YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GENETİK ALGORİTMA

Nurdan ÇETİN

F.B.E. Matematik Anabilim Dalı Matematik Programında Hazırlanan

YÜKSEK LİSANS TEZİ

Tez Danışmanı : Doç. Dr. Fatma TİRYAKİ

Prof. Dr. Mehmet AHLATÇIOĞLU

Prof. Tahir ŞİŞMAN

İSTANBUL, 2002

İÇİNDEKİLER

SİMGE LİSTESİ

E	Çözüm Uzayı (Cevap Yüzeyi)
f_k	k. bireyin uyum değeri
f'	Ölçeklendirilmiş Uyum
$\overline{f}$	Popülasyonun ortalama uyumu
G	Yığın (kuşak) aralığı
L	Her bir yapıdaki (dizideki) bit pozisyonu sayısı
N	Popülasyonun Genişliği
n	Kromozomun Uzunluğu
P_k	k. bireyin seçilme olasılığı
P_c	Çaprazlama Olasılığı
P_m	Mutasyon Olasılığı
$P(t)$	t. kuşaktaki (jenerasyondaki) popülasyon
$P(t+1)$	(t+1). kuşaktaki popülasyon
S	Seçim Stratejisi
W	Ölçeklendirme Fonksiyonu

KISALTMA LİSTESİ

ANN	Yapay Neural Şebekeler
ARMA	Karma Auto Regresyon
B.G.A.	Basit Genetik Algoritma
DP	Dinamik Programlama
EOQ	Ekonomik Sipariş Sayısı
FLC	Fuzzy Mantık Kontrolü
FNN	Fuzzy Neural Şebekeler
G.A.	Genetik Algoritma
GFDF	Gri Fuzzy Dinamik Programlama
GFLP	Gri Fuzzy Lineer Programlama
GFNN	Genetik Algoritma Bazlı Neural Şebekeler
GIP	Gri Tamsayı Programlama
GLP	Gri Lineer Programlama
JWI	XYZ Şirketi
KV	Karar Verici
LP	Lineer Programlama
MIP	Karma (Karışık) Tamsayı Programlama
PAGA	Paralel Tavlamalı Genetik Algoritma

ŞEKİL LİSTESİ

Sayfa

ÇİZELGE LİSTESİ

Sayfa

ÖNSÖZ

Genetik Algoritma adında da anlaşılabileceği gibi doğadaki evrim mekanizmasına yani genetik ve doğal seleksiyon kurallarına dayanan; kompleks problemlerin çözümünde kullanılan genel amaçlı araştırma algoritmalarıdır. Genetik Algoritmalar, doğada geçerli olan en iyinin yaşaması kuralına dayanarak sürekli iyileşen çözümler üretir. Bunun için "iyi"nin ne olduğunu belirleyen bir uygunluk (fitness) fonksiyonu ve yeni çözümler üretmek için yeniden kopyalama (recombination), değiştirme (mutation) gibi operatörleri kullanır. Genetik Algoritmalar'ın bir diğer önemli özelliği de bir grup çözümle uğraşmasıdır. Bu sayede çok sayıda çözümün içinden iyileri seçilip kötüleri elenmektedir.

G.A., ilk önce aday (rastgele) çözümlerin oluşturulmasıyla başlatılır. Bu çözüm kümesine "popülasyon" denir. Aday çözümler, aynı sayıda elemandan oluşan bir bit dizisi formunda oluşturulur. Bir dizi olarak gösterilen her çözüm, birey ya da "kromozom" olarak bilinir. Böylece her bir kromozom n uzunluğunda bir bit dizisidir.

Kromozomun her biri probleme bir çözüm sunar. Her çözüme, popülasyondaki diğer çözümler ile karşılaştırıldığında ne kadar iyi bir çözüm olduğunu gösteren bir uyum değeri (fitness value) atanır. Bu değer, doğada bir organizmanın yaşayabilmek için mücadele ederken ne kadar başarılı olduğunun belirlenmesine karşılık gelir. Bir bireyin uyum değeri ne kadar büyükse, sonraki kuşakta yaşama ve üreme (çözümün seçilme) şansı o kadar fazla olur. Daha uygun bireylere, popülasyondaki diğer bireyler ile eşlenerek üreme (reproduction) fırsatı verilir. Bu ise, her atanın bazı özelliklerini taşıyan yeni bireylerini meydana getirir. Popülasyondaki düşük uyumlu bireylerin, üreme için seçilme şansları azdır. Bu nedenle yok olurlar. Mevcut kuşağın en iyi bireylerini seçip, yeni bireyler elde etmek için bunları eşleyerek, olası çözümlerden oluşan yeni bir popülasyon yaratılır. Yeni kuşak, önceki kuşağın iyi bireylerinin sahip olduğu özelliklerin büyük bir kısmını içerir. Böylece iyi özellikler, kuşaklar boyu popülasyona dağılır. Eğer, G.A. iyi düzenlenirse, popülasyon problemin optimal çözümüne yakınsar. Bu döngü, optimizasyon kriterleri uyuşuncaya veya belirli sayıda tekrarlamalar yapılıncaya kadar, yani istenen çözüm bulununcaya kadar uygulanır.

Bu çalışmada, yukarıda tanımının ve ana hatlarıyla işleyişinin verildiği G.A.'nın kullanıldığı araştırmalara da yer verilmiştir.

Tezimi hazırlarken bana yardımcı olan ve yönlendiren hocam, Sayın Doç. Dr. Fatma Tiryaki'ye, benim her zaman yanımda olan ve beni destekleyen aileme; çalışmamda bana yardımcı olan arkadaşlarıma teşekkürlerimi sunuyorum.

ÖZET

Bu tez, Darwin 'in evrim teorisini yani "Doğada en iyi uyum sağlayanın yaşaması, neslini devam ettirmesi" temel alan, optimizasyon problemlerinin çözümlerinde kullanılan Genetik Algoritma'yı ele almıştır. Genetik Algoritma, önerilen çözüm adaylarından çaprazlama ve mutasyon operatörlerini kullanarak, yeni çözümler; yani sürekli iyileşen çözümler üreterek optimum çözüme ulaşmaya çalışır.

Genetik Algoritma'nın ele alındığı bu çalışma, dört bölümden oluşmaktadır.

Birinci bölümde, Genetik Algoritma'ya giriş yapılmış, tarihçesi hakkında bilgi verilmiş, uygulama alanlarından ve bir araştırma algoritması (çözüm tekniği) olarak kullanılma nedenlerinden bahsedilmiştir.

İkinci bölümde, öncelikle Genetik Algoritma ana hatlarıyla gözden geçirilmiş ve daha sonra işleyişi ele alınmıştır. İşleyişindeki tüm adımlar; başlangıç popülasyonunun oluşturulması, kodlama, seçme ve genetik operatörler ayrıntılarıyla bulunmaktadır. Bu bölümün son kısmında da Genetik Algoritma parametrelerine (kontrol parametreleri) yer verilmiştir.

Üçüncü bölümde, Genetik Algoritma'nın kullanıldığı çalışmalardan genel çerçevede bahsedilmiştir. Bu araştırmaların herbiri Genetik Algoritma ve işleyişini ana hatlarıyla ele almışlardır.

Son bölümde ise, bir araştırma yöntemi olan G.A.'nın özellikleri özetlenmiştir.

Anahtar Kelimeler: Genetik, popülasyon, kodlama, seçim, genetik operatörler.

ABSTRACT

This thesis deals with Genetic Algorithm (G.A), which is based on Darwin's Evolution Theory, in that; "The survival of the fittest" and which is used in the solution of optimization problems. Genetic Algorithm tries to reach to the solutions by producing new solutions from the proposed solutions.

This study, in which Genetic Algorithm is discussed, has 3 parts.

In the first part; an introduction to G.A. is made, brief history of G.A. is presented, its application fields and the reasons why they are used as a research algorithm (solution techniques) are discussed.

In the second part; first, G.A. is briefly overviewed next its process is told. All the steps in its process; setting up the initial population, coding, selection and genetic operators are presented in details. G.A. parameters (control parameters) are explained at the end of this section.

In the third part; studies in which G.A.'s are used are examplified in general context. Each of these studies outlines G.A. and its process.

In the last part; the characteristics of G.A., a search technique, are summirized.

Keywords: Genetic, population, encoding, selection, genetic operators.

1. GENETİK ALGORİTMAYA GENEL BAKIŞ

Genetik Algoritma (G.A.) adından da anlaşılabileceği gibi doğadaki evrim mekanizmasını örnek alarak oluşturulmuş çözüm tekniğidir (Louis, 1993). G.A.'lar ilk defa Charles Darwin tarafından ortaya atılan, "doğal sistemde güçlülerin hayatta kalması" (en iyinin hayatta kalması) diye özetlenebilecek doğal seleksiyon ve doğal genetik mekanizmasına (genetiğin evrim tezine dayandırılan) dayanan araştırma algoritmalarıdır (Goldberg, 1989).

1.1. Giriş

Genetik Algoritmalar, bazı doğal olayları modelleyen stokastik algoritmalardır. Bu algoritmalar biyolojik evrimin işleyiş biçimini taklit eder. Doğal popülasyonlar (topluluklar), Charles Darwin'in, Türlerin Kökeni (The Origin of Species) adlı kitabında belirttiği gibi doğal seçme (natural selection) ve en iyi uyum yapanın yaşaması (survival of the fittest) prensibine göre kuşaklar boyunca gelişmektedir. Doğada, bireyler arasında yiyecek, su ve barınak gibi kıt kaynaklar veya eşler için yapılan mücadeleler, yüksek uyumlu (highly adapted) ya da uygun (fit) bireylerin daha zayıf olanlara üstünlüğü ile sonuçlanır. İyi uyum yapan bireyler yaşar ve daha fazla yavru sahibi olur. Düşük uyumlu bireyler az sayıda yavru sahibi olacak, belki de hiç yavru sahibi olamayacaktır. Bu, uygun bireylerin genlerinin sonraki her kuşakta (generation) daha fazla bireye dağılması demektir. Farklı atalardan gelen iyi özelliklerin birleşimi ile bazen her bir ataya göre daha büyük uygunluğa sahip süper uygun yavrular türüyebilir. Böylece türler, bulunduğu çevre için giderek daha uygun hale gelirler.

Örnek olarak tavşan topluluğunu (popülasyonu) düşünelim. Bazı tavşanlar diğerlerine göre daha hızlı ve daha açıkgözdür. Böyle tavşanların tilkiler tarafından yenilme olasılığı daha düşüktür. Dolayısıyla çoğu yaşamını sürdürür. Kuşkusuz, yavaş ve durgun olan tavşanların bazıları da şanslı oldukları için yaşayacaklardır. Bu yaşayan tavşan popülasyonu üremeye başlayacaktır. Popülasyonun çoğunluğu, tilkilerden kurtulan daha hızlı ve açıkgöz tavşanlar olduğundan, yavru tavşanlar önceki popülasyonlardakilerden ortalama olarak daha hızlı ve açıkgöz olacaktır. G.A.'lar bu süreci taklit eden stokastik iteratif yöntemlerdir (Yeniay, 1999).

Genetik Algoritmalar'ın mühendislik, tasarım, sinir ağı tasarımı, ekonomi ve finans, yapay hayat (artificial life), haberleşme ağları, tahmin (hava, deprem, at yarışı vb.), suç şüphelilerinin teşhisi, müzik besteleme, kombinatoryal optimizasyon (kutu paketleme, sıralama ve çizelgeleme, steiner ağacı, ulaştırma vb) gibi birçok alanda kullanıldığı yüzlerce çalışma mevcuttur.

1.2. Genetik Algoritmanın Tarihçesi

İlk defa 1960'larda John Holland tarafından ortaya atılan; kendisi, öğrencileri ve meslektaşları tarafından geliştirilen (Louis, 1993) G.A.'lar, mühendislik dünyasında birçok alanda genişçe işlenmiş, deneyleri yapılmış ve uygulanmıştır (Mitchell ve Forrest, 1993). Ayrıca 1975'de G.A.'nın teorik çatısının verildiği John Holland'ın "Doğal ve Yapay Sistemlerde Adaptasyon" (Adaptation in Natural and Artificial Systems) isimli kitabı yayınlandı (Louis, 1993). Holland'ın G.A.'sı Basit Genetik Algoritma (B.G.A.) olarak adlandırılmaktadır (Yeniay,1999). Holland'ın asıl amacı, adaptasyon olgusunu tabiatta meydana geldiği şekliyle resmi olarak incelemek ve özel problemleri çözmek için algoritmalar düzenlemek yerine, doğal adaptasyon mekanizmalarının bilgisayar sistemlerine aktarılabileceği yollar geliştirmekti (Tanrıseven, 2000).

1992'de, John Koza Genetik Algoritma'yı, verilen bazı görevleri gerçekleştirecek programlar geliştirmek için kullandı. Koza, bu yönteme genetik programlama ("genetic programming" (GP)) adını vermiştir (Louis, 1993).

1.3. Genetik Algoritmaların Uygulama Alanları

G.A.'lar Bilgisayar Bilimi, Mühendislik, Yöneylem Araştırması, Sosyal Bilimler, Tıp, Matematik vb alanlarda karşılaşılan çeşitli problemlerin çözümünde kullanılmaktadır. G.A.'lar global optimal çözümü bulmayı garanti etmezler, ancak kabul edilebilir hızla, kabul edilebilir ölçüde iyi çözümler bulunmasında genel olarak başarılıdırlar. Belirli problemlerin çözümünde özel teknikler varsa, bunların hem sonucun doğruluğu, hem de hız açısından G.A.'lara göre daha iyi işlemesi olasıdır. G.A.'ların esas alanı, bu tür tekniklerin olmadığı alanlardır. Mevcut tekniklerin iyi işlediği yerlerde bile, bu teknikleri G.A.'lar ile birleştirerek ilerlemeler sağlanmıştır (Cantoni vd., 1999).

G.A. uygulamalarında başarının anahtarı, G.A.'yı etkili bir şekilde kullanmak ve zindelik, güçlülük değerlendirmesini anlamlıca yapabilmektir. G.A.'ların cazibesi, güçlü araştırma algoritmalarında kolay ve zarif olmasının yanısıra çok boyutlu problemlerden iyi sonuçlar çıkarabilme gücünden de kaynaklanmaktadır (Mitchell ve Forrest, 1993; Bodnovich, 1995).

G.A.'lar problem çözmede alternatif metodlar sağlamakla kalmaz, aynı zamanda problemlerin çoğunda diğer geleneksel yöntemleri de tutarlıca gerçekleştirir. Mesela optimal parametreler bulmada yer alan gerçek dünya problemleri geleneksel metodlar için zor olabilir; fakat G.A.'lar için idealdir. G.A.'lar, optimizasyondaki etkili performanslarından dolayı yanlış

kanıyla işlem optimizeri olarak görülmektedirler. Aslında Genetik Algoritma'lara çok fazla bakış açısı vardır. Birçok kullanıcı, G.A.'ya problem çözücü olarak bakmaktadır. Fakat bu kısıtlayıcı bir bakış açısıdır.

G.A.'lar,

- Araştırma alanı geniş, kompleks ve anlaşılması zayıfsa,
- Konudaki bilgi azsa ya da eldeki uzman bilgi, araştırma alanını daraltmada zorlanıyorsa,
- Matematiksel analiz elde edilemiyorsa,
- Geleneksel araştırma metodları başarısız olmuşsa

faydalı ve etkilidirler.

G.A. yaklaşımının avantajı, zorluklarla ve hedeflerle başa çıkmadaki rahatlığıdır.

G.A.'lar, problem çözme ve modellemede (taslak oluşturmada) kullanılmaktadır. G.A.'lar, bilimsel mühendislik problemlerinde, iş hayatında, ekonomi ve finansta, yapay hayatta (artificial life), haberleşme ağlarında ve eğlencede uygulanmaktadır. Bunlar:

Optimizasyon: G.A.'lar, içinde sayısal optimizasyon ve seyyar satıcı problemi gibi bütünleştirici optimizasyonların, devre dizaynlarının iş ve dükkan programlamalarının, video ve ses kalitesi optimizasyonlarının yer aldığı bir çok optimizasyon işinde kullanılmaktadır (Goldstein, 1991; Louis, 1993).

Otomatik Programlama: G.A.'lar, spesifik işler için bilgisayar programı geliştirerek, hücresel otomasyon ve network gibi bilgisayarla ilgili yapıları geliştirmekte kullanılmaktadır.

Bilgisayar ve robot öğrenimi: G.A.'lar, içerisinde sınıflama, tahmin ve protein yapısı tahmininin yer aldığı birçok bilgisayarlı öğrenim uygulamalarında kullanılmaktadır. Aynı zamanda networkler tasarlamak, sembolik üretim sistemlerine yönelik öğrenme kurallarını geliştirmek, robot tasarlamak ve kontrol etmekte kullanılmaktadır.

Ekonomik Modeller: G.A.'lar, açık artırmayla satış stratejilerini, yenilik süreçlerini, ekonomik piyasaların önceliklerini tasarlamakta kullanılmaktadır.

Bağışıklık Sistemi Modelleri: G.A.'lar, içerisinde bireyin yaşamındaki somatik mutasyon da olan doğal bağışıklık sisteminin çeşitli yönlerini tasarlamada (düzenlemede) kullanılmaktadır.

Ekolojik Modeller: G.A.'lar, biyolojik silahlanma yarışı, ortak yaşam ve ekolojilerdeki kaynak akışı gibi ekolojik fenomenlerin tasarlanmasında kullanılmaktadır.

Popülasyon Genetiği Modelleri: G.A.'lar, "Bir gen hangi şartlar altında tekrar birleşim (rekombinasyon) için evrimsel olarak yaşayabilir?" gibi popülasyon genetiği sorularına cevap bulmada model oluşturmaktadır.

Evrim ile öğrenim arası etkileşimde: G.A.'lar, bireyin öğrenmesiyle türlerin evriminin birbirini nasıl etkilediğini incelemede kullanılmaktadır.

Sosyal Sistem Modellerinde: G.A.'lar, işbirliğinin gelişmesini (evrimleşmesi), iletişimin gelişmesini ve karıncalardaki birbirini takip etme davranışları gibi sosyal sistemlerin evrimle ilgili yönlerini inceler.

1.4. Genetik Algoritmaların Kullanılma Nedenleri

- G.A., doğadaki evrim mekanizmasını örnek alan bir arama metodudur ve bir veri grubundan özel bir veriyi bulmak için kullanılır (Goldstein, 1991; Valenzuela, 1995).

- G.A.'lar özellikle araştırmacının kesin, konu uzmanı olmadığı zamanlarda çok yardımcıdır. Çünkü G.A.'lar, kendi alanlarını araştırma ve o alandan bilgi edindirmede yeteneklidirler.

- G.A.'lar değerlendirme için yeni ve daha iyi sonuçlar üretmenin yanı sıra varolan potansiyel sonuçları değerlendirmek için de tasarlandıklarından dolayı başka alternatifler için büyük yardım sağlarlar (Mitchell ve Forrest, 1993).

- G.A.'lar klasik yöntemlerin çok uzun zamanda yapacakları işlemleri kısa bir zamanda çok net olmasa da yeterli bir doğrulukla yapabilir.

- G.A., doğadaki evrimin yöntemlerini kullanan bir arama türüdür. Bu yöntem sayesinde, klasik yöntemlerle çözülmesi çok zor kimi zaman, imkansız olan problemler (NP-tam, NP-complete) çözülebilmektedir (Goldberg, 1989; Lawrence, 1990).

- G.A.'lar, çeşitli mühendislik problemlerini çözmek için kullanılmışlar ve optimizasyon problemleri için oldukça etkili olmuşlardır.

- G.A., gerçek optimal değeri bulmayı garanti etmez, fakat uzun nesiller sonunda optimal değere çok yakın çözümler bulunmasını sağlar (Goldstein, 1991; Valenzuela, 1995).

- Problem çözmedeki kullanılışlığı ve zarifliği, G.A.'yı diğer sıralı araştırma (graded search), rasgele araştırma (random search) vb geleneksel metodlar arasından tercih edilir hale getirmiştir.

- Çok amaçlı problemleri çözerken G.A., hedefler açısından bir çok doyurucu çözümler sunar ve karar verecek kişinin en iyisini seçmesine müsaade eder. G.A., çok amaçlı problemlerde karar vermenin taslak aşamasında yardımcı olur (Mitchell ve Forrest, 1993).

