3. GENETİK ALGORİTMALAR

3.1. Giriş

Günlük olayların ve araştırmaların en önemli sorularından birisi yapılan çaba ve çalışmaların en iyileme (Eİ) ile sonuçlandırılabilir olup olmayacağıdır. Bunun nedeni elde bulunan imkânların (malzeme, zaman, mekân) kısıtlı olması dolayısı ile hizmetin Eİ biçiminde verilmesi ile önemli kazançların sağlanmasıdır. Bu, endüstride önemli olduğu kadar doğal kaynakların (su, enerji, gıda, vb.) en iyi üretimde kullanılması konularında da aynıdır [27].

Karmaşık bir sistemin en iyilemesinde karşılaşılan iki problem vardır. Birincisi, en iyileme algoritması sistem için uygun olmalıdır. İkincisi ise en iyileme algoritmasının çeşitli parametreleri verimliliği artırmak için ayarlanabilmelidir. Buna göre, karmaşık bir sistemin geniş bir aralıkta en iyilemesinin yapılabilmesi için kullanılan en uygun yöntemlerden birisi genetik algoritmadır [1].

Genetik algoritmalar (GA), evrim teorisinden esinlenerek türetilen hesaplama modelleridir. Makine öğrenmesi konusunda çalışan Holland (1975), canlılardaki genetik işlemleri sanal ortamda gerçekleştirmeye çalışarak bu işlemlerin etkinliğini açıklamıştır. Başlangıçta pratik bir yararı olmadığı düşünülen GA'lara olan ilgi, Holland'ın öğrencisi olan Goldberg'in yaptığı doktora teziyle National Science Foundation tarafından verilen Genç araştırmacı ödülünü alması ve dört yıl sonrada klasik eserini yayınlamasıyla çoğalmıştır. (Goldberg 1989).

Bu algoritmalarda, belirlenen bir problemin potansiyel çözümleri, ikili (binary) ya da ikili olmayan sistemlere dayalı veri yapısında basit diziler olarak şifrelenir ve kritik bilgileri saklamak için bu dizilere bir takım işlemler uygulanır. GA yöntemi, evrim teorisi esaslarına göre çalışarak verilen bir sorun için en iyi çözüm veya çözümleri arayarak bulmaya yarar. Bu esaslar ortama en fazla uyum sağlayan canlıların hayata devam etmesi ve uyum sağlayamayanların da elenmesi olarak algılanmalıdır. GA'lar bu iki kuralı bir arada kullanarak en iyiyi aramayı hedef edinen bir Eİ yöntemidir [27].

Oldukça geniş uygulama alanına sahip olan genetik algoritmalar daha çok bir en iyileme fonksiyonu olarak görülür. Genel olarak, en iyileme (en küçükleme-en büyükleme) sorunlarının çözümü için uygun olan GA kullanımı, diğer yöntemlerle kıyas edildiğinde çok daha iyi çözümlere en kısa zamanda sayısal örgünlük ve rasgele düzen içinde gidebilmektedir. Genetik algoritmalar aynı zamanda oldukça geniş bir araştırma uzayında, bölgesel araştırma

tekniklerinin (gradiyent yöntemi vb.) uyguladığı birçok ihmali eleyen global en iyileme teknikleri olarak da görülebilir.

Genetik algoritma, mühendislik, bilim, ekonomi gibi çok değişik alanlardaki problemler için gürbüz bir en iyileme aracı olarak son yıllarda büyük bir önem kazanmıştır. Genetik algoritmanın uygulama alanlarından bazıları, haberleşme şebekeleri tasarımı, elektronik devre tasarımı, gaz boruları şebeke en iyilemesi, görüntü ve ses tanıma, veri tabanı sorgulama en iyilemesi, uçak tasarımı, fiziksel sistemlerin kontrolü, gezgin satıcı problemlerinin çözümü, ulaşım problemleri ve optimal kontrol problemleridir [1, 24].

3.2. Genetik Algoritmaların Diğer En İyileme Tekniklerine Göre Farklılıkları

Genetik algoritmalar gürbüzlük araştırmalarında diğer geleneksel en iyileme yöntemlerine göre daha üstündür. Genetik algoritmaların faklılığı dört ana başlık altında toplanabilir:

1) Genetik algoritmalar, parametrelerin kendisiyle değil parametre setini kodlayarak çalışır:

Genetik algoritmalar, sonlu bir alfabe üzerinden sonlu uzunlukta diziler kodlamak için, en iyileme probleminin doğal parametre setini kullanır. Örneğin, $f(x) = x^2$ fonksiyonunun [0,31] aralığında en büyüklemesi yapılmak istendiğinde, diğer en iyileme yöntemleri x parametresini küçük küçük değiştirerek amaç fonksiyon değerinin en yüksek olduğu yeri bulmaya çalışırlar. Genetik algoritmalarla en iyileme işleminin yürütülmesinde ise ilk adım, x parametresinin sonlu uzunlukta dizi olarak değişik şekillerde kodlanması ve kodlanan parametre ile çalışılarak en iyilemenin sağlanmasıdır. Karar değişkenlerinin karakterleri topluca karar uzayında bir noktayı temsil eder. Pek çok durumda ikili (binary) kodlama kullanılır. Fakat genetik algoritma için bu bir zorunluluk değildir. Gerçel sayı kodlama, ağaç yapılı kodlama (tree coding) gibi farklı kodlama biçimleri de kullanılabilir. Parametrelerin kodlanması sonucunda, diğer yöntemleri sınırlayan birtakım özelliklerde de büyük ölçüde serbestlik sağlanmış olur (süreklilik, türevin varlığı, vb.)

