Çankaya University Journal of Science and Engineering
Volume 8 (2011), No. 1, 51–62

Fidan Gelişim Algoritması Yardımı ile DNA Motiflerinin Keşfi

Murat Demir[1,*], Ali Karcı[2] ve Mehmet Özdemir[3]

[1] *Muş Alparslan Üniversitesi, Meslek Yüksekokulu, 49000 Muş, Türkiye*
[2] *Fırat Üniversitesi, Bilgisayar Mühendisliği Bölümü, 23119 Elazığ, Türkiye*
[3] *Fırat Üniversitesi Elektrik-Elektronik Mühendisliği Bölümü, 23119 Elazığ, Türkiye*
[*] *Corresponding author: murat2107@yahoo.com*

Özet. Motif, DNA dizilerinde önemli bir görevi olan veya düzenli DNA parçalarının yerleşiminde önemli bir role sahip olan DNA parçacığıdır. Motif keşfi ise verilen DNA dizisi içerisinde potansiyel motif olabilecek DNA parçacıklarının bulunması işlemidir. Bu çalışmada fidan gelişim algoritması yardımıyla DNA ardışılları üzerinde motif keşfi işlemi gerçekleştirilmiştir.

Fidan gelişim algoritması fidanların gelişiminden esinlenerek geliştirilmiş bir algoritmadır. Bu yöntemde problemin çözümünü teşkil edebilecek olan değerler, fidan olarak adlandırılan çözüm dizilerine yerleştirilir. Fidan ekimi, eşleştirme, dallanma ve aşılama birer operatör olarak ele alınır. Fidan ekimi, arama uzayına düzgün dağılmış yeni fidanların (çözümlerin) oluşturulmasını sağlar. Dallanma yerel arama, eşleştirme küresel aramayı sağlar. Aşılama ise benzer fidanlar arasında bilgi değişimini sağlar.

Literatürde motif keşfi üzerinde yapılan çalışmalarda kullanılan yöntemlerin bir kısmı AlignACE, MEME, MEME3, MotifSampler, Consensus, Weeder v.b. çalışmalardır. Bu çalışmada elde ettiğimiz sonuçlar AlignACE, MEME, MEME3, MotifSampler, Consensus, Weeder yöntemlerinin sonuçlarıyla sonuç bölümünde karşılaştırılmıştır. Veriler TRANSFAC veri tabanından elde edilmiştir.

Anahtar Kelimeler. DNA, DNA motifleri, fidan gelişim algoritması.

Abstract. Motif, a DNA particle, has an important role in the formation of DNA sequences or in the placement of the regular DNA particles. The discovery of motif is the operation of finding out the potential DNA particles that are able to transform into motifs in a given DNA sequence. In this study, with the help of Sapling Growing up Algorithm, Motif discovery has been realized on the DNA sequences.

Sapling Growing up Algorithm is an algorithm developed as a result of the study concerning sapling growth. In this method, the data that may be of help in solving the problem are put into strings of solution that are called "sapling". Sowing of the saplings, mating,

Received August 23, 2010; accepted February 14, 2011.

Bu makale, 24-25 Nisan 2008 tarihlerinde Çankaya Üniversitesi'nin Ankara yerleşkesinde yapılmış olan *1. Çankaya Üniversitesi Mühendislik ve Teknoloji Sempozyumu*'nda sunulan ve sadece geniş bildiri özeti bölümü hakem sürecinden geçerek bu sempozyum kitapçığında yayımlanan bir makalenin revize edilmiş şekli olup Sempozyum Değerlendirme Komitesi tarafından yayımlanmak üzere *Çankaya University Journal of Science and Engineering* dergisine gönderilmesi önerilmiş ve derginin bağımsız hakem değerlendirmeleri sonucunda yayıma kabul edilmiştir.

ISSN 1309 - 6788

branching, and vaccinating are taken as operators. Sowing of the saplings provides the formation of new saplings (solutions) in the search space. Branching provides the local searching, and mating provides the global searching. Vaccinating, however, provides the exchange of information between similar saplings.
In literature, some of the methods on motif discovery studies are as follows: AlignACE, MEME, MEME3, MotifSampler, Consensus, Weeder, etc. The results attained in this study have been compared with AlignACE, MEME, MEME3, MotifSampler, Consensus, Weeder methods' results in the conclusion part of the paper. The data in this paper have been obtained form TRANSFAC database.