2. GENETİK ALGORİTMANIN DOĞUŞU, GELİŞİMİ VE İŞLEYİŞİ

2.1. Genetik Algoritmaya Giriş

Çözüm uzayının çok büyük olduğu gerçek hayat problemleri için eniyi çözümün bulunması geliştirilen özel algoritmalarla bile çok uzun zaman almaktadır. Bu nedenle bu tür problemler için eniyiye yakın çözüm veren sezgisel tekniklerin geliştirilmesi önem kazanmaktadır. Sezgisel teknikler, makul zamanda iyi bir çözüme ulaşmak için problemdeki bilgiyi kullanırlar. Ancak, klasik eniyileme tekniklerinin aksine bu yaklaşımlar global eniyiyi bulmayı garanti etmezler. Sezgisel teknikler, "çözüm kurucu" ve "çözüm iyileştirici" olmak üzere iki ayrı sınıfta incelenmektedirler. Çözüm kurucular çeşitli kuralları kullanarak problem için bir çözüm elde ederken çözüm iyileştiriciler, elde edilen bir başlangıç çözümünü bitirme koşulu sağlanıncaya kadar adım adım iyileştirmeye çalışırlar. Bilinen çözüm iyileştirici sezgiseller bir problem için global eniyiyi bulmada çok başarılı değildirler. Son yıllarda çözüm iyileştirici sezgisel teknikler sınıfında bulunan ve global eniyiyi bulmada başarılı olan yeni teknikler geliştirilmiştir. Bu tekniklerden yaygın olarak kullanılanları, Tabu Arama, Genetik Algoritmalar , Sinir Ağları ve Tavlama Benzetimi dir (Dengiz ve Altıparmak, 1998).

SEZGİSEL TEKNİKLER

1. Çözüm Kurucu

2.Çözüm İyileştirici

Çözüm iyileştirici olarak geliştirilen yeni teknikler:

- Tabu Arama (TA)
- Genetik Algoritma (GA)
- Sinir Ağları (SA)
- Tavlama Benzetimi (TB)

2.1.1. Genetik Algoritmanın Tanımı

G.A.'ların oldukça başarılı olduğu alanlardan biri, optimizasyondur. Bir optimizasyon problemi için Genetik Algoritma'lar, doğada geçerli olan en iyinin yaşaması kuralına dayanarak sürekli iyileşen çözümler üretir. Bunun için "iyi" nin ne olduğunu belirleyen bir *uygunluk* (fitness) fonksiyonu ve yeni çözümler üretmek için *yeniden kopyalama* (recombination), *değiştirme* (mutation) gibi operatörleri kullanır. Genetik Algoritmaların bir diğer önemli özelliği de bir grup çözümle uğraşmasıdır. Bu sayede çok sayıda çözümün içinden iyileri seçilip kötüleri elenebilir.

G.A., ilk önce aday (rasgele) çözümlerin oluşturulmasıyla başlatılır. Bu çözüm kümesine "popülasyon" denir (Goldberg, 1989; Biegel ve Davern, 1990; Lawrence, 1990). Aday çözümler, aynı sayıda elemandan oluşan bir bit dizisi formunda oluşturulur. Bir dizi olarak gösterilen her çözüm, birey ya da "kromozom" olarak bilinir. Böylece her bir kromozom n uzunluğunda bir bit dizisidir. Kromozom üzerindeki yerlerde bulunan değişkenlere gen, genin olası değerlerine o genin allelleri adı verilir. Bir genin kromozomda bulunduğu yere lokus denir. Biyolojiden bilinen bir örnek verilirse, göz rengi bir gen tarafından belirlenmektedir. Farklı göz renkleri, genin allelleridir. Göz renginin kromozom içerisindeki yeri, genin lokusudur.

Bir optimizasyon probleminde, kromozomlardan diziler, vektörler ya da çözümler olarak söz edilir. Değişkenlere genler, genin olası değerlerine alleller ve değişkenin pozisyonuna lokus denilir. Basit örneklerde değişkenin (genin) lokusu genellikle önemsizdir, ancak daha karmaşık problemlerde önemli olmaktadır (Yeniay, 1999).

Kromozomun her biri probleme bir çözüm sunar. Her çözüme, popülasyondaki diğer çözümler ile karşılaştırıldığında ne kadar iyi bir çözüm olduğunu gösteren bir uyum değeri (fitness value) atanır. Bu değer, doğada bir organizmanın yaşayabilmek için mücadele ederken ne kadar başarılı olduğunun belirlenmesine karşılık gelir. Bir bireyin uyum değeri ne kadar büyükse, sonraki kuşakta yaşama ve üreme (çözümün seçilme) şansı o kadar fazla olur. Daha uygun bireylere, popülasyondaki diğer bireyler ile eşlenerek üreme (reproduction) fırsatı verilir. Bu ise, her atanın bazı özelliklerini taşıyan yeni bireyleri meydana getirir. Popülasyondaki düşük uyumlu bireylerin, üreme için seçilme şansları azdır. Bu nedenle yok olurlar. Mevcut kuşağın en iyi bireylerini seçip, yeni bireyler elde etmek için bunları eşleyerek, olası çözümlerden oluşan yeni bir popülasyon yaratılır. Yeni kuşak, önceki kuşağın iyi bireylerinin sahip olduğu özellikleri, kuşaklar boyu popülasyona dağılır. Daha uygun bireylerin eşleşmesi sağlanarak uzayının en umut verici bölümleri incelenir. Eğer, G.A. iyi düzenlenirse, popülasyon problemin optimal çözümüne yakınsar. Bu döngü optimizasyon kriterleri uyuşuncaya veya belirli sayıda tekrarlamalar yapılıncaya kadar, yani istenen çözüm bulununcaya kadar uygulanır (Goldberg, 1989; Biegel ve Davern, 1990; Lawrence, 1990).

2.1.2. Ana Hatlarıyla Genetik Algoritma

1. **[Başlat]** Problem için rastgele n kromozomlu populasyon oluşturulur.
   Problem için n tane çözüm önerilir.
2. **[Fitness]** Populasyondaki her bir x kromozomu için f(x) (fitness) fonksiyonu hesaplanır.
3. **[Yeni populasyon]** Yeni bir populasyon oluşuncaya kadar aşağıdaki adımlar tekrar edilir:

a) **[Seleksiyon]** Fitness durumuna göre populasyondan iki tane kromozom çaprazlanmak (crossover) amacıyla seçilir. Fitness derecesi yüksek olanın seçilme şansı yüksektir.

b) **[Çaprazlama]** Seçilmiş olan ebeveyn kromozomlar, çaprazlama oranına (crossover probability) göre yeni yavrular oluşturmak üzere çaprazlanırlar. Eğer çaprazlama uygulanmazsa bireyler atalarının tamamen kopyası olacaklardır.

c) **[Mutasyon]** Kromozom üzerindeki bazı strınglerin (DNA dizilerinin) yerleri ile oynanarak belirli mutasyon oranına göre değişiklikler yapılır.

d) **[Kabul]** Oluşturulan yeni bireyler yardımıyla yeniden bir populasyon oluşturulur.

4. **[Değiştirme]** Oluşturulan yeni populasyon eskileriyle yer değiştirilir.
5. **[Test]** Programı bitirme şartı gerçekleşiyorsa program durdurulur ve populasyondaki en iyi çözüm best çözüm olarak alınır.
6. **[Döngü]** Aksi takdirde ikinci adım tekrarlanır.

Yukarıdaki algoritmadan da anlaşılacağı üzere G.A. oldukça genel prensiplerle çalışır.

2.1.3. Genetik Algoritmaların Çalışma Prensibi

Genetik Algoritmanın çalışmasını aşağıdaki gibi özetleyebiliriz.

Çizelge 2.1 G.A.'nın İşleyişi

Adım 1	Olası çözümlerin kodlandığı bir çözüm grubu oluşturulur (çözüm grubu, biyolojideki benzerliği nedeniyle, *toplum* (population), çözümlerin kodları da kromozom olarak adlandırılır).
Adım 2	Her kromozomun ne kadar iyi olduğu bulunur.
Adım 3	Bu kromozomlar eşlenerek yeniden kopyalama ve değiştirme operatörleri uygulanır. Bu sayede yeni bir toplum oluşturulur.
Adım 4	Yeni kromozomlara yer açmak için eski kromozomlar ortadan kaldırılır.
Adım 5	Tüm kromozomların uygunlukları tekrar hesaplanır.
Adım 6	Çalışmayı sonlandırma şartı sağlanmamış ise 3. adıma gidilir.
Adım 7	O ana kadar bulunmuş en iyi kromozom, sonuçtur.

Yukarıdaki algoritmayı (adımları) 4 adımda açıklayalım:

1) Bir çözüm grubu oluşturulur. Bu grubun oluşturulması tamamen rasgele olabileceği gibi probleme göre özelleşmiş de olabilir. Tamamen rasgele bir çözüm grubu genetik algoritmaya tüm problem uzayını arama şansı verir; probleme göre özelleşme ise işlemi önemli oranda hızlandırabilir. Çözüm grubunun büyüklüğü de önemli bir faktördür. Büyük çözüm grupları çok işlem gerektirir. Küçük çözüm grupları, yerel maksimum değerlerine takılabilir. Başarılı bir çözümün küçük bir grupta baskın hale geçmesi çok daha kolaydır. Bu nedenle küçük çözüm grupları çeşitliliklerini çok çabuk kaybederler.

2) Eldeki çözüm grubunun içinden başarılı çözümler seçilir. Seçme işleminin temel mantığı, doğadaki gibi başarılı olan çözümlere üstel çoğalma imkanı vermekle açıklanabilir. Bilgisayar ortamında, bellek ve işlem zamanı sınırlı olduğu için, başarılı çözümleri çoğaltmak diğer bireyleri azaltmak anlamına gelir. Bu yolla, çözüm grubunun büyüklüğü sabit tutulabilir. Kullanılan seçme yöntemlerinin amacı, başarılı bireylere üstel çoğalma imkanı vermekle çeşitliliği korumayı en verimli şekilde dengelemektir. Sıklıkla kullanılan yöntemler arasında, rulet tekerleği seçimi, turnuva seçimi, sigma ölçeklendirmesi, Boltzman seçimi, derece seçimi, kararlı durum (steady state) seçimi sayılabilir. Yardımcı olarak kullanılan elitist seçim ise, en başarılı bireylerin bir sonraki çözüm grubuna değiştirilmeden aktarılmasıdır. Bazı uygulamalarda başarılı olduğu görülmüştür.

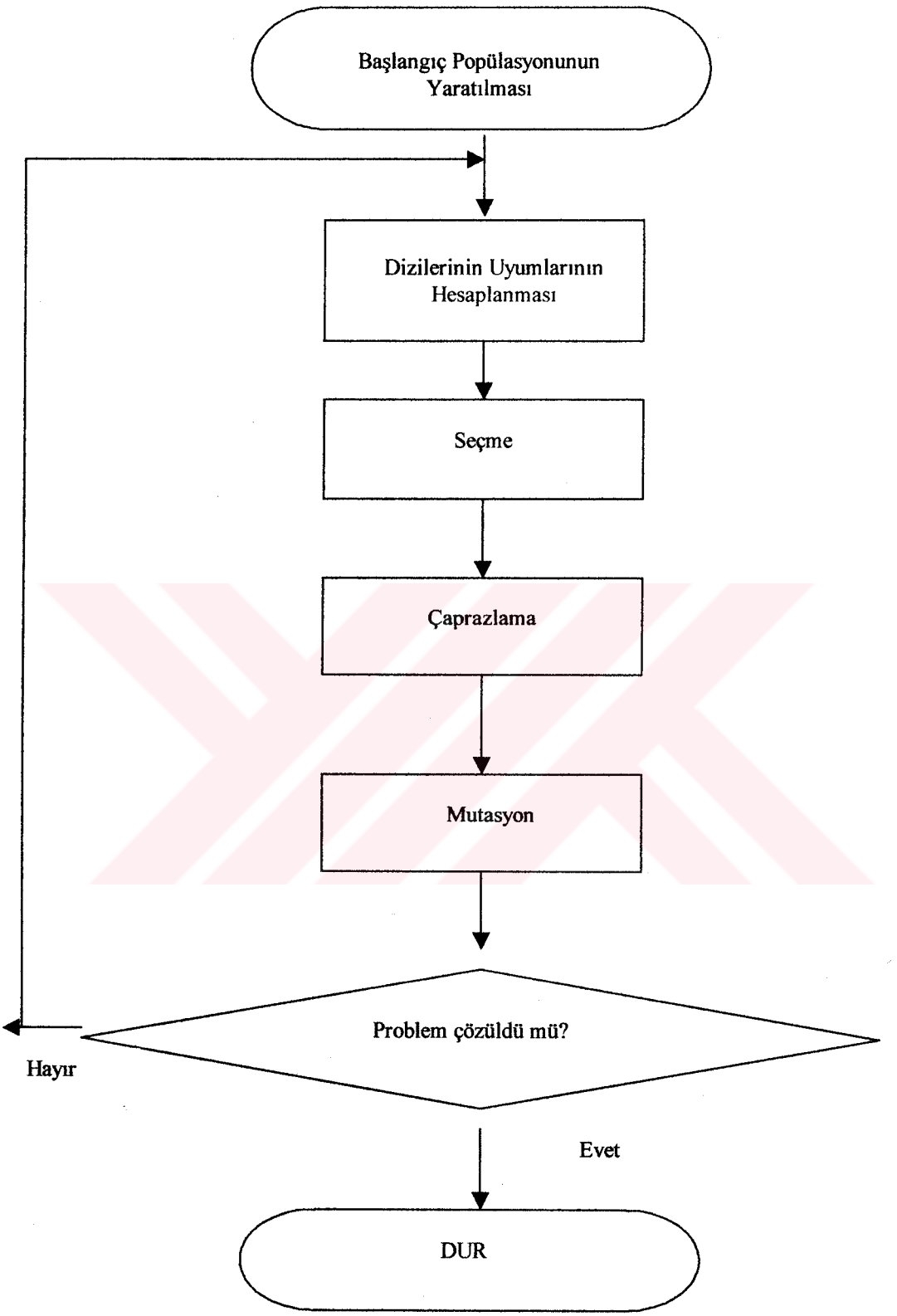
Başarılı bireyler kullanılarak yeni çözüm grubu oluşturulur. Yeni çözüm grubunu oluşturmak için kullanılan işlemciler 'crossover' ve mutasyondur. Bu işlemciler de yapılan işleme göre değişse de genel yöntemleri pek farklı değildir.

3) Çaprazlama (crossover) iki çözümün yapıtaşları kullanılarak yeni bir çözüm oluşturulması esasına dayanır. Bu işlem doğada görülen 'crossingover' olayının analoğudur. 'crossover' işlemi genel olarak ikili dizilerin parçalarının değiş tokuşu şeklinde gerçekleştirilir. Farklı uygulamalarda, farklı kodlama yöntemleri kullanıldığı için farklı 'crossover' yöntemleri kullanılır.

Mutasyon bir çözümün, çoğunlukla rasgele olarak, değiştirilmesidir. Bu işlem çok değişik şekillerde kullanılır. Problemin yapısı bu aşamada çok önemlidir. Örneğin sıralama problemlerinde sıralamayı değiştirmek, ikilik dizi gösteriminde bitleri değiştirmek, çözüm ağaçlarında parçaları değiştirmek işlem olarak tanımlanır.

4) 2. ve 3. işlemler belirlenen bir şart (optimizasyon kriterleri uyuşuncaya kadar) sağlanana kadar tekrarlanır. Bu şart tekrar sayısının belirlenen bir sayıya ulaşması, belirlenen bir başarım değerine ulaşılması, Genetik Algoritmanın daha başarılı çözümler oluşturamaması vb. olabilir (Goldberg, 1989; Biegel ve Davern, 1990; Lawrence, 1990).

2.2. İŞLEYİŞİ



Şekil 2.1 Genetik Algoritmanın Akış Diyagramı (Yeniay,1999)

2.2.1. Başlangıç Popülasyonunun (Yığınının) Oluşturulması

G.A.'nın uygulamasında ilk adım, kromozomların bir başlangıç popülasyonunun oluşturulmasıdır. Başlangıç popülasyonu, çoğunlukla rasgele olarak oluşturulur. Ancak, özellikle kısıtlı optimizasyon problemlerinde, rasgelelik, uygun olmayan (infeasible) çözümlere neden olabilir. Bu durumdan kaçınmak için, genellikle incelenen probleme özgü sezgisel yöntemlerden yararlanılır. Örneğin, Grefenstette 1987'de yaptığı çalışmada, gezgin satıcı problemi için greedy sezgisellerinden; Kapsalis ve arkadaşları 1993'de yaptığı çalışmada, Steiner ağaç problemi için minimum ağaç yaklaşımından; Thiel ve Vass 1994'de yaptığı çalışmada, 0-1 sırt çantası problemi için ekleme çıkarma sezgiselinden; Chen ve arkadaşları ise 1995'de çizelgeleme problemi için Campbell-Dudek-Smith ve Dannenbrig sezgisellerinden yararlanarak başlangıç popülasyonunu oluşturmuşlardır.

G.A., bir popülasyonu değerlendirir ve uygulanan bir süreç sonunda yeni bir popülasyon elde eder. Birbirini izleyen her popülasyona, kuşak denir.

G.A.'nın herhangi bir uygulamasında, popülasyon genişliğinin (N) belirlenmesi gerekir.Büyük popülasyonlarda, çözüm uzayı iyi örneklendiği için aramanın etkinliği artar, ancak makul bir sürede yüksek kalitede çözüme yeterince ulaşamadan ağır bir hesaplama külfetiyle karşılaşılabilir. Küçük popülasyonlar ise çözüm uzayını yeterli bir biçimde örnekleyememe riskini taşır ve zamansız yakınsama (premature convergence) görülebilir. Popülasyon genişliği için, 50-100 arasında kalan değerler yaygın olarak kullanılmaktadır. Ancak 30 diziye sahip popülasyonlar ile tatmin edici sonuçlara ulaşan birçok uygulama da vardır (Yeniay, 1999).

2.2.2. Uygunluk (Uyum-Değerlendirme) Fonksiyonu (Fitness Function)

Uygunluk değeri, yeni yığına taşınacak dizilerin belirlenmesinde kullanılan bir araçtır. Bu nedenle, algoritmanın her çevriminde, yığındaki dizilerin bir değerlendirme fonksiyonu yardımıyla uygunluk değerleri hesaplanır. G.A.'da kullanılan değerlendirme fonksiyonu problemin amaç fonksiyonudur (Dengiz ve Altıparmak, 1998).

Kromozomların ne kadar iyi olduğunu bulan fonksiyona uygunluk fonksiyonu denir. Bu fonksiyon işletilerek kromozomların uygunluklarının bulunmasına ise *hesaplama* (evaluation) adı verilir. Bu fonksiyon, Genetik Algoritma'nın beynini oluşturmaktadır. Genetik Algoritmada probleme özel çalışan tek kısım bu fonksiyondur. Uygunluk fonksiyonu, kromozomları problemin parametreleri haline getirerek onların bir bakıma şifresini *çözmekte* (decoding), sonra bu parametrelere göre hesaplama yaparak kromozomların uygunluğunu

bulmaktadır. Çoğu zaman Genetik Algoritma'nın başarısı, bu fonksiyonun verimli ve hassas olmasına bağlı olmaktadır.

G.A. ile çözülecek her problem için uyum fonksiyonu (fitness function) gerekir. Belirli bir kromozom verildiğinde, uyum fonksiyonu, o kromozom ile temsil edilen bireyin yararı (utility) ya da gücüyle (ability) orantılı olan bir sayısal değer verir. Uyum fonksiyonu, çevre rolünü oynamaktadır. G.A., bir kromozomun uyum fonksiyonunu, o kromozomun çevreye uyum derecesini incelemede ve istenmeyen kromozomların yaşama ve üreme olasılığını azaltmada kullanır. Böylece algoritma, başarısız olan aday çözümlerin elenmesini sağlar. Bu nedenle, uyum fonksiyonunun seçimi, algoritmanın etkinliği üzerinde büyük öneme sahiptir. Çoğu problemde uyum fonksiyonuna karar vermek kolaydır (Yeniay, 1999).

2.2.3. Kodlama (Şifreleme) (Encoding)

G.A.'nın başarılı bir biçimde uygulanabilmesi için yapılması gereken ilk iş, bağımsız parametrelerin kromozomlar içerisinde kodlanmasıdır. Bu kromozom, belirli bir çözüm hakkındaki tüm bilgileri taşır. Kodlama mekanizması, problemin değişkenlerinin yapısına bağlıdır ve yöntemin performansında önem taşır.