2) Genetik algoritmalar tek bir noktada değil bir noktalar kümesinde en iyileme araştırması yapar:

Birçok en iyileme probleminde, tanım aralığı içindeki tek bir noktadan hareketle bazı geçiş kurallarına göre bir sonraki inceleme noktası bulunur. Bu, noktadan noktaya yönelme yöntemi, çok sayıda tepe noktası bulunan araştırma uzayı için risklidir. Çünkü bölgesel tepe değerine yaklaşma hatası oluşabilir. Genetik algoritmalarda, çok sayıda noktalardan oluşan bir veri tabanı ile (dizilerin nüfusu) çalışıldığından aynı anda pek çok tepe noktasına atlanabilir.

Böylece noktadan noktaya geçme yöntemindeki yanlış tepe noktasının bulunma olasılığı azalır. Sonuç olarak, genetik algoritma dizilerden oluşan bir nüfus ile başlar ve bu dizilerden daha başarılı nüfuslar üretir. Bu nüfus, problemin bütün olası çözümlerini temsil eden uzayı oluşturur. Başlangıç nüfusu genellikle rasgele üretilen bireyleri içerir.

3) Genetik algoritmalar, türevler ya da diğer yardımcı bilgiler değil sadece amaç fonksiyon bilgisini kullanır:

Çoğu araştırma tekniklerinin, doğru bir şekilde çalışması için yardımcı bazı bilgilere ihtiyaçları vardır. Örneğin, gradient tekniklerinde, tepe değerine doğru yükselmenin olup olmadığını anlamak için, nümerik ya da analitik olarak hesaplanan türev bilgisi gereklidir. Ayrıca, farklı araştırma problemleri için de çok farklı yardımcı bilgilere ihtiyaç olabilir. Bunun aksine, genetik algoritmalarda hiçbir yardımcı bilgiye ihtiyaç yoktur. Verimli bir araştırma yapmak için gerekli olan, her bir dizinin değerlendirileceği bir amaç fonksiyonudur. En iyileme sırasında problemle ilgili özel birtakım bilgilerin kullanılmaması, genetik algoritmaların performansını yükseltmede son derece etkilidir.

4) Genetik algoritmalar kesin bilinen kuralları değil olasılığa dayalı (stokastik) kuralları kullanır:

Genetik algoritmalar araştırmaya yön vermek için, birçok en iyileme tekniğinin aksine, olasılığa dayalı geçiş kuralları kullanır. Bunun sonucunda araştırmanın, araştırma uzayının hangi bölgesine doğru yöneleceğine karar vermek için rasgele seçim tekniği kullanılır.

Bahsedilen bu dört farklılığın bir arada bulunması, genetik algoritmaların gürbüzlüğüne ve sonuca ulaşma üstünlüğüne olumlu yönde katkıda bulunur.

3.3. Genetik Algoritma Terminolojisi

Genetik algoritmalar, hem doğal genetikler hem de bilgisayar alanında kullanıldığı için genetik algoritma literatüründe kullanılan terminoloji, doğal ve yapay ortamdaki terimlerin karışımından oluşmaktadır. Bu nedenle, genetik algoritmalarla çalışan araştırmacıların kullanacağı temel bazı terimlere ihtiyaç vardır. Bilgisayar ortamında tanımlanan diziler, alfabeler, dizi pozisyonları ve buna benzer bazı terimlerin doğal genetiklerde kullanılan karşılıkları çizelge 3.1'de görülmektedir. Örneğin; yapay genetik sistemlerdeki diziler, biyolojik sistemlerdeki kromozomlara benzemektedir. Doğal sistemlerde, bir ya da daha fazla kromozom, bir organizmanın yapısı ve çalışması için toplam genetik bilgileri oluşturacak şekilde birleşir. Bu toplam genetik paket genotip olarak adlandırılır. Yapay genetik sistemlerde ise dizilerin toplamı genetik yapı paketi olarak adlandırılır. Doğal sistemlerde,

toplam genetik paketin içinde bulunduğu şartlara göre birbirini etkilemesiyle şekillenen organizma fenotip (phenotip) olarak tanımlanır. Yapay genetik sistemlerde ise yapılar, belli bir parametre setini, olası bir çözümü ya da çözüm uzayında bir noktayı düzenlemek için şifreli durumu çözer. Bir yapay genetik sistem tasarımcısı hem nümerik hem de nümerik olmayan parametreleri kodlamak için çeşitli alternatiflere sahiptir.

Doğal terminolojide kromozomlar, belli sayıda değer alabilen genlerin birleşiminden oluşmaktadır. Bir gen'in pozisyonu onun gen olarak fonksiyonundan ayrı tanımlanır. Örneğin; bir hayvanın göz rengi geni ele alındığında bu genin pozisyonu ya da yeri 10, bu kodun değeri de mavi göze karşılık gelebilir. Yapay genetik araştırmalarda ise diziler, farklı değerler alan özellikler ya da karakterlerden meydana gelmektedir. Özellikler, dizi üzerinde farklı pozisyonlara yerleştirilebilir.