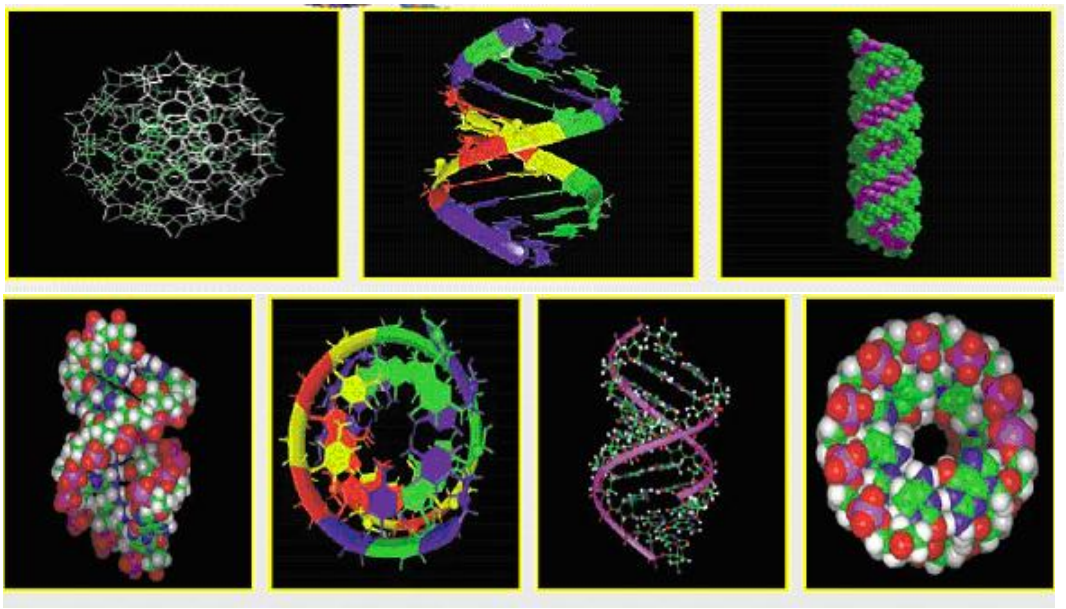
Keywords. DNA, motifs of DNA, saplings growing up algorithm.

1. DNA'nın Yapısı ve Motif Kavramı

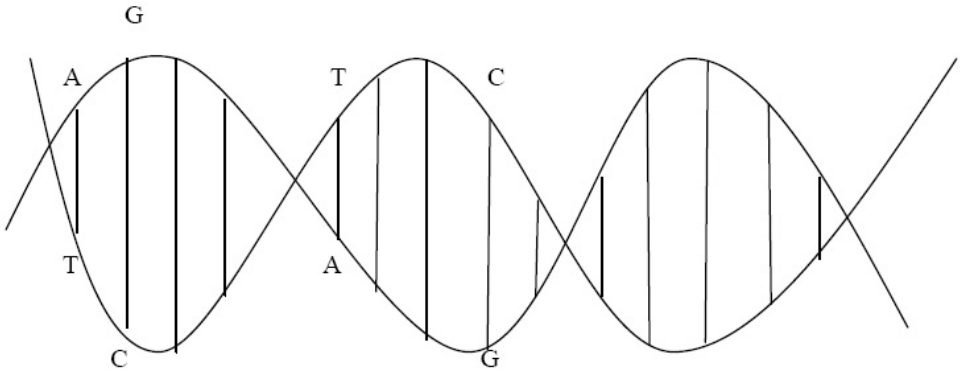
1.1. DNA'nın yapısı. DNA, bir nükleik asit olup Deoksi Ribo Nükleik asit kelimelerinin kısaltmasıdır. Tüm hücreli canlıların ve bazı virüslerin yapısında bulunur. Biyolojik gelişim için gerekli olan genetik bilgiyi taşır. Örneğin saç rengi, göz rengi, el ve ayak yapısı kişiye ait protein şifreleri v.s. tüm bilgiler DNA'da saklıdır. Yani canlının alfabesidir. Tüm hayatsal olayları DNA'da gizli olan şifrelerde saklıdır. Ayrıca DNA kalıtım olaylarını da içinde barındırdığı için kalıtımsal olayların (soya çekim) şifreleri de DNA içerisinde gizlidir. Bakterilerde ve diğer basit hücreli canlılarda DNA hücre içinde dağınık halde bulunur. Hayvanlar ve bitkilerde ise DNA'nın çoğu hücre çekirdeğindeki kromozomlarda bulunur.

DNA'nın yapı taşları nükleotid denilen moleküllerdir. Nükleotidler üç bölümden oluşur: Bir fosfat grubu, beş karbonu bulunan bir şeker ve bir organik baz (adenin, guanin, sitozin ya da timin) [1]. Adenin (A) her zaman timinle (T) birleşir ve aralarında 2 hidrojen bağı kurulur. Guanin (G) ise her zaman sitozinle (C) birleşir ve aralarında 3 hidrojen bağı vardır. DNA'da iki zincirde sıralı halde bulunan A, G, C ve T nükleotidlerinden A ile T birbirinin karşısına G ile C ise birbirlerinin karşısına gelirler (Şekil 1.1.1.). Yani birinci zincirde A veya T'den biri gelirse ikinci zincirde de A'ya karşılık T, T'ye karşılık ise A gelir. Birinci zincirde G veya C'den biri gelirse ikinci zincirde de G'ye karşılık C, C'ye karşılık ise G gelir (Şekil 1.1.2.).