Kodlama (Şifreleme) problemin tipine bağlı olarak çözüm adaylarını temsil eden yapılardır. Kodlamanın çeşitli yolları vardır:

1. İkili Kodlama (Binary Encoding)
2. Permütasyonlu Kodlama (Permutation Encoding)

2.2.3.1. İkili Kodlama (Binary Encoding):

En yaygın kullanılan kodlama yöntemidir. Sayıların ikilik sistemde gösterimine dayanır. Bu yöntemde kromozomlar sıfır ve birlerden oluşan dizilerdir. Her parametre için n-bit kullanılır. n, her parametre için farklı olabilir.

İkili düzende kodlama, çok sık kullanılan bir kodlama tipi olmasına rağmen bazı sakıncaları vardır. Örneğin, çok değişkenli fonksiyon eniyilemesi için değişkenlerin alt ve üst sınırlarına bağlı olarak elde edilen dizi çok uzun olabilir. Aynı zamanda gezgin satıcı, çizelgeleme, karesel atama gibi kombinatoryal eniyileme problemlerinde ikili düzende kodlama, araştırma uzayını tam olarak temsil edememektedir. Bu nedenle literatürde, permütasyonlu kodlama (sayısal kodlama) daha sık kullanılmaktadır.

2.2.3.2. Permütasyonlu Kodlama (Permutation Encoding):

Sıralama problemlerinde permütasyonlu kodlama kullanılır. Bu kodlamada, her kromozom, dizideki bir sırayı temsil eden sayılardan oluşur.

Örneğin, gezici satıcı probleminde (travelling salesman problem) "verilen şehirler arasındaki uzaklıklar bilinirken, bütün şehirleri birleştiren minimum yol uzunluğu ne olmalıdır?" sorusuna cevap aranmaktadır. Burada kromozomlar, şehirlerin hangi sırada ziyaret edilebileceğini gösteren permütasyonlar şeklinde kurulurlar (Goldstein, 1991; Valenzuela, 1995).

2.2.4. Seçme

Kodlamaya karar verilip başlangıç popülasyonu oluşturulduktan sonra, yeni popülasyona geçebilmek için seçimin nasıl yapılacağı belirlenmelidir. Seçme, bireylerin üreme için seçildiği bir işlemdir. İlke olarak, yüksek uyuma sahip bireyler daha büyük bir olasılıkla seçilmelidir. Seçim operatörünün, yüksek uyuma sahip bireylerin sonraki kuşağa kopyalanma olasılığını artırması istenir. Böylece seçim, arama uzayının başarılı olacağı umulan bölgelerinde yoğunlaşır. Sonraki kuşak için, yavru meydana getirecek bireylerin popülasyondan nasıl seçileceği ve her bir bireyin kaçar tane yavruya sahip olacağı belirlenmelidir. Sonraki kuşak için yavru meydana getirecek bireyler ile üremehavuzu (mating pool) adı verilen bir ara popülasyon oluşturulur. Daha sonra, üreme için bu havuzdan rasgele bireyler seçilir. Seçimde iki önemli konu vardır: Popülasyon çeşitliliği (population diversity) ve seçicilik baskısı (selective pressure). Bu faktörler birbirleriyle ilişkilidir. Seçicilik baskısındaki bir artış, popülasyonun çeşitliliğini azaltır. Bunun aksi de doğrudur. Yani güçlü seçicilik baskısı, G.A.'nın zamansız yakınsamasını destekler, zayıf seçici baskı ise aramanın etkinliğini azaltabilir. Bu nedenle bu iki faktör arasında bir denge kurulmalıdır. Örnekleme mekanizmaları bu hedefi gerçekleştirmeye çalışır.

G.A. literatüründe çeşitli seçim planları önerilmiştir. Holland'ın orjinal G.A.'sında popülasyon seçimi, eniyi uyum yapanın yaşaması ilkesine dayanır. G.A.'da uygun bir dizi çok sayıda yavru elde eder ve bunun sonucunda, sonraki kuşakta daha yüksek bir yaşama şansına sahip olur. Mevcut popülasyondan bireyleri uygunluk değerlerine göre seçen yöntemler, uyuma orantılı seçim (fitness proportionate selection) yöntemleri grubu olarak adlandırılırlar (Yeniay, 1999).

Goldberg ve Deb (1991) literatürde mevcut ve çok sık kullanılan seçim mekanizmalarını Çizelge 2.2'de görüldüğü gibi, orantılı yeniden üretim (proportionate reproduction) mekanizması, sıralı (ranking), turnuva (tournament), denge durumu (steady state, diğer adıyla Genitor) yeniden üretim mekanizmaları olmak üzere 4 ana sınıfta toplamışlardır.

Çizelge 2.2 (Yeniden Üretim Mekanizmaları)

Mekanizmalar	Açıklama
Orantılı Yeniden Üretim Mekanizması	Yığındaki her bireyin seçilme olasılıkları belirlenir ve bu olasılıklar kullanılarak bir sonraki yığın oluşturulur. Bu grupta bulunan yeniden üretim mekanizmaları: rulet çemberi, stokastik artan ve stokastik üniversal metodudur.
Sıralı Yeniden Üretim Mekanizması	Yığındaki bireyler, uygunluk değerlerine göre küçükten büyüğe doğru sıralanır. En iyiden başlayarak bir azalan fonksiyon yardımı ile dizilere kopya sayısı atanır ve orantılı seçim mekanizmalarından birisi kullanılarak yeni yığın elde edilir.
Turnuva Yeniden Üretim Mekanizması	Yığından rassal olarak bir grup dizi (yerine koyarak/yerine koymadan) seçilir ve grup içindeki en iyi uygunluk değerine sahip dizi, yeni yığına kopyalanır. Bu işleme, yığın genişliğine ulaşıncaya kadar devam edilir. Grup genişliği en az ikidir.
Denge Durum Yeniden Üretim Mekanizması	Doğrusal sıralı seçim mekanizması kullanılarak seçilen bir ya da iki bireye genetik operatörler uygulanır. Elde edilen yeni diziler, mevcut yığındaki uygunluk değeri en küçük diziler ile yer değiştirerek yeni yığın oluştururlar.

Literatürde, orantılı, sıralı ve turnuva yeniden üretim mekanizmaları ile birlikte elitist stratejisinin de çok sık kullanıldığı görülmektedir. Elitist stratejisi ile mevcut yığındaki eniyi bir ya da birkaç dizi bir sonraki yığına taşınır. Amaç, elde edilen eniyi uygunluk değerine sahip dizinin (veya dizilerin) örnekleme hatası ya da genetik operatörler kullanımı sonucunda kaybolmasını önlemektir (Dengiz ve Altıparmak, 1998).

2.2.4.1. Orantılı Yeniden Üretim Mekanizması

1°) Yerine Koyarak Stokastik Örnekleme (Rulet Çemberi Seçimi)

Yerine koyarak stokastik örnekleme, rulet çemberi (roulette whell) seçimi olarak da adlandırılmaktadır. Bu metod ilk defa Holland tarafından ortaya çıkarıldı ve muhtemelen Genetik Algoritmada kullanılan en eski ihtimale dayanan seçim metodudur. Muhtemel seçim metodları hayatta kalma ihtimali üzerine kurulan herbir kromozom kopyasının gerçek sayısını belirler. Böylece seçim bölümü herbir kromozomun beklenen değerinin kararlaştırılması ve bu değerin döl miktarına dönüştürülmesinden oluşur. Ana düşünce, herbir kromozomun seçilme ihtimalinin uygunluk değeriyle orantılı olduğudur. f_k uygunluk (fitness) değerindeki herbir kromozomun seçilme olasılığı (ihtimali) P_k dır (Aksoy, 1998). Öncelikle $P_k = \frac{f_k}{\sum_{k=1}^{N} f_k}$ eşitliği yardımıyla her bir bireyin P_k seçilme olasılığı hesaplanır. Bu sayılar bir tabloda tutulur. Tablodaki olasılık değerleri birbirine eklenerek rasgele bir sayıya kadar ilerlenir. Bu rasgele sayıya ulaşıldığında ya da geçildiğinde son eklenen sayının ait olduğu birey seçilmiş olur. Bu işlem N kez tekrar edilir. Bu yönteme rulet çemberi seçimi ismi, bir daireyi, bireylerin uyum değerlerine orantılı olarak dilimleyip çevirdiğimizde olacakların benzeşimi olduğu için verilmiştir.

Rulet tekerleği seçimini adım adım aşağıdaki gibi yazabiliriz:

1- Tüm bireylerin uygunluk değerleri bir tabloya yazılır.
2- Bu değerler toplanır.
3- Tüm bireylerin uygunluk değerleri toplama bölünerek [0,1] aralığında sayılar elde edilir. Bu sayılar bireylerin seçilme olasılıklarıdır. Sayıların hepsi bir tabloda tutulur.
4- Seçilme olasılıklarını tuttuğumuz tablodaki sayılar birbirine eklenerek rastgele bir sayıya kadar ilerlenir. Bu sayıya ulaşıldığında yada geçildiğinde son eklenen sayının ait olduğu çözüm seçilmiş olur.

Rulet çemberinde stokastik hatalar nedeniyle bir bireye pay edilen yavru sayısı, beklenen yavru sayısından önemli ölçüde farklı olabilir. Bu nedenle rulet çemberi seçimi büyük örnekleme hataları doğurabilmektedir. İterasyon sayısı arttıkça, örneklemehatası da artmaktadır. Pay edilen yavru sayısı, sadece çok büyük popülasyonlar için beklenen yavru sayısına yaklaşmaktadır. Rulet çemberi seçimindeki bu örnekleme hatası nedeniyle, arama

tahmin edilenden farklı yönlerde devam etmektedir. Bu ise algoritmanın muhtemelen yerel optimuma zamansız yakınsamasına neden olmaktadır. Bu sorunu ortadan kaldırmak için çeşitli seçim yöntemleri önerilmiştir.

2°) Yerine Koymadan Stokastik Örnekleme

Yerine koymadan stokastik örnekleme, De Jong'un beklenen değer modelinin öteki adıdır. Bu yöntem, rulet çemberi seçiminin stokastik hatalarını azaltmaktadır. f_k, k. bireyin uyum değerini; $\sum_{k=1}^{N} f_k$, popülasyonun toplam uyumunu ve $\overline{f} = \frac{\sum_{k=1}^{N} f_k}{N}$, popülasyonun ortalama uyum değerini göstermektedir. Her birey için $\frac{f_k}{\overline{f}}$ beklenen yavru sayısı hesaplanır. Bu sayı, birey çaprazlamaya her seçildiğinde 0.5, çaprazlama olmadan üremeye seçildiğinde ise 1 azaltılır.

3°) Kalanı Stokastik Örnekleme

Kalanı stokastik örnekleme yöntemi, yerine koyarak ve koymadan uygulanabilir. Yöntem, beklenen birey sayılarını bilinen biçimde hesaplar ve tamsayılı kısmını bireye atar. Popülasyon genişliğine ulaşılmadıysa, beklenen değerlerin kesirli kısımlarından yararlanılır. Yerine koyarak kalanı stokastik örneklemede, beklenen değerlerin kesirli kısımları, rulet çemberi seçim yönteminde bireylere pay edilen ağırlıkları hesaplamak için kullanılır. Yerine koymadan kalanı stokastik örneklemede, beklenen değerlerin kesirli kısımları kullanılarak birer birer ağırlıklı para atışları (Bernoulli denemeleri) yapılır. Örneğin, beklenen kopya sayısı 1.36 olan bir birey, bir kopyayı kesin olarak, diğer kopyayı ise 0.36 olasılıkla elde eder. Popülasyon tamamlanana dek bu süreç devam eder.

4°) Stokastik Evrensel Örnekleme

James Baker, bireye atanan gerçek yavru sayısı ile beklenen yavru sayısı arasındaki farkı minimize eden stokastik evrensel örnekleme adında bir yöntem önermiştir. Temel düşünce tek bir seçimle tüm N bireyi örneklemektir. Bu yöntemi gerçekleştirmek için, rulet çemberine seçim çemberi (selection whell) denilen bir ek parça eklenir. Bu parça eşit aralıklı N göstergeye sahiptir. Çember bir kez çevrilir ve durduğunda göstergelerin bulunduğu yerler bireyleri gösterir. Böylece N birey, bir adımda seçilir. Göstergeler eşit aralıklı olduğu için, bir bireyin küçük popülasyonlarda bile tüm popülasyona hakim olma tehlikesi yoktur. Bu

yöntemle, her bireyin beklenen yavru sayısı kadar (daha çok değil) üremesi garanti edilir. Bu algoritma ile, seçim planının belirli bir işleyişinin sonucu beklenen davranışa mümkün olduğunca yakındır, yani ortalama değişim minimumdur.

2.2.4.2. Sıralı Yeniden Üretim Mekanizması (Rank Selection)

Sıralama seçimi, ilk olarak Baker tarafından uyuma orantılı seçimin dezavantajlarını yok etmek üzere önerilmiştir. Sıralama seçiminde bireyler (kromozomlar) uyum değerlerine göre (fitness durumlarına göre) sıralanır. Her bireyin beklenen değeri, sahip olduğu sıraya bağlıdır.

Bu sıra dikkate alınarak seçme yapılır. Bu seçme işleminde tüm kromozomların seçilme ihtimali mevcuttur. Fakat bu yöntem çok farklı olmayan bireylerin seçilmesine daha çok imkan tanıyacağından istenen tarama bölgesini çabuk taramayabilir.

2.2.4.3. Turnuva (Tournament) Seçim Yöntemi

Turnuva seçim yöntemi, seçim baskısı yönünden sıralama seçimlerine benzer, ancak işlemsel olarak daha verimli olup, paralel uygulamaya daha yatkındır. Turnuva seçiminde, popülasyondan yerine koyarak ya da yerine koymadan rasgele t birey seçilir. t büyüklüğüne turnuva genişliği adı verilir. Bu gruptaki en iyi birey, yeni popülasyona kopyalanır. Bu işlem N kez yinelenir. Büyük t değeri, yöntemin seçicilik baskısını artırır. Turnuvalarda çoğunlukla t=2 alınır ve ikili turnuva olarak adlandırılır. Bu yöntemde zamansız yakınsama, durağanlaşma ve açık uyuma gereksinim yoktur.

Stokastik turnuva seçimi, Wetzel tarafından önerilmiştir. Bu yöntemde seçim olasılıkları hesaplanır ve rulet çemberi yardımıyla ard arda birey çiftleri belirlenir. Bir çift belirlendikten sonra, yüksek uyuma sahip olanı alınarak yeni popülasyona dahil edilir ve diğer çift seçilir. Süreç, popülasyon tamamlanana dek sürer.

2.2.4.4. Denge Durum Yeniden Üretim Mekanizması (Steady State) (Denge Durumu Seçim Yöntemi)

G.A.'nın her iterasyonunda mevcut popülasyondan bütünüyle yeni bir popülasyon yaratılmaktadır. Bütün popülasyonu değiştiren G.A.'lara kuşaksal (generational) G.A.'lar denir. Bazı seçim planlarında (elitist plan gibi) birbirlerini izleyen kuşaklar bir derece örtüşmekte, yani önceki kuşağın bir bölümü yeni popülasyonda korunmaktadır. Her kuşakta popülasyonun değiştirilen oranına kuşak aralığı (generation gap) denir ve G ile gösterilir.

Böylece t. kuşağın N(1-G) adet bireyi, (t+1). kuşakta aynen yaşamaya devam eder. Kuşaksal Genetik Algoritmalarda G=1'dir.

Denge durumu (steady state) seçiminde, her kuşakta sadece birkaç birey değiştirilir. Bu seçimde, ata olarak iki bireyin nasıl seçileceğine ve gelecek yavrulara yer açmak için hangi iki bireyin silineceğine karar verilir. Örneğin; Whitley'in GENİTOR algoritması ataları sıralanan uyum skorlarına göre seçmekte ve yavrular en kötü iki bireyin yerini almaktadır.

Elitist Seçim Yöntemi

İlk olarak Kenneth De Jong tarafından önerilmiştir. Bu yöntem, popülasyonun en iyi bir (ya da iki) bireyini koruyup, popülasyonun geri kalan elemanlarını uyuma orantılı seçim yöntemlerinden birini kullanarak yeni bireyler ile değiştirir. Böylece en iyi bireylerin yaşaması sağlanır. Yani her nesilde en iyi olan en az bir çözüm, olduğu gibi bir sonraki nesile aktarılır. Yöntemin avantajı, en iyi uyum değerine sahip bireyin, örnekleme hatası ya da genetik operatörlerin kullanılması ile kaybolmasını önlemektir.

Ölçeklendirme Fonksiyonları

Orantılı seçme planı, bireyin uyum değerinin popülasyonun ortalama uyum değerine oranına bağlı kalarak yavruları paylaştırmaktadır. G.A.'nın başlangıç kuşaklarında popülasyon genellikle düşük ortalama uyum değerine sahiptir. Yüksek uyum değerine sahip birkaç bireyin olması orantılı seçim planının bu süper bireylere çok sayıda yavru pay etmesine neden olur ve bu bireyler popülasyona hakim olur. Bu ise zamansız yakınsamaya yol açar.

G.A.'nın sonraki aşamalarında ise farklı bir problem ortaya çıkar. Popülasyonda hala önemli çeşitlilik olabilir, ancak popülasyonun ortalama uyumu, popülasyonun en iyi uyumuna yaklaşabilir. Eğer bu durum kendi haline bırakılırsa, ortalama elemanlar ile en iyi elemanlar sonraki kuşaklarda hemen hemen aynı sayıda birey elde ederler. Her iki durumda da uyum ölçekleme yardımcı olabilir.

Yararlı bir ölçekleme prosedürü, doğrusal ölçeklemedir. Burada ham uyum f, ölçeklendirilmiş uyumu f' ile tanımlanırsa,

$$f' = af + b \tag{2.1}$$

dönüşümü kullanılır. Her kuşakta a ve b sabitlerinin, f'_{ort} ortalama ölçeklenmiş uyumu, f_{ort} ortalama ham uyumuna eşit olacak biçimde seçimi yapılır. Böylece seçim prosedürünün

kullanımı, her bir ortalama popülasyon elemanının sonraki kuşağa bir yavru verebilmesini garanti eder. Maksimum ham uyuma sahip popülasyon elemanına verilen yavru sayısını denetleyebilmek için, ölçeklendirilmiş maksimum uyumu veren

$$f'_{\max} = c \cdot f_{ort} \tag{2.2}$$

bağıntısı kullanılır. Burada c, en iyi popülasyon elemanı için istenilen kopya sayısıdır. Küçük popülasyonlarda c=1.2 ile 2 başarıyla kullanılmıştır.

Bazen ölçeklenmiş uyum değerleri negatif bulunabilir. Bu durum popülasyonun çoğu elemanı yüksek uyuma, bazıları ise çok düşük uyuma sahip ise, sıklıkla karşılaşılan bir problemdir. Bu sorunu önlemek için Forrest, popülasyon değişim bilgisini kullanmayı önermiştir. Popülasyonun standart sapma bilgisini kullandığı için, bu sürece, sigma kesim planı (sigma truncation scheme) adı verilir. Sigma kesim planı,

$$f' = f - (\bar{f} - c\sigma) \tag{2.3}$$

eşitliğini kullanır. Burada sigma, popülasyondaki uyum değerlerinin standart sapmasıdır. c sabiti, popülasyon standart sapmasının makul bir katı olarak (genellikle 1 ile 3 arasında) seçilir. Uyum değerleri $(f' - c\sigma)$'dan daha küçük olan bireyler göz ardı edilir.

Gillies, üs kuralı ölçeklemeyi (power law scaling) önermiştir. Burada ölçeklenen uyum, ham uyumun belirli bir üssü olarak alınır:

$$f' = f^k \tag{2.4}$$

k, bire yakın bir sayıdır. k parametresi, f fonksiyonunu ölçeklendirir. Bazı çalışmalarda k'nın seçiminin probleme bağlı olması gerektiği sonucuna varılmıştır.

2.2.5. Genetik Operatörler

Seçim prosedürleri bizi herhangi bir yeni çözüme götürmez. Seçilen bireyler, üreme havuzuna alınırlar ve üreme havuzu popülasyonun büyüklüğüne ulaştığı an, seçme işlemi son bulur. Seçilen bireyler yeni kuşağın atalarıdır. Gelişmiş yeni çözümler, bu aşamada genetik operatörler yardımıyla elde edilirler. Yeni çözümler elde etmede görev alan iki genetik operatör vardır. Bunlar çaprazlama ve mutasyondur.