Çizelge 3.1. Doğal ve GA terminoloji karşılaştırması [28]

Doğal terminoloji	**Genetik algoritma**
kromozom	**dizi**
gen	**özellik, karakter veya dedektör**
allele	**özellik değeri**
konum	**dizi pozisyonu**
genotip	**yapı**
fenotip	**parametre seti, alternatif çözüm, kodlanmış yapı**

Belli bir karakter ve onun pozisyonu arasındaki farkı ayırt etmek gerekirse; bir dizi içinde bir bitin pozisyonu, nüfusun her yerinde ve her zaman o bitin nasıl çözüldüğü ile bulunabilir. Örneğin; ikili sayı sistemine göre 10000 dizisi, işaretsiz tamsayı olarak onlu sayı sistemine göre 16 olarak çözülebilir. Burada 1, onlu sayı sisteminde 16'nın yerine geçmektedir [1,28,29].

Basit bir genetik algoritma (Simple Genetic Algorithm), problem en iyilemesi için son derece uygun ve güçlü bir yöntemdir. Literatürde genetik algoritma tekniği içerisinde, SGA (Simple Genetic Algorithm)'dan başka, Mikro Genetik Algoritma (μGA), Kararlı Durum Genetik Algoritma (KDGA), Hiyerarşik Genetik Algoritma (HGA) ve Dağınık Genetik Algoritma (mGA) gibi farklı modeller ve paralel çalışan genetik algoritma modelleri tanımlanmıştır.

Genetik algoritma, en uygun (optimal) çözümü bulmak için birden çok noktadan aramaya başladığı için bu arama noktalarının başlangıç değerlerini oluşturmak önemlidir. İlk popülasyon genellikle rastlantısal olarak oluşturulur. Fakat yapılan bazı çalışmalarda ilk popülasyonun daha önceden elde edilen bir bilgiye veya sezgiye (heuristic) bağlı olarak üretildiği görülmektedir [26]. Varılması istenilen Eİ noktasının yaklaşık konumu hakkında ipuçları bulunuyorsa dizi değerlerinin böyle bir çözüm noktası etrafında rasgele ama daha dar bir karar uzayından seçilmesi uygundur.

3.4. Basit Genetik Algoritma

Birçok problemin çözümünde iyi sonuçlar veren basit bir genetik algoritma (SGA); kopyalama (reproduction), çaprazlama (crossover) ve mutasyon (mutation) olmak üzere üç genetik işlemin birleşiminden oluşur. Bir jenerasyon boyunca yürütülen bu işlemler en büyükleme ya da en küçükleme probleminde jenerasyonlarda elde edilen optimal değerler arasındaki fark sıfırlandığında ya da belli bir değere yakınsadığında sona erdirilir. Ayrıca bir genetik algoritma, programın başında belirlenen bir jenerasyon sayısı kadar tekrarlanıp bitirilebilir. Bu sayının yeterince büyük olması sonuçta elde edilecek değerin, fonksiyonun optimal çözümü olma şansını artırır.

3.4.1. Kopyalama

Kopyalama, nüfus içindeki çözüm seçenekleri olan diziler arasında bir sonraki aşamada daha da iyiye gidebilecek olanların seçilmesi işlemidir. Kopyalama diziyle özdeş dizi üreten bir operatördür. Başka bir deyişle her bir dizinin amaç fonksiyon değerine (f) göre, gelecek jenerasyona kopyalanmasını sağlayan bir işlemdir. (f) fonksiyonu en büyükleme yapılmak istenen uygunluk, yararlılık ya da iyiliğin bir ölçüsü olarak düşünülebilir. Uygunluk değerlerine göre dizilerin kopyalanması, yüksek değere sahip olan dizilerin gelecek jenerasyona bir ya da daha fazla ürün olarak katkıda bulunmasıdır. Bu işlem doğal seçimin yapay bir versiyonudur. Darwin'e göre de, varlıklar arasında en iyilerin yaşama şansı her zaman daha yüksektir. Klasik yöntemlerin tümünde her an yegâne diyebileceğimiz bir tane en iyi çözüm olmasına karşılık, GA'larda çözüm olmaya aday bir nüfus vardır. Bunlar arasında sadece bir tanesi en iyidir. Diğerleri de bulundukları karar değişkeni bölgesinde en iyi olmaya

adaydır. Yaygın olarak kullanılan kopyalama teknikleri, bütünüyle yer değiştirme, jenerasyon aralığı ve kararlı durum yöntemleridir.

Bütünüyle yer değiştirme yönteminde, eski jenerasyondaki bütün diziler yeni jenerasyondaki dizilerle yer değiştirir. Yeni dizi seçilince, genetik operatörler diziyi şekillendirmeye başlar. İşlemler, dizi sayısı popülasyon büyüklüğüne eşit oluncaya kadar devam eder. Bütünüyle yer değiştirme en eski kopyalama tekniklerinden biridir ve genetik algoritma uygulamalarının çoğu bu tekniği kullanmaktadır.

