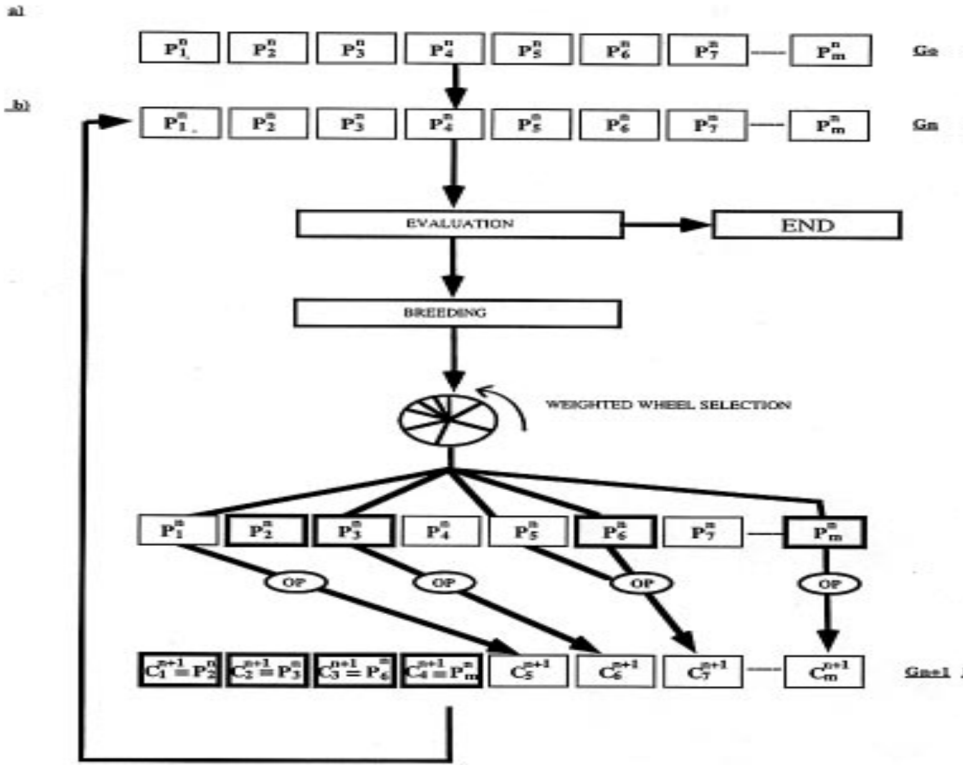
BAliBASE, SABmark, SMART ve PREFAB dört teste tabi tutmuşlardır ve araştırmacılar MUSCLE algoritmasının çok büyük boyutlardaki diziler üzerinde doğruluğu ve hızının en yüksek olduğu sonucuna ulaşmışlardır. Diğer çoklu dizi hizalama programlarından farklı olarak MUSCLE'da ne sadece aşamalı ne de sadece sezgisel yöntemi kullanmaktadır. Hizalamayı bulurken, aşamalı hizalama ve profil hizalamanın avantajlarını birleştirerek en olası yönlendirilmiş grafi bulmaktadır. MUSCLE, graf bulmayı çözümlenebilir bir şekilde gerçekleştirir [33]. MUSCLE ile iteratif yaklaşım, iki temel hizalama turu ile başlar ve sonra ağaç eşliğinde grup-grup hizalama, diziler arası uyum sağlanıncaya kadar tekrarlanır [34].

MUSCLE, tekrarlama temelli bir yöntem kullanır. İki dizinin yakınlığını belirlemek için daha doğruluklu bir uzaklık değeri hesaplayarak, aşamalı yöntemlerden daha yüksek bir başarı gösterir. Uzaklık ölçütü, tekrarlama aşamaları arasında yenilenir. Üç aşamada işlemlerini yapar. İlk aşamada aşamalı yöntem kullanılarak hizalama yapılır. Dizi çiftleri arasındaki benzerlik oranı ölçülür. İkinci aşamada uzaklık matrisi oluşturulur. Bu matrise göre ağaç oluşturulur. Bu ağaç üzerinden hizalamalar yeni fonksiyonlarla yeniden kümelenir. Üçüncü aşamada ise iteratif yöntemler kullanılarak ağaç parçalara ayrılarak yeni hizalamalar sağlanır. Aslında MUSCLE hem aşamalı yöntemleri, hem de iteratif yöntemleri beraber kullanarak sonuca ulaşmaya çalışmaktadır. Aslında bir çok aşamadan geçmesine rağmen süre açısından iyi sonuçlar vermektedir [33].

2.1.5. SAGA

Notrademe ve Higgins [35] tarafından çoklu dizi hizalama için evrimsel algoritma SAGA'yı 1996 yılında tanıtmışlardır. Temelde SAGA standart evrimsel algoritmalara benzer, hizalamaya aday popülasyonlar varyasyon ve seçime tabi tutulur. Seçim sırasında optimum bireyler bir sonraki nesle aktarılır. Diğer evrimsel algoritmalarla çoklu dizi hizalamanın aksine, SAGA boşluk bölgeleri ve umut verici bölgeleri çeşitli işlevlerle değiştirerek, toplamda 25 varyasyon(19 mutasyon ve 6 crossover) operatörü sağlar. SAGA içinde varsayılan başlatma modu ile her bir dizideki boşluk sayılarını random seçilmesiyle tüm bireyleri random olarak başlatır [36].

SAGA operatörlerin kullanımına ilişkin diğer tüm çoklu dizi hizalamalardan farklıdır. Kullanılacak operatörlerden hangisinin seçileceği 1989 yılında Davis [37] tarafından tanımlanan operatör planlama stratejisi kullanılır [36].



Şekil 2.5. SAGA algoritmasının planı a) başlangıç popülasyon, b) bir nesil döngüsü [35]

2.1.6. PRALİNE

PRofileALIgNEment (Praline) tamamen özelleştirilebilir çoklu dizi hizalama uygulamasıdır. Kullanılabilir hizalama stratejilerine ek olarak, Praline homojen çoklu dizi hizalamayı oluşturmak için homojen araştırma veri tabanlarından bilgi entegre edebilir. Praline ayrıca hizalama sürecinde yapısal bilgilerin bütünleştirilmesi için ayrı ayrı yada birlikte kullanılabilir yedi farklı ikincil yapı tahmini programlarından bir seçim sağlamaktadır [38].

2.1.7. PROBCONS

Olasılıksal modelleme ve tutarlılık amaçlı hizalama tekniklerini beraber kullanır ve hizalama yöntemleri içinde tutarlılık açısından en iyi sonuçları bulabilmektedir. PROBCONS algoritmasının temelinde çift posterior olasılık matrislerinin hesaplanması vardır. Her bir hizalamanın doğruluk olasılığı hesaplanır. P(xi ~ yj | x, y), x ve y dizileri hizalanarak xi ve yj karşılaştırılarak doğruluk olasılığının hesaplanmasıdır. PROBCONS

olasılıkların verimli hesaplanabilmesine olanak sağlayan basit bir olasılıksal model kullanır. Daha sonra posterior matrislerine olasılık tutarlılığı dönüşümü uygulanır. Bu değerler kullanılarak filogenetik ağaç oluşturulur. Bu filogenetik ağaç kullanılarak aşamalı olarak diziler hizalanır [39].

2.2. Mevcut Çoklu Dizi Hizalama Yöntemlerinin Performanslarının Karşılaştırılması

ÇDH yöntemlerini ve programlarının performans karşılaştırması BAliBASE, OXBENCH, PREFAB, SABMARK, IRMBASE gibi benchmark veritabanları kullanılarak yapılabilir. Aşağıdaki çeşitli sitelerden alınmış test sonuçları bulunmaktadır [40]:

IRMBASE 2 (Protein)

Sonuçlara bakılınca en yüksek değerlere PROBCONS'un ulaştığı görülmektedir. Ama yüksek değerler elde etmese bile süre açısından en avantajlı olan CLUSTALW olmuştur.

Korunmuş gül motifine ref1'de uzunluğu 400 olan ref2, ref3'te 500 ve ref4'te 600 olan rastgele diziler eklenmiştir.

Tablo 2.1. IRMBASE2 veritabanı kullanılarak ÇDH algoritmalarını karşılaştırma

Metot	ref1	ref2	ref3	ref4	Toplam	Süre
CLUSTAL W2	07.13 / 00.00	10.63 / 00.00	19.87 / 00.11	26.17 / 02.86	15.95 / 00.74	**1.86 sn**
T-COFFEE 5.56	72.67 / 34.84	77.80 / 40.87	83.03 / 43.62	83.48 / 49.56	79.24/ 42.22	26.41 sn
MUSCLE 3.7	32.67 / 04.65	34.82 / 06.87	54.19 / 14.80	57.84 / 19.65	44.88/ 11.49	6.34 sn
PROBCONS 1.12	**78.78 / 36.77**	**86.82 / 43.47**	**87.29 / 41.89**	**87.69 / 43.56**	**85.15 / 41.42**	28.27 sn

DIRMBASE 1 (DNA)

Sonuçlara bakılınca en yüksek değerlere MUSCLE ulaşmıştır. Ama yüksek değerler elde etmese bile süre açısından en avantajlı olan CLUSTALW olmuştur.

BALIBASE 3 (Protein)

BAliBASE 3 protein veritabanı üzerindeki benchmark sonuçlarına bakıldığında PROBCONS daha başarılı ama süre açısından bakıldığında yine en başarılı CLUSTALW olmuştur.