2.2.5.1. Çaprazlama

G.A.'ya dayalı çoğu uygulamada en önemli operatör, çaprazlamadır. Çaprazlama, farklı çözümler arasında bilgi değişimini sağlayarak arama uzayının benzer, ancak araştırılmamış bölgelerine ulaşmayı sağlayan bir arama operatörüdür. Çaprazlamanın bireylerdeki iyi özellikleri birleştirerek daha iyi çözümler yaratması beklenir. Literatürde en çok kullanılan çaprazlama operatörleri, Tek Nokta Çaprazlama (single point crossover), İki Nokta Çaprazlama (two point crossover), Çok Nokta Çaprazlama (multi point crossover) ve Üniform Çaprazlamadır (uniform crossover).

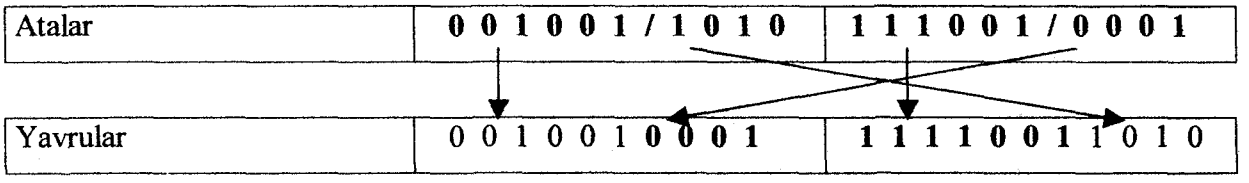
1°) Tek Nokta Çaprazlama

G.A.'nın kullanıldığı en basit çaprazlama türüdür. Öncelikle tüm popülasyon rasgele eşlenerek, olası ataları içeren N/2 küme elde edilir. P_c çaprazlama olasılığı yardımıyla çaprazlama uygulanacak çözüm çiftleri belirlenir. Çoğunlukla 0.6 ile 1.0 arasında bir P_c olasılığı kullanılmaktadır. Eğer [0,1] aralığından rasgele seçilen bir sayı, P_c den küçük ise o çifte çaprazlama uygulanır, aksi durumda bu çözümler popülasyonda değişmeden kalır. Her yeni popülasyonda yaklaşık olarak P_cN tane bireye çaprazlama uygulanır.

Çaprazlama için,

0010011010 ve 1110010001

dizilerinin çaprazlamaya seçildiğini düşünelim. Öncelikle 1 ile n-1 arasında bir tamsayı (çaprazlama noktası) rasgele seçilir. Bu nokta 6 olsun. Bu nokta, Şekil 2.2 de "/" ile gösterilmiştir. Daha sonra, çaprazlama noktasından sonraki iki alt dizi atalar arasında yer değiştirilerek, yavru adı verilen iki yeni çözüm elde edilir (Şekil 2.2).



Şekil 2.2 Tek Nokta Çaprazlama

Birinci ve ikinci yavru, her iki atasından farklı olma eğilimindedir, ancak iki atanın bazı özellikleri korunur. Eğer her iki ata da yüksek uyuma sahipse, yavrulardan enaz birinin atalardan daha iyi olma şansı yüksektir. Böyle bir durum olursa, seçim, bu yavruların üremelerini; aksi halde, yok olmalarını destekleyecektir. Çaprazlama ile tümü ya da çoğu

düşük uyuma sahip yavrular da yaratılabilir. Bu olasılığı da ortadan kaldırmak için, P_c çaprazlama olasılığından yararlanılır. Çaprazlama olmadan, hiçbir ilerleme belirtisi olmayacağından P_c genellikle yüksektir.

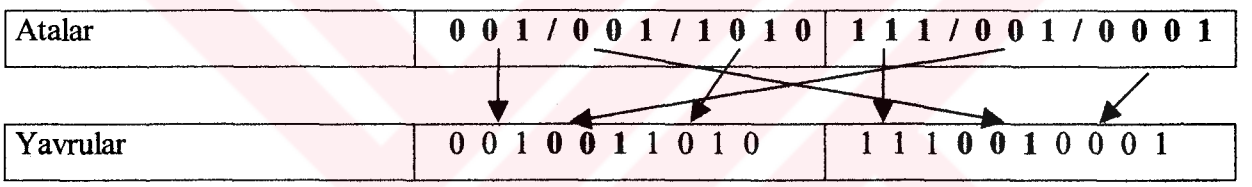
2°) İki Nokta Çaprazlama

İki nokta çaprazlamada, iki çaprazlama noktası vardır ve bu iki nokta arasında kalan alt dizilerin değiştirilmesiyle iki yeni yavru elde edilir.

Şekil 2.3'de

0010011010 ve 1110010001

dizilerine iki nokta çaprazlamanın uygulanışı gösterilmektedir. Burada da çaprazlama noktaları, 1 ile n-1 arasından rasgele seçilmektedir.



Şekil 2.3 İki Nokta Çaprazlama

3°) Çok Nokta Çaprazlama

Çok nokta çaprazlama, iki nokta çaprazlamanın bir uzantısıdır. Herbir çözüm, k çaprazlama noktası ile k parçaya ayrılır. Bir atlanarak elde edilen allel blokları, çiftler arasında değiştirilerek yavrular elde edilir.

De Jong, çok nokta çaprazlama operatörlerini test edip, daha fazla alt dizi değiştirdiğinden performansın gerilediği sonucuna varmıştır. İlerlemeyi sürdürmek için bazı değişiklikler kazandırmak önemlidir. Ancak, çok fazla değişim yapmak, bir çözümün iyi özelliklerine zarar verme olasılığını yükseltmektedir. Bazı araştırmacılar ise, çok nokta çaprazlamanın dizilerdeki bazı iyi özellikleri birleştirmeye daha uygun olduğunu düşünmektedir. Bu nedenle, mevcut çözümlerdeki iyi özelliklerin işletilmesi ve yeni özelliklerin kazanılması arasındaki denge, etkin bir G.A. araması için çok önemlidir.

4°) Üniform Çaprazlama

Üniform çaprazlamada iki ata verildiğinde, birinci yavrunun her geni, rasgele olarak yaratılan bir çaprazlama maskesi (crossover mask) uyarınca, birinci veya ikinci atanın karşılık gelen geninin kopyalanmasıyla yaratılır. Çaprazlama maskesindeki 1, o genin birinci atadan kopyalanacağını, 0 ise o genin ikinci atadan kopyalanacağını ifade eder.

Rasgele yaratılan

1001011100

çaprazlama maskesi uyarınca,

1010001110 ve 0101010011

atalarına üniform çaprazlamanın uygulanışı, Şekil 2.4'de gösterilmiştir.

Çaprazlama Maskesi	1 0 0 1 0 1 1 1 0 0
1.Ata	1 0 1 0 0 0 1 1 1 0
1.Yavru	**1 1 0 0 0** 0 1 **1 1 1**
2.Ata	**0 1 0 1 0 1 0 0** 1 1

Şekil 2.4 Üniform Çaprazlama

Birçok gerçek uygulama için, probleme özel çözüm gösterimleri ve çaprazlama operatörleri geliştirilmiştir. Bu esneklik G.A.'ların çekiciliklerinden biridir.

Çaprazlama ile oluşan yavrular popülasyonda olmayan bilgileri alamazlar. Örneğin, mevcut popülasyonun tüm elemanları dizinin 1 pozisyonunda 1 içeriyorsa, çaprazlama bu pozisyonda 0 olan bir bireyi hiçbir zaman yaratamaz.

Son yıllarda G.A. literatürü çeşitli teknikleri karşılaştırmıştır. Özellikle tek nokta ve iki nokta çaprazlama ile tek nokta ve üniform çaprazlama karşılaştırılmaktadır. Teknikleri sınıflandırmak amacıyla pozisyon ve dağılım eğilimi (positional and distributional bases) kavramları kullanılmaktadır. Eğer bir bitin değiştirilme olasılığı, dizideki pozisyonuna bağlıysa, çaprazlama operatörü pozisyon eğilimine sahiptir. Dağılım eğilimi, çaprazlama

operatörüyle değiştirilen bitlerin sayısı ile ilişkilidir. Bu sayının dağılımı üniform değilse, çaprazlama operatörü dağılım eğilimine sahiptir.

Tek nokta çaprazlama en fazla pozisyon eğilimi, en az dağılım eğilimi göstermektedir. Çünkü çaprazlama noktası, üniform dağılım kullanılarak rasgele seçilmektedir. Ancak eğilimin bu eksikliği, iyi birşey değildir. Çünkü atalar arasında bilgi değişimini sınırlamaktadır.

Üniform çaprazlama, bitlerin pozisyonlarını göstermeksizin değiştirmektedir, ancak yüksek bozucu yapısı sık sık sorun olmaktadır. Tek nokta ve iki nokta çaprazlama, popülasyon homojen olduğunda, aramaya daha az yardımcı olmaktadır.

Diğer bir konu da popülasyon büyüklüğü ile çaprazlama türü arasındaki etkileşimdir. Deneysel sonuçlar, üniform çaprazlamanın küçük popülasyonlar için, iki nokta çaprazlamanın büyük popülasyonlar için daha iyi olduğunu göstermektedir. Üniform çaprazlamanın bozuculuğu, küçük popülasyonlarda oldukça keşifsel bir arama yapmaya yardımcı olmaktadır. Büyük popülasyonlarda var olan çeşitlilik keşif gereksinimi azaltmakta ve iki nokta çaprazlamayı daha uygun kılmaktadır (Yeniay, 1999).

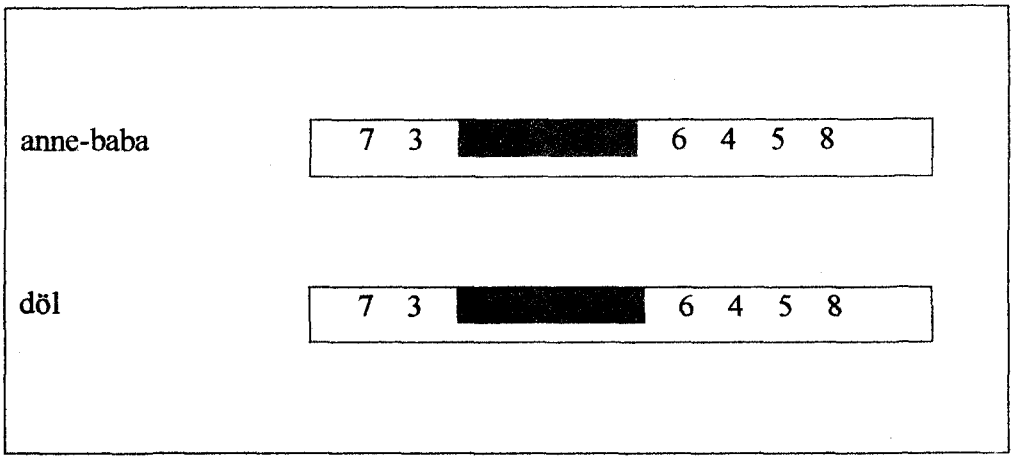
2.2.5.2. Mutasyon

G.A.'larda diğer önemli operatör, mutasyondur. Mutasyon doğada olduğu gibi G.A. uygulamalarında da geri planda kalan bir operatördür (Yeniay, 1999). G.A.'ın çalışmasında ikinci dereceden rol oynar. G.A.'da mutasyon operatörü, küçük bir olasılıkla bir dizi içindeki bir veya birkaç değeri rassal olarak değiştirerek, yığında yeni dizilerin (yani arama uzayında yeni çözüm noktalarının) elde edilmesini sağlar (Goldberg, 1989). Her yığına mutasyon operatörü, P_m olasılığı ile uygulanır. İkili düzende kodlamanın kullanıldığı bir dizide, mutasyon operatörü ile rassal olarak seçilen eleman değeri 1 ise 0, 0 ise 1 olarak değiştirilerek yeni bir dizi elde edilir.

Özellikle G.A.'ın son çevrimlerinde mutasyonun etkinliği artmaktadır. Çünkü son çevrimlerde yığın iyi çözümlere yakınsadığından diziler birbirine çok benzemektedir. Bu durum, çaprazlama operatörünün farklı yeni diziler oluşturmasını engelleyerek aramayı kısıtlar. Bu aşamada mutasyon, yığındaki değişkenliği gerçekleştirerek arama uzayında yeni çözüm noktalarının elde edilmesini sağlamaktadır (Dengiz ve Altıparmak, 1998).

1°) **Ters Çevirme**

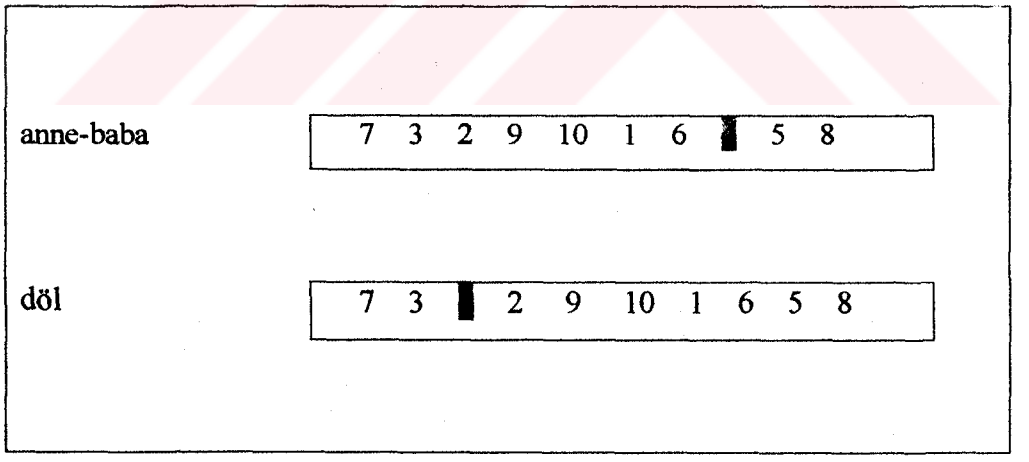
Ters çevirme mutasyonu bir alt dizi oluşturmak için bir kromozomda rastgele iki pozisyon seçer. Daha sonra bu alt dizi Şekil 2.5'de olduğu gibi iki ucu arasında ters çevrilir.



Şekil 2.5 Ters çevirme mutasyonu

2°) **Ekleme**

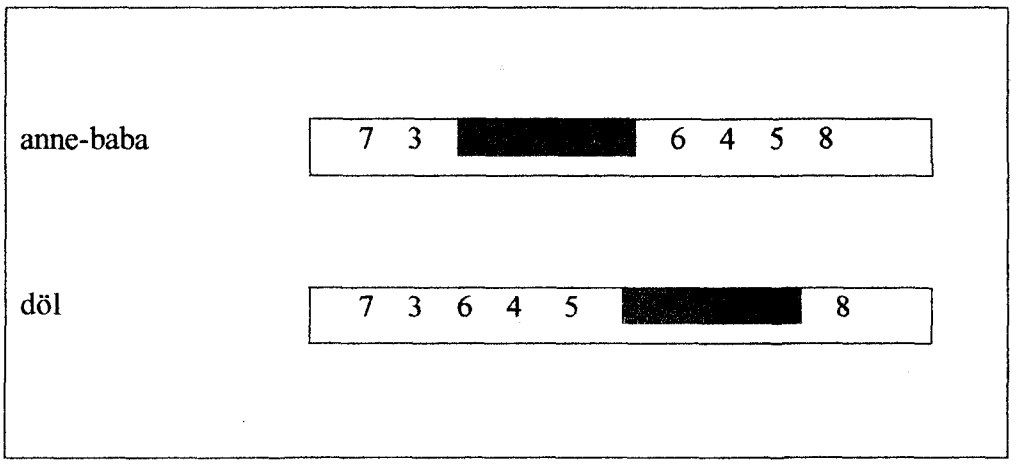
Ekleme rastgele bir parça seçer ve onu rastgele bir yere yerleştirir.



Şekil 2.6 Ekleme mutasyonu

3°) **Yer Değişikliği Mutasyonu**

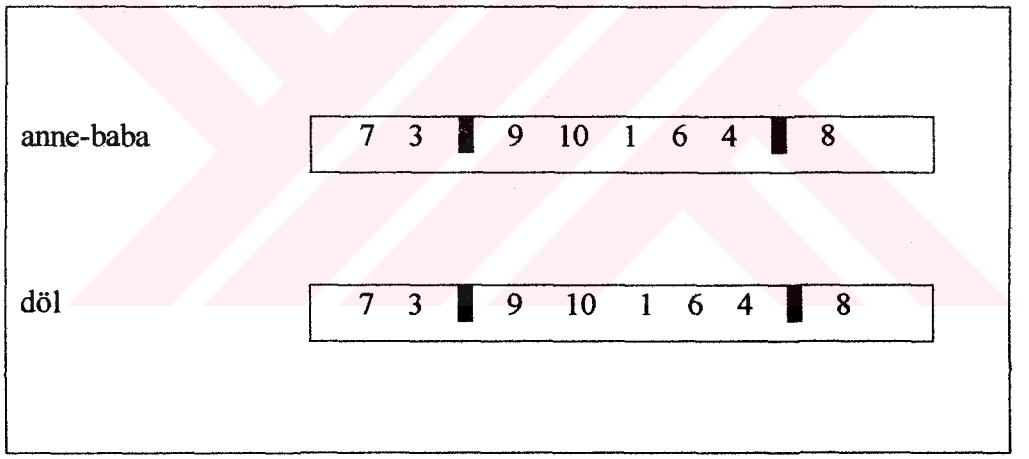
Yer değişikliği mutasyonu rastgele bir alt dizi seçer ve onu Şekil 2.7'de görüldüğü gibi rastgele bir yere yerleştirir.



Şekil 2.7 Yer değişikliği mutasyonu

4°) **Karşılıklı Değişim Mutasyonu**

Karşılıklı değişim mutasyonu, rastgele iki gen seçer ve yerlerini değiştirir (Aksoy,2001).



Şekil 2.8 Karşılıklı değişim mutasyonu

2.3. GA PARAMETRELERİ (KONTROL PARAMETRELERİ)

G.A.'ların işleyişi, arama uzayından daha önce örneklenmiş olan bölgelerin işletilmesi ve bu uzaydan yeni bölgelerin araştırılmasının dengeli birleşimi olarak düşünülebilir. GA'ların performansını kontrol eden bu denge, kontrol parametrelerinin doğru seçilmesi ile belirlenmektedir (Yeniay, 1999). Bu kontrol parametreleri:

1. Yığın Genişliği (Popülasyon büyüklüğü) (N)
2. Çaprazlama Oranı (P_c)
3. Mutasyon Oranı (P_m)
4. Yığın Aralığı (Jenerasyon Boşluğu) (G)
5. Seçim Stratejisi (S)
6. Ölçeklendirme Fonksiyonu (Ölçekleme Penceresi) (W)

1. Popülasyon Büyüklüğü (N) : Bu parametre, G.A.'ların hem etkinliğini hem de nihai performansını etkiler. G.A.'lar, küçük popülasyonlarda fazla etkili değildirler. Çünkü bu popülasyonlar çoğu hiperdüzlem (hyperplane) için yetersiz sayıda örnek sunar. Geniş bir popülasyonun ise, fazla sayıda hyperplane'lerden temsilciler içermesi daha muhtemeldir. Böylece G.A.'lar, daha detaylı bir arama gerçekleştirebilirler. Sonuç olarak da, geniş bir popülasyon, optimal olmayan çözümlere yaklaşılmasını engeller. Diğer yandan büyük bir popülasyon, herbir jenerasyon için daha fazla değerlendirmeye ihtiyaç duyacağından daha yavaş ilerleme ihtimali vardır.

2. Çaprazlama Oranı (P_c) : Çaprazlama oranı, çaprazlama operatörünün uygulandığı sıklığı kontrol eder. Herbir yeni popülasyonda $P_C \cdot N$ tane yapı, çaprazlama işlemine tabi olur. P_c çaprazlama oranı ne kadar yüksek ise, popülasyona o kadar çabuk yeni yapılar girebilir. Eğer bu oran çok fazla yüksekse, yüksek performanslı yapılar, seçimin sunacağı iyileştirmelerden daha hızlı bir şekilde atılır. Eğer oran çok fazla düşük ise, o zaman da tarama işlemi, düşük araştırma oranından dolayı durgunlaşabilir.

3. Mutasyon Oranı (P_m) : Mutasyon oranı, kromozomlara ne oranda mutasyon uygulanacağını belirler. Mutasyon, popülasyonun değişkenliğini artıran ikincil bir tarama operatörüdür. Seçim işleminden sonra, yeni popülasyondaki her bir yapının bit pozisyonları (L, her bir yapıdaki bit pozisyonu sayısı), mutasyon oranına eşit bir olasılıkla rasgele bir değişime tabi olur. Sonuç olarak da, herbir jenerasyon için yaklaşık olarak $P_m \cdot N \cdot L$ mutasyonları meydana gelir. Düşük bir mutasyon seviyesi herhangi bir bit pozisyonun bütün popülasyon içerisinde tek bir değere yaklaşmasını engellerken, yüksek seviyedeki bir mutasyon oranı ise önemli bir rasgele arama imkanı sağlar.