1.2. Motif Kavramı. DNA zincirlerinde bulunan dört farklı nükleotid birbiri ardınca dizilerek DNA zincirlerini oluşturur. İnsanın her bir DNA'sında on binlerce nükleotid bulunur. Gen DNA zincirindeki belli bir uzunluktaki birimdir. Kromozom DNA'nın özel bir şekilde paketlenmesi sonucu ortaya çıktığına göre her kromozomda çok sayıda gen var demektir. DNA nükleotidlerinin bazı dizilimleri olay tetikleyici olduğundan bunlara motif denilmektedir. Bu DNA parçalarından motif adı verilen



Şekil 1.1.1. DNA'nın farklı formlardaki yapıları [1].



Şekil 1.1.2. DNA'daki nükleotidlerin karşılıklı yerleşmeleri.

kısa birimlerin keşfedilmesi özellikle bioinformatik adı verilen ve biyoloji ve bilgisayar bilimlerinin ortak çalışma alanı olan bilim dalı açısından çok önemlidir ve literatürde bir NP-zor problem olarak geçer [2,3,4].

ATCGTTACTGGTCAATCTGCATGCGTATTACTGGTCTCCGTTACCC
TTACTGGTCGGAACTGCGGTACGAACGTGCATATCGTACGGTTCCA
ACTACTTCAGCGCTGCAGTCGTACGGTTTACTGGTCCGGTCCTTGG

Yukarıdaki üç ardışılda da TTACTGGTC alt dizisi vardır. Birinci ardışılda iki kez tekrar ediyor. Bu alt dizinin sadece 2 kez birinci ardışılda geçmesi bile motif olması için yeterlidir. Fakat diğer iki ardışılda bulunması motifin sıklığını arttırdığından

daha iyi bir motif olduğunu gösterebilir. Motif keşfinde deterministik ve olasılıksal yöntemler kullanılmaktadır. Bunlarda pozisyon ağırlık matrisleri (PWM) ve rasgele seçilen alanlar kullanılarak işlemler yapılır [5]. Motifin sık tekrar etmesinin yanında uzunluğu da önemli bir kavramdır. Motif keşfi için kullanılan yöntemler: Genetik Algoritma [6], AlignACE [7,8,15], Bioprospector [8,15], MEME [7,15], Gibbs [9,10], Weeder [15].

2. Fidan Gelişim Algoritması

Fidan gelişim algoritması fidanların gelişiminden esinlenerek geliştirilmiş bir algoritmadır. Fidanların gelişim süreçlerini en iyi bir şekilde taklit eder [11,12,13]. Bu yöntemde problemin çözümünü teşkil edebilecek olan değerler fidanlara yerleştirilip daha sonra yapılacak olan işlemler bu fidanlar üzerine uygulanmaktadır.

Geliştirilmiş olan yöntemdeki operatörler, fidan gelişiminin mümkün olduğunca en iyi bir şekilde taklit edilmesi amacıyla fidanların gelişim esnasında nasıl bir değişikliğe uğradıkları operatörler olarak ele alınmıştır. Bundan dolayı, fidan ekimi bir operatör, eşleştirme bir operatör, dallanma bir operatör ve aşılama da bir operatör olarak ele alınmıştır. Fidan ekimi operatörü arama uzayına düzgün dağılmış fidanların (çözüm) oluşturulmasını sağlamaktadır. Dallanma operatörünün yerel arama, eşleştirme operatörünün ise küresel aramayı sağlaması beklenmektedir. Aşılama operatörü ise benzer fidanlar arasında bilgi değişimini sağlamaktadır [11,12,13].

Böylece geliştirilmiş yöntemde, aday çözümlerin başlangıçta arama uzayında düzgün dağıtılması sağlandıktan sonra hem yerel ve küresel arama gerçekleştirilmekte ve benzer çözümler arasında bilgi değişimi imkânı olmaktadır.

2.1. Fidanların ekimi. Düzenli Populasyon : Çözüm uzayı Z olarak kabul edilsin ve bu uzay s elemanını içermiş olsun. Z uzayı $Z = \{z_0, z_1, \ldots, z_{s-1}\}$ şeklinde yazılabilir. İlk olarak çözüm uzayının ikili tabanda olduğu kabul edilsin ve çözüm bir fidan veya fidan kümesi olacaktır. Eğer her fidanın iki dalı varsa,

$$Z = \{00, 01, 10, 11\}$$

şeklinde olur. Eğer her fidanda n tane dal varsa, çözüm uzayının boyutu 2^n olur ve s elemanı 2^n tane farklı durumu olur. Z uzayının bazı $\{01, 10\}$ olur ve diğer durumların bunların lineer kombinasyonundan elde edilebilir [11,12,13].