Tablo 2.2. DIRMBASE 1 veritabanı kullanılarak ÇDH algoritmalarını karşılaştırma

Metot	ref1	ref2	ref3	ref4	Toplam	Süre
CLUSTALW2	06.79 / 00.00	08.27 / 00.00	18.51 / 02.19	29.09 / 04.99	15.66 / 01.80	**1.36 sn**
T-COFFEE 5.56	14.71 / 00.00	18.88 / 00.18	32.08 / 04.01	43.39 / 08.44	27.62/ 03.16	365.88 sn
MUSCLE 3.7	**48.17 / 14.18**	**54.40 / 16.18**	**56.57 / 19.62**	**60.24 / 30.43**	**56.84/ 20.10**	4.87 sn
PROBCONSRNA 1.10	13.00 / 00.73	12.94 / 00.05	20.28 / 01.34	32.56 / 04.31	19.69/ 01.61	18.54 sn

Tablo 2.3. BAliBASE 3 veritabanı kullanılarak ÇDH algoritmalarını karşılaştırma

Metot	**RV11**	**RV12**	**RV20**	**RV30**	**RV40**	**RV50**	**Total**	**CPU-Time**
CLUSTALW2	50.06 / 22.74	86.43 / 71.59	85.16 / 21.98	72.50 / 27.23	78.93 / 39.55	74.24 / 30.75	75.36 / 37.35	**8.72 sn**
T-COFFEE 5.56	58.22 / 31.34	92.27 / 81.18	90.93 / 37.81	79.09 / 36.57	86.03 / 48.20	86.09 / 50.63	82.41/ 48.54	315.78 sn
MUSCLE 3.7	57.90 / 33.03	91.67 / 80.46	89.17 / 35.22	80.60 / 38.77	87.26 / 45.96	83.39 / 44.94	82.19/47.58	10.49 sn
PROBCONS 1.12	**66.99 / 41.68**	**94.12 / 85.52**	**91.68 / 40.49**	**84.61 / 54.37**	**90.24 / 52.90**	**89.28 / 56.50**	**86.40/55.66**	168.65 sn

Tablo 2.4. BAliBASE v.3 veritabanı kullanılarak ÇDH algoritmalarını karşılaştırma

Metot	**Ref11**	**Ref12**	**Ref2**	**Ref3**	**Ref4**	**Ref5**	**Toplam**	**Süre (s)**
Tutarlılığa Dayalı Yöntemler								
ProbCons 1.10 (varsayılan)	66.99/ 41.68	94.12/ 85.52	91.67/ 40.54	84.60/ 54.30	90.52/ 54.37	89.28/ 56.50	86.46/ 55.99	43,000
ProbCons 1.10 (düzenlenmiş)	66.73/ 41.47	94.13/ 85.38	91.85/ 42.00	84.47/ 54.03	89.79/ 51.94	89.34/ 57.69	86.27/ 55.71	(44,000)
TCoFfee 2.46	61.48/ 33.63	93.04/ 82.36	91.71/ 39.68	81.61/ 48.87	89.22/ 52.90	89.03/ 57.13	84.56/ 52.76	(210,000)

Yinelemeli Arıtma Yöntemleri								
Muscle 3.52 (doğruluk amaçlı)	56.62/ 30.87	90.96/ 79.59	88.90/ 35.17	81.07/ 37.87	85.90/ 45.06	85.17/ 46.19	81.67/ 46.79	3,400
ClustalW 2.0 (Yinelemeli ağaç)	49.94/ 25.08	88.91/ 75.32	85.80/ 21.61	72.78/ 30.43	81.20/ 40.84	76.49/ 35.06	76.67/ 39.58	(58,000)
Aşamalı Yöntemler								
Muscle 3.52 (hız amaçlı)	53.36/ 26.97	88.79/ 72.32	86.39/ 29.37	77.74/ 32.93	79.38/ 34.47	76.59/ 35.56	77.63/ 39.71	160
ClustalW 1.83	50.06/ 22.74	86.43/ 71.14	85.20/ 21.98	72.50/ 27.23	78.82/ 39.55	74.244/ 30.75	75.34/ 37.35	2,000

BAliBASE

Önceliklere göre bazı uygunluk değerlerini değiştirerek farklı sonuçlar elde edebilmektedir. Bu değerler değiştirildiğinde programların verdiği sonuçlar değişebilmektedir. Aynı zamanda hizalama yöntemlerindeki değişikliklerde sonucu etkilemektedir.

PREFAB v. 3

Benzerlik düşük olduğunda metotlar arasındaki farklar artmaktadır. Düzgün hizalanmış sitelerin algoritma karşılaştırmaları mevcut metotlar için gösterilmiştir.

Tablo 2.5. PREFAB v.3 veritabanı kullanılarak ÇDH algoritmalarını karşılaştırma

Metot	**%id: 0-20**	**%id: 20-40**	**%id: 40-70**	**%id: 70-100**	**Toplam**	**Zaman (s)**
ProbCons 1.10	47.58	83.43	96.14	98.10	69.43	210,000
TCoffee 2.46	44.51	81.84	95.46	98.63	67.42	
Muscle 3.52 (most accurate option)	43.06	80.60	94.90	98.44	66.26	8,600
ClustalW 1.83	33.96	74.14	93.54	97.85	59.45	13,000

HOMSTRAD

Düzinelerce ve yüzlerce dizi beraber hizalanmak istendiğinde yine farklı sonuçlar çıkacaktır. HOMSTRAD veritabanı 130 protein ailesini ve 590 hizalı yapıyı içerir. Homolog ailelerden oluşur. n homolog sayılarını belirtir. Burada hizalamanın doğruluğunu belirlemek için referans hizalamayla ve bulunan hizalama karşılaştırılarak puanlama yapılmıştır.

Tablo 2.6. HOMSTRAD veritabanı kullanılarak ÇDH algoritmalarını karşılaştırma

Metot	**n=0**	**n=20**	**n=50**	**n=100**
ProbCons 1.10	**46.65**	**50.17**	**51.07**	**51.62**
TCoffee 2.46	42.20	47.39	48.80	48.50
Muscle 3.52 (doğruluk amaçlı)	43.14	46.39	47.05	48.87
ClustalW 1.83	36.77	36.57	37.33	36.77

OXBENCH

OXBENCH veri ve yazılımı içinde barındırarak ÇDH için doğruluk hesaplamalarında kullanılan bir araçtır. Farklı hizalama yöntemleri için test kriterleri olan bir benchmark paketidir. OXBENCH kullanılarak elde edilen sonuçlar.

Tablo 2.7. OXBENCH veritabanı kullanılarak ÇDH algoritmalarını karşılaştırma

Metot	Ortalama ~5.7 sequence	Ortalama ~99 sequence	Ortalama ~6.0 sequence
Muscle 3.52	84.37 / 7.306	86.76 / 7.339	74.22 / 6.750
ProbCons 1.10	84.12 / 7.261	87.79 / 7.108	75.14 / 6.753
ClustalW 1.83	83.80 / 7.300	83.29 / 7.178	72.68 / 6.691

ÇDH hizalama için farklı web tabanlı programları kullanabilmektedir. Bu siteden [41] aşağıdaki dizileri kullanarak CLustalW2, Muscle ve TCoffee kullanarak elde edilen sonuçlar verilmiştir. Diziler kısa olduğu için sonuçlar birbirine yakın elde edilmiştir.