4. Jenerasyon Boşluğu (G) : Yığın genişliği, algoritmanın yakınsaması ile doğrudan ilişkilidir. Bu parametre, her jenerasyon boyunca değiştirilecek popülasyon oranını kontrol eder. Şöyle ki; t. kuşaktaki popülasyon yani P(t)'nin $N \cdot (1-G)$ tane yapısı, (t+1). kuşaktaki

popülasyonda yani P(t+1) içinde eksiksiz olarak kalması için rasgele seçilir. G=1.0 değeri, bütün popülasyonun herbir jenerasyon boyunca değiştirileceği anlamına gelir. G=0.5 değeri ise, her popülasyondaki yapıların yarısının bir sonraki jenerasyona aktarılacağı anlamına gelir.

5. Seçim Stratejisi (S) : Deneyler iki seçim stratejisini karşılaştırmıştır. Eğer S=P ise, sade bir seçim stratejisi; eğer S=E (E, çözüm uzayı (cevap yüzeyi)) ise, daha seçici bir strateji kullanılmıştır. İlk olarak uygulanan seçim stratejisinden sonra daha seçici strateji, en iyi performansa sahip yapının bir sonraki jenerasyona eksiksiz olarak geçmesini şart koşar. Böyle bir stratejinin olmadığı durumlarda örnekleme hatası, çaprazlama veya mutasyondan dolayı en iyi yapının kaybolması muhtemeldir.

6. Ölçümleme Penceresi (Ölçeklendirme Fonksiyonu) (W) : G.A.'nın çalışması süresince yığındaki değişkenlerin korunması önemlidir. Özellikle orantılı seçim mekanizmalarında aramanın etkin bir şekilde yürütülmesi zorlaşmaktadır. Algoritmanın ilk birkaç çevrimi sonunda elde edilen yığında uygunluk değeri yüksek birkaç dizi bulunabilmektedir. Bu dizilerin seçim olasılıkları yüksek olduğundan arama bu diziler etrafında yoğunlaşır. Bu olay algoritmanın zamansız yakınsamasına sebep olmaktadır. Diğer taraftan, algoritmanın son çevrimlerinde yığındaki dizilerin uygunluk değerleri birbirine yaklaşmaktadır. Dizilerin seçim olasılıkları da birbirine çok yakın olacağı için yeni yığınlarda iyi dizilerin korunması zorlaşmaktadır. Bu durum aramanın etkinliğini azaltmaktadır. Orantılı seçim mekanizmalarında bu iki sorunu ortadan kaldırabilmek için, uygunluk değerlerinde bir düzenlemenin yapılması gerekmektedir. Literatürde bunu gerçekleştirebilmek amacıyla ölçeklendirme fonksiyonları geliştirilmiştir. En çok bilinen ölçeklendirme fonksiyonları; doğrusal ölçeklendirme, standart sapma kadar azaltma ve üs yaklaşımıdır.

Bir G.A. ile nümerik bir $f(x)$ fonksiyonu maksimize edildiğinde, bir x yapısının $u(x)$ performans değerini $u(x) = f(x) - f_{\min}$ olarak tanımlamak oldukça yaygındır. Burada $f_{\min}$, belli bir araştırma uzayında $f(x)$ in kabul edebileceği minimum değerdir. Bu dönüşüm $f(x)$ in karakteristiğine bakılmaksızın $u(x)$ performansının pozitif olmasını garantiler. Sıklıkla $f_{\min}$ başlangıçta mevcut değildir. Bu durumda $u(x) = f(x) - f(x_{\min})$ olarak tanımlamak mantıklıdır. Burada $f(x_{\min})$ şimdiye kadar değerlendirilmiş herhangi yapının minimum değeridir. $u(x)$'in hiçbir tanımı x'in iyi değerlerinin ayırt edilmesini etkilemez. Örneğin, $f_{\min} = 0$ olsun. Birkaç jenerasyon sonra, mevcut popülasyon, yalnızca $105 < f(x) < 110$ olan x yapılarını içerebilir. Bu noktada, popülasyondaki hiçbir yapının ortalamadan fazla sapan bir performansı olmayacaktır. Bu da, seçim baskısını daha iyi yapılara doğru azaltır. Bir çözüm

yeni bir f'_{min} parametresini, örneğin 100 değeri ile tanımlamaktır ve her bir yapıda oran bir standarda karşıdır. Örneğin $f(x_i)=110$ ve $f(x_j)=105$ $u(x_i)=f(x_i)-f'_{min}=10$ ve $u(x_j)=f(x_j)-f'_{min}=5$ dir; şimdi x_i nin performansı x_j nin performansının iki katı kadar iyi olarak gözükmektedir.

Deneyimlerimiz, 3 ölçekleme modunu, Ölçekleme Penceresi (W) denen bir parametreye dayanarak araştırmıştır. Eğer W=0 ise ölçekleme aşağıdaki gibi oluşturulmuştur:

f'_{min}, 1. iterasyonda minimum $f(x)$ olarak alınmıştır. Sonraki her bir jenerasyonda değerlendirmeleri f'_{min}'den daha küçük olan böyle yapılar, seçim prosedüründen ihmal edilmiştir. f'_{min}, belli bir popülasyondaki bütün yapılar, f'_{min}'den daha büyük değerlendirmelere her sahip olduğunda genelleştirilmiştir. Eğer 0<W<7 ise, son W jenerasyonlarında oluşan en küçük $f(x)$ değerini, f'_{min} alınır. W=7 değeri bir sonsuz pencere yani hiçbir ölçeklemenin yapılmadığını gösterir (Grefenstette, 1986, s.122-128).

Bitirme Koşulu

G.A.'da üretim-değerlendirme-seçim çevrimi, önceden belirlenen çevrim sayısına ulaşıncaya ya da yığınının ortalama uygunluk değeri yığındaki en iyi dizinin uygunluk değerine önceden belirlenen oranda yaklaşıncaya kadar devam eder (Goldberg, 1989).

3. GENETİK ALGORİTMA KULLANILARAK YAPILAN ÇALIŞMALARDAN BAZILARI

Bu kısımda, Genetik Algoritma'nın kullanıldığı çalışmalardan genel çerçevede bahsedilmiştir. Bu araştırmaların herbiri Genetik Algoritma ve işleyişini ana hatlarıyla ele almışlardır.

1. **Zhao, C. ve Wu, Z., "Çok Yönlü Çok Amaçlı Hücre Formasyonu Üretmede Genetik Algoritma (A genetic algorithm for manufacturing cell formation with multiple routes and multiple objectives)"** adlı çalışmasında; makine (işlem) parçalarının çok amaçlı gruplanması problemine Genetik Algoritma yaklaşımını sunmuşlardır. Ayrıca bu çalışmada özel genetik operatörler geliştirilip, çoklu deneyler gerçekleştirilmiş, denenen problemler için önerilen algoritmalardan elde edilen sonuçlarla diğer yöntemler karşılaştırılmıştır (Zhao ve Wu, 2000).

2. **İris, E.S.** ve **Moghaddain R.T., "Facilities Layout Design By Genetic Algorithms"** adlı çalışmasında eşit ve eşit olmayan büyüklükteki tesis düzenleme problemlerinin ikinci dereceden formülasyonunu çözmede bir G.A. tasarlanması ile ilgili araştırma sunmuşlardır (İris ve Moghaddain, 1998).

3. **Cantoni, M., Morsequerra M. ve Zio, E., "Genetik Algoritmalar ve Optimal İşletme Tasarımı İçin Monte Carlo Modeli (Genetic Algorithms and Monte Carlo Simulation for optimal plant design) "** adlı çalışmalarında tesis güvenliği ve ekonomik performansı belirlemek için Monte Carlo benzeşim metodu ile optimal sistem tasarımını belirlemek için yapılan Genetik Algoritmaları birleştiren bir yaklaşım sunmaktadırlar (Cantoni vd., 2000).

4. **Hasşerbetçi, K.H.** tarafından hazırlanan **"Genetik Algoritmaların Yöneylem Araştırmasında Kullanımı"** adlı çalışmasında, Genetik Algoritmaların Gelişimi, Genetik Algoritmaların Kullanımı, Genetik Algoritmalarda Sıralama-Çizelgeleme verilmiş ve Olmuksa Firması için Sıralama-Çizelgeleme uygulaması ele alınmıştır (Hasşerbetçi, 1997).

5. **Tanrıseven, D.** tarafından **"Genetik Algoritmalar Kullanarak Peptidlerde Motif Bulma (Identifying Peptide Motifs Using Genetic Algorithms)"** isimli çalışma yapılmıştır (Tanrıseven, 2000).

Alıcı moleküllere bağlanan peptid motiflerini bulmak, aşı ve ilaç dizaynında çok önemlidir. Bunun en önemli uygulaması ise MHC-peptid problemidir. Tanrıseven, çalışmasında, belirli MHC moleküllerine bağlanan peptid motiflerine karar vermek için regresyon analizini kullanmıştır. Geleneksel regresyon analizi metodları ile her zaman optimum sonuç

yakalanamadığından optimum regresyon doğrusunu bulmak için G.A. teknikleri kullanılmıştır. G.A. ile bulunan optimum regresyon doğrusu peptid motifini belirlemekle beraber MHC moleküllerinde, peptidlerin bu moleküllere bağlanması için gerekli olan etkenlerde bulunmaktadır.

G.A.'nın yeterliliği değişik uygulama teknikleri ile test edilmiş ve bu problem için optimum parametre seti belirlenmiştir. Sonuçlar, peptid motifinin ikinci pozisyonunun bulunmasında yüzde 95 birebir uyumluluk ve yüzde 100, bir standart sapma ile uyumluluk göstermiştir. Son pozisyonu iki regrasyon doğrusuyla açıklayabilmek için veri ikiye bölünmüştür. İlk regrasyon doğrusu ile peptid motifinin son pozisyonunun bulunmasında yüzde 80, ikinci regrasyon doğrusu ile ise yüzde 75 doğru tahmin yapılabilmiştir.

6. **Bayraktar, B.,** tarafından **"Neural Network Topology Optimization With Genetic Algorithms"** adlı çalışma yapılmıştır (Bayraktar, 1996).

Yapay sinir ağlarının boyutları karmaşıklığı performansını büyük ölçüde etkiler. Bir ağ için en uygun yapı, eğitim verisini öğrenirken deneme verisini de genelleyebilen bir yapıdır. Ne var ki, bu yapıyı bulmak oldukça zordur. Sinir ağlarının yapılarının bulunmasının otomatikleştirilmesi için teknikler geliştirilmesi son yıllarda büyük ilgi çekmektedir. Tekniklerin geliştirilmesinin yanı sıra, herhangi bir öğrenme yordamından bağımsız tekniklerin geliştirilmesi oldukça önemlidir.

Genetik yordamların, sinir ağlarının yapılarının eniyilemesindeki başarısı, ağ içindeki yararlı bölümlerin korunması ve yararlı olmayan bölümlerin iyileştirilmesi esasına dayanır. Bir yapay sinir ağının yapısını betimlemek basit değildir. Bu çalışmada, genetik algoritmalar için uygun olan bir yapay sinir ağı yapısı gösterimi geliştirilmiştir. Uygun olan genetik algoritma işlemleri uygulanmış ve öğrenme yordamlarından bağımsız bir genetik yordam yazılımı kurulmuştur. Kurulan algoritmalar, dışlamalı-veya öğesi, sinüs fonksiyonu, düzensiz zaman serileri tahmini ve göğüs kanseri tahmini problemleri üzerinde denenmiştir. Bahsedilen problemler için kurulan algoritmaların hem basit, hem de uygun ve etkili sinir ağı yapıları bulduğu görülmüştür.

7. **Eren, A.M.'**in **"Optimization of Virtual Paths In Atm Networks Using A Parallel Annealed Genetic Algorithm (PAGA)"** isimli çalışması, ATM ağlarında sanal yolların paralel tavlamalı Genetik Algoritma (PAGA-Parallel Annealed Genetic Algorithm) kullanılarak eniyilemesi üzerinedir (Eren, 1999).

Günümüz bilgisayar ağlarında herbiri ayrı servis kalitesi (QoS-Quality of service) gerektiren birçok değişik hizmetin bir arada varolabilmesi, kaynakların etkili kullanımını temel bir konu durumuna getirir. Sanal yol kavramı çok esnek bir araçtır ve eniyilemesi ağ performansını önemli ölçüde etkiler. Bu eniyilemenin amacı, ağ topolojisi, bağ kapasiteleri ve trafik isterleri verilmişken ağdaki maksimum kullanım oranını en aza indirmektir. Bu problemi çözmek için Paralel Tavlamalı Genetik Algoritması kullanılmıştır. Genetik Algoritmaların performansı parametrelerine çok bağlı olduğundan her bilgisayardaki Genetik Algoritma değişik parametrelerle paralel olarak çalıştırılmıştır. Programın çalışması sırasında da bilgisayarlar arasında bazı bilgilerin alışverişi gerçekleştirilmiş, sonuçların iyiliğini ve çözüm yönteminin performansını ölçmek için algoritma, çeşitli ağ topolojilerinde ve çeşitli trafik isterlerinde çalıştırılmış, bu koşullarda PAGA'nın performansı bir dizi sonuçla karşılaştırılmıştır. Tüm bu ölçümler, algoritmanın çok iyi çözümler sağladığını göstermiştir.

8. **Aksoy, B.** tarafından **"Döviz Piyasalarında Teknik Analiz Temelli Bir Alım/Satım Stratejisi ve Genetik Algoritmalar İle Optimizasyonu (A Technical Analysis Based Trading Strategy in Foreign Exchange Markets and Its Optimization Using Genetic Algorithms)"** isimli çalışma yapılmıştır (Aksoy, 2001).

Teknik Analiz, finansal piyasalarda alım/satım kararı vermede kullanılan bir çok yaklaşımdan biridir. Temel olarak, fiyatları, arz/talep ilişkilerini, grafik ve endikatörler gibi araçlarla öngörü maksatlı olarak niceliksel araştırmaya tabi tutmayı ve piyasa hareketlerini analiz etmeyi amaçlar. Buradan da bu araçların tanımlama ve ölçümlerini temel alan karar kuralları üretir.

Bu çalışmada özel bir tip teknik analiz stratejisi incelenmiştir. Bu stratejinin özelliği belli sayıda tanımlanmış operatör vasıtası ile değişik teknik analiz araçlarından gelen sinyalleri bir araya getirmek ve değerlendirmektir.

USD/DM, USD/Y, USD/Sfr. ve üçgensel arbitraj olanağının olmadığı varsayımı altında bu döviz kurlarından üretilmiş altı diğer zaman serisi tipinde veri kümeleri üzerinde, belli bir sınıf Genetik Algoritmalar kullanılarak, bahsedilen tipteki optimal strateji bulunmaya çalışılmıştır.

Elde edilen sonuçlar, bir faiz getirili döviz referansıyla karşılaştırılmıştır. Sonuçlar, bu çalışmada kullanılan tipte kurallar referans ve incelenen diğer kurallara göre yüksek performans elde edebildiğini göstermektedir. Ayrıca, başka veri küme parçaları üzerinde genelleştirilmesini sağlayacak ortak örgülerin bulunmasının karmaşıklığını da göstermiştir.

9. **Paşaoğlu, G.** tarafından **"A Genetic Algorithm Application for the Cutting Wrapping Problem İn The Textile Industry"** isimli çalışma yapılmıştır (Paşaoğlu,1999).

Bu çalışmada, Genetik Algoritması kullanılarak tekstil sektörünün üretimin son aşamasında karşılaştığı büyük ebatlardaki kumaşlardan daha küçük parçaların kesildiği ve daha sonra bu parçaların kendi kalite kategorilerine göre toplara sarılmasıyla ilgilenen bir problem çözülmüştür. Kumaş toplarının kalitesi, topa sarılan kumaşlara bağlıdır. Bir kumaş parçasının kalitesi ise onun karakteristiklerinin ilgili kalite spesifikasyonlarını (kumaş parçasının uzunluğu, bir kumaş parçasına düşen kritik hata sayısı, kumaş parçasına düşen her hatanın boyu, kumaş parçasında cm başına düşen toplam hata puanı) dahilinde olup olmamasına bağlıdır. Bir metre kumaşın satış fiyatı, kumaşın kalitesiyle orantılıdır. Problem, optimum parça ve hata lokasyonunun kesişiminin bulunmasıyla ilgilenen sürekli atama problemidir. Bu problemde amaç, bir metre kumaş başına düşen kârı en büyüklemektir. Problemin matematik formülasyonu, çeşitli ikili değişkenleri ve sürekli değişkenleri içerir. Geliştirilen G.A., rassal olarak üretilen ve 100 adet veri içeren bir test kümesinde denenmiştir. Sonuçlar, amaç fonksiyonları için hesaplanan üst limitlerle, bu problem için geliştirilen ardışık algoritmanın (sequential algorithm) ve mutative simulated annealing yaklaşımının aynı test problemleri için verdiği sonuçlarla karşılaştırılmıştır. Test sonuçları, G.A.'nın üstün performansını desteklemekte ve G.A.'nın, tekstil sektöründe kesim aşamasında güvenilir ve verimli bir gerçek zamanlı planlama aracı olarak kullanılabilir olduğunu göstermiştir.

10. **Yeniay, M.Ö.** tarafından **"Taguchi Deney Tasarımı Problemlerine Genetik Algoritma Yaklaşımı (A genetical algorithm apporoach to the problems of Taguchi's experimental design)"** isimli çalışma yapılmıştır (Yeniay, 1999).

Özellikle üretim sektöründe, üretilen ürünün kalitesini en iyi yapabilmek için girdilerin çeşitli düzeylerinin belirlenmesi önemli bir problemdir. Bu problemin çözümü için önerilmiş çeşitli yöntemler vardır. Bu çalışmada, bu yöntemlerden biri olan Taguchi Deney Tasarım Yöntemi ile Genetik Algoritma Yöntemi karşılaştırılmıştır. Karşılaştırma 3 yapay test problem kümesi üzerinde gerçekleştirilmiştir. Karşılaştırma amacıyla, bir durağan durum G.A. geliştirilmiş ve Visual Basic 3.0 for Windows yardımıyla programlanmıştır. Geliştirilen bu Genetik Algoritmadan, çalışmada GENMAK adıyla söz edilmiştir.

11. **Özuysal, M.** ve **Suysal, O.** tarafından **"Çok Değişkenli Fonksiyonların Eniyilemelerinde Genetik Algoritmaların Davranışları"** isimli çalışma yapılmıştır (Özuysal ve Suysal, 1997).

Çalışmada G.A.'nın, doğadaki evrimin yöntemlerini kullanan bir arama yöntemi olduğu bu yöntem sayesinde klasik yöntemlerle çözülmesi çok zor, kimi zaman da imkansız olan problemlerin çözülebildiği ifade edilmiştir. Bu tip problemlerin bir örneği çok değişkenli fonksiyonların en iyilemesi problemidir. Bu tip problemlere klasik yaklaşım, türev hesaplarını kullanmayı gerektirir; ancak, çok değişkenli fonksiyonlar için bu hesaplar sonuç vermeyebilir. Probleme bir diğer yaklaşım da tepe tırmanma yöntemleri ile getirilmiştir. Tepe tırmanma yöntemlerinin en büyük sorunu, yerel optimum noktalarına takılmalarıdır. Ayrıca tepe tırmanma yöntemleri, probleme özel bazı iyileştirmeler içerebilir ve çoğu zaman başarılarının temel nedeni bu iyileştirmelerdir. Bu iyileştirmeler çözüm yöntemlerin genel geçerliliğini ortadan kaldırır.

Bu projede kullanılan Genetik Algoritma'nın, fonksiyon iyilemesini nasıl gerçekleştirdiği incelenmeye çalışılmıştır. Genetik Algoritma'nın sonucu bulurken izlediği yolu bulabilmek için çeşitli istatistikler tutulmuştur. Bu sonuçlar grafik haline getirilmiştir.

Fonksiyon en iyilemesini gerçekleştirmek için bir Genetik Algoritma yazılmış, hazırlanmıştır. Ayrıca fonksiyon bilgisinin daha rahat kullanılabilmesi için bir hesaplama kütüphanesi yazılmıştır. Bu kütüphane bir girdi dosyasıyla verilen fonksiyon bilgisini programın kullanabileceği *sona eklemeli* hale dönüştürmektedir. Programın yazımında hızdan çok güvenilirliğe ve taşınabilirliğe önem verilmiştir. Program iki değişkenli fonksiyonların incelenmesini sağlayan bir grafik programını ve veri dosyalarının kolay oluşturulmasını sağlayan bir editör programını da içermektedir.