Aslında fidan gelişim algoritmasının diğer optimizasyon problemlerinden üstün olan taraflarından olan düzenli popülasyonun sağladığı şey, mümkün olabilecek çözüm

aralığını oluştururken en küçük ve en büyük değerleri ve bunlardan oluşan bir popülasyon oluşturmak ve bu sayede çözüm aralığını mümkün olduğunca en iyi şekilde dağıtmaktır. Bu sağlanınca en iyi çözümlere yakınsamak daha kısa sürede ve daha iyi bir başarı ile olabilecektir. Diğer algoritmalarda bazen yerel minimum veya maksimumlara takılabilmeler olmaktadır. Fidan gelişim algoritmasının başlangıç popülasyonunu oluşturmada kullandığı bu yöntem bu problemleri ortadan kaldırabilmektedir. Buradaki maksimum ve minimum denilen değerler fidan gelişim algoritmasında alt sınır ve üst sınır diye tabir edilmektedir [11,12,13]. Bu amaçla popülasyon oluşturulurken aşağıdaki formüller kullanılmaktadır. Eğer S_0 üst sınır (maksimum çözümler) ve S_1 alt sınır (minimum çözümler) ise;

$$S_0 = \{u_1, u_2, \ldots, u_n\},$$

$$S_1 = \{l_1, l_2, \ldots, l_n\},$$

ve bunlardan oluşacak olan diğer çözüm fidanları aşağıdadır:

$$\begin{aligned} S_3 =\big\{ & l_1 + (u_1 - l_1) * r, l_2 + (u_2 - l_2) * r, \ldots, l_{n/2} + (u_{n/2} - l_{n/2}) * r, \\ & l_{n/2+1} + (u_{n/2+1} - l_{n/2+1}) * (1 - r), l_{n/2+2} + (u_{n/2+2} - l_{n/2+2}) * (1 - r)\big\}, \end{aligned}$$

$$\begin{aligned} S_4 =\big\{ & l_1 + (u_1 - l_1) * (1 - r), l_2 + (u_2 - l_2) * (1 - r), \ldots, \\ & l_{n/2} + (u_{n/2} - l_{n/2}) * (1 - r), l_{n/2+1} + (u_{n/2+1} - l_{n/2+1}) * r, \\ & l_{n/2+2} + (u_{n/2+2} - l_{n/2+2}) * r\big\}, \end{aligned}$$

$$\begin{aligned} S_5 =\big\{ & l_1 + (u_1 - l_1) * r, l_2 + (u_2 - l_2) * r, \ldots, l_{2n/3} + (u_{2n/3} - l_{2n/3}) * r, \\ & l_{2n/3+1} + (u_{2n/3+1} - l_{2n/3+1}) * r, \ldots, l_n + (u_n - l_n) * r\big\}, \end{aligned}$$

$$\begin{aligned} S_6 =\big\{ & l_1 + (u_1 - l_1) * r, l_2 + (u_2 - l_2) * r, \ldots, l_{n/3} + (u_{n/3} - l_{n/3}) * r, \\ & l_{n/3+1} + (u_{n/3+1} - l_{n/3+1}) * (1 - r), \ldots, l_{2n/3} + (u_{2n/3} - l_{2n/3}) * (1 - r), \\ & l_{2n/3+1} + (u_{2n/3+1} - l_{2n/3+1}) * r, \ldots, l_n + (u_n - l_n) * r\big\}, \end{aligned}$$

$$\begin{aligned} S_7 =\big\{ & l_1 + (u_1 - l_1) * r, l_2 + (u_2 - l_2) * r, \ldots, l_{n/3} + (u_{n/3} - l_{n/3}) * r, \\ & l_{n/3+1} + (u_{n/3+1} - l_{n/3+1}) * (1 - r), \ldots, l_n + (u_n - l_n) * (1 - r)\big\}, \end{aligned}$$

$$S_8 = \{l_1 + (u_1 - l_1) * (1 - r), l_2 + (u_2 - l_2) * (1 - r), \ldots, l_{n/3} + (u_{n/3} - l_{n/3}) * (1 - r), \\ l_{n/3+1} + (u_{n/3+1} - l_{n/3+1}) * r, \ldots, l_n + (u_n - l_n) * r\},$$

$$S_9 = \{l_1 + (u_1 - l_1) * (1 - r), l_2 + (u_2 - l_2) * (1 - r), \ldots, l_{n/3} + (u_{n/3} - l_{n/3}) * (1 - r), \\ l_{n/3+1} + (u_{n/3+1} - l_{n/3+1}) * r, \ldots, l_{2n/3} + (u_{2n/3} - l_{2n/3}) * r, \\ l_{2n/3+1} + (u_{2n/3+1} - l_{2n/3} + 1) * (1 - r), \ldots, l_n + (u_n - l_n) * (1 - r)\},$$

$$S_{10} = \{l_1 + (u_1 - l_1) * (1 - r), l_2 + (u_2 - l_2) * (1 - r), \ldots, \\ l_{2n/3} + (u_{2n/3} - l_{2n/3}) * (1 - r), l_{2n/3+1} + (u_{2n/3+1} - l_{2n/3+1}) * r, \ldots, \\ l_n + (u_n - l_n) * r\}.$$

2.2. Fidanların büyümesi. Fidanların büyümesi işlemleri eşleşme, dallanma ve aşılama işlemleri yardımıyla olmaktadır. Bunlar kullanılarak aslında çözüm uzayı içerisinde farklı çözüm değerleri oluşturulmakta ve mümkün olan en geniş çözüm uzayı elde edilmeye çalışılmaktadır. Bu sayede olabilecek en iyi çözüme ulaşılabilme olanağı arttırılmaktadır.