>plas_horvu

DVLLGANGGVLVFEPNDFSVKAGETITFKNNAGYPHNVVFDEDAVPSGVDVSK ISQEEYL

TAPGETFSVTLTVPGTYGFYCEPHAGAGMVGKVTV

>plas_chlre

VKLGADSGALEFVPKTLTIKSGETVNFVNNAGFPHNIVFDEDAIPSGVNADAISR DDYLN

APGETYSVKLTAAGEYGYYCEPHQGAGMVGKIIV

>plas_anava

VKLGSDKGLLVFEPAKLTIKPGDTVEFLNNKVPPHNVVFDAALNPAKSADLAK SLSHKQL

LMSPGQSTSTTFPADAPAGEYTFYCEPHRGAGMVGKITV

>plas_proho

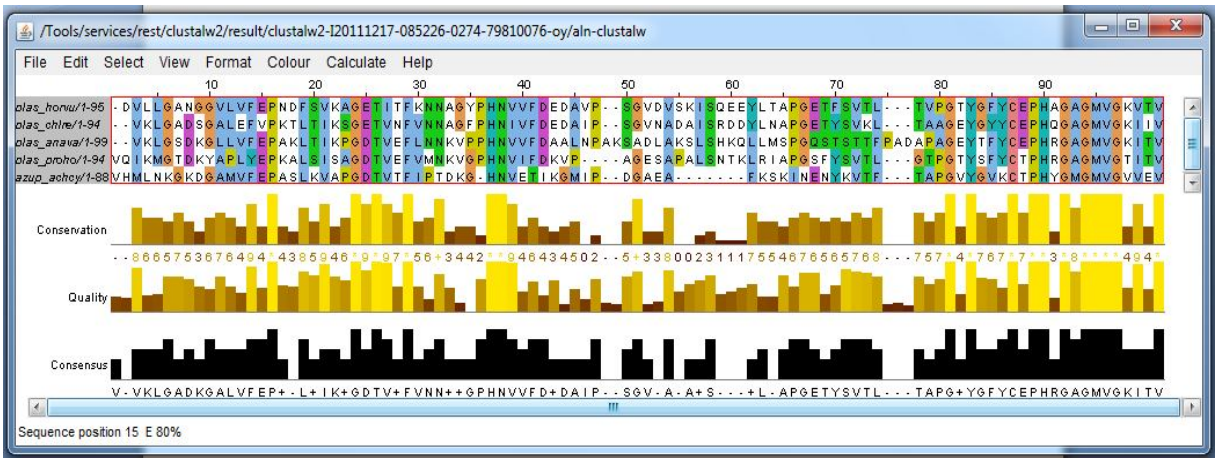
VQIKMGTDKYAPLYEPKALSISAGDTVEFVMNKVGPHNVIFDKVPAGESAPAL SNTKLRI

APGSFYSVTLGTPGTYSFYCTPHRGAGMVGTITV

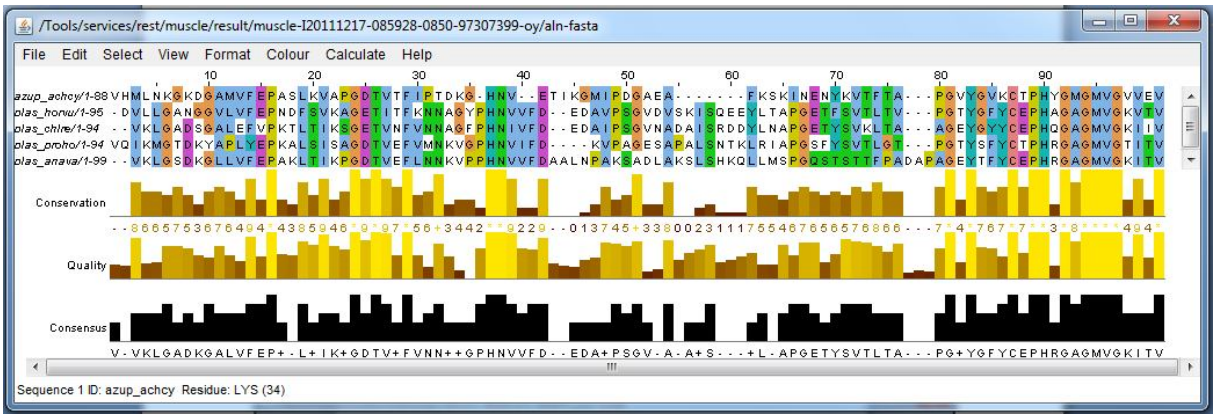
>azup_achcy

VHMLNKGKDGAMVFEPASLKVAPGDTVTFIPTDKGHNVETIKGMIPDGAEAFK SKINENY

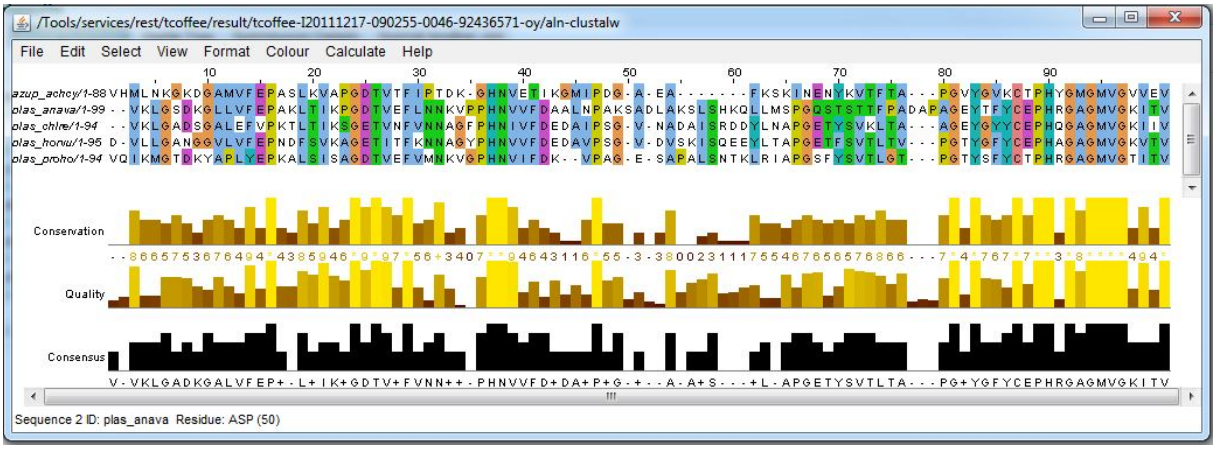
KVTFTAPGVYGVKCTPHYGMGMVGVVEV



Şekil 2.6 CLUSTALW2 Algoritması ile hizalama



Şekil 2.7. MUSCLE Algoritması ile hizalama



Şekil 2.8. T-COFFEE Algoritması ile hizalama

3. ÇOK AMAÇLI GENETİK ALGORİTMALAR

3.1. Optimizasyon

Doğal seçim ve optimizasyon arasındaki benzerlikler; evrimsel sürecin bilgisayar ortamında gerçekleştirilmesi "Evrimsel algoritmaların" geliştirilmesinde büyük etkisi vardır. Çok amaçlı optimizasyon yöntemine dayalı olan algoritmaların kullanımı gün geçtikçe yaygınlığı artmaktadır. Çok amaçlı optimizasyonla tek amaçlı optimizasyon arasında büyük farklılıklar vardır. Tek- amaçlı optimizasyonda, bir amaç için tek bir tasarım veya karar elde edilmeye çalışılırken, çok amaçlı optimizasyonda sonuç kümesi elde edilerek çözümler arasından seçim yapılır [2].

3.1.1. Tek Amaçlı Optimizasyon

Tek amaçlı optimizasyon verilen bir karar değişken maliyet fonksiyonu olarak tanımlanan karar değişkenlerinin bir dizi optimizasyonunu içerir.

Matematiksel olarak genel bir single-objective problem(SOP) aşağıdaki gibi tanımlanabilir:

$$Min\ z = f(x), \quad x=(x_1,x_2,.....,x_n) \in X, \tag{3.1}$$

x bir n-boyutlu kara vektör veya çözüm ise, tüm çözüm ifadelerinin X karar uzayıdır. Amaç fonksiyon f $\mathbb{R}$ içinden X haritalar ve z=f(x) amaç değeri çağırır. X amacının uzay görüntüsü, Z'de ulaşılabilen tüm amaç değerlerin kümesidir. En az bir amaç değer z, elde edilen bir veya birkaç çözümde vardır [42].

3.1.2. Çok Amaçlı Optimizasyon

Çok amaçlı optimizasyon, bir grup karar değişkeninin amaçlı fonksiyonlarının eş zamanlı optimizasyonunu içerir. Genel anlamda, amaçlı fonksiyonlar, çözümlerin farklı açılarının tahmin eder; bu nedenle kıyaslanamaz ve sık sık(kısmen veya tamamen) çelişki içindedir. Genel (serbest) çok amaçlı optimizasyon şöyle belirtilir:

‘minimize’ z=f(x)=(f1(x), f2(x),…., fm(x))

ve $x=(x_1, x_2, \ldots..x_n) \in X$ (3.2)

x n-boyutlu karar vektörü veya çözümü ve X kararlar uzayı, başka bir deyişle tüm anlatılabilir çözümlerin grubudur. Amaçlı fonksiyon vektörü f(x), amaçların sayısının $m \geq 2$ olduğu yerde X’i R^m’in içinde eşleştirir. z=f(x) vektörü amaçlı vektör veya noktadır. Amaç uzayındaki X’in görüntüsü tüm elde edilebilir noktaların grubu Z’dir. Burada ‘minimize’ dönemi alıntı işaretlerin üstünde belirir çünkü anlamı henüz belirtilmemiştir. Başka minimizasyon sorunları bulunur, kapsamı: amaçların düzenlenmesi ve sözlük sıralaması. (Örneğin ana kademesi belirlenmiş altın madalyaların sayısında bulunan ve bu düzenleniş içinde berabere bittiğine karar verilmiş gümüş madalyaların sayısında bulunan ve son olarak bronz madalyaların sayısında bulunan olimpik oyunlar madalya tabloları); en kötü amaçlı fonksiyonu en iyi şekilde kullanmak bunlardan ikisinin veya diğerlerinin kombinasyonudur [43]. Bununla birlikte, ’minimize’nin en alışılmış anlayışından uzak, dışında Pareto ölçütüdür. Pareto ölçütü, eş zamanlı kötüleşen diğer amaçların dışında bazı amaçları geliştirmesi mümkün olmayan tüm bu çözümler grubundan oluşur [42].

$$X^*=\{x^* \in X \mid \nexists x \in X, f(x) \leq f(x*)\}$$

$$f(x^1) \leq f(x^2) \; eğer \; \forall i \in 1 \ldots. m, fi(x^1) \leq fi(x^2) \wedge$$

$$\exists j \in 1 \ldots. m, fj(x^1) < fj(x^2) \quad (3.3)$$

Amaç uzayında Pareto’nun yerini tutan noktalar, belirlenmiş baskın olmayan ve Pareto cephesi biçimidir [42]. Bazı durumlarda Pareto grubu birden fazla unsur içerir çünkü sorunlara farklı anlaşmalar öneren farklı takas çözümleri bulunur. Böylece gerçekte, çok amaçlı sorun çözümü bazen Pareto’yu çözüm olarak seçen karar alan insanları içerir. Karar verme metodu, geniş olarak bilim/operasyon yönetim dalında incelenmiş olup, çok ölçütlü karar verici olarak bilinir bunun yanında verileri görüntüleme metotları içerebilir [44]. Sonradan gelen ve etkileşimli karar verme metotlarının genellikle karar verici takas çözümlerinin mümkün olduğunu görmekle yardımcı olarak daha etkileyici olduğu düşünülmektedir. Bu durum, tüm Pareto setinin üretimi için metotların filizlenmesine yol

açar. Giderek literatürde genel bir anlam olan çok amaçlı optimizasyon, Pareto kümesinin oluşumu anlamında kullanıldığı halde aslında çok amaçlı programlama diye adlandırılacaktır [45]. Bazen ilgi optimizasyon problemi tam bir yaklaşımı kabul etmeyebilir veya Pareto kümesi katlanarak genişleyebilir, bu durumda Pareto kümesini numaralandırmak mümkün değildir. Benzer durumlarda Pareto kümesine yakın bir şey, istenenin yerine [42]:

$$A^*=\{x^* \in A \subseteq X | \nexists x \in A, f(x) \leq f(x^*)\}. \quad (3.4)$$

Bu tanımdan eğer A=X ise bunun doğru pareto kümesi olduğu, fakat A⊂X olduğunda bunun gereksiz olduğunu görülmektedir. Buna yakın kümeler niteliklerinin dönemlerine göre kısmen düzenlenmiştir [42].

Son olarak gözden geçirilecek olunursa, çok amaçlı optimizasyonun hiçbir metodunun ayrı tutulmaması gerekmektedir, bunu yapmak bakış açımızı boş yere daraltmak olacaktır. Bununla birlikte, tüm Pareto kümesini meydana getiren yaklaşımlar, bizim burada dikkate aldığımız literatürdeki uygulama alanlarında kullanılmıştır. Bu nedenle, "çok amaçlı optimizasyon" terimi kullanıldığında aksi belirlenmedikçe bu durum gerçekleşecektir [42].

3.2. Çok Amaçlı Genetik Algoritmalar (MOGA)

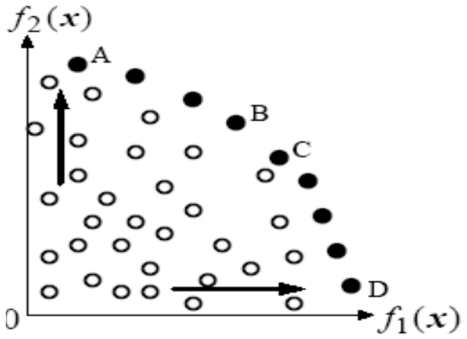
Popülasyon-tabanlı bir yaklaşım olarak, GA'ları ve çok amaçlı optimizasyon problemlerini çözmek için Çok Amaçlı Genetik Algoritmalar kullanılmaktadır. Genel tek amaçlı GA'lar tek bir vadede birden fazla baskın olmayan bir dizi çözüm bulmak için değiştirilebilir [46]. Çok amaçlı optimizasyon problemlerin çözümünde genetik algoritmaları kullanmak oldukça avantajlıdır. Bu yüzden çok amaçlı optimizasyon problemlerinin gün geçtikçe kullanımı yaygınlaşmaktadır [2].

Çok amaçlı GA'ların ilki Vektör Değerlendirmeli Genetik Algoritma (VEGA), daha sonra geliştirilen algoritmaları Çok Amaçlı Genetik Algoritmalar (MOGA), Hücrelendirilmiş Pareto Genetik Algoritması (NPGA), Ağırlık Tabanlı Genetik Algoritma (WBGA), Rastgele Ağırlıklandırılmış Genetik Algoritma (RWGA), Bastırılmamış

Sınıflandırmalı Genetik Algoritma (NSGA), Kuvvet Pareto Evrimsel Algoritması (SPEA) , Pareto-Arşivlenmiş Evrim Stratejisi (PAES), Pareto Zarflama-Temelli Seçim Algoritması(PESA), Bölge Tabanlı Seçim Evrimsel Çok Amaçlı Optimizasyon (PESA-II), Bastırılmamış Sınıflandırmalı Genetik Algoritma II (NSGA- II), Çok Amaçlı Evrimsel Algoritma (MOEA), Micro-GA, Rank-Yoğunluk Tabanlı Genetik Algoritma (RDGA), ve Dinamik Çok Amaçlı Evrimsel Algoritma (DMOEA) şeklinde sıralanabilir. Literatürde çok amaçlı genetik algoritmaların birçok türü vardır, ancak burada listelenenler birçok çalışmada uygulanmış güvenilir algoritmalardır [46]. Bu algoritmalardan bazıları aşağıdaki gibidir:

1. *Vektör Değerlendirmeli Genetik Algoritma (VEGA):* J boyutlu popülasyonda I tane amaç fonksiyonu olan bir problemin, her biri J/I boyutu olan alt popülasyonlarla birlikte ele alınır ve yeni bir J boyutlu popülasyon elde edilir. Çaprazlama mutasyon gibi genetik operatörler bu yeni J boyutu olan popülasyon üzerinden uygulanır.