Jenerasyon aralığı (λ) yöntemi, bütünüyle yer değiştirme yöntemi ile oldukça benzerdir. Aralarındaki fark, eski jenerasyondaki dizilerin belli bir yüzdesinin yeni jenerasyona taşınmasıdır. Örnek olarak λ=0.5'lik jenerasyon aralığı, eski jenerasyondaki bireylerin %50'sinin yeni jenerasyona taşındığı anlamına gelir. %100'lük bir jenerasyon aralığı, bütünüyle yer değiştirme yöntemini oluşturur. Yer değiştirilecek diziler rasgele seçilebilir veya bir tür elistik model kullanılabilir. Elistik model'de jenerasyondaki en yüksek uygunluğa sahip olan dizinin bir sonraki jenerasyonda devamı sağlanır. Fakat jenerasyon aralığı yöntemi, dizilerin bir önceki jenerasyondan daha yüksek uygunluk değerlerine sahip olmasını garanti etmez. Böylece özellikle küçük jenerasyon aralığı değerlerinde jenerasyon aralığı yöntemi, bütünüyle yer değiştirme yönteminden oldukça farklı sonuçlar verir.

Kararlı durum kopyalama yöntemi ise kararlı durum genetik algoritmada (KDGA) kullanılan kopyalama yöntemidir. Bu kopyalama yönteminde, her jenerasyondaki popülasyona ait dizilerden sadece 1 adedi silinir ve yeni 1 adet dizi üretilir. Böylece her jenerasyonda sadece 1 adet dizi üretilmiş olur.

Seçim yöntemleri ise genel olarak en iyi olan yaşar prensibine göre oluşturulur. Amaç, yeni jenerasyonda daha yüksek uygunluğa sahip bireylerin sayısını arttırmaktır. Genetik algoritmada kullanılan seçim yöntemlerini aşağıdaki gibi 3 ana grupta toplayabiliriz.

1) Uygunluk değeri orantılı seçim yöntemleri (Fitness proportionate selection)
2) Turnuva seçim yöntemi (Tournament selection)
3) Sıralı seçim yöntemi (Rank selection)

Uygunluk değeri orantılı seçim yöntemlerinin genel özelliği, nüfusu oluşturmak için kullanılacak dizileri seçmek için, dizilerin uygunluk değerlerine bakılmasıdır. Bu yöntemlerden, rulet çarkı seçim yönteminde nüfus içindeki her dizi, rulet çarkı üzerinde uygunluk değeri ile orantılı olacak kadar yer kaplar. Örneğin, $f(x) = x^2$ fonksiyonunun

[0,31] tam sayı aralığında en büyükleme probleminin genetik algoritma ile çözümünün elle simülasyonu yapılırsa algoritma, belirlenen aralıktan rasgele seçilen x değişken değerlerinden başlangıç nüfusunun oluşturulmasıyla başlar. Başlangıç nüfusu sayısı $n = 4$ olarak verildiğinde, verilen aralıktan rasgele seçilen başlangıç nüfusu,

0 1 1 0 1
1 1 0 0 0
0 1 0 0 0
1 0 0 1 1

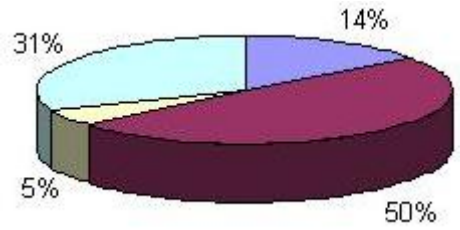
olabilir. Bu dört diziden oluşan örnek nüfusun amaç ya da uygunluk fonksiyonu, (f) , değerleri ve nüfusun toplam uygunluğuna göre her dizinin yüzdesi, Çizelge 3.2.'de gösterilmektedir.

Çizelge 3.2. Örnek problemin dizileri ve uygunluk değerleri [1]

No.	Diziler	Uygunluk	% Toplam
1	0 1 1 0 1	169	14
2	1 1 0 0 0	576	50
3	0 1 0 0 0	64	5
4	1 0 0 1 1	361	31
Toplam		1170	100.0

Örnek problemin nüfusunu oluşturan dört dizinin toplam uygunluk değeri 1170 olarak bulunmaktadır. Ağırlıklı rulet çarkına göre bu jenerasyonun kopyalanması Şekil 3.1'de gösterilmektedir. Dizilerin, gelecek jenerasyondaki kopyalanmasını bulmak için, ağırlıklı rulet çarkının dört kez döndürüldüğü varsayılır.

Örnek problem için, 1 numaralı dizinin uygunluk değeri 169'dur ve bu değerin toplam uygunluk değerine göre yüzdesi 14'dür. Sonuç olarak, 1 nolu dizi rulet çarkının yüzde 14'lük kısmını kaplar. Rulet çarkı bir defa döndürüldüğünde 1 nolu dizinin gelme olasılığı 0.14'dür. Uygunluk değeri yüksek olan diziler rulet çarkı üzerinde daha büyük alan oluşturacağından, rulet çarkının bir defa döndürülmesinde bu dizilerin gelme olasılığı daha yüksek dolayısıyla bir sonraki jenerasyonda görünmesi şansı da daha fazladır.