Eşleştirme. Eşleştirme operatörünün amacı, fidanlar arasında genetik bilginin değişimini sağlamaktır. Bu çalışmada bir eşleştirme faktörü vardır ve bu faktör fidanlar arasındaki uzaklık ile ilgili olarak tanımlanan bir olasılıktır [11,12,13].

$G = g_1 g_2 ... g_i ... g_n$ ve $H = h_1 h_2 ... h_i ... h_n$ iki tane fidan olsun. $P(G, H)$, G ve H fidanlarının eşleşmeme, $P_m(G, H)$ ise, eşleşme olasılıkları olsun. Bu durumda bu olasılıklar şu şekilde hesaplanır:

$$P(G, H) = \frac{\left(\sum_{i=1}^{n} (g_i - h_i)^2\right)^{1/2}}{R} \quad \text{ve} \quad R = \left(\sum_{i=1}^{n} (u_i - l_i)^2\right)^{1/2} \tag{2.1}$$

u_i üst sınır ve l_i alt sınırdır.

$$P_m(G, H) = 1 - \frac{\left(\sum_{i=1}^{n} (g_i - h_i)^2\right)^{1/2}}{R}. \tag{2.2}$$

Eşleştirme işlemi sonucunda n adet fidandan $2n$ adet fidan oluşur.

Karıştır-Geliştir. Dallanma operatörünün değiştirme olasılığının çok yüksek olması ve aşılamanın ise birbirinden uzak olan çözümler üzerinde işlem yapmasından kaynaklanan istenilen çözümden uzaklaşma sıkıntıları vardır. Bu nedenle dallanma ve aşılama operatörleri yerine karıştır-geliştir operatörü kullanıldı [14]. Fidan gelişim algoritması ilk tanımlanan şekli ile üç operatörlü iken (seçme hariç), yukarıda belirtilen sebeplerden dolayı iki operatörlü bir algoritma şeklini almıştır.

Karıştır-geliştir işlemine operatörü iki bireyi giriş olarak alır ve yeni iki bireyi de çıktı olarak verir. Bu amaçla uzayın maksimum Öklit uzaklığı hesaplanır ve operatörün uygulanacağı iki birey arasında Öklit uzaklığı hesaplanır (Denklem (2.4)). R maksimum Öklit uzaklığı ve R_1 ise iki fidan arasındaki Öklit uzaklığı olmak üzere

$$R_{\max} = \sqrt{\sum_{i=1}^{n} (F_{\max,i} - F_{\min,i})^2}, \tag{2.3}$$

$$R_1 = \sqrt{\sum_{i=1}^{n} (F_1[Di] - F_2[Di])^2} \tag{2.4}$$

şeklinde hesaplandıktan sonra bu iki fidandan yeni iki fidan elde edilir (Denklem (2.5) ve Denklem (2.6)). Denklem (2.3)'deki $F_{\max}$ en büyük fidan değerlerini, $F_{\min}$ ise en küçük fidan değerlerini göstermektedir:

$$F_{\text{yeni},i1} = \frac{R_1}{R} F_{1,i} + \left(1 - \frac{R_1}{R}\right) F_{2,i}, \tag{2.5}$$

$$F_{\text{yeni},i2} = \left(1 - \frac{R_1}{R}\right) F_{1,i} + \frac{R_1}{R} F_{2,i}. \tag{2.6}$$

3. Geliştirilen Yöntem ve Elde Edilen Sonuçlar

3.1. Geliştirilen yöntem. Bu çalışmada TRANSFAC veri tabanından elde edilen 4 farklı dosya üzerinde çalışıldı [15]. Bunlar dm01r.fasta, dm01g.fasta, hm15r.fasta, mus05r.fasta. Bunlardan dm01g.fasta, hm15r.fasta, mus05r.fasta dosyaları üzerinde özette bahsedilen diğer yöntemlerin sonuçları mevcuttur. Uyguladığımız yöntem ile diğer yöntemlerin karşılaştırmaları sonuç verilmiştir. mus01r.fasta isimli dosyanın üzerinde diğer yöntemler ile çalışma mevcut değildir. Bu amaçla uyguladığımız yöntem ile bu dosya üzerinde de motif keşfi yapılmıştır.