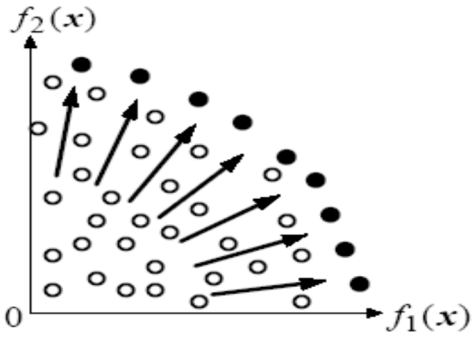
   Algoritmanın n tane alt popülasyonunu şekillendirecek olan seçim işlemi n amacın her birine göre ayrı ayrı gerçekleştirilir. Bu nedenle algoritma n tane arama yönüne sahiptir. Maksimize edilecek olan iki amaçlı bir optimizasyon için arama yönleri şekil 3.1'de gösterilmektedir. Bu yaklaşımda A ve D çözümü kolaylıkla bulunurken B ve çözümlerinin bulunması kolay değildir [2].



Şekil 3.1. Vega Yaklaşımında Arama Yönleri [2]

VEGA bazı problemlerde başarılı sonuçlar verse de, amaç uzayın eksenlerine paralel olarak arama yapmasından dolayı sadece uç değer çözümleri bulabilmiştir.

2. *Çok Amaçlı Genetik Algoritma* (*MOGA*): Çok kriterli optimizasyon problemlerinde amaç, birbiriyle çelişen çok amaçlı fonksiyonlar arasında çok yönlü arama gerçekleştiren Tadahiko Murata [47] çok amaçlı genetik algoritmayı önermiştir. Bu yaklaşımın tek amaçlı genetik algoritmadan farkı seçim süreci ve seçkinliğini koruma stratejisidir [2].



Şekil 3.2. MOGA'nın farklı yönlerde arama yapması [47]

n- kriterli optimizasyon problemi için GA kullanıldığında, her bir çözüm için n tane amaç fonksiyon değerlendirilir ve n tane amaç fonksiyonun değerleri kullanılarak bireylerin uygunluk değerleri tanımlanmalıdır. GA'lar tek amaçlıdaki optimizasyondaki gibi en iyi uygunluk değerine sahip olan bireyi ararlarken, amaç fonksiyonun değerlerini her bir bireyin uygunluk değerine dönüştürmek için sayısal fonksiyon içinde birleştirme yöntemi kullanırlar [2].

$$f(x)=w_1f_1(x)+w_2f_2(x)+.....+w_nf_n(x) \quad (3.5)$$

f(x)'in uygunluk fonksiyonu ve $w_1,\ldots w_n$ amaç için negatif olmayan ağırlıkları aşağıdaki gibi tanımlayabiliriz:

$$w_i\geq 0 \quad i=1,2,\ldots n \text{ için}$$

$$w_1+w_2+....+w_n=1 \quad (3.6)$$

3. *Hücrelendirilmiş Pareto Genetik Algoritması (NPGA):* Bu algoritma Jeffrey Horn ve Nicholas Nafpliotis [48] tarafından önerilmiştir. Genetik algoritmalarda seçim işlemi popülasyondaki bireylerden hangilerinin seçilip genotip bilgisinin sonraki jenerasyona aktarılacağı ile ilgilidir [2]. Burada sadece iki kişi rastgele karşılaştırma için seçilir. Kazanan çözüm, popülasyondaki diğer bireylerin bir dizi karşılaştırma kümesinden random seçilir. Sonra, karşılaştırma

kümesi ile ilgili olarak her iki aday hakimiyeti test edilir. Sadece bir aday, karşılaştırma kümesinde baskın çıkarsa, o kazanan olarak seçilir. Aksi takdirde kazanan adayı belirlemek için paylaşım prosedürü uygulanması gerekmektedir.

4. *Ağırlık Tabanlı Genetik Algoritma (WBGA):* Hajela ve Lin [49] tarafından 1993 yılında geliştirilen çok amaçlı optimizasyon için kullanılan genetik algoritmadır. Her bir amaç fonksiyon f_i, ağırlık w_i tarafından çarpılır. Bir GA dizisi sunulurken karar değişkeni x_i, onların ağırlığı w_i ile ilişkilidir. Ağırlıklı amaç fonksiyon değerleri, çözümün fitness değerleri hesaplanırken eklenir. GA popülasyonundaki her bir birey farklı ağırlık vektörü tarafından atanır. WBGA'da en önemli unsur populasyon üyeleri arasında ağırlık vektörlerin çeşitliliğini muhafaza etmektir. WBGA'in ağırlığı vektör içinde çeşitliliği iki yolla sağlanır [50]:

- Sadece sub-stringlerin ağırlık vektörlerinin sunulduğu metot
- Alt-popülasyonlardan, önceden tanımlanan farklı ağırlık vektörleri özenle seçilen, VEGA benzeri bir yaklaşım

5. *Bastırılmamış Sınıflandırmalı Genetik Algoritma (NSGA):* 1994'te Srivinas ve Deb [51] tarafından geliştirilmiştir. Basit genetik algoritmalardan ayrıln tek özelliği çok amaçlı genetik algoritmalarda olduğu gibi seçim işleminin uygulanmaktadır. Çaprazlama ve mutasyon işlemleri bilinen şekliyle uygulanır. Seçme işleminden önce popülasyon bireylerin Pareto üstünlüklerine göre sıralanır ve popülasyondaki bastırılmamış bireyler eldeki popülasyondan tespit edilir ve bu bireylerin ilk pareto yüzeyleri oluşturduğu varsayılır. Daha sonra bireylere sahte uygunluk değeri atanarak, aynı uygunluk değerinin tüm bireylere eşit üretkenlikte potansiyel verdiği varsayılır. Popülasyondaki çeşitliliği elde ederken bu sınıflandırılmamış olan bireyler daha sonra sahte uygunluk değerleri ile paylaşılırlar. Paylaşım; bireyin asıl uygunluk değerinin birey sayısına bölünmesinden elde edilir. Daha sonra indirgenmiş uygunluk değeri kullanılarak seçme işlemi gerçekleştirilir, akabinde bastırılmamış bireyler geçici olarak ihmal edilir. Tekrar aynı süreçle ikinci bir bastırılmamış yüzey elde edilir. Bütün popülasyonun bireyleri bir yüzey içinde sınıflandırılıncaya kadar bu süreç devam eder [2].

6. *Kuvvet Pareto Evrimsel Algoritması(SPEA):* Eckart Zitzler [52] tarafından 1999 yılında önerilen bu algoritma pareto-optimal kümeye yakınsamak için bilinen

eski yöntemlerle yenilerini birleştirmektedir. Bu yöntemin özellikleri aşağıda sıralanmıştır [2]:

- Ele alınan tüm bireyler arasındaki bastırılmamış yüzeyi gösteren bireyler harici olarak depolanmaktadır.
- Sayısal uygunluk değerlerini atamak için pareto üstünlük tanımını kullanılmaktadır.
- Pareto optimal yüzeyin özelliklerini yok etmeden harici olarak depolanan bireylerin sayısını azaltmada kümeleme tekniği kullanılır.

7. *Pareto-Arşivlenmiş Evrim Stratejisi(PAES):* J. D. Knowles ve D.W. Corne [53] tarafından 1999 yılında önerilen bir evrim algoritmasıdır. PAES'te bir yavru, bir ebeveynden mutasyonla üretilir. Yavruları ebeveynle karşılaştırır. Yavrular eğer ebeveynden baskın çıkarsa, yavru sonraki nesil için ebeveyn olarak kabul edilir ve yineleme devam eder. Eğer ebeveyn yavrudan baskın çıkarsa, yavru atılır ve yeni bir mutasyona uğramış bir çözüm (yeni bir yavru) üretilir. Eğer yavrular ve ebeveyn diğerlerinden baskın değilse, önceki baskın olmayan bireylerin karşılaştırma kümesi kullanılır. Pareto nüfus çeşitliliğini korumak için, baskın olmayan çözümleri arşivi olarak kabul eder. Yeni üretilen yavruların bu arşivin herhangi bir üyesinden baskın olmadığını doğrulamak için arşivle karşılaştırır. Yavru birey arşive girer ve yeni bir ebeveyn olarak kabul edilir. Baskın çözümler arşivden elenir, Eğer yavru arşivdeki bireylerden baskın değilse ebeveyn ve yavru arşivdeki çözümlerle yakınlıkları kontrol edilir. Yavru birey arşiv üyeleri arasında parametre uzayında en az kalabalık bölgede bulunuyorsa, ebeveyn olarak kabul edilir ve bir kopyası arşive eklenir [54].

8. *Pareto Zarflama-Temelli Seçim Algoritması(PESA):* PESA çaprazlama ve mutasyon oranları gibi standart parametreleri kullanır ve ayrıca popülasyon boyutuyla ilgili olarak iki ve bir parametre de hiper-grid kalabalıklık stratejisi ile ilgili olarak kullanmaktadır. PESA algoritmasını aşağıdaki gibi açıklanabilmektedir [55]:

Popülasyon tabanlı parametreler *IP* dahili popülasyon boyutu P_I ve *EP* harici popülasyon max boyutu P_E olarak gösterilmektedir:

- *IP* dahili popülasyonunun tüm kromozomlarını üret ve *EP*'yi boş küme olarak oluştur.

- *IP*'ye ait bastırılmamış bireyleri *EP*'ye ekle.
- Eğer sonlandırma kriteri oluşmuşsa dur ve *EP*'deki kromozom kümesini sonuç olarak ver, değilse *IP*'nin şimdiki içeriğini sil ve P_I yeni aday çözümler üretilinceye kadar aşağıdakileri tekrarla:
    - p_c olasılığı ile *EP*'den iki ebeveyn seç, çaprazlama işlemiyle tek bir çocuk üret ve buna mutasyon işlemini uygula. $1\text{-}p_c$ olasılığı ile bir ebeveyn daha seç, bir çocuk üretmek için buna mutasyon uygula.