Şekil 3.1. Her bir dizinin uygunluk değerine göre rulet çarkında kapladığı alan

Turnuva seçim yönteminde, diziler rasgele olarak gruplanır ve gruptaki diziler aralarında seçim işlemi için rekabete sokulur. Grup içinde, en yüksek uygunluk değerine sahip olan dizi, bir sonraki jenerasyondaki nüfusu oluşturmak için dizi olarak seçilmiş olur. Bu işlem, toplam popülasyon sayısı oluşturuluncaya kadar devam eder. Bu seçim yönteminde grup büyüklüğü önemlidir ve seçim yönteminin performansını önemli ölçüde etkiler. Bazı uygulamalarda grup büyüklüğü 2 olarak seçilirken bazı uygulamalarda çok daha büyük gruplar oluşturulur. Turnuva seçim yöntemi küçük popülasyon kullanan uygulamalarda uygunluk değeri orantılı seçim yöntemlerinden daha iyi sonuç verir. Bu seçim yönteminde bireyin seçilme olasılığı eşitlik (3.1) ile verilir [24] .

$$p_i = \frac{1}{N^q}\left((N-i+1)^q - (N-i)^q\right) \qquad (3.1)$$

Eşitlik (3.1)'de; N popülasyon büyüklüğü, p_i $i.$bireyin seçilme olasılığını ve q turnuva büyüklüğünü temsil eder.

Sıralı seçim yöntemi ise Baker tarafından gerçekleştirilmiştir [24]. Bu yöntemde diziler uygunluk değerlerine göre büyükten küçüğe sıralanır. Dizinin seçim şansı uygunluk değerinden ziyade oluşturulan liste içindeki yerine bağlıdır. Oluşturulan liste içindeki dizilere kopya sayısı atanır. Baker'in geliştirdiği Doğrusal Atama Fonksiyonunu kullanan yöntemin algoritması aşağıda verilmiştir.

1) Popülasyondaki her dizi uygunluk değerine göre küçükten büyüğe doğru sıralanır.

2) Kullanıcı son sıradaki bireyin beklenen seçim değerini hesaplar (Max).

3) t adımda, i inci bireyin beklenen seçim şansı Eşitlik (3.2)'den bulunur.

4) İşlem N popülasyon büyüklüğüne ulaşıncaya kadar devam eder.

$$Min + (Max - Min)\left((Sira_{(i,t)} - 1)N - 1\right) \qquad (3.2)$$

Eşitlik (3.2)'de Min=1.bireyin seçilme şansıdır ve $(Min = 2 - Max)$ olarak seçilir. $Max = (1 \le Max \le 2)$ olarak seçilir. Baker bu değeri 1.1 olarak seçmiştir.

Bir başlangıç nüfusundan başarılı nüfuslar üreten basit genetik algoritmanın performans ve verimliliğini etkileyen en önemli faktörlerden birisi nüfusun boyutudur. Algoritmanın başında belirlenen nüfus sayısının az olması genetik algoritmaları genellikle başarısız yapar. Çünkü nüfus sayısının az olması problemin çözümü için araştırma uzayından verimsiz örnek sayısı sağlar. Nüfus sayısının çok olması araştırma uzayında daha çok ve etkili örnek seçilmesi açısından uygundur. Böylece genetik algoritmalar daha çok bilgi içeren araştırmalar yapabilir ve alt optimal çözümlere hatalı yaklaşımlar önlenir. Diğer taraftan, büyük nüfus sayılarında her jenerasyonda daha çok değerlendirme yapıldığından olası sonuca yaklaşım daha yavaş olmaktadır.

3.4.2. Çaprazlama

Kopyalama işleminden sonra her jenerasyonda, mevcut nüfus dışında, aynı uzay içinde farklı noktalara ulaşmak ya da araştırma uzayının diğer noktalarını da incelemek için yeniden düzenleyici genetik işlemler uygulanarak yeni nüfus içinde bazı değişimler ortaya çıkarılır. Çaprazlama, dizilerin karakterleri birbirleriyle değiştirilerek farklı dizilerin elde edilmesi işlemidir. Çaprazlama işlemi de farklı yöntemlerle (tek kesimli, çift kesimli, çok kesimli, tekdüze, tersleme) gerçekleştirilebilmektedir.

Genetik algoritmada geleneksel olarak, çaprazlama nokta sayısı 1'dir. Algoritmanın başında belirlenen çaprazlama oranı (p_c), bir dizi için uygulanacak çaprazlama işleminin olasılığını gösterir. Bu oran tüm algoritma boyunca sabit olabileceği gibi, jenerasyonlara göre değişebilir. Bu oranın çok yüksek olması nüfus içinde yeni dizilerin daha hızlı oluşmasını sağlar. Dolayısıyla yüksek performanslı diziler de hızlı bir şekilde atılabilir. Diğer taraftan, çaprazlama oranı çok düşük olursa değişime uğrayacak dizi sayısı az olacağından araştırma durgunlaşıp yavaşlayabilir.