Bu çalışmada kullanılan"*Genetik Parser*" programı, Genetik Algoritma yöntemini kullanarak polinomların belli aralıklardaki reel köklerini, minumum ve maksimum noktalarını bulmakta ve belli bir x değeri için polinomun değerini hesaplamaktadır. Ayrıca program polinomların belli aralıklardaki grafiğini çizmektedir. Programın yazılma amacı, polinomlar üzerinde bazı işlemleri kısa zamanda gerçekleştirmek dışında Genetik Algoritmaların bir uygulamasını yapmaktır. Program bir "standart C" programıdır. Ancak long double işlemleri desteklemesi ve Windows ortamında çalışabilmesi nedeniyle Borland C for Windows 3.1 altında yazılmış ve derlenmiştir. Kaynak kodu derlenirken derleyicinin sağladığı en iyileme seçeneklerinin büyük bir bölümü kullanılmıştır. Kod içerisinde kullanılan çeşitli kısaltmalar programın bir standart C derleyicisiyle derlenmesini engellemektedir.

"Genetik Parser" programı Genetik Algoritmaları kullamanın yanında polinom işlemlerini kolaylaştırmayı da hedeflemektedir. Bu nedenle bir arabirim içerir. Arabirim Visual Basic 3.0 for Windows kullanılarak tasarlanmış olup Windows 95 altında çalışmaktadır. Arabirim tasarlanırken kullanıcıya en rahat ortamı hazırlamak amaçlanmıştır.

Program ek olarak, temel bir grafik çizim programı içerir. Bu program, belli aralıklar içerisinde bir tek değişkenli fonksiyonun grafiğini çizer. Bu programa da arabirim yoluyla ulaşılabilmektedir.

Programın kullanımı:

Polinom girişleri arabirimde yer alan metin kutuları ile yapılır. Kutulara girilebilecek karakterler bir polinomda yer alabilecek karakterlerle sınırlıdır. Polinom üç ayrı giriş kutusundan birisine yazılır. Polinom üzerinde işlem yapmak için önce hesapla çerçevesinden yapılacak işlem seçilir. Daha sonra yapılacak işleme göre değişken değeri ya da aralığı girilir. "Polinom" menüsündeki "Hesapla" seçeneği ya da araç çubuğundaki hesapla tuşu kullanılarak işlem başlatılır. Burada kontrol "Proje.exe" programına geçer. Gerekli hesaplamalar yapılarak sonuçlar arabirimde belirtilen çıktı dosyasına ve ekrana yazılır. Kontrol arabirime geri döner. Eğer seçilen işlem grafik çizdirme ise kontrol hesaplama işleminin başında "Proje.exe" dosyası yerine "Graph.exe" dosyasına geçer. Bu program MS-DOS altında çalışmakta ve polinomun grafiğini çizmektedir. Bir tuşa basılmasıyla beraber Windows ortamına geri dönülür. Arabirim polinomların düzenlenmesi ve saklanması amacına yönelik pek çok işleve sahiptir.

Program üç ayrı polinom giriş kutusu için üç ayrı polinom listesi tutar. Bu polinom listeleri içinde araç çubuğundaki düğmeler ve "Polinom" menüsündeki seçenekler yardımıyla dolaşılabilir, polinomlar düzenlenebilir ve polinomların hata denetlemesi yapılabilir. Ayrıca arabirim pano işlemleri için "Düzen" menüsünde seçenekler ve araç çubuğunda düğmeler yer almaktadır.

Gözlemler

1. Genetik Algoritmanın çalışması sırasında, bit pozisyonlarındaki bir veya sıfırların frekansları azalan anlamlılık sırasında bire yakınsamaktadır.
2. Bazı komşu basamakların frekansları bağımlı olarak artmaktadır. Bu basamaklar ve bağıntıları Genetik Algoritmanın ilerleyişine göre değişmektedir.

3. Fonksiyonun değişik olması, basamaklarda bir değerin baskın olma hızını etkiler.

4. Çok sayıda çözüm içeren fonksiyonların en iyilemesinde frekansların baskın hale gelme hızı küçüktür.

5. En iyileme sırasında bazı noktalarda bir değer sıradan bağımsız olarak (genellikle beklenenden daha erken) baskın hale geçer.

6. Fonksiyon değişkenlerine göre simetrik değilse fonksiyonun değişkenlerinin bitlerinin baskın hale geçme hızları farklıdır.

Sonuçlar ve Tartışma

Genetik Algoritma'nın başarısı, kısaca, problemi parçalayarak çalışmasıyla açıklanabilir. Bu aşamada bir dinamik programlama yöntemi gibi çalışmaktadır. Genetik Algoritmanın bu yöntemlere göre daha başarılı olmasının en önemli nedeni problemin boyutundan bağımsız olmasıdır. Değişken sayısı ve bu değişkenlerin değerlerinin sonuçlara etkisi araştırma yöntemini değiştirmez. Araştırma arama uzayını dinamik olarak parçalar. Algoritmanın çalışması sırasında çevre etkenlerine (biçim dosyasında belirtilen özellikler, rasgele sayı üreticinin ilk değeri) bağlı olarak arama uzayı farklı şekilde taranır. En alt düzeyde işlemler oldukça karmaşık ve değişkendir; ancak üst düzey bir bakışla değişik yollarla da olsa Genetik Algoritma sonuca ulaşır.

Genetik Algoritmalar ayrıca bazı geri dönüşler içerir. Bu geri dönüşler sayesinde daha kamaşık problemler çözülebilir ve Genetik Algoritma yerel maksimum noktalara takılmaktan kurtulur. Bu geri dönüşlere Genetik Algoritma'nın içinde bulunan çeşitlilik neden olur. Baskın değerin yanında bazı farklı değerler de denenmeye devam eder. Kimi durumlarda bu farklı değerler baskın değerden daha başarılı olur. Bu olay Genetik Algoritma'nın frekans grafiklerinde baskın değerini değiştiren noktalar olarak kendini gösterir.

Çalışmanın genişletilmesiyle Genetik Algoritma'nın çalışma adımları hakkında daha kapsamlı bilgi edinilebilir.

12. **Yılmazbayan, A.** tarafından **"Genetik Algoritma ile Nükleer Yakıt Yöntemi Optimizasyonu (Nuclear Fuel Management Optimization Using Genetic Algorithm"** isimli çalışma yapılmıştır (Yılmazbayan, 1999).

Çalışmada kordaki yakıt demetlerinin dizilişi çok farklı olduğundan nükleer yakıt yönetimi optimizasyon problemini klasik optimizasyon teknikleri ile çözmenin oldukça güç olduğu ifade edilmiştir. G.A.'lar ise popülasyon adı verilen bir grup çözümden yola çıkarak en iyi

çözümlerin bulunduğu gruba ulaşmaya çalışır. Bu çalışmada da nükleer yakıt yönetimi optimizasyon problemi, G.A. kullanılarak çözülmüştür.

13. **Oğuz, F.** tarafından **"Performance Enhancement of An Air-To-Air Missile Autopilot Controller Via Genetic Algorithms (Havadan Havaya Bir Füze Otopilot Denetleyicisinde Genetik Algoritmalar İle Başarım Artırımı)"** isimli çalışma yapılmıştır (Oğuz, 1998).

Modern optimal kontrol teorisi, fiziksel yapısı belirlenen ve dizayn özelliklerini karşılayan bir sistemin performans ölçütünü maksimize veya minimize ederek en uygun denetleyiciyi belirler. Gürbüz çok değişkenli geribesleme dizayn alanındaki son ilerlemeler, sistem duyarlılığı ve kararlılık kriterleri gibi klasik yöntemleri adresleyen ek başarım sınırlamalarının gerekliliğini ortaya koyar. Bu önemli gelişmelere rağmen, varolan teorik teknikler ile arzu edilen sistem başarım sınırlamaları giderilemeyen birçok pratik dizayn problemi artış göstermektedir.

Havadan havaya bir füzenin denetleyici tasarımı gürbüz kontrol problemi olarak ele alınır. Denetleyici parametrelerindeki varyasyonları, modellenmemiş dinamik yapıları, modelleme anında yapılan hataları ve atmosfer içindeki bilinmeyen karmaşık sinyalleri hesaba katarak sabit bir kontrol sistemi kurmak bu problemde ulaşılması gereken noktadır. Doğrusal ve doğrusal olmayan gürbüz kontrol teknikleri bu problemin çözümünde sürekli hal geri beslemesi baz alınarak tartışılmaktadır.

Genetik Algoritmalar plant karakteristiğinden bağımsız olarak analitik başarım oluşumuna ihtiyaç duymadan, olasılıksal geçiş kurallarını kullanan nümerik bir araştırma metodudur.

Doğrusal geri besleme denetleyicisi sayesinde doğrusal bir füze iskelet dinamiği ve aktuatör durum uzay modeli geliştirilir ve ayrık zaman simülasyonu gerçekleştirilir. Basit bir Genetik Algoritma, ağırlıklı doğrusal kuadratik başarım indeksini temel alarak oluşturulan uygunluk fonksiyonu ile doğrusal denetleyici parametrelerini optimize etmek için yapılandırılır. Farklı denetleyiciler için gerçekleştirilen simülasyon sonuçları, Genetik Algoritmaların yükselme zamanı, yerleşme zamanı ve aşım değeri gibi sınırlayıcı koşullar karşısında en iyi geri besleme kazanç parametrelerini elde ederek, bu raporun amacını karşılamaktadır.

14. Kuo, R.J., Chen, C.H. ve **Hwang, Y.C.** tarafından "**Genetik Algoritma Bazlı Fuzzy Neural Şebeke ve Yapay Neural Şebeke Entegrasyonu Kullanan, Hisse Senedi Alım Satım Kararları İçin Zeki Destek Sistemleri (An intelligent stock trading decision support system through integration of genetic algorithm based fuzzy neural network and artificial neural network)**" isimli çalışma yapılmıştır (Kua vd., 2001).

Hisse senetleri borsası, birçok değişik araştırmacı tarafından incelenen, oldukça karmaşık bir ortamdır. Araştırmaların çoğu, politik etkiler gibi niteliksel faktörler yerine, teknik endeksler gibi (nicel) sayısal faktörlerle ilgilidir. Buna rağmen sayısal (nicel) faktör, hisse senetleri piyasasında kritik bir rol oynar. Bu nedenle, çalışmada Genetik Algoritma Bazlı Fuzzy Neural Şebekeler (GFNN) geliştirilerek, hisse senetleri piyasası üzerindeki sayısal (nicel) etkileri ölçmeye yarayan karmaşık anlamlandırma kuralları bilgi tabanı formüle edilmiştir. Daha sonra bu etkiler, Yapay Neural Şebekeler (ANN) kullanılarak teknik endekslerle birleştirilmiştir. Önerilen zeki sistemin bir örneği Taiwan Hisse Senetleri piyasasında kullanılarak, etkinliği saptanmaya çalışılmıştır. Değerlendirme sonuçları nitel ve nicel faktörleri gözönüne alan Neural Şebekenin, sadece nicel faktörleri gözönüne alan Neural Şebekelerden hem alım satım noktalarının belirlenmesinde, hem de alım satım performansında daha üstün olduğunu ortaya çıkarmıştır.

Hisse senetleri piyasası, yüksek kâr beklentisi nedeniyle, en gözde yatırım araçlarından biridir. Bu nedenle yatırımcıların mümkün olan en iyi tahmini yapabilmelerini sağlamak için yapılan araştırmalar çeşitli karar destek sistemlerinin doğmasına yol açar. Geleneksel araştırma yöntemleri, zaman seri analiz teknikleri kullanır. Örneğin çoklu regresyon modelleri ve karma auto regresyon (ARMA) gibi. Bu araştırma yöntemlerinin çoğu, sadece nicel faktörleri, teknik endeksler gibi, gözönüne alır. Son zamanlarda, Yapay Neural Şebekeler (ANN) ve Genetik Algoritmalar gibi yapay zeki teknikleri bu alanda kullanılmaya başlanmıştır. Makro-Ekonomik ya da politik etkiler gibi nitel faktörler hisse senetlerinin eğilimleri üzerinde önemli etkiler yapar. Hatta araştırmacının psikolojisi bile endeks alımına etki edebilir. Öyleyse bu karmaşık ortamda, hisselerin eğilimleri bazı teknik endekslerdeki sınırlandırmalarla tahmin edilemez. Bu, "tecrübeli" hisse senedi uzmanlarının ya da brokerların, sıradan yatırımcılardan daha doğru kararlar alabilmelerinin yanıtıdır.; çünkü onlar sadece teknik endeksleri değil aynı zamanda tecrübelerine dayanarak nitel faktörleri de gözönünde tutarlar. Bu nedenle bu düzensiz bilgiye sahip olmak önemlidir.

Bulanık (Fuzzy) mantık, kontrol alanında kullanılmış ve gelecek vadeden sonuçlar vermiştir. Bu kavram uzmanların belli belirsiz düzensiz bilgisini kavramaya çalışır. Dolayısıyla Bulanık

(Fuzzy) mantık, hisse senedi uzmanının bilgisini taklit eden, oldukça gelecek vadeden bir aday haline gelir. Uzmanın bilgi dağarcığı oldukça özneldir. Çünkü karmaşık kümelerin elemanlarının fonksiyon yapısında kaynaklanır.

R.J. Kuo, C.H. Chen ve Y.C. Hwang, bu araştırmalarında Genetik Algoritma Bazlı Fuzzy Neural Şebeke (GFNN) içinde yer alacak öğrenme algoritması (learning algorithm) geliştirerek hisse senetleri uzmanlarının bilgilerine ulaşmayı amaçlamışlardır.

Temelde bu araştırma, nicel ve nitel faktörlere dayanan zeki bir hisse alış satış karar destek sistemi geliştirerek yatırımcılara doğru karar vermelerinde yardımcı olmuştur. Bu sistem;

1. Faktörlerin Tanımlandırılması,
2. Nitel Model (GFNN)
3. Karar Entegrasyonu (ANN)

nundan oluşmaktadır.

Araştırmanın birinci bölümünde; sistem, nitel yada nicel borsayı etkileyebilecek faktörleri toplamış; sonra GFNN uzmanlarının bilgilerini ya da borsa üzerindeki nitel etkileri organize ederek, karmaşık anlamlandırma kuralları olan bir bilgi tabanı oluşmuştur. Sonuçta, hata geri bildirimli öğrenen algoritmalı bir feed forward Neural şebeke kullanan GFNN' den gelen borsa üzerindeki nitel etkilerin bilgisi, borsa teknik endeksleri ile birleştilerek son kararın verilmesi sağlanmıştır. İkinci bölümde genel altyapı verilmiş, üçüncü bölümde de sistem tartışılmıştır. Dördüncü bölümde Taiwan Borsası hakkındaki tartışmalar ve değerlendirme sonuçları özetlenmiş, beşinci bölümde de sonuç ve öneriler verilmiştir. İkinci bölümde, birinci bölümde açıklanan sistemde Yapay Neural Şebekeler (ANN), Fuzzy Neural Şebekeler ve G.A.'ların uygulanmasının genel altyapısı açıklanmıştır.

Yapay Neural Şebeke (ANN) , Nöropsikoloji modellerini örnek olarak, girdi ve çıktıları biraraya getirerek bir şebeke oluşturan basit online hesaplama elemanlarının toplamından oluşur. Son zamanlarda, bilgisayarların hızlarındaki artış sonucu, Yapay Neural Şebekeler (ANN) kontrol, imge prosesing ve tahmin gibi birçok alanda uygulanmıştır.

Son zamanlarda belirli bazı hisselerin gelecekteki performansını tahmin edebilmek için yeni bir Genetik Algoritma bazlı sistem tanıtılmış ve uygulanmaya başlamıştır. Bu sistem, kurulu Neural sistemle karşılaştırılmıştır. Bulgularda, G.A. bazlı sistemin Neural Şebekeden daha üstün performans gösterdiği görülmüştür. Bu araştırma büyük çoğunlukla, nitel faktörler gözönüne alınmadığı halde, bu sistemin çok iyi sonuçlar verebileceğini göstermiştir. Yine de nitel faktörlerin bazen borsa üzerinde en önemli etkileri yarattığı unutulmamalıdır.

Genetik Algoritma Bazlı Fuzzy Neural Şebeke (GFNN) bu araştırmada hisse senedi uzmanlarının bilgisini anlamak ve fuzzy database oluşturmak için kullanılmıştır. GFNN'nin girdi çıktıları Fuzzy sayılar olduğuna göre, Fuzzy Delph Metodu uzmanların bilgilerini anlamak için kullanılmış ve kabul edilebilir bir GFNN formatına dönüştürülmüştür.

Temelde G.A. kullanılmasının amacı, lokal minimaldan kaçınmaktır. Böylece G.A. önce çözümü kaba hatlarıyla verir daha sonra FNN sonuçları detaylandırır.

G.A.'nın FNN ile entegrasyonu çalışmasında G.A.'la FNN için başlangıç ağırlıklarını sağlar. Bu sadece eğitim süresini kısaltmakla kalmayıp, aynı zamanda lokal minimumdan da sakınılmasını sağlamaktadır.

G.A. adımları şöyledir:

1.Adım: Rasgele n- sayıda popülasyon yapıları üret ve nesil uygunluk fonksiyon numaralarını kur.

2.Adım: Her bir kromozom için bir uygunluk fonksiyon değerini sapta.

3.Adım: Kromozom operatörlerini, (seçme, çaprazlama ve mutasyonu) çalıştır.

4.Adım: Her yeni kromozom için bir uygunluk fonksiyonu değerlendir.

5.Adım: Düşük uygunluk fonksiyon değerli kromozomları yok et yeni ve daha yüksek uygunluk fonksiyon değerli kromozomlar ekle.

6.Adım: Bitirme kriterine ulaşınca dur, aksi taktirde 3.adıma geri dön.

Bu çalışmada uygunluk fonksiyonu şöyle tarif edilmiştir:

$$F = \frac{N}{\sum_{i=1}^{N} (T_i - Y_i)^2} \qquad (3.1)$$

N, populasyon sayısı; T_I, arzu edilen i-inci çıktı; Y_I; ise elde edilen i-inci çıktıdır.

Kodlama metodu olarak da en çok kullanılan binary kodlama metodu kullanılmıştır. Örneğin; 12 sayısı, 8 rakamlı değerde iki bazında 00001100 olarak sunulur. Popülasyon sayısı da bu çalışmada 50'dir.

15. **Chang, N-B.** ve **Wei, Y.L.,** tarafından **"Genetik Algoritma Bazlı Fuzzy Çok Amaçlı Nonlineer Tamsayılı Programlama Modeli ile Şehirlerde Yeniden Değerlendirme İstasyonlarının Yerlerinin Belirlenmesi (Siting recycling drop-off stations in urban area by genetic algorithm-based fuzzy multi objective nonlinear integer programming modeling)"** isimli çalışma yapılmıştır (Chang ve Wei, 2000).

Boş arazilerin hızla azalması ve yeni yakma merkezlerine yer bulunup inşa edilmeleri gibi zaman alan işlemlere bağlı olarak, katı atık yönetim stratejileri geri kazanım, iyileştirme ve ikincil maddelerin yeniden kullanımı ışığında yeniden organize edilmelidir. Katı atık dönüşüm programlarının etkili planlanması ise bir çok katı atık yönetim sistemlerinin karşılaştığı önemli bir zorluktur. Bu çalışmalardan biri de katı atık toplama şebekesindeki yeniden değerlendirme istasyonlarının uygun boyutlarda ve etkili bir biçimde paylaştırılmasına ve yeniden değerlendirme ürünlerinin artırılırken harcamaların azaltılmasına yönelik olanıdır.

Ni-Bin Chang ve Y.L. Wei, bu araştırmalarında G.A. kullanarak problemleri çözen bir Fuzzy Çok Amaçlı Nonlineer Tamsayı Programlama modeli yardımıyla, yerleştirme ve maksimum yönlendirme çalışmalarına yeni bir yaklaşım getirmiştir. Taiwan'ın Kaohsiung şehrinin ilçelerinden birinde uygulanan bu çalışma, böyle bir planlama metodolojisinin potansiyel uygulamalarını sunmuştur.