4. Adım 2'ye dön.

4. ÇOKLU DİZİ HİZALAMADA ÇOK AMAÇLI GENETİK ALGORİTMA KULLANIMI

Bu tezde NSGA II [56] olarak adlandırılan ve herkesçe bilinen yüksek performansları ve çok amaçlı bir algoritma, iki amaca bağlı olan çoklu dizilerde olası en iyi sıralamayı bulmak için kullanılacaktır. NSGA II algoritması aynı zamanda üç amaçlı durumlarda da kullanılmaktadır. Diğer bir yandan tekli amaç formülasyonu durumunda, tekli elit çözümüyle beraber standart tekli-amaç GA'sı kullanılır. Önerilen algoritma GA tabanlı olduğu için çalışmanın devamında bunun geçmişi hakkında bilgi verilecektir.

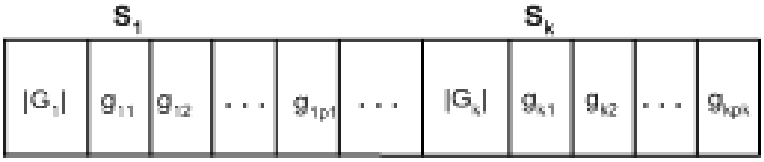
GA, Holland [3] tarafından icat edilen ve evrimsel hesaplama alanında yer alan bir arama ve optimizasyon metodolojisidir. GA, Darwin'in en güçlünün hayatta kaldığını ifade eden doğal seleksiyon kavramı tabanlıdır ve mühendislik ile bilgisayar bilimleri alanında zor problemlerin çözümünde kullanılır. GA popülasyon tabanlı bir metottur ve popülasyonun her bir bireyi, hedef problem için bir çözümü temsil eder. Bu çözümler popülasyonu, genel olarak rastgele üretilen birey yardımıyla, birkaç nesil boyunca evrimleşir. Bu evrimsel sürecin her bir neslinde, popülasyon içindeki her bir birey bir uygunluk fonksiyonuyla evrimleşir. Bu fonksiyon her bir bireyin hedef problemin çözümünü ne kadar doğru temsil ettiğini gösterir. Darwin'in doğal seleksiyon yöntemine uygun olarak verilen herhangi bir nesilden bir sonraki nesile, bazı ebeveyn bireyler (genellikle en yüksek uygunluğa sahip olanlar) çocuklar oluşturur yani ebeveynlerinden bazı özellikleri taşıyan yeni bireyler oluşur ve aynı zamanda uygunluğu düşük olan bireyler de çıkarılır. Ebeveynlerin seçilmesi olasılık tabanlı bir süreçtir ve uygunluk değerleriyle alakalıdır. Bu prosedür takip edilerek, genel olarak popülasyonun uygunluğunun bir sonraki nesilde azalmayacağı beklenir. Seçilen ebeveynlerden yeni çocukların oluşumu genetik operatörler yardımıyla yapılır. Bu işlemler arzu edilen bir nesil bulunana kadar veya bir ölçüte ulaşılana kadar (Örneğin maksimum nesil sayısı) tekrarlamalı bir şekilde devam eder.

4.1. Bireylerin Yapısı

İlk aşamada popülasyon, bireylerin her bir genine bir dizinin rastgele yüklenmesiyle oluşturulur. Bu yükleme işlemi en geniş dizinin uzunluğu belirlenmesiyle başlar. Her bir dizi, en geniş yüklü dizinin uzunluğu artı en geniş yüklü dizinin uzunluğunun %0 ile %20'si boyutuna ulaşacak şekilde rastgele boşluklarla doldurulur. Bu boşluklar, diziler

içerisine rastgele olarak konumlandırılır. %20 seçimi, çoklu dizi hizalama problemlerinin çözümlerinin sadece çok az bir kısmında %20'den fazla boşluk bulunması gözlemine bağlı olarak belirlenmiştir.

Önerilen yöntemde her bir birey genel olarak bütün hedef dizilerdeki boşlukların yerini tutar. Bu anlamda her bir birey Şekil 4.1'de görüldüğü gibi k adet gene bölünmüştür. Burada her bir gen sırasıyla, boşlukların sayısı ve boşlukların pozisyonundan oluşur. Şekil 4.1'de l_i den l_k'ye değişen uzunluklarda k adet dizinin hizalanışının temsilini göstermektedir.



Şekil 4.1. Bir Bireyin Temsili

$|G_i|$ alanı i = 1,2,…,k olmak kaydıyla i-ninci dizideki boşlukların sayısını göstermek için kullanılır. Bu alanın değeri $(m = 1,2, \dots, k)$, $\max(l_m)$ en uzun dizinin uzunluğu ve l_i de i-ninci dizinin uzunluğu olmak üzere, $(\max(l_m) - l_i)$ ve $(\max(l_m) + 20\%.\max(l_m) - l_i)$ arasında değişir. $g_{i1}, g_{i2}, \dots, g_{ip_i}$ alanları, $0 \leq g_{ij} \leq l_i$ ve $p_i = (\max(l_m) + 20\%.\max(l_m) - l_i)$ olmak üzere i-ninci dizideki boşlukların yerlerini gösterir.

$g_{ij} = 3$ ifadesi üçüncü nükleotidden sonra boşluk olduğu anlamına gelir. Eğer g_{ij} alanında sırayı göz önüne almaksızın, iki adet 3 var ise ikinci nükleottiden sonra 2 adet boşluk olduğu anlamına gelir.

Şu unutulmamadır ki g_{ij} alanlarının sayısı p_i olsa da, bunların $|G_i|$ sayıları dikkate alınmaktadır. Örneğin, sıralamak için üç dizimiz var olduğunu farz edelim. Uzunlukları 10,15 ve 20 olsun. Bu durumda en uzun dizi artı en uzun dizinin %20'si 24 eder. Bu yüzden ilk dizinin sahip olacağı boşluklar 10 ve 14 aralığında değişecektir. Bu aralık ikinci dizi için [5,9] üçüncü dizi için [0,4] olacaktır.

Yukarıdaki kodlama dizilerin sayısına göre oldukça esnektir. Geleneksel bir GA bu yönden sınırlıdır zira sadece sabitlenmiş bir sayı diziyle çalışabilir. Bizim yaklaşımımızda, her bir birey sabit bir uzunluğa sahip olsa da genlerin ($|G_i|$ değerini baz alarak) farklı bir

şekilde yorumlanmasıyla her bir fenotip (hizalanma) farklı bir uzunluk değeri almaktadır. Bu yüzden farklı bireyler farklı uzunluktaki hizalanmalara denk gelmektedir.

İlk popülasyonun başlangıcı isteğe bağlı olarak oluşturulan sabit sayıdaki bireylerin evrimi ile gerçekleşir.

4.2. Çoklu Amaçlar ve Seçim

Önerilen yöntemde her bir bireyin uygunluğu benzerlik ve hizalanmış uzunluk olmak üzere iki farklı parametre vasıtasıyla değerlendirilmiştir.

Benzerlik: Bireyi tanımlayan bütün dizilerin benzerlik ölçümünü gerçekleştirir. Bunu hesaplamak üzere her bir dizi için sunulan yöntemde bulunan hizalama desenlerinden konum ağırlık matrisi oluşturulur. Ardından her bir sütundaki baskın nükleotid için baskınlık değeri (d_v) şu şekilde hesaplanmaktadır:

$$dv(i) = \max_{b}\{f(b,i)\},\ i = 1,\ldots,l \tag{4.1}$$

burada f(b,i), pozisyon ağırlık matrisinin *i* sütunundaki *b* nükleotidinin skoru *dv(i), i* sütunundaki baskın nükleotidin baskınlık değeri ve *l* ise hizalama uzunluğudur.

A hizalamasının benzerlik hedef fonksiyonunu, pozisyon ağırlık matrisindeki bütün sütunların baskınlık değerlerinin ortalaması almakla bulunur.

$$Benzerlik(M) = \frac{\sum_{i=1}^{l} dv(i)}{l} \tag{4.2}$$

Benzerlik *(M)* değeri *1*'e yaklaştıkça, aday hizalama A'nın en iyi hizalama olma olasılığı yükselir. Sıradaki örnek verilen iki adet farklı boyutlardaki konum ağırlık matrislerinden benzerlik değerinin elde edilmesini göstermektedir.

Örnek:

İlk matris (Tablo 4.1), bir veri kümesinde hedef dizi sayısının 5 olduğu ve hizalanmış uzunluğunun 4 olduğu bir örnektir. Böyle bir durumda sütun 1 için dominant karakter T ve dominant değer 0.6'dır. Diğer sütunlardaki dominant değerler sırasıyla 1,0.8,1 'dir. Böyle bir hizalama örneğinin benzerlik değeri aşağıdaki gibi hesaplanır:

$$\frac{(0.6 + 1 + 0.8 + 1)}{4} = 0.85$$

Tablo 4.1. 4 Uzunluklu Hizalanmış Pozisyon Ağırlık Matrisi

	1	2	3	4
A	0.2	1	0	0
C	0.2	0	0.8	1
T	0.6	0	0	0
G	0	0	0	0
-	0	0	0.2	0

Benzer biçimde, ikinci matriste (Tablo 4.2), hedef dizilerin sayısı 4 ve hizalanma uzunluğu 5 olarak belirlenmiştir. Benzerlik değeri daha uzun hizalama için 0.85 olarak hesaplanmıştır.

Tablo 4.2. 5 Uzunluklu Hizalanmış Ağırlık Matrisi

	1	2	3	4	5
A	0.25	0	0.75	1	0
C	0.5	0	0	0	0
T	0	1	0	0	1
G	0	0	0	0	0
-	0.25	0	0.25	0	0

***Hizalanmış uzunluk**:* Çoklu dizi hizalamada her zaman için daha kısa hizalanma uzunluğu aranır.

Genel olarak çoklu dizi hizalama problemi aşağıdaki iki aşamalı optimizasyon problemine dönüştürülür.

***Maksimizasyon** Benzerlik(A), **Minimizasyon** Hizalanmış Uzunluk(A)*

Popülasyondaki bireyler, ilk olarak baskın olmayan sıralama prosedürleri kullanılarak kendi aralarındaki baskınlık durumlarına göre sıralanır. Burada bireyler popülasyonun ilk formundaki herhangi bir üye tarafından baskın değildir ve rank 1 olarak alınır. Daha sonra bu bireyler dikkate alınmaz ve bundan sonra baskın olmayan bireylere rank 2 atanır. Tüm süreç bireylere bir rank atanıncaya kadar tekrarlanır. Bireyler, sıralanmış ranklarına göre sıralanırlar. [56]'da ayrıntılı olarak tarif edilen baskın olmayan sıralama algoritmalarının genel karmaşıklığı $O(MN^2)$ olarak gösterilmiştir, burada M amaç sayısı N ise popülasyondaki birey sayısıdır. İlk ön çözüm mevcut popülasyona göre baskın olmayan kümeyi temsil eder, bu ön çözümlerden hiçbiri popülasyondaki diğer çözümlerden herhangi birinden baskın değildir.