Basit bir çaprazlama işlemi iki adımda yürütülür. İlk olarak çaprazlama işlemine uğrayacak diziler rasgele belirlenir. İkinci adımda ise çaprazlama işlemini uygulamak için dizi üzerinde bir k tam sayı pozisyonu yine rasgele seçilir. Diziyi oluşturan karakterlerin sayısı ya da dizi uzunluğu l ise, k çaprazlama pozisyon değeri 1 ile l-1 arasında olmalıdır. Buna göre rasgele seçilen bir dizi çiftinde, k+1 ile l arasındaki bütün karakterler karşılıklı olarak yer değiştirerek iki yeni dizi elde edilir. Örnek problemin başlangıç nüfusundan A_1 ve A_2 dizileri seçilmiş olsun.

$$A_1 = 0 \quad 1 \quad 1 \quad 0 \quad | \quad 1$$

$$A_2 = 1 \quad 1 \quad 0 \quad 0 \quad \vert \quad 0$$

Çaprazlama işleminin uygulanacağı nokta k=4 olarak seçildiğinde, işlem sonucu elde edilen yeni diziler, (çaprazlama işleminin uygulanacağı nokta "$\vert$"sembolüyle gösterilmektedir.)

$$A_1' = 0 \quad 1 \quad 1 \quad 0 \quad \vert \quad 0$$

$$A_2' = 1 \quad 1 \quad 0 \quad 0 \quad \vert \quad 1$$

olur ve yeni jenerasyonda kopyalama işlemi için değerlendirilir.

Çift kesimli çaprazlama işleminde tek kesimli çaprazlama işlemine benzer işlem, kesimin dizi boyunca iki yerde yapılması ile aynen tekrarlanır.

$$A_1 = 1 \quad \vert \quad \mathbf{0} \quad \mathbf{0} \quad \mathbf{1} \quad \mathbf{0} \quad \vert \quad 1 \quad 1$$

$$A_2 = 1 \quad \vert \quad \mathbf{1} \quad \mathbf{1} \quad \mathbf{0} \quad \mathbf{1} \quad \vert \quad 0 \quad 0$$

Burada dizi çiftleri iki farklı yerden kesilerek çaprazlama uygulanır. Koyu renkli genler çaprazlanmıştır.

$$A_1' = 1 \quad \vert \quad \mathbf{1} \quad \mathbf{1} \quad \mathbf{0} \quad \mathbf{1} \quad \vert \quad 1 \quad 1$$

$$A_2' = 1 \quad \vert \quad \mathbf{0} \quad \mathbf{0} \quad \mathbf{1} \quad \mathbf{0} \quad \vert \quad 0 \quad 0$$

Görüleceği üzere bu kez de yeni şekil alan diziler sonucunda temsil ettikleri sayılar yine değişmiştir. İki noktalı çaprazlamada her dizi üç parçaya bölünür. Bu parçalardan karşılıklı her ikisinin çaprazlama yer değiştirmesi ile birbirinden farklı 6 tane yeni dizi elde edilir. Aşağıda iki noktalı çaprazlamaya maruz bırakılan dizilerde düz, italik ve koyu kısımlar olmak üzere üç parça gösterilmiştir.

000*10111*0**0110**

001*11110*00**1001**

Çaprazlamanın sırası ile düz, italik ve koyu parçalar arasında yapılması ile birbirinden farklı kromozomlar düz kısımlar arasında,

001*10111*0**0110**

000*11110*00**1001**

italik kısımlar arasında,

000*11110*00**0110**

001*10111*0**1001**

ve son olarak da koyu kısımlar arasında,

000*10111*0**1001**

001*11110*00**0110**

şeklinde olmak üzere 6 tanedir. Yeni dizilerin tümünün kullanılması gerekli değildir. Bu seçimde ya düz, ya italik veya koyu parçaların değiştirilmesinden elde edilen yeni dizilerden istenilen kadarı rasgele seçim ile nüfusa katılabilir.

Çok kesimli çaprazlama, dizilerin çapraz olarak ikiden daha fazla yerden rasgele kesilerek karakterlerin yer değiştirilmesi ile sağlanır.

A_1= 1 | **0 0** | **1 0** | 1 | 1

A_2= 1 | **1 1** | **0 1** | 0 | 0

Bu çaprazlama sonucunda oluşan diziler

A_1' = 1 **1 1 1 0** 0 1

A_2' = 1 **0 0 0 1** 1 0

gibidir. Burada da karşılıklı parçalar arasında çaprazlama ile çok sayıda yeni dizi elde edilebilir.

Tekdüze çaprazlama işleminin aslı rasgele hanelerin iki dizi arasında yer değiştirmesi ile olur.

1 **0** 0 **1** 0 1 **1**

↑ ↓ ↑ ↓ ↑ ↓

1 **1** 1 **0** 1 0 **0**

Burada her hane için yazı tura atılır. Örneğin yazı gelirse çaprazlama yapılsın, tura gelirse çaprazlama yapılmasın kuralına uyulur. Koyu renkli karakterler için yazı geldiğinden bu karakterler için çaprazlama uygulanarak karakterlerin yeni şekilleri

1 1 0 0 0 1 0

1 0 1 1 1 0 1

biçiminde oluşmuştur.