Boş arazilerin hızla azalması ve yeni yakma merkezlerine yer bulunup inşa edilmeleri gibi zaman alan işlemlere bağlı olarak, geri dönüşüm, birçok ülkede, entegre katı atık yönetim sistemleri için önemli bir alternatif olmuştur. Geri dönüşüm amaçlı araçlar ve dikdörtgen toplama tankları kullanan çöp toplama planlarının maliyet etkinliğini geliştirme yöntemleri sistematik bir yaklaşımla Ni-Bin Chang ve Y.L. Wei'nin bu araştırmasında incelenmiştir.

Bu araştırma, katı atık toplama şebekesindeki güzergah ve yerleştirme konularının iyileştirilmesine yeni bir yaklaşım getirmiştir. Planlama aşamasındaki belirsizlikler düşünülünce Fuzzy Küme Teorisi böylece, amaçların doğrudan planlanması aşamasındaki eksikliklerin giderilmesinde ve parametre değerlerinin dolaylı olarak tahmin edilmesindeki eksikliklerin giderilmesinde alternatif bir yaklaşım olmuştur. Genetik Algoritma Bazlı Fuzzy Çok Amaçlı Nonlineer Tamsayı Programlama Modelini kullanan analitik metod, şehirlerdeki etkili geri dönüşüm alternatiflerinin üretilmesinde kullanılmıştır. Fuzzy Setleri ve Genetik Algoritma Birleşimi Teknikleri kullanan bu çözüm prosedürü, çevre sistemleri analizi alanında öncü bir çalışma olarak kabul edilmiştir. Genetik Algoritma Bazlı Fuzzy Matematik

Programlama modeli kullanılarak, amaçlarla ilgili kesin olmayan mesajlar, çok amaçlı optimizasyon işlemleri ile birleştirilmiş ve daha esnek, gerçekçi en iyi çözüm üretilmiştir.

Taiwan'ın Kaohsiung belediyelerinde yapılan araştırma, ev atıklarının geri dönüşüm programları ekonomisi ve etkinliğinin arttırılması için tanıtım amaçlı pratik bir örnek olarak gösterilmiştir. Bu çalışmada, servis verilen alanın nüfusu, evlerin geri dönüşüm istasyonlarına ortalama yürüyüş mesafesi ve toplama araçlarının güzergah mesafesi, üç ana Fuzzy Planlama amacı olarak düzenlenmiştir. Kaohsiung şehrinde seçilen bölgedeki geri dönüşüm faaliyetlerinin verimliliğini arttırmak için düzenlenen toplama tanklarının yerleşim şeması, tamamen entegre bir geri dönüşüm şemasında maliyet etkinliği ve sosyal uygulanabilirlikleri açısından bir çok kendi gelişen performans indisi göze alınarak geliştirilip yönetim tarafından uygulanmıştır.

Katı atık yönetim sistemlerinin planlanmasında değişik deterministik matematiksel programlama modelleri kullanılagelmiştir. Bu deterministik matematiksel programlama modelleri içinde lineer programlama (LP) , Karışık (Karma) Tamsayı Programlama (MIP), Dinamik Programlama (DP) ve Çok Amaçlı Programlama sayılabilir. Örneğin Hsieh ve Ho (1993) ve Lund ve diğerleri, katı atık yok edilmesi ve geri dönüşüm sistemlerinin lineer programlama teknikleri kullanılarak geliştirilmesini tartışmışlardır. Fakat uygulamada, Karma Tamsayı Programlama Tekniklerini (MIP) kullanan yerel modeller daha çok kullanıla gelmiştir. Ekonomik odaklı yerel modelde risk, olabilecek hava kirliliği, toprağa sızıntı etkileri, gürültü kontrolü ve trafik sıkışıklıkları gibi çevresel etkileri birleştirme çabaları Chang ve arkadaşları tarafından ortaya konulmuştur. Perlak ve Willis (1985) atık yok etme planlamasında, çok amaçlı karar alma analizini geliştirmiştir. Chang ve Wang (1996,1997), Uzlaşık Programlama ve Hedef Programlama Tekniklerini, büyüyen bir metropolitan bölgede, toprak dolgu, yakma ve geri dönüşüm üzerinde olabilecek potansiyel fikir ayrılıklarını çözmek için kullanmışlardır. Diğer yandan atık toplama, belediye katı atık yönetim harcamalarının büyük bir bölümüdür. Böylece çöp toplama faaliyetlerinin iyileştirilmesi, büyük tasarruflar sağlayabilir. Sonuç olarak araç rotası tayini ve tarife problemlerinin kısa dönem planlaması, uzun dönem bölgesel planlamanın tamamlanması değerli bir alt analiz olabilir.

Belirsizlik, katı atık yönetim problemlerinde sık sık önemli bir rol oynar. Katı atık üretimini yöneten rasgele karakter, parametre değerlerindeki tahmin hataları ve planlama hedeflerindeki belirsizlikler ve kısıtlamalardır ve bu belirsizliklerin mümkün olan kaynaklarıdır. Sistematik belirsizliklerle ilgilenen Fuzzy Matematiksel Programlama yaklaşımlarının kullanılması son

birkaç yılda geniş ölçüde dikkat çekmiştir. Örneğin Koo ve diğerleri (1991) Kore'de bir Fuzzy Çok Amaçlı Programlama Algoritmasını bir bölgesel tehlikeli atık işleme merkezinin yerleşim planlamasında kullanarak başarılı olmuştur. Huang ve diğerleri (1992,1995) Kanada'da varsayımsal katı atık yönetim problemlerinin çözümünde Gri Lineer Programlama (GLP), Gri Fuzzy Lineer Programlama (GFLP), Gri Fuzzy Dinamik Programlama (GFDF) ve Gri Tamsayı Programlama (GIP) yaklaşımlarını geliştirmişlerdir. Son zamanlarda Chang ve diğerleri (1996,1997) Taiwan'da entegre katı atık yönetim sistemlerindeki çeşitli önemli konuların çözümünde Fuzzy Hedef Programlama kullanmışlardır. Katı atık yönetimine ek olarak, hava kirliliği kontrolü, su kalitesi yönetimi ve su kaynak sistemleri analizi konularında Fuzzy Matematiksel Programlama tekniklerinin bir çok uygulamaları merkez olarak kullanılmıştır.

Bu geniş çaplı sistemlerin nonlineer karakteristikleri ele alınırken, kompleks çevresel planlama ve yönetim konularının çözümünde G.A. geniş ölçüde ilgi çekmiştir. G.A. genelde en iyi global çözüm arayışında "yeniden üretim, geçişme ve ortama uyum" gibi üç klasik genetik operatör kullanır. Daha sofistike olan geçişme ve ortama uyum operatörleri literatürde basılı yayınlarda bulunabilir. Araç güzergahı ve Gezgin Satıcı Problemlerinin çözümünde G.A. teknolojisi kullanımı, aslında sistem analizinde yeni bir konudur. Ama, en iyi küresel çözüm arayışında, tekrarlayan prosedürün genetik evrim kriterini uygular. G.A.'nın nitelikleri en azından şunları kapsar:

1. G.A., içinde lingarizasyon varsayımların ve parçalı türevlerin gerekli olmadığı basit bir algoritma kullanarak en iyi küresel çözümü için etkili bir küresel metottur;
2. G.A., geleneksel matematiksel programlama algoritmalarında sık sık karşılaşılan taban matrisinin tersiyle ilgili sayısal kararsızlıklardan kaçınılmasını sağlar.
3. G.A., geleneksel Monte Carlo Simülasyonu ve Nonlineer Programlama Modellerinin çözümünde kullanılan önceki optimizasyon algoritmaları ile karşılaştırıldığında çok daha etkili ve sağlamdır.

Bu araştırma geri dönüşüm istasyonlarının yerlerinin belirlenmesi, aynı zamanda toplama taşıtlarının etkili güzergahının belirlenmesinde, G.A. Bazlı Çok Amaçlı Nonlineer Tamsayı Programlama Modelinin kullanılmasını özellikle öngören katı atık yönetim analizinin devamı niteliğindedir.

Geri dönüşüm istasyonlarının yerleştirilmesi, toplama taşıtlarının güzergahlarının belirlenmesi ve tarifelendirilmesi dizayn süreci, aşağıdaki dört analitik adımla açıklanan açık ve kapalı bilgi entegrasyonunu gerektirir. Analizin ilk adımı, seçilen bölgedeki katı atık toplama şebeke

işleyişini belirlemektedir. Bu SUN SPARC 20/50 çalışma istasyonunun şebeke veri tabanının yaratılmasıyla elde edilmiştir. Her bir şebeke bağlantısı için güncel nüfus dağılımı ve toplama noktalarının yerleri kaydedilmiştir. Seçilen bölgedeki katı atık miktarı ve niteliğinin background bilgisi, ilgili veri tabanı yönetim sistemine girilmiştir. İkinci adım ise şebekedeki bütün bağlantılardaki katı atık dolaşım ortalamasının belirlenmesidir. Böylece veri tabanı, sistemdeki bütün atık dolaşım, üretim sürecini özetlemede kullanılır. Atık üretim seviyesi, geri dönüşebilirlerin türü, şebekedeki bağlantıların nüfus dağılımı gibi sosyal, ekonomik ve çevresel parametre değerlerinin incelenmesiyle external alt model, tüm hizmet bölgesindeki potansiyel atık geri dönüşümünü tahmin edebilir. Böyle bir pre-optimal analiz kurulması amacıyla bağımsız bir bilgisayar programı geliştirilmiştir. Üçüncü adımda da optimum yerleşim güzergah ve tarifelendirme için çok amaçlı bir programlama modeli uygulanmış ve varolan veri tabanı ve Genetik Algoritma tekniklerinden faydalanılmıştır. Daha sonra tavsiye edilen alternatiflere başvurularak farklı tolerans seviyeleri yoluyla Fuzzy Üyelik Fonksiyonları belirlenmiştir. Son olarak dördüncü adımda ise, G.A. yoluyla Fuzzy Çok Amaçlı Modeli çözülmüştür. Çalışma analitik sonuçları, interaktif bir yolla gösterebilir, ilgili bilgi istatistiklerini analiz edebilir, performans endeksini değerlendirebilir ve istenen sonuçların çıktısı verilebilir.

Sonuç olarak Ni-Bin Chang ve Y.L. Wei, araştırmalarında seçilen bir bölgedeki geri dönüşüm istasyonlarının yerleşim ve güzergah bakımından optimize edilmesi için bir Fuzzy Matematiksel Programlama Modeli önermiştir. Böyle bir analitik yapı, yüksek hızlı bilgisayarlar kullanılarak çok büyük miktarda bir bilgi entegrasyonu, analiz ve display performansı gerektirir. Geri dönüşüm istasyonlarının ve araç güzergah stratejilerinin optimal büyüklükleri, bu araştırmayla tahmin edilebilir. Optimal çözümün bulunmasında G.A. etkin bir araç olarak kullanılmıştır.

16. **Arslan, A.** ve **Kaya, M.** tarafından **"Genetik Algoritma Kullanan Fuzzy Mantık Üyelik Fonksiyonlarının Saptanması (Determination of fuzzy logic membership functions using genetic algorithms)"** isimli çalışma yapılmıştır (Arslan ve Kaya, 2001).

Bir Fuzzy set, üyelik fonksiyonları ile tanımlanır. Birçok kontrol uygulamasında tanımlanacak setler kolaylıkla adlandırılabilir. Fakat diğer uygulamaların bir ya da bir grup uzmanın bilgisi ile saptanması mümkündür. Fuzzy set bir defa kurulduğunda onlarla alakalı üyelik fonksiyonları da göz önüne alınmalıdır. Üyelik fonksiyonlarının en iyi saptanabilmesi için birinci soru çözümlenmelidir.

Herhangi bir üyelik fonksiyonun şeklini elde etmek için uyarlanan yaklaşım sık sık uygulamaya bağlıdır. Çoğu Fuzzy Mantık Kontrol Parametreleri için üyelik fonksiyonlarının lineer ve genellikle üçgen şeklinde olduğu kabul edilir. Böylece uygulamadaki değişik önemli faktörlerin tanımlanmasında kullanılan setler ve saptanacak konular bu üçgeni tanımlayan parametrelerdir. Bu parametreler genellikle ya kontrol mühendisinin tecrübelerine dayanır ya da otomatik olarak üretilir. Fakat birçok diğer uygulama için üçgen üyelik fonksiyonları uygun değildir. Çünkü bunlar bir istatistik yaklaşım ya da şekillerin otomatik üretilmesiyle elde edilen uzman bilgisine dayanan modellenmiş linguistik terimler ile üyelik fonksiyonlarının şekillerini doğru olarak temsil etmezler.

G.A. metotları, fonksiyon optimizasyonu, güzergah problemleri, tarifelendirme, Neural Şebeke Dizaynı, sistem tanımlaması, dijital sinyal projesi, computer görüntüsü Kontrol ve Makine Eğitimi gibi birçok farklı problemin çözümünde kullanılan rasgele araştırma teknikleridir.

G.A., Karr (1991) tarafından ilk defa üyelik fonksiyonlarının tanımında kullanılmıştır. Karr G.A.'ları taşıma çubuğu probleminde Fuzzy Mantık Kontrolü (FLC) dizaynı için kullanmıştır. İki örnek sunmuştur. Biri, Non-Adaptif G.A. Dizaynlı FLC ve diğeri G.A. Dizaynlı Adaptif FLC dir. Buradaki üyelik fonksiyonları Gaussian'dır ve amaç ise, bir yandan çubuğu dengede tutarken, cart ve cartın bulunduğu yönün ve merkezinin arasındaki farkın karesinin minimize edildiği bir amaç fonksiyonunu (G.A. terminolojisinde uygunluk fonksiyonu) minimize etmektir.

Meredith ve arkadaşları, (1992) bir helikopter için bir FLC'deki üyelik fonksiyonlarının ince ayarları için (fine tuning), G.A.'ları uygulamıştır. Üyelik fonksiyonlarının başlangıç tahminleri, kontrol mühendisleri tarafından yapılır ve bunları kullanarak tanımlanan, parametreleri ayarlayan G.A.'lar, havadaki (hovering) helikopterlerin hareketini minimize eder. Üçgensel üyelik fonsiyonları kullanılmıştır.

Lee ve Takagi (1993) de "cart problemini" ele almışlardır. Onlar tüm sistemi dizayn etmek için, G.A.'ları kullanarak bir holistic yaklaşım vermişlerdir. Bu üyelik fonksiyonları da üçgenseldir.

G.A.'lar, üyelik fonksiyonlarının modellenmesinde uygun bir yöntem sunarlar. Yine de üyelik fonksiyonlarının üçgensel olması diye bir zorunluluk yoktur. G.A.'lar kullanılarak matematik modeli bilinen üyelik fonksiyonları için önerilen metod uygundur.

Araştırmanın ikinci bölümünde G.A.'ların basit prensipleri ve özellikleri tartışılmış ve basit bir G.A. süreci incelenmiştir. Üçüncü bölümde verilen şekildeki üyelik fonksiyonlarının uygun olarak nasıl yerleştirilebileceği tartışılmıştır. Bir tek Girdi-Çıktı Fuzzy sistem için C++ da G.A. programı ve sonuçları hakkında tartışma da verilmiştir. G.A. sürecinin sonuç olarak üyelik fonksiyonlarının şekli saptanmıştır.

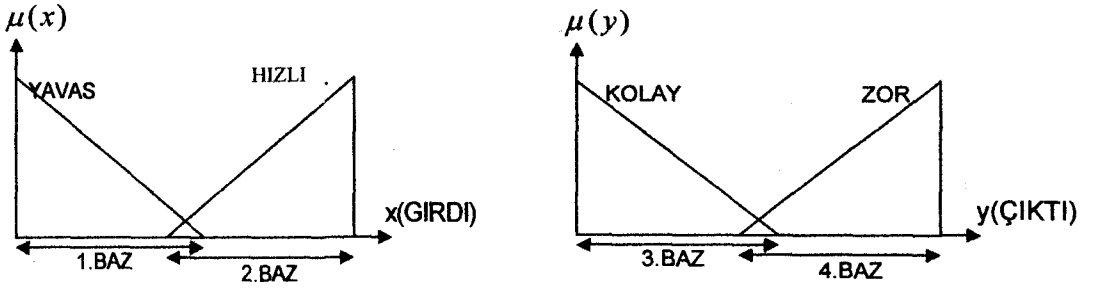
Optimum problem için optimum çözümün bulunması, aynı zamanda Fuzzy Mantık Sistemindeki üyelik fonksiyonlarının hesaplanmasında G.A. kullanılır. Fuzzy değişkenlerini dikkate alan üyelik fonksiyonlarının en uygun yerleştirmesi kural tablosu ve üyelik fonksiyonlarının şekli daha önceden verilen Fuzzy Sistemde bulunur. Üyelik fonksiyonlarının şekli için herhangi bir kısıtlama yoktur. Her yerde kullanılabilirler. Sadece matematik fonksiyonunun modeli bilinmektedir. Bu arada en uygun çözüm aynı Fuzzy sistemin üyelik fonksiyonlarının biçim değişimi ile ilgili bilinen referans değerlerinin karşılaştırılması ile bulunur. Bu yolla referans değerleri için en uygun üyelik fonksiyonları da bulunabilir. Tek girdi-çıktı sistemli C++ da uygulanan programın ne kadar uygun olduğu ve girdi-çıktı eksenindeki üyelik fonksiyonlarının amaca ne kadar hizmet ettiği aşağıda sunulmuştur.

Düşünülen sistemin girdi ve çıktıları için iki farklı üyelik fonksiyonun olduğu ve bunlardan birinin dik üçgen olduğu kabul edilir. Üyelik fonksiyonları (x) girdisi ve (y) çıktısı olarak adlandırılır. x hızlı ve yavaş, y ise kolay ve zor kullanır. Bu durumda linguistik kurallar şöyledir:

1. Kural : Eğer x yavaşsa y kolaydır

2. Kural : Eğer x hızlıysa y zordur

Eğer x girdi değişkeni aralığının 0-7 ve çıktı değişken aralığının 0-49 olduğu kabul edilirse, girdi ve çıktı değişkenleri için Fuzzy Sistem üyelik fonksiyonları Şekil 3.1'de gösterilenler gibidir.



Şekil 3.1 Girdi ve Çıktı İçin Fuzzy Sistemin Üyelik Fonsiyonları

Girdi ve Çıktı için Fuzzy Sistemin Üyelik Fonksiyonları Dik üçgenlerin dik kenarları alt ve üst limitlerle sınırlandırılmıştır. Çıktı değerleri uygunluk fonksiyonunun hesaplanmasında girdi değerleri arasında bilinmelidir. Sistem için gerekli optimum seçim referans değeri sayısı arttıkça iyileşir.

Girdi : $x = \{1,3,5,7\}$

Çıktı : $y = \{1,9,25,49\}$, $i = 1,2,3,4$

G.A.'dan beklenen sonuç dik üçgenin taban uzunluğunun bulunmasıdır. Eğer her bir üyelik fonksiyonunun taban uzunluğu 6 parça olarak ölçülüyorsa problemin çözümünü içeren 1,2,3 bazlar 6x4=24 parçadır. Bu durumda her bazın alacağı maksimum değer 2^6-1=63'tür. Girdi ve çıktı değişkenleri için olan aralıkları sırasıyla 0-7 ve 0-49 'dur. Taban değerleri bu kurallar altında gösterilir ve şöyle formüle edilir:

$$x_i = x_{min} + \frac{d}{(2^L - 1)} \cdot (x_{max} - x_{min}) \qquad (3.2)$$

L, ilgili değişkenin (burada gendir) parça olarak uzunluğu; d, bu değişkenin ondalık değeri, x_{min} değiştirilecek alanın en alt değeri ve x_{max} ise bu alanın maksimum değeridir. Böylece x_i, bu değişkenin değiştirilmiş şeklidir.

Bu tanımlara dayanarak ele alınan Fuzzy Sistemin girdileri, x_{min} = 0, x_{max} = 7, çıktılar için x_{min} = 0, x_{max} = 49' dur. Girdi ve çıktı alan aralıkları devamlı ve bazların parça sunumları genişletilmiş ise değişkenlerin yüzme noktası kesinliği artar. Ama G.A.'nın işlem süresini de uzatır. Burada hem girdi aralığı (0-7) hem de çıktı aralığı (0-49) 6 parça ile ifade edilir. Bu girdi aralığının 1 parçada değişime daha duyarlı olduğu anlamına gelir. Eğer hem girdi hem de çıktı aynı seviyede duyarlı ise farklı parça boylarıyla kodlanmalıdırlar. Örnek olarak burada çıktı parça uzunluğu 8'e çıkarılmıştır. Baz uzunluğunu temsil eden parça uzunluğu uygulama için uygun bir değerle değiştirilebilir. Fakat şu unutulmamalıdır ki parça uzunluğunun azaltılması doğruluğun azalmasına ve parça uzunluğunun arttırılması ise G.A. performansının azalmasına neden olur. Öyleyse yapılması gereken birincil nüfusun rasgele seçilmiş 10 kromozomdan oluşması ve G.A. sürecinin başlamasının sağlanmasıdır. Çizelge 3.1, birincil nüfusu tarif eder. 2-5 kolonlar ilgili kromozomların uzunluklarının onluk değerleridir. 6-9 kolonlar alan aralıklarındaki uzunluklardır. Uygunluk fonksiyonun kompütasyonundan önce toplam hata şöyle hesaplanır:

$$\sum_{i=1}^{4}(y_i - y_{GA_i})^2 \qquad (3.3)$$

Burada y_i , G.A. tarafından elde edilen i-inci referans girdisinin çıktısıdır.