Baskın olmayan sıralama algoritması uygulandıktan sonra kalabalıklaştırma mesafesi atanır ve seçim kalabalıklaştırılmış turnuva seçimi kullanılarak yapılır.

Kalabalıklaştırılmış mesafe önceki çözümlerin amaç değerlerinin farklarının toplamı olarak hesaplanır ve her bir hedef için mevcut çözümü (söz konusu bireye karşılık gelen) izlemektedir. Bu popülasyon içinde belirli bir noktadan çevreleyen çözüm yoğunluğu bir ölçüsünü sağlar. Kalabalıklaştırılmış turnuva seçiminde bireylerden bir çift rastgele seçilir ve alt rankı ile birlikte seçilir. Bireylerin rankları eşitse daha büyük bir kalabalıklaştırma mesafe ile birey seçilir. Büyük kalabalıklaştırma mesafe çözümleri pareto- optimal yayılabilir olmasını sağlar.

4.3. Genetik Operatörler

Önerilen yöntemde alışılmış bir noktadan çaprazlama operatörü, havuzdan seçilen 2 birey kullanılarak önceden tanımlanmış bir olasılıkla uygulanır. Çaprazlama noktası, stringin çaprazlama kırılması birey uzunluğunun başlangıç noktasından itibaren nereden bölüneceğinin yüzdesidir. Biz deneylerde aritmetik çaprazlama yöntemini kullandık. Kullanılan yöntem aşağıdaki gibi çalışmaktadır. $C_1 = (c_1^1, \ldots, c_n^1)$ ve $C_2 = (c_1^2, \ldots c_n^2)$ olan iki kromozom düşünülsün. C1 ve C2 üzerinde çaprazlama operatörü uygulanarak iki yavru $H_1 = (h_1^1, \ldots, h_i^1, \ldots, h_n^1)$ ve $H_2 = (h_1^2, \ldots, h_i^2, \ldots, h_n^2)$ oluşturulur, burada for i=1 to n için $h_i^1 = \lambda c_i^1 + (1-\lambda)c_i^2$ ve $h_i^2 = \lambda c_i^2 + (1-\lambda)c_i^1$ dir. Bizim deneylerimizde, λ düzgün olmayan aritmetik çaprazlama olarak, nesillerin üretilen sayısına göre değişmektedir. Mutasyon operatörü arama uzayını daha fazla araştırmaya teşvik etmek ve bazı yerel minimum için erken yakınsamansa neden olan genetik materyalin kurtarılamayan kaybını önlemek için kullanılır. Genel olarak verilen bir olasılık, mutasyon olasılığı cinsinden olan tek bir belirli konumda değerini değiştirerek mutasyon uygulanmaktadır. Önerilen yöntemde, bizim genom gösterimi için hazırlanmış üç mutasyon operatörleri geliştirdi:

Boşluğun konumunu sağa doğru kaydırma: rastgele seçilen bir genin değeri bir arttırılır.

Boşluğun konumunu sola doğru kaydırma: rastgele seçilen bir genin değeri bir azalır.

Rastegele değişim: Mutasyonun bu türünde küçük bir tamsayı üretilir sonra boşluk konumu, ağırlık veya uzunluk değerlerinin herhangi birinin o anki içeriğine eklenir ya da çıkarılır. Bu alt ve üst sınır alanını aştığı zaman olduğu gibi bir şekilde uygulanmaktadır.

Özetle bu çalışmanın görev süreci aşağıdaki algoritma ile özetlenebilir:

Algoritma:

Giriş: Populasyon boyutu N, jenerasyonun maksimum sayısı G, çaprazlama olasılığı p_c, Mutasyon oranı p_m

Çıkış: baskın olmayan küme

Adım 1: P:=Başlangıç populasyonunu oluştur (P)

Adım 2:*while* sonlandırma kriteri gerçekleşmediği sürece *do*

Adım 3: C:=Seçme işlemi yap (P)

Adım 4: C^I:=Genetik Operatörler (C)

Adım 5: P:=Yerine Geçme (PUC^I)

Adım 6: *end while*

Adım 7: return (P)

İlk olarak, Adım 1'de P başlangıç populasyonu üretilir. Adım 3'te mevcut P popülasyonundan ebeveyn çözüm çiftleri seçilir. Adım 3'te ebeveyn çözüm çiftlerinden seçilen küme C ile gösterilir. Adım 4'te herbir C çiftinden yavru populasyon C^I üretmek için çaprazlama ve mutasyon işlemleri uygulanır. PUC^I birleşiminden en iyi çözümlerden seçilerek yeni populasyonu oluşturulur. Pareto- dominant ve toplanma ölçümü adım 5'teki PUC^I birleşiminden ve Adım 3'teki mevcut P populasyonunun herbir çözümlerinin hesaplamak için kullanılır. Elitizim PUC^I birleşiminden yeni popoulasyon üyelerinin en iyi çözümlerinin seçilmesi için Adım 5'te uygulanır.

5. DENEYSEL SONUÇLAR

Tezin bu bölümünde çoklu dizilerin hizalanması için önerdiğimiz çok amaçlı genetik algoritmanın performansını ve etkinliğinin analiz etmek için bazı uygulamalar gerçekleştirildi. Önerilen algoritmanın sonuçları daha önceden bilinen etkin iki farklı algoritma ile karşılaştırılmıştır. Bu algoritmalardan ilki aşamalı yöntemlerin en etkin olanlarından biri olan ClustalW diğeri ise en tanınmış genetik algoritma tabanlı yaklaşım olan SAGA'dır. Uygulamanın tamamı 2.0 GHz işlemcili, 4 GB hafıza ve Windows 7 işletim sistemine sahip bir bilgisayarda C++ kullanılarak gerçekleştirilmiştir.

Veritabanı olarak BAliBASE [57] kullanılmıştır. Bu veritabanı 5 referansa ayrılmıştır. Toplam 141 karşılaştırma alt verisinden oluşur. Referans1, 3-6 dizilik 82 karşılaştırma alt verisinden meydana gelir. Bu referans daha küçük 3 alt gruba bölünür. Grup 1, ortalama %25'den daha küçük benzerlikli karşılaştırma alt verisini içerir. Grup 2 ortalama %20 ile %40 benzerlikli karşılaştırma alt verisine sahiptir. Grup 3 ise ortalama %30'tan daha büyük benzerlikli karşılaştırma alt verisinden ibarettir. Referans 2, 23 karşılaştırma alt verisine sahiptir ve bunların her biri en az 15 yakın ilgili, 3 taneye kadar da ilgisiz diziye sahiptir. Referans 3,4 ve 5'de ise 12 karşılaştırma alt verisi mevcuttur.

Bu tezde yapılan bütün uygulamalarda önerdiğimiz çok amaçlı genetik algoritma tabanlı yöntem 200 bireyli bir popülasyonla sürece başlar. Çaprazlama olasılığı 0,8 olarak belirlenmişken mutasyonun her bir çeşidi için 0,3'lük bir mutasyon oranı belirlenmiştir. Her bir jenerasyonda en iyi çözüm kaydedilir. Eğer 100 ardışık jenerasyonda en iyi çözüm aynı kalırsa algoritma sonlandırılır. Bu sonlandırma şartı deneysel gözlemlere dayalıdır.

Biyolojik bakış noktasından çözüm kalitesini değerlendirmek için iki ölçü kullanılmıştır. Ölçülerden biri "çiftlerin toplamıdır (the Sum-of-Pairs Score (SPS))". Bu ölçü çoklu dizi hizalamadaki çözümleri değerlendirmek için kullanılan etkili değerlerden biridir. Hizalamadaki her bir kolonun değeri, her bir sembol çiftlerinin değerlerinin toplamıyla elde edilir. Bütün hizalamanın değerine ise aşağıdaki denklem kullanılarak mevcut kolonların değerlerinin toplamıyla ulaşılır.

$$SPS = \sum_{l=1}^{L} S_l \qquad (5.1)$$

Burada,

$$S_l = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} Cost(A_i, A_j) \quad (5.2)$$

Yukarıdaki denklemde L, hizalamanın uzunluğu (kolon sayısı); S_l, l'ninci kolonun hizalama değeri; N ise dizi sayısıdır. $Cost(A_i, A_j)$ ise iki hizalanmış dizi A_i ve A_j arasındaki hizalanma değeridir.

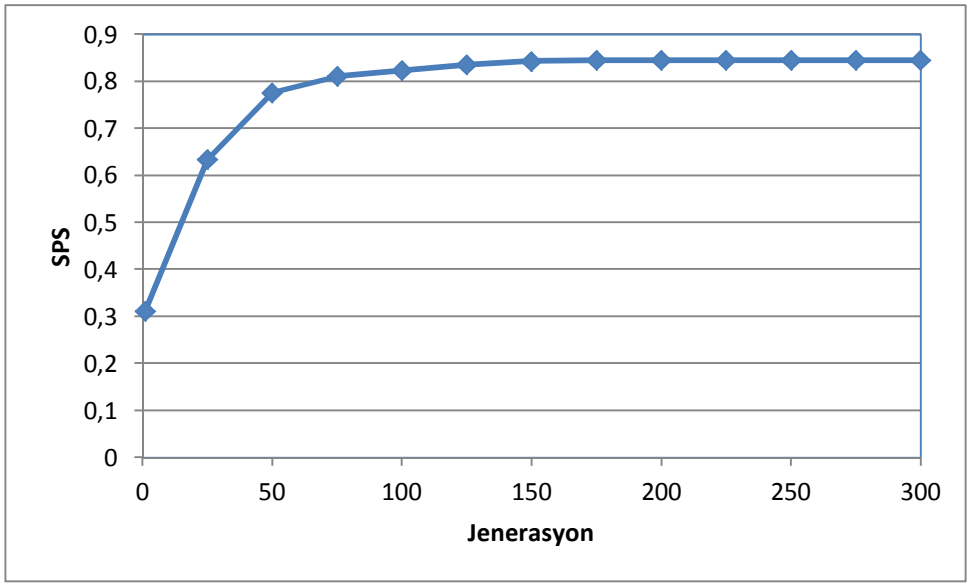
Karşılaştırmada kullanılacak diğer ölçü, doğru bir şekilde hizalanmış kolonların oranını gösteren "kolon değeridir (Column Score (CS))".