Bu 4 farklı dosyanın her birinde aşağıda temsil edildiği gibi 4 farklı DNA ardışılı mevcuttur:

ATCGTTACTGGTCAATCTGCATGCGTATTACTGGTCTCCGTTACCC
TTACTGGTCGGAACTGCGGTACGAACGTGCATATCGTACGGTTCCA
ACTACTTCAGCGCTGCAGTCGTACGGTTTACTGGTCCGGTCCTTGG
ATCGTTACTGGTCAATCTGCATGCGTATTACTGGTCTCCGTTACCC

Uygulanan yöntemde oluşturulan fidanlar motif olabilme ihtimali olabilecek rasgele pozisyon değerlerini tutmaktadır. Bu pozisyonların kodlama temsili aşağıdaki gibidir:

P_1	P_2	P_3	P_4

Maksimum P değerleri her dosyadaki nükleotid sayısı kadardır. Minimum P değerleri ise 1 dir. Maksimum ve minimum fidanlar aşağıdaki gibidir:

1	1	1	1

max	max	max	max

Bu maksimum ve minimum değerler kullanılarak 4 nitelikten $2^4 = 16$ adet başlangıç fidanı oluşturulur. Bu oluşturulan fidanlar Tablo 3.1.'de verilmiştir. Buradaki r değeri 0-1 aralığında rasgele oluşturulan bir değerdir.

Tablo 3.1.'deki L değerleri minimum değerleri, U değerleri ise maksimum değerleri temsil etmektedir.

TABLO 3.1. Başlangıç popülasyonu oluşturma.

Min 1	Min 2	Min 3	Min 4
Max 1	Max 2	Max 3	Max 4
$L_1 + (u_1 - l_1) * r$	$L_2 + (u_2 - l_2) * r$	$L_3 + (u_3 - l_3) * r$	$L_4 + (u_4 - l_4) * r$
$L_1 + (u_1 - l_1) * r$	$L_2 + (u_2 - l_2) * r$	$L_3 + (u_3 - l_3) * r$	$L_4 + (u_4 - l_4) * (1 - r)$
$L_1 + (u_1 - l_1) * r$	$L_2 + (u_2 - l_2) * r$	$L_3 + (u_3 - l_3) * (1 - r)$	$L_4 + (u_4 - l_4) * r$
$L_1 + (u_1 - l_1) * r$	$L_2 + (u_2 - l_2) * r$	$L_3 + (u_3 - l_3) * (1 - r)$	$L_4 + (u_4 - l_4) * (1 - r)$
$L_1 + (u_1 - l_1) * r$	$L_2 + (u_2 - l_2) * (1 - r)$	$L_3 + (u_3 - l_3) * r$	$L_4 + (u_4 - l_4) * r$
$L_1 + (u_1 - l_1) * r$	$L_2 + (u_2 - l_2) * (1 - r)$	$L_3 + (u_3 - l_3) * r$	$L_4 + (u_4 - l_4) * (1 - r)$
$L_1 + (u_1 - l_1) * r$	$L_2 + (u_2 - l_2) * (1 - r)$	$L_3 + (u_3 - l_3) * (1 - r)$	$L_4 + (u_4 - l_4) * r$
$L_1 + (u_1 - l_1) * r$	$L_2 + (u_2 - l_2) * (1 - r)$	$L_3 + (u_3 - l_3) * (1 - r)$	$L_4 + (u_4 - l_4) * (1 - r)$
$L_1 + (u_1 - l_1) * (1 - r)$	$L_2 + (u_2 - l_2) * r$	$L_3 + (u_3 - l_3) * r$	$L_4 + (u_4 - l_4) * r$
$L_1 + (u_1 - l_1) * (1 - r)$	$L_2 + (u_2 - l_2) * r$	$L_3 + (u_3 - l_3) * r$	$L_4 + (u_4 - l_4) * (1 - r)$
$L_1 + (u_1 - l_1) * (1 - r)$	$L_2 + (u_2 - l_2) * r$	$L_3 + (u_3 - l_3) * (1 - r)$	$L_4 + (u_4 - l_4) * r$
$L_1 + (u_1 - l_1) * (1 - r)$	$L_2 + (u_2 - l_2) * r$	$L_3 + (u_3 - l_3) * (1 - r)$	$L_4 + (u_4 - l_4) * (1 - r)$
$L_1 + (u_1 - l_1) * (1 - r)$	$L_2 + (u_2 - l_2) * (1 - r)$	$L_3 + (u_3 - l_3) * r$	$L_4 + (u_4 - l_4) * r$
$L_1 + (u_1 - l_1) * (1 - r)$	$L_2 + (u_2 - l_2) * (1 - r)$	$L_3 + (u_3 - l_3) * r$	$L_4 + (u_4 - l_4) * (1 - r)$
$L_1 + (u_1 - l_1) * (1 - r)$	$L_2 + (u_2 - l_2) * (1 - r)$	$L_3 + (u_3 - l_3) * (1 - r)$	$L_4 + (u_4 - l_4) * r$
$L_1 + (u_1 - l_1) * (1 - r)$	$L_2 + (u_2 - l_2) * (1 - r)$	$L_3 + (u_3 - l_3) * r$	$L_4 + (u_4 - l_4) * (1 - r)$

Yukarıdaki tabloda verilen kodlama yöntemi ile ilk maksimum ve minimum fidanlarda dahil olmak üzere 18 adet elimizde fidanımız olmuş oldu. Eşleştirmeler şu