3.4.3. Mutasyon

Araştırma uzayının diğer noktalarını da incelemek için uygulanan genetik işlemlerden bir diğeri de mutasyondur. Algoritmanın başında belirlenen mutasyon oranı p_m, tüm nüfus içinde değişime uğrayacak karakter oranını belirler. Bu oran tüm algoritma boyunca sabit alınabileceği gibi jenerasyonlara göre farklı değerlerde de seçilebilir. Bu işleme göre, bir jenerasyonda yaklaşık olarak $p_m * n * l$ karakterde mutasyon meydana gelir. Burada *n* nüfus

sayısı, l ise dizi uzunluğudur. Mutasyon tüm nüfus içinden rasgele seçilen bir karakterin değerinin değiştirilmesidir. Örneğin, binary olarak kodlanan bir problemde mutasyon işlemi, seçilen bir karakterin (bitin) değerini 1 ise 0, 0 ise 1 olarak değiştirmektir. Mutasyon oranının düşük olması, verilen herhangi bir karakter pozisyonunun tüm nüfus içinde daima tek bir değere yakınsayarak kalmasını önlemeye yardımcı olur. Yüksek oranda mutasyon ise tamamen rasgele bir araştırmayı baştan kabul etmek anlamına gelir.

Muhlenbein ve Backin'in yaptığı çalışmalar, mutasyonun olasılığının p_m, optimal oranının dizinin uzunluğu ve problemin çözüm uzayı ile orantılı olduğunu ortaya koymuştur [24]. Çaprazlama olasılığı, genellikle 0.25 ila 1 arasında, mutasyon olasılığı 0.01 ila 0.001 arasında seçilir. Özellikle küçük nüfus sayılarında sistemin performansını çaprazlama olasılığından çok mutasyon olasılığı belirler.

Yapılan bazı çalışmalarda ise dinamik mutasyon ve çaprazlama olasılıkları kullanılmıştır. Bu çalışmalarda, jenerasyon sayısı arttıkça mutasyon olasılığı artmakta fakat çaprazlama olasılığı azalmaktadır. Dinamik çaprazlama ve mutasyon olasılıkları aşağıdaki gibi ifade edilebilir [24].

$$P_c = e^{(-T_G / M)}$$
$$P_m = e^{(0.005*T_G / M)} \qquad (3.3)$$

Eşitlik (3.3)'de T_G o andaki jenerasyon değerini, M maksimum jenerasyon sayısını göstermektedir.

Genetik algoritmanın yürütülmesi sırasında seçilen bu parametreler en iyileme probleminin performansını etkileyen en önemli faktörlerdir.

3.5. Genetik Algoritmanın Elle Yürütülmesi

Basit genetik algoritma işleyiş adımları aşağıdaki gibi sıralanabilir:

1) Başlangıç: Probleme aday çözümler için n tane l bit uzunluğunda dizilerden oluşmuş başlangıç nüfusunun rasgele oluşturulması,

2) Uygunluk: Nüfus içindeki her x dizisi için $f(x)$ uygunluk değerinin hesaplanması,

3) n tane diziye sahip yeni nüfus oluşuncaya kadar aşağıdaki adımların tekrar edilmesi,

4) Belirlenen seçim yöntemine göre başlangıç nüfusu içinden dizilerin seçilmesi. (Yüksek uygunluk değeri seçilme şansını artırır)

5) Yeni diziler oluşturmak için çaprazlama olasılığına göre, rasgele seçilen dizi çiftlerinin, rasgele belirlenen noktalarından çaprazlama işlemine tabi tutulması, (Eğer çaprazlama yapılmazsa yeni diziler eski dizilerin kopyası olacaktır)

6) Mutasyon olasılığına göre rasgele dizilerde mutasyon işleminin uygulanması,

7) Eldeki nüfusun yeni nüfusla yer değiştirilmesi,

8) Test: Bir jenerasyon boyunca yürütülen bu işlemlerin programın başında belirlenen bir jenerasyon sayısı kadar tekrarlanıp bitirilmesi veya en büyükleme ya da en küçükleme probleminde jenerasyonlarda elde edilen optimal değerler arasındaki fark sıfırlandığında ya da belli bir değere yakınsadığında sona erdirilmesi,

9) Döngü:2. adıma geri dönülmesi.

Basit bir genetik algoritmanın bir en iyileme problemine adım adım uygulanmasını görmek için bölüm 3.4.1'de verilen $f(x) = x^2, x \in [1,31]$, fonksiyonunun en büyüklemesi ele alınsın. Genetik algoritmanın uygulanmasında öncelikle, problemdeki değişkenleri sonlu uzunlukta dizilerle göstermek için kullanılacak koda karar vermek gerekir. Bu problem için, x değişkeni 5 bit uzunluğunda işaretsiz tam sayı olarak ikili sayı sisteminde kodlanabilir. Böylece x değişkeninin alt ve üst sınır değerleri 00000 (0) ve 11111 (31) olarak gösterilebilir. Başlangıç nüfusu, $n = 4$ için, bölüm 3.5.1'de verilen başlangıç nüfusu ile aynı seçilirse, algoritmanın bir sonraki safhası başlangıç nüfusundan yeni jenerasyonun seçilmesidir. Uygunluk fonksiyonuna göre aldıkları değere bağlı olarak, gelecek jenerasyona geçmesi beklenen dizileri belirlemek için ağırlıklı rulet çarkının dört kez çevrildiği kabul edilir. Çizelge 3.3'de gösterildiği gibi 1 ve 4 nolu diziler bir sonraki jenerasyonda bir kez görünmesine karşın 2 nolu dizi iki kez tekrarlanarak görülmektedir. 3 nolu kromozomun etkisi ise uygunluk değerinden dolayı sıfırdır.