Her bir karşılaştırma verisi 10 defa test edilmiştir ve burada ortalama ölçüler verilmiştir. Tablo 5.1'de Referans 1 karşılaştırma alt verisi için önerdiğimiz yöntem ile ClustalW ve SAGA'nın sonuçları gösterilmiştir. Çoklu diziler arasında %25'in altında benzerlik olması durumunda önerilen yöntem dizi uzunluklarına bakılmaksızın SPS ve CS olarak diğer iki yönteme göre daha kötü sonuçlar vermiştir. %25'in altındaki benzerlik durumunda en iyi çözümü ClustalW bulmuştur. Diğer durum %20-%40 arası benzerlik durumudur ki burada en iyi çözüm dizi uzunluğuna göre değişmektedir. Kısa dizilerde ClustalW, orta uzunluklu dizilerde SAGA ve uzun dizilerde ise önerilen yöntemin sonuçları ön plana çıkmaktadır. Son durum ise dizilerin % 30'dan daha büyük benzerlik halidir. Bu durumda en iyi çözümler her üç dizi uzunluğu için önerilen yöntem tarafından bulunmuştur. SAGA ise ikinci en iyi çözümleri bulan yöntemdir.

Bu tezdeki diğer uygulama ise jenerasyon sayısına göre SPS'deki değişimi gözlemlemektir. Bu maksatla 4 dizili *1ycc* karşılaştırma alt verisi için bir test yapılmıştır. Jenerasyon sayısına göre SPS deki değişim Şekil 5.1'de verilmiştir. Bu şekil'e göre yaklaşık 150'nci jenerasyondan sonra SPS değerinde bir değişiklik gözlenmemektedir. Genetik algoritma süreci ise sonlandırma kriteri olarak 100 ardışık jenerasyonda bir değişiklik olmaması durumunda sonlanacağından 250'nci jenerasyonda bitecektir.

Tablo 5.1. Önerilen Yöntem, ClustalW ve SAGA'nın SPS ve CS ölçüleri

Benzer uzunluklu diziler	**% 25'in altındaki benzerlik**			**%20-%40 arası benzerlik**			**% 30 'den daha büyük benzerlik**		
	Önerilen Yöntem	**Clustal W**	**SAGA**	**Önerilen Yöntem**	**ClustalW**	**SAGA**	**Önerilen Yöntem**	**ClustalW**	**SAGA**
	SPS/ CS	**SPS/ CS**	**SPS/ CS**	**SPS/ CS**	**SPS/ CS**	**SPS/ CS**	**SPS/ CS**	**SPS/ CS**	**SPS/ CS**
Kısa (<100 rezidü)	0.512/ 0.354	**0.566/ 0.384**	0.498/ 0.314	0.780/ 0.648	**0.799/ 0.677**	0.784/ 0.652	**0.948/ 0.883**	0.930/ 0.876	0 915/ 0.856
Orta (200~300 rezidü)	0.458/ 0.258	**0.472/ 0.276**	0.415/ 0.233	0.824/ 0.724	0.824/ 0.726	0.805/ 0.703	**0.956/ 0.919**	0.949/ 0.910	0.923/ 0.887
Uzun (>400 rezidü)	0.456/ 0.286	**0.497/ 0.332**	0.427/ 0.273	**0.843/ 0.742**	0.825/ 0.734	0.775/ 0.702	**0.938/ 0.896**	0.930/ 0.883	0.902/ 0.879



Şekil 5.1. Önerilen yöntemde jenerasyona göre SPS değişimi

Tezin son uygulaması ise yine Referans 1 karşılaştırma alt verileri için gerekli çalışma zamanının elde edilmesi için yapılmıştır. Tablo 5.2'de gösterildiği gibi önerilen yöntemin çalışma zamanı ClustalW'nin değerlerinden daha kötü olmasına rağmen SAGA'dan daha iyidir. Diziler arasındaki benzerlik oranı arttıkça SAGA'ya olan üstünlük daha açık görülmektedir.

Tablo 5.2. Her üç yöntemin çalışma zamanı (saniye)

Ref. 1	**% 25'in altındaki benzerlik**			**%20-%40 arası benzerlik**			**% 30'dan daha büyük benzerlik**		
	Önerilen Yöntem	**ClustalW**	**SAGA**	**Önerilen Yöntem**	**ClustalW**	**SAGA**	**Önerilen Yöntem**	**ClustalW**	**SAGA**
	1423	924	1514	1154	745	1278	986	514	1139

6. SONUÇLAR

Doğal seçim ilkelerine dayanan bir arama ve optimizasyon yöntemi olan genetik algoritmalar parametre kümeleri üzerinde değil parametre kümelerinin kodlanmış biçimlerini kullanmaktadırlar. Genetik algoritmalar sadece amaç fonksiyonuna ihtiyaç duyar ve çözüm uzayının bir bölümünde arama yaparak çözüme en kısa sürede ulaşırlar.

Genetik algoritmaların arama uzayının büyük oluşunun yanında problemleri çözerken bir aday çözüm kümesi ile çalışması onu güçlü ve etkin kılmaktadır. Genetik algoritmaların uygulandığı problemlerde biri olan ÇDH'da diziler arasında benzerlik ve ilişkileri belirlemek ve ölçmek için kullanılmaktadır. Amaç nükleotid veya aminoasit arasındaki benzerliği en üst düzeye çıkarmaktır. Diziler arasında benzerliğin olması dizilerin yapısal, işlevsel ve evrimsel ilişkilerinin olduğunu göstermektedir.

Bu tez çalışmasında çoklu dizi hizalama problemi için çok amaçlı genetik algoritma tabanlı bir yöntem önerilmiştir. Daha önce bu alanda kullanılan yöntemler incelenerek yöntemlere farklı bir yaklaşım getirilmiştir. Bu yaklaşım genetik algoritmanın alt yapısını kullanmaktadır. Çoğu genetik algoritma tek bir amacı gerçekleştirmek için çalışırlar. Ama bazen çözüm sadece bir amaca göre şekillenmeyebilir. Bu tür durumlarda sistemi etkileyen tüm parametreler çözüm üretmek için kullanılmalıdır.

Bu tezde birden fazla parametreyi optimize etmeye çalışan bir yöntem anlatılmıştır. Çoklu dizi hizalama problemi üzerinde çalışan diğer algoritmalar ile benzerlik gösterse de birden fazla parametre kullanması algoritmanın kullanışlılığını arttırmaktadır. Geliştirilen bu yöntemde bir çözüm havuzu sunulmaktadır. Algoritma çözümler içerisinden en iyi olanları bulmaya çalışır.

Algoritma kaynak olarak tek bir diziyi almaz. Birden fazla diziyi aynı anda hizalamayı bekler. Algoritmada dizilerin benzerliği diğer parametre olan hizalanmış uzunluktan daha baskındır. Uygunluk hesabında kullanılan yöntemin geliştirilmesi algoritmanın güvenirliğini arttıracaktır. Önerilen yöntemde kullanılan çoklu-amaçlar benzerlik ve hizalanma uzunluğudur. Benzerliğin olabildiğince yüksek olmasına çalışılırken, uzunluk minimize edilmeye çalışılmıştır. Geliştirilen yöntem etkin iki çoklu dizi hizalama yöntemiyle karşılaştırılmıştır. Bunlar aşamalı yöntemlerin en tanınmış olanı ClustalW ve

genetik algoritma tabanlı bir algoritma olan SAGA'dır. BAliBASE veritabanı üzerinde yapılan deneysel sonuçlarda önerilen yöntemin doğruluk ve çalışma zamanı bakımından mevcut iki yaklaşıma üstünlükleri görülmüştür. Özellikle diziler arasında %30'dan daha büyük benzerlik durumunda her iki yönteme de SPS ve CS değerlerine göre daha üstündür. Ancak %25'in altındaki benzerlik durumunda daha kötü sonuçlar vermiştir ki, bu da başlangıçta hizalanmış dizi uzunluğunun maksimum dizinin %20 fazlasına eşit olacağını kabulünden kaynaklanmıştır. Çalışma zamanı bakımından da önerilen yöntem ClustalW ve SAGA ile karşılaştırılmıştır. Her ne kadar tezde yapılsa da aslında ClustalW ile karşılaştırma uygun değildir. Çünkü her iki yöntem de probleme farklı boyutlardan bakmaktadır. Bunun yerine SAGA gibi genetik algoritma tabanlı bir yaklaşımla zamanları karşılaştırma yerinde olacaktır. Bu bağlamda önerilen yöntem SAGA'dan daha hızlı çalışmaktadır.

Bundan sonraki çalışmalarda, önerilen çok-amaçlı genetik algoritmanın amaç fonksiyonları ve genetik algoritma süreci geliştirilerek daha uygun çözümler yakalanabilir. Bu sayede daha fazla sayıda dizinin aynı anda hizalama işlemi kolaylaşabilir.

KAYNAKLAR

[1] **Fatumo, S.A., Akinyemi İ. O. ve Adebiyi E.F.** (2009). Aligning Multiple Sequences with Genetic Algorithm. International Journal of Computer Theory and Engineering, Vol. 1, No.2,June2009 1793-8201.

[2] **Sağ, T.,** (2008). Çok Kriterli Optimizasyon İçin Genetik Algoritma Yaklaşımları, Yüksek Lisans Tezi, Selçuk Üniversitesi Fen Bilimleri Enstitüsü, Konya.

[3] **Holland, J.,** (1975). Adaptation in natural and artificial system. Ann Arbor, MI: University of Michigan Press.

[4] **Emel, G.G. ve Taşkın, Ç.,** (2002). Genetik Algoritmalar ve Uygulama Alanları. U.Ü. İktisadi ve İdari Bilimler Fakültesi Dergisi Cilt XXI, Sayı 1, s:129-152.

[5] **Güngör S.,** (2007). Hastalık teşhisi için çok-amaçlı genetik algoritma kullanarak çoklu-bulanık sınıflandırıcıların geliştirilmesi. Yüksek Lisans Tezi, Fırat Üniversitesi Fen Bilimleri Enstitüsü, Elazığ.

[6] **Sasanian, Z.,** (2011). Genetic Algorithms. http://webhome.cs.uvic.ca/~mgbarsky/COMPBIO/ZahraReport.pdf. (07.04.2011 tarihinde erişilmiştir.)

[7] **Liu, F. F. M., Tsai, J. J.P., Chen, R.M, Chen, S.N. ve Shih, S.H., (2004).** FMGA: Finding Motifs by Genetic Algorithm, Proc. of BIBE'04, page:459-466.

[8] **Baloğlu, U.B.,** (2006). Dna Sıralarındaki Tekrarlı Örüntülerin Ve Potansiyel Motiflerin Veri Madenciliği Yöntemiyle Çıkarılması. Yüksek Lisans Tezi, Fırat Üniversitesi Fen Bilimleri Enstitüsü, Elazığ.

[9] **Mount, D.M.**, (2004). Bioinformatics: Sequence and Genome Analysis, 2, Cold Spring Harbor Laboratory Press: Cold Spring Harbor, NY. ISBN 0-87969-608-7.