şekilde yapılmıştır: 1-3 fidanlar arası eşleştirme 2. noktadan itibaren, 2-4 fidanlar arası eşleştirme 1. noktadan itibaren, 5-7 fidanlar arası eşleştirme 3. noktadan itibaren, 6-8 fidanlar arası eşleştirme 2. noktadan itibaren, 9-11 fidanlar arası eşleştirme 3. noktadan itibaren, 10-12 fidanlar arası eşleştirme 1. noktadan itibaren, 13-15 arası eşleştirme 2. noktadan itibaren, 14-16 fidanlar arası eşleştirme 1. noktadan itibaren, 17-18 fidanlar arası eşleştirme 3. noktadan itibaren. Daha sonra da karıştır geliştir operatörü uygulanmıştır. Bu amaçla 1-2, 3-4, 5-6, 7-8, 9-10, 11-12, 13-14, 15-16, 17-18 fidan çiftleri birer birer karıştır geliştir operatörüne tabi tutularak her birinden 2 fidan elde edilmiştir. Bu işlemin sonucunda da 18 yeni fidan elde edilmiştir.

Başlangıç popülasyonu oluşturma, eşleştirme ve karıştır geliştir işlemleri sonucunda elimizde 54 adet fidanımız olmuştur. Buradan sonrasında amaç şudur: Fidanların göstermiş olduğu pozisyon değerlerine bakarak 4 ardışıl içinde kontrol yapıp o pozisyonlarda motif olup olmadığını kontrol etmektir. İşaret edilen bir pozisyondaki motifin tüm ardışıllarda geçmesi şart değildir. Fakat bir motifin uzunluğu, tüm ardışıllarda geçmesi (desteği) o motifin sağlamlığını arttırır [6]. Yine de bir motifin aynı ardışılda da olsa 2 kez tekrar etmesi yeterlidir.

3.2. Elde edilen sonuçlar. Uygulanan diğer yöntemler motif keşfi için benzer motifler sunmaktadır. Uyguladığımız yöntem tam motifleri keşfetmektedir. Bu nedenle karşılaştırma yapılırken tam motifler ölçü alınmıştır. Ayrıca uyguladığımız yöntemin tam motifleri keşfetmesi daha sağlıklı bir sonuçtur. Çünkü benzer motif elde etmektense tam bir motif keşfedebilmek daha iyi bir sonuçtur. Aşağıda özette bahsedilen diğer yöntemlerle birlikte fidan gelişim algoritmasının (FGA) sonuçları karşılaştırılmıştır. Tabloda verilen değerler bulunan maksimum uzunluktaki motif değerleri ve sayılarıdır. "-" değeri o dosya için o yöntemin motif keşfedemediğini "*" değeri ise 7 den uzun motif bulunamadığını göstermektedir. Bulunan motifin anlamlı olması için en az 6 uzunluklu olması gerekmektedir. Bu çalışmada bu değer daha anlamlı olabilmesi açısından 7 olarak seçilmiştir.

Tablo 3.2. Sonuçlar.

Dosya adı	AlignACE	Consensus	MEME	MEME3	MotifSampler	Weeder	FGA
dm01g.fasta	-	8	10	*	13	-	9
mus05r.fasta	-	-	8	9	*	8	8
hm15r.fasta	-	Çalışılmamış	-	21	14	10	18

Tablodan da görüldüğü gibi fidan gelişim algoritması motif uzunluğu bakımından tablodaki diğer yöntemlerden dm01g.fasta dosyası için MEME ve MotifSampler hariç diğerlerinden iyidir. mus05r.fasta dosyası için MEME3 hariç diğerlerine eşit veya diğerlerinden iyidir. hm15r.fasta dosyası için ise yine MEME3 hariç diğerlerinden iyidir.

Fidan gelişim algoritmasının avantajlı olan bir yönü ise bu çalışılan yöntemlere göre daha çok sayıda tam motif keşfedebilmesidir. Tablo 3.3'te bulunan tam motif sayıları karşılaştırılmıştır.

Tablo 3.3. Motif sayıları.

Dosya adı	AlignACE	Consensus	MEME	MEME3	MotifSampler	Weeder	FGA
dm01g.fasta	-	1	1	-	1	-	14
mus05r.fasta	-	-	2	3	-	1	7
hm15r.fasta	-	-	-	4	2	1	11

Tablo 3.3.'ten de görüldüğü gibi diğer yöntemlerin hepsinin bulduğu toplam tam motif sayısından da fazla motif keşfi yapılmıştır. Ayrıca uygulama sırasında keşfedilen tüm motifler alınmamıştır. Yani fidan gelişim algoritmasının keşfettiği motif sayısı gerçekte daha fazladır. Buna rağmen sonuç çok iyidir. Aşağıda fidan gelişim algoritmasının bulduğu motifler sunulmuştur.

dm01g.fasta dosyası için bulunan tam motifler :

TTAAAAA, GCGGTTA, AACAACA, GCCGACT, AAGTGCG, AGTAAAG, AATTTAAA, GAAATAAA, AATAATAG, TAAATTTG, AAATAAAC, AAAAAAAAA, ACAACAACA, TGGCCGTGA.

mus05r.fasta dosyası için bulunan tam motifler:

AGATTTA, AGAAGGC, CCCCCAT, TATGAGG, TTGAAAAT, AGGTAAAA, GTGTAATA.

hm15r.fasta dosyası için bulunan tam motifler:

CAATAAA, AAAAAAG, ATAAAAA, AGGACTGA, GCTCTATA, CCTTTCTCA, CCAGGAAAGG, CTGGGATTACA, AAAAAAAAAAAAA, TTTTTTTTTTTTTTTTTT.