Genetik algoritma içinde kopyalamadan sonra yürütülecek ikinci işlem çaprazlamadır. Çarprazlama oranı $p_c = 1.0$ seçildiğinde tüm diziler çaprazlama işlemine uğrar. Seçilen 1 ve 2 nolu diziler için çaprazlama yeri 4 seçildiğinde, 01101 ve 11000 dizilerinden farklı olarak 01100 ve 11001 dizileri elde edilir. Benzer şekilde, 3 ve 4 nolu diziler için çaprazlama yeri 2 olarak seçilirse, çaprazlama işleminden sonra iki farklı dizi elde edilmiş olur. Son işlem bit bit yürütülen mutasyon işlemidir. Örnek problem için mutasyon oranı 0.001 olarak verildiğinde,

tüm nüfus için toplam bit sayısı 20 olduğundan mutasyona uğrayacak bit sayısı 20*0.001'den 0.02 bit olarak bulunur. Sonuç olarak nüfus içindeki hiçbir bitin değeri 0'dan 1'e ya da 1'den 0'a değişmeden kalır.

Genetik algoritmanın tüm işlemlerinin ardından yeni bir jenerasyon nüfusunun dizileri belirlenmiş olur ve yeni jenerasyon test edilmeye hazırdır. Basit bir genetik algoritmayla yaratılan yeni diziler, onlu sayı sistemine göre x değerlerine dönüştürülerek uygunluk fonksiyonuyla değerlendirilir.

Çizelge 3.3. Genetik algoritmanın elle yürütülmesi [1]

Dizi No.	**Başlangıç Nüfusu**	x **Değeri**	$f(x)$ x^2	**Seçilme Olasılığı**	**Beklenen Gerçek-Sayı**
1	**0 1 1 0 1**	**13**	**169**	**0.14**	**0.58**	**1**
2	**1 1 0 0 0**	**24**	**576**	**0.49**	**1.97**	**2**
3	**0 1 0 0 0**	**8**	**64**	**0.06**	**0.22**	**0**
4	**1 0 0 1 1**	**19**	**361**	**0.31**	**1.23**	**1**
Toplam			**1170**	**1.00**	**4.00**	**4.0**
Ortalama			**293**	**0.25**	**1.00**	**1.0**
Maksimum			**576**	**0.49**	**1.97**	**2.0**

Çaprazlanacak Diziler	**Çaprazlanacak Dizi No.**	**Çaprazlama Noktası**	**Yeni Nüfus**	x **Değeri**	$f(x)$ x^2
0 1 1 0 | 1	**2**	**4**	**0 1 1 0 0**	**12**	**144**
1 1 0 0 | 0	**1**	**4**	**1 1 0 0 1**	**25**	**625**
1 1 | 0 0 0	**4**	**2**	**1 1 0 0 1**	**27**	**729**
1 0 | 0 1 1	**3**	**2**	**1 1 0 0 1**	**16**	**256**
Toplam					**1754**
Ortalama					**439**
Maksimum					**729**

Bir jenerasyon boyunca yürütülen tüm işlemlerin gösterildiği Çizelge 3.3'e bakıldığında dizilerin performansının bir jenerasyon sonra bile oldukça iyi derecede arttığı, uygunluk fonksiyonunun toplam, ortalama ve maksimum değerlerindeki artıştan gözlenebilmektedir. Bir jenerasyon boyunca yürütülen bu işlemler en büyükleme ya da en küçükleme probleminde

jenerasyonlarda elde edilen optimal değerler arasındaki fark sıfırlandığında ya da belli bir değere yakınsadığında sona erdirilir. Ayrıca bir genetik algoritma, programın başında belirlenen bir jenerasyon sayısı kadar tekrarlanıp bitirilebilir. Bu sayının yeterince büyük olması sonuçta elde edilecek değerin, fonksiyonun optimal çözümü olma şansını artırır [1, 24, 27].

KAYNAKLAR

[1] Kahvecioğlu, A., *Uçuş kontrol sistem tasarımında katlı- model yaklaşımı ve genetik algoritma tekniğinin uygulanması*, Doktora Tezi, Anadolu Üniversitesi, Fen Bilimleri Enstitüsü, Eskişehir, 2000.

[24] Topuz, V., Bulanık genetik proses kontrolü, Doktora Tezi, Marmara Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul, 2002.

[26] Chen, G., Pham, T. T., Introduction to Fuzzy Sets, Fuzzy Logic and Fuzzy control Systems, CRC Pres LLC, Florida, U.S.A., 2001.

[27] ŞEN, Z., Genetik algoritmalar ve en iyileme yöntemleri, Su Vakfı Yayınları, İstanbul., 2004.

[28] Goldberg, D.E., Genetic algorithms in search, optimization, and machine learning, Addison-Wesley Publishing Company, Inc, New York, U.S.A., 1989.

[29] KURT, M., SEMETAY, C., Genetik algoritma ve uygulama alanları, Marmara Üniversitesi Teknik Eğitim Bölümü, İstanbul. http://www.mmo.org.tr/muhendismakina/arsiv/2001/ekim/Genetik_Algoritma.htm (Temmuz 2006)