[10] **Needleman, S. B. ve Wunsch, C. D.,** (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology* 48 (3): 443–53.

[11] **Smith, T. F. ve Waterman, M. S.,** (1981). Identification of Common Molecular Subsequences. *Journal of Molecular Biology* 147: 195–197.

[12] **Notredame, C. ve Higgins, D. G.,** (1996). SAGA: sequence alignment by genetic algorithm. Nucleic Acids Res., 24, 1515–1524.

[13] **Lee, C.,** (2003). Generating consensus sequences from partial order multiple sequence alignment graphs. Bioinformatics, 19, 999–1008.

[14] **Gupta, S. K., Kececioglu, J. D. ve Schaffer, A. A.,** (1995). Improving the practical space and time efficiency of the shortest-paths approach to sum-of-pairs multiple sequence alignment. J. Comput. Biol., 2, 459–472.

[15] **Wallace, M., O'Sullivan, O. ve Higgins ,D.G.,** (2005). Evaluation of iterative alignment algorithms for multiple alignment , Bioinformatics.

[16] **Karadimitriou, K. ve Kraft D. H,** (1996). "Genetıc algorıthms and the multıple sequence Alıgnment problem ın bıology" Proceedings of the Second Annual Molecular Biology and Biotechnology Conference, Baton Rouge, LA.

[17] **Tompa, M.,** (2000). Multiple Sequence Alignment. http://www.cs.washington.edu/education/ courses/ cse527/00wi/lectures/lect06.pdf. (15.05.2012 tarihinde erişilmiştir.)

[18] **Edgar, R.C. ve Batzoglou S.,** (2006). Multiple sequence alignment. Current Opinion in Structural Biology, 16:368–373.

[19] **Chen, Y., Pan,Y., Chen,L. Ve Chen, J.,** (2006). Partitioned optimization algorithms for multiple sequence alignment. AINA '06 Proceedings of the 20th International Conference on Advanced Information Networking and Applications ,Volume 02 ,pages 618 – 622.

[20] **Notradame, C.,** (2007). Recent Evolutions of Multiple Sequence Alignment Algorithms. PLoS Computational Biology,Volume 3, Issue 8.

[21] **Diamantis, S. ve Anna C.,** (2012). Comparison of Multiple Sequence Alignment programs.http://www.ceng.metu.edu.tr/~tcan/ceng465/Spring2006/Schedule/ÇDHComparison.pdf 10.05.2012 tarihinde erişildi.

[22] **Mount, D.M.,** (2001). Bioinformatics, Sequence and Genome Analysis. Cold Spring Harbor Laboratory Press.

[23] **Wang L, Jiang T.,** (1994) On the complexity of multiple sequence alignment. Journal of Computational Biology, 1(4): 337-348.

[24] **Higgins, D. G.,** (1988). CLUSTAL: a package for performing multiple sequence alignment on a microcomputer. Gene 73: 237-244.

[25] Sequence Alignment and Modeling System. http://compbio.soe.ucsc.edu/sam.html. (21.05.2012 tarihinde erişilmiştir.)

[26] **Feng, D-F. ve Doolittle, R.F.,** (1987). Progressive sequence alignment as a prerequisite to correct phylogenetic trees. J Mol Evol 25: 351-360.

[27] **Taylor, W. R.,** (1988). A flexible method to align large numbers of biological sequences. J. Mol. Evol., 28, 161–169.

[28] **Thompson, J.D., Higgins, D.G., Gibson, T.J.,** (1994). CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. Nucleic Acids Research 22(22):4673-4680.

[29] **Diamantis S., Anna C.** (2006).”Comparison of Multiple Sequence Alignment programs.http://www.ceng.metu.edu.tr/~tcan/ceng465/Spring2006/Schedule/MSAComparison.pdf . (10.05.2012 tarihinde erişilmiştir.)

[30] **Notradame, C., Holme, L., Higgins, D.G., Herınga, J., O'sullıvan, O., Suhre, K. ve Abergel, C.,** (2010). Tcoffee ©: Multipurpose sequence

alignments program. Journal of Cell and Molecular Biology 7(2) & 8(1): 71-72, 2010 Software Review Haliç University, Printed in Turkey.

[31] **Subramanian, A.R., Weyer-Menkhoff, J., Kaufmann, M. ve Morgrenstern, B.,** (2005). DIALIGN-T: An improved algorithm for segment-based multiple sequence alignment. BMC Bioinformatics, 6:66.

[32] Multiple Sequence Alignment: A Critical Comparison of Four Popular Programs http://biochem218.stanford.edu/Projects%202008/Sutton2008.pdf (15.05.2012 tarihinde erişilmiştir.)

[33] **Hang, N.,** (2008). Comparıson of Multıple Sequence Alıgnment Programs in Practise. Unıversıty Of Arhus, Yüksek Lisans Tezi.

[34] **Llyod, G.,** (2010). Parallel Multiple Sequence Alignment: An Overview. http://dna.cs.byu.edu/ÇDH/overview.pdf. (04.06.2012 tarihinde erişilmiştir.)

[35] **Notredame,C. and Higgins,D.G.**, (1996) SAGA: sequence alignment by genetic algorithm. *Nucleic Acids Res.*, 24, 1515–1524.

[36] **Thomsen, R. ve Boomsma, W.,** (2004). Multiple Sequence Alignment Using SAGA: Investigating the Effects of Operator Scheduling, Population Seeding, and Crossover Operators. Applications of Evolutionary Computing Lecture Notes in Computer Science Volume 3005, pp 113-122.

[37] **Davis, L.,** (1989). Adapting operator probabilities in genetic algorithms. In: Proceedings of the Third International Conference on Genetic Algorithms (ICGA III). 61–69

[38] **Simossis, V. A. ve Heringa, J.,** (2005). PRALINE: a multiple sequence alignment toolbox that integrates homology-extended and secondary structure information. Nucleic Acids Research, 2005, Vol. 33, Web Server issue, 289–294.

[39] **Do C.B. , Brudno M. ve Batzoglou S.,**(2005) PROBCONS: Probabilistic Consistency-based Multiple Alignment of Amino Acid Sequences. http://www.cs.utoronto.ca /~brudno /AAAI04-111.pdf (22.05.2012 tarihinde erişilmiştir.)

[40] http://dialign-tx.gobics.de/dialign_tx_results (10.06.2012 tarihinde erişilmiştir.)

[41] http://www.ebi.ac.uk/ (10.06.2012 tarihinde erişilmiştir.)

[42] **Handl, J., Kell, B.D. ve Knowles, J.,** (2006). Multiobjective optimization in bioinformatics and computional biology .

[43] **Ehrgott, M.,** (2000). Multicriteria Optimization. Number 491 in Lecture Notes in Economics and Mathematical Systems. Springer- Verlag,Berlin, Germany, 2000.

[44] **Wright, H., Brodlie, K. ve David, T.,**(2000). Navigating high- dimensional spaces to support design steering. İn proceedings of the 11th IEEE visualization conference, pages 291-296. IEEE pres , Anaheim,CA.

[45] **Steuer, R.,** (1986). Multiple criteria optimization, theory, computation and applications. Wiley, New York, NY.

[46] **Konak, A., Coit, D.W, Smith A.E.,** (2006). Multi-objective optimization using genetic algorithms: A tutorial. Reliability Engineering & System Safety In Special Issue - Genetic Algorithms and Reliability, Vol. 91, No. 9. (September 2006), pp. 992-1007.

[47] **Murata, T.,** (1997). Genetic Algorithms or Multi-Objective Optimization, Doctoral Thesis at Osaka Prefecture University.

[48] **Horn, J. Ve Nafpliotis, N.,** (1993) Multiobjective optimization using the niched pareto genetic algorithm. ILLiGAL Report 93005, Illionis GeneticAlgorithms Laboratory, Universty of Illionis,Urbana,Chamaign.

[49] **Hajela, P . ve Lin, C.Y.,** (1993). Genetic search strategies in large scale optimization. AIAA/ASME/ASCE/AHS/ASC Structures, Structural

Dynamics, and Materials Conference, 34th and AIAA/ASME Adaptive Structures Forum, La Jolla, CA, Apr. 19-22, 1993, Technical Papers. Pt. 4 (A93-33876 13-39), p. 2437-2447.

[50] **Ghosh, A. ve Dehuri, S.,** (2004). Evolutionary algorithms for multi-criterion optimization: A survey. Intl. J. Comp. Inform. Sci., 2: 38–57.

[51] **Srinivas N., Deb K.,** (1994). Multiobjective Optimization Using Nondominated Sorting Genetic Algorithms, Journal of Evolutionary Computation, Vol. 2, No.3, pages 221-248.

[52] **Ziztler,E.,** (1999). Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications PhD thesis, ETH Zurich, Switzerland.

[53] **Knowles, J. D., ve Corne, D. W.,** (1999) . Local Search, Multiobjective Optimization and the Pareto Archived Evolution Strategy. Proceedings of Third Australia-Japan Joint Workshop on Intelligent and Evolutionary Systems. University of New South Wales and Ashikaga Institute of Technology: pages: 209-216.

[54] **Oltean, M., Grosan, C., Abraham A. ve Koppen, M.,** (2005). Multiobjective Optimization Using Adaptive Pareto Archived Evolution Strategy , Intelligent Systems Design and Applications, 2005. ISDA '05. Proceedings. 5th International Conference on.

[55] **Corne, D.W., Knowles, J. D. ve Oates, M. J.,** (2000). The Pareto Envelope-based Selection Algorithm for Multiobjective Optimization, Proceedings of the Parallel Problem Solving from Nature VI Conference.

[56] **Deb, K.,** (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. Kanpur Genetic Algorithms Lab., Indian Inst. of Technol., Kanpur. Volume**:**6, Issue:2 Page(s): 182 – 197.

[57] http://bips.u-strasbg.fr/fr/Products/Databases/BAliBASE/

ÖZGEÇMİŞ

Hülya HARK

hulyahark@hotmail.com

Fırat Üniversitesi
Bilgisayar Mühendisligi Bölümü
23119, Elazıg

Hülya HARK, 1986 yılında Kahramanmaraş'ta dünyaya geldi. İlkokul, ortaokul ve liseyi burada tamamladı. 2004 yılında başladığı Fırat Üniversitesi Bilgisayar Mühendisliği'nden 2008 yılında mezun oldu. 2010 yılında F.Ü Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Ana Bilim Dalında yüksek lisans eğitimine başlamış ve halen devam etmektedir.

Çalışma Yerleri:

2009-2010 Karayolları 17.Bölge Müdürlüğü'nde Ağ ve Sistem Yön. Tek. Elm.

2010-…. Karayolları 11.Bölge Müdürlüğü'nde Tünel Bakım İşletme Şefi