Ayrıca hiçbir yöntemin üzerinde çalışmadığı bir dosya olan dm01r.fasta dosyası üzerinde uyguladığımız yöntem 10 uzunluklu tam motife kadar motif keşfetmiştir. Bu dosya için keşfedilen motiflerde aşağıdaki gibidir :

GAGTTCA, CATCATA, TTCCTGA, GCATCGAA, CTACTAAT,
TAAATAAA, AAATAAATA, ATTGAAGGA, AAAAAAAAAA,
ACCAAAATAA, AACAAAAACA.

4. Sonuç

Fidan gelişim algoritması fidanların gelişimini taklit eden, başlangıç popülasyonunun düzgün dağıtılması sebebiyle iyi çözümlere daha çabuk yakınsayabilen bir algoritmadır. Eşleştirme ve karıştır geliştir operatörlerinin uygulanacağı fidanların seçiminin kullanıcı tarafından rasgele verilmesi yerine bunların uygunluk değerlerinin hesaplanarak en uygun olanlarının işleme tabi tutulması daha iyi motiflerin keşfine sebep olabilecektir. Yine de fidan gelişim algoritmasının keşfettiği bu sonuçlar diğer yöntemlere göre gayet başarılı sonuçlar olmuştur.

Teşekkür. Bu çalışma, TÜBİTAK-EEEAG-105E144 no.'lu proje kapsamında desteklenmektedir. Bundan dolayı TÜBİTAK-EEEAG grubuna teşekkür ederiz.

Kaynaklar

[1] DNA. `http://www.biltek.tubitak.gov.tr/bdergi/poster/icerik/dna.pdf`, 1999. Son erişim: 23-Mayıs-2011.

[2] S. Mahony, P. V. Benos, T. J. Smith and A. Golden, Self-organizing neural networks to support the discovery of DNA-binding motifs, *Neural Networks* **19** (2006), 950–962.

[3] T. H. Cormen, C. E. Leiserson and R. L. Rivest, *Introduction to Algorithms*, The MIT Press, Massachusetts 1990.

[4] R. Rivière, D. Barth, J. Cohen and A. Denise, Shuffling biological sequences with motif constraints, *Journal of Discrete Algorithms* **6** (2008), 192–204.

[5] P. D'haeseleer, How does DNA sequence motif discovery work? *Nature Biotechnology* **24** (2006), 959–961.

[6] M. Kaya, MOGAMOD: Multi-objective genetic algorithm for motif discovery, *Expert Systems with Applications* **36** (2009), 1039–1047.

[7] J. Hu, B. Li and D. Kihara, Limitations and potentials of current motif discovery algorithms, *Nucleic Acids Research* **33** (2005), 4899–4913.

[8] X. Liu, D. L. Brutlag and J. S. Liu, BioProspector: discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes, *Pacific Symposium on Biocomputing* **6** (2001), 127–138.

[9] W. Thompson, E. C. Rouchka and C. E. Lawrence, Gibbs Recursive Sampler: finding transcription factor binding sites, *Nucleic Acids Research* **31** (2003), 3580–3585.

[10] G. Thijs, K. Marchal, M. Lescot, S. Rombauts, B. De Moor, P. Rouzé and Y. Moreau, A Gibbs sampling method to detect overrepresented motifs in the upstream regions of coexpressed genes, *Journal of Computational Biology* **9** (2002), 447–464.

[11] A. Karci, M. Yiğiter and M. Demir, Natural inspired computational intelligence method: Saplings growing up algorithm, *International Kazakh-Kyrgyz Electronics and Computer Conference (IKECCO'2007)*, Almaty 2007.

[12] M. Demir, M. Yiğiter and A. Karci, Application of saplings growing up algorithm to clustering medical data, *International Kazakh-Kyrgyz Electronics and Computer Conference (IKECCO'2007)*, Almaty 2007.

[13] M. Demir ve A. Karcı, Veri kümelemede fidan gelişim algoritmasının kullanılması, *Elektrik Elektronik Bilgisayar Biyomedikal Mühendisliği 12. Ulusal Kongresi ve Sergisi*, Eskişehir 2007.

[14] S. Arslan Tuncer, *Fidan Gelişim Algoritması ile Protein İkincil Yapı Tahmini*, Yüksek Lisans Tezi, Fırat Üniversitesi, Elazığ 2008.

[15] E. Wingender, P. Dietze, H. Karas and R. Knüppel, TRANSFAC: A database on transcription factors and their DNA binding sites, *Nucleic Acids Research* **24** (1996), 238–241.