Çizelge 3.1 Birincil nüfus

1 Kromozomlar	2 Baz 1	3 Baz 2	4 Baz 3	5 Baz 4	6 Baz 1 (ref)	7 Baz 2 (ref)	8 Baz 3 (ref)	9 Baz 4 (ref)	10 Fitness (Uygunluk)	11 Kopya Sayısı
11001111000010010000 1011	51	48	36	11	5.66	5.33	28.0	8.55	4430.46	2
01101010111100110001 1101	26	47	12	29	2.88	5.22	9.33	22.5	3722.68	1
01111000100111110011 0001	30	9	60	49	3.33	1.0	46.6	38.1	2597.0	0
10111110101101010101 0110	47	43	21	22	5.22	4.77	16.3	17.1	4387.68	1
01011110000001110101 1111	23	32	29	31	2.55	3.55	22.5	24.1	4236.58	1
11100001111100110111 1101	56	31	13	61	6.22	3.44	10.1	47.4	4450.03	2
10100100110011011111 1000	41	12	55	56	4.55	1.33	42.7	43.5	3417.08	0
00010110010001001010 1110	5	36	18	46	0.55	4.0	14.0	35.7	4360.65	1
00100101111011110001 0110	9	30	60	22	1.0	3.33	46.6	17.1	4209.39	1
11001111111111110001 1000	51	63	60	24	5.66	7.0	46.6	18.6	3919.29	1

En az uygunluk: 2597.0; en yüksek uygunluk: 4450.03, ortalama uygunluk: 3973.08

Amaç toplam hatayı sıfıra mümkün olduğu kadar çok yakınlaştırmaktır. Çünkü toplam hata sıfıra yakın, y_{GA_i} de y'ye yakın ise bu durum uygun bir çözümdür. Uygunluk fonksiyonu 4480 toplam hatadır. Bu yolla minimizasyon süreci maksimizasyon sürecine dönüşür. (10. kolon) Uygunluk fonksiyonunun eksi değerlere düşmesini engellemek için, 4480 maksimum sayısı kullanılır. Bu aynı zamanda maksimum hatadır. Bu 49'un en büyük çıktı değeri olduğu;

$$\sum_{r=1}^{4}(y_i - 49)^2 \qquad (3.4)$$

ile değerlendirilir.

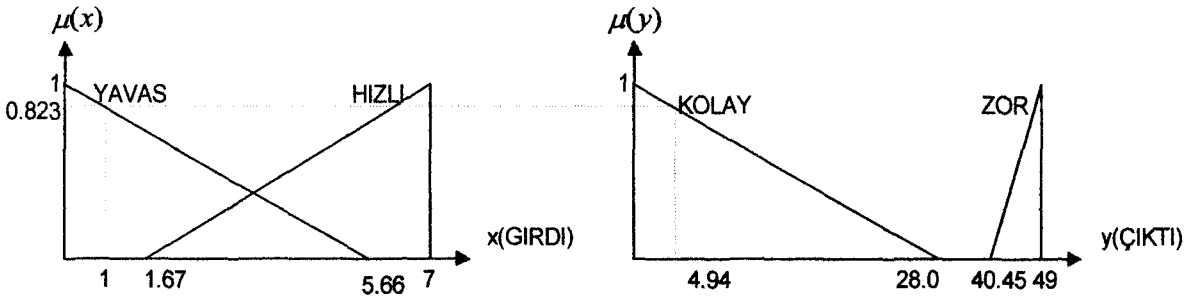
Baz 1 < ilk Ref.Girdi (x=1) ve (7-Baz 2)

< ilk Ref.Girdi (x=1) ve Baz 4= 0

Uygunlukları değerlendirildikten sonra kromozomlar pencereleme ile derecelendirilirler. 11. kolondaki bir sonraki neslin kopya sayıları da burada bulunur. Şöyle formüle edilir:

$$\frac{(\text{Derecelendirilen uygunluk x nüfus büyüklüğü})}{\text{Toplam derecelendirilen uygunluk}} \qquad (3.5)$$

Birincil nüfustaki ilk kromozomun çözümü Şekil 3.2'de görülür.



Şekil 3.2 Birincil nesildeki ilk kromozomun çözümü

x'in baz değerleri 5,66 ve 5,33'tür. Aynı yolla y'nin baz değerleri ise 28,0 ve 8,55'tir.

Bu olayda, çıktı her girdi referansı için değerlendirilir öyle ki girdi üyelik seviyesi, girdi referansı x=1 iken 0,823'tür. Verilen Fuzzy Sistem kurallarında "Eğer x yavaş ise y kolaydır." Görülür. Eğer y değeri doğrudan defuzifikasyon süreci olarak kabul edilirse, bu olaya göre derecesi 0,823 olarak bulunan kolay üyelik fonksiyonunun y noktası, gerçek çıktı değeri 1 iken 4,94'tür. Bu noktada hata $(1- 4,94)^2 = 15,52$'dir.

Bu G.A. programında iki geçişme noktası kullanılmış ve olabilirlik derecesi 1 olarak seçilmiştir. Mutasyon olabilirliği "Yeni neslin ortalama uygunluğu < eski neslin ortalama uygunluğudur." şartıdır.

Bu şartlar altında bir zamandaki evrimin birincil nüfusu Çizelge 3.2'de gösterilmiştir. Bu tablonun diğerinden farkı çıktıları girdi referansı olarak gösterilmesidir.

Çizelge 3.2 Nesil1

Fenotip				Yansıyan Fenotip				Çıktı değeri				Uygunluk Fitness	Kopya sayısı
Baz 1	Baz 2	Baz 3	Baz 4	Baz 1 (ref)	Baz2 (ref)	Baz 3 (ref)	Baz 4 (ref)	y1 (x=1)	y2 (x=3)	y3 (x=5)	y4 (x=7)		
51	48	4	11	5.66	5.33	3.11	8.55	0.54	1.64	45.79	49.0	3993.4	1
56	31	29	61	6.22	3.44	22.55	47.44	3.62	10.87	21.45	49.0	4457.0	2
26	47	44	29	2.88	5.22	34.22	22.55	11.86	31.72	40.36	49.0	3610.0	0
41	12	23	56	4.55	1.33	17.88	43.55	3.92	11.78	0.0	49.0	3838.7	0
56	31	29	61	6.22	3.44	22.55	47.44	3.62	10.87	21.45	49.0	4457.0	2
5	36	50	46	0.55	4.0	38.88	35.77	0.0	0.0	31.11	49.0	4360.6	1
47	43	5	22	5.22	4.77	3.88	17.11	0.74	2.23	41.83	49.0	4150.6	1
9	30	60	22	1.0	3.00	46.66	17.11	0.0	0.0	38.73	49.0	4209.3	1
23	32	13	31	2.55	3.55	10.11	24.11	3.95	0.0	35.43	49.0	4281.3	1
51	63	60	24	5.66	7.0	46.66	18.66	8.23	24.70	41.17	49.0	3919.3	1

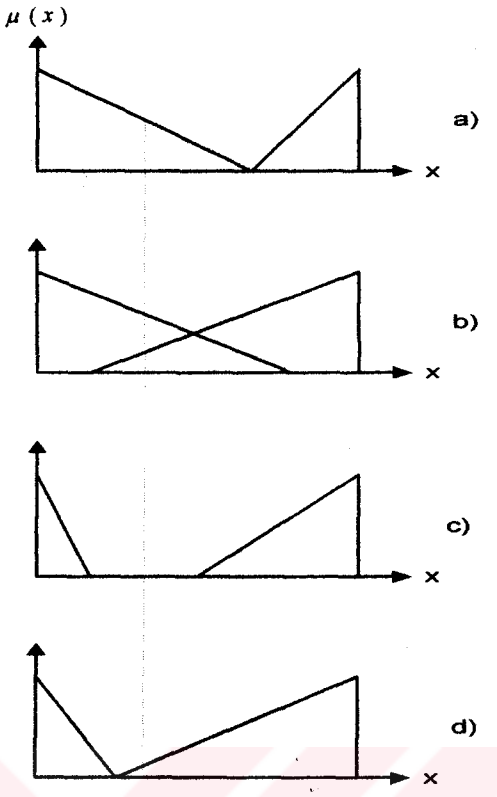
min uygunluk: 3610.02 ; max uygunluk: 4457.03 ; ortalama uygunluk: 4127.75.

1.neslin ürettiği nüfus max ve ortalama uygunluk bakımından bir öncekinden daha iyidir.

Program akışındaki herhangi bir girdi referansı için üyelik fonksiyonları Şekil 3.3 a-d şeklinde görülmektedir. 17. nesildeki optimal çözüm Çizelge 3.3'de gösterilmiştir. Burada önemli bir noktaya dikkat edilmelidir. Optimal çözümü olan (4476,4) birden çok birey vardır.

Çizelge 3.3 Nesil 17

Kromozomlar	Baz 1 (ref)	Baz 2 (ref)	Baz 3 (ref)	Baz 4 (ref)	y1 (x=1)	y2 (x=3)	y3 (x=5)	y4 (x=7)	Uygunluk
10001101110000011111110001	3.88	3.11	11.66	38.11	3.0	9.0	24.5	49.0	4475.7
10001101110000011111110000	3.88	3.11	11.66	37.33	3.0	9.0	25.0	49.0	4476.0
10001101110000011101110000	3.88	3.11	10.88	37.33	2.8	8.4	25.0	49.0	4476.4
11100101101101011101110	6.33	3.0	17.88	35.77	2.8	8.4	25.1	49.0	4476.4
11100101101001011101111	6.33	2.88	17.88	36.55	2.8	8.4	23.6	49.0	4474.6
11100101101101011101110	6.33	3.0	17.88	35.77	2.8	8.4	25.1	49.0	4476.4
10001101110000011101110000	3.88	3.11	10.88	37.33	2.8	8.4	25.0	49.0	4476.4
11100101101101011101110	6.33	3.0	17.88	35.77	2.8	8.4	25.1	49.0	4476.4
11000010001101000111011	5.33	3.88	13.22	45.88	2.4	7.4	25.4	49.0	4475.2
10001101110000011101110000	3.88	3.11	10.88	37.33	2.8	8.4	25.0	49.0	4476.4



Şekil 3.3 Mümkün olan üyelik fonksiyon şekilleri

a) Bu durumda 1. kural işletilir

b) 1. ve 2. kurallar ayrı ayrı işletilir, hatası en az olan seçilir.

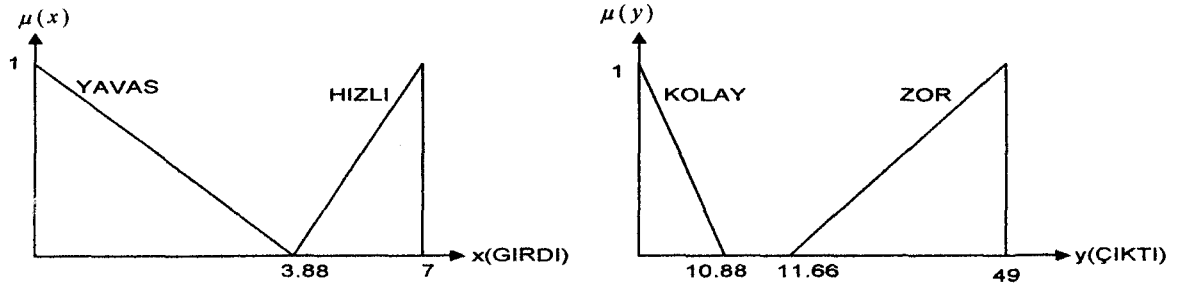
c) Kural işletilmez y çıktısı 0'dır.

d) Sadece 2. kural işletilir.

Girdi referansından anlaşılan Fuzzy Sistem 0 ve 7 arasındaki sayıların karesini alır. Çizelge 3.3'te de görüldüğü gibi 3. sıradaki kromozom en yüksek uygunluğa sahiptir. Fakat en iyi çözüm değildir. Çünkü Şekil 3.4'te görüldüğü gibi

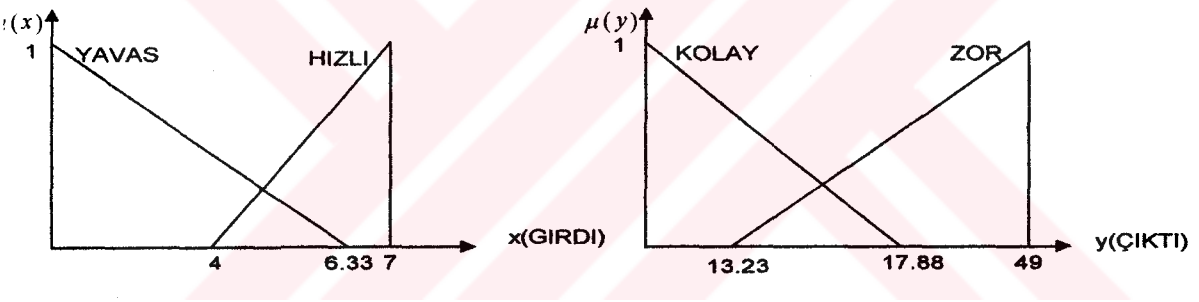
Baz 1 > (7-Baz 2) ve Baz 3 > (49-Baz 4)

kuralını taşımaz. Bu da x=3,88 de ki çıktının sıfır olduğunu gösterir.



Şekil 3.4 Üçüncü sıra kromozom

Aynı yolla çıktı 10,88 ve 11,66 arasında bir sonuç sağlamaz. Böylece optimal sonuç Şekil 3.5'deki dördüncü sıradaki uygunluk değeri Şekil 3.4'deki üçüncü kromozomuyla aynı olan kromozomdur. Fuzzy Sistem üyelik fonksiyonlarının saptanması için bilgisi daha önce verilen geliştirilmiş G.A. programının (-) (nm (k+r)) dır. Burada n nesil sayısı, m, nüfus büyüklüğü, r, referans numarası ve k, kromozom uzunluğudur.



Şekil 3.5 En iyi çözüm

G.A. süreci evrim geçiren kromozomlar birbirine benzemeye başladığında durur. Rasgele mutasyon nüfusun en iyi çözüme ulaşıp ulaşmadığı anlaşılana kadar kullanılabilir. Eğer mutasyona uğrayan nüfus birkaç nesil sonra aynı çözüme ulaşıyorsa çözümün en iyisi olduğu kabul edilir.

Girdi ve çıktı üyelik fonksiyonları üçgen olan bir başka Fuzzy Sistem örneği daha göz önüne alınmıştır.Bu üyelik fonksiyonlarının baz değerlerini belirleyen kurallar ve referans bilgileri 7. şekilde gösterilmiştir.

Kurallar 1) Eğer a g^1 ise b 01'dir

2) Eğer a g^2 ise b 02'dir

3) Eğer a g^3 ise b 03'tür.

Referans bilgisi;

$a_1 = (1,2,3,4,5)$ $b_1 = (4,6,9,16,20,26)$

$i = 1,2,3,4,5$

g2 ve o2 fonksiyonları parça sırasıyla gösterildiğinden iki baz değeri bu fonksiyonların tanımlanmasında saptanmalıdır. A1 ve A2 iki dik üçgen olarak ele alınan iki üçgenin ters tabanları dikeylerinin, yatay değerleridir. A1 ve A2 bu üçgenlerin yerlerini tespit amacıyla G.A.'ya da dahil edilmiştir. Böylece bir kromozom genlerden oluşur.

Baz 1, Baz 2, 3, 4, 5, 6, 7, 8.

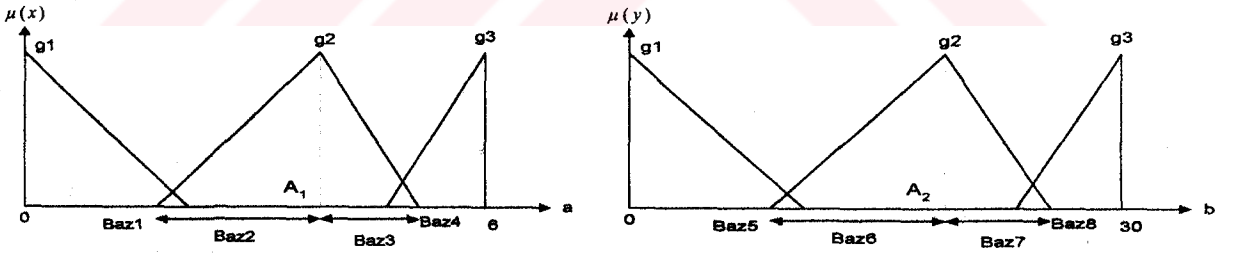
Girdi ve çıktı alan aralıklarının baz değerlerinin nasıl yansıtıldıklarına dikkat etmek gerekir. Örneğin girdi değişkeninin üst ve alt sınırları aşağıdaki gibidir:

Baz 1 : $x_{max} = 6$, $x_{min} = 0$

Baz 2 : $x_{max} = A_1$, $x_{min} = 0$

Baz 3 : $x_{max} = 6$, $x_{min} = A_1$

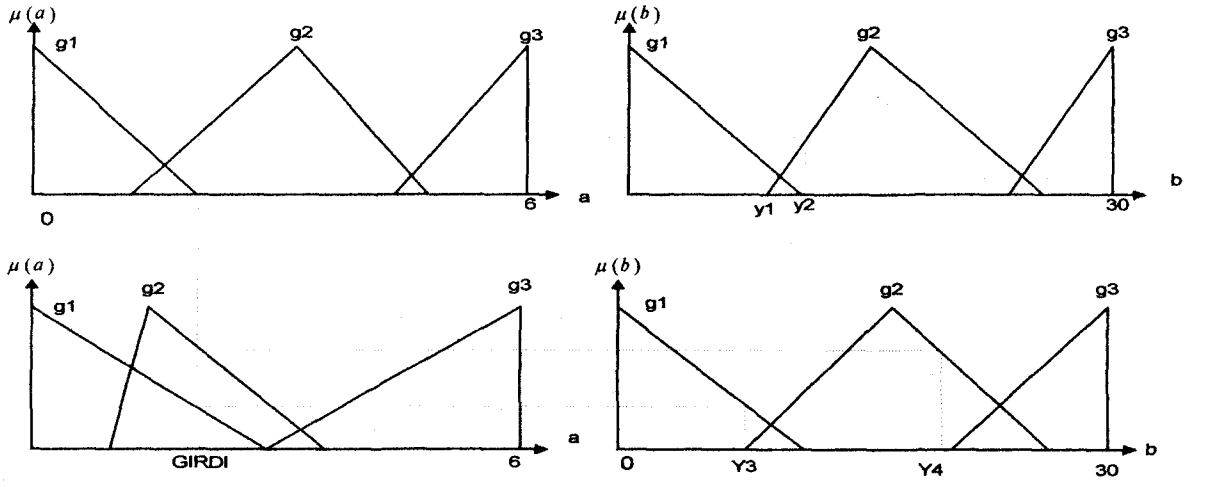
Baz 4 : $x_{max} = 6$, $x_{min} = 0$



Şekil 3.6 Fuzzy Sisteme Ait Girdi ve Çıktı Üyelik Fonksiyonları

Referans değerleriyle çakışan girdi üyelik fonksiyonlarının bir önceki örnekteki kural tablosuna bakıldığında çıktı yansıdığı görülür. Eğer referans değeri hiçbir üyelik fonksiyonuyla çakışmıyorsa (Baz 1< Ref girdi ve (A1-Baz 2> Ref girdi) ve (6-Baz 4) > Ref girdi) ise b=0 dır. Diğer yandan birden fazla kurala uyma kuralı varsa az hatalı kural kullanılır. Çünkü A1 ve A2 sınır değerleri arasındadır. Bazdan alınan değer – Baz 2 – üyelik

fonksiyonu g2, girdi olursa az üyelik fonksiyonlu Baz 6 da çıktı değeri görülür. Aynı durum Baz 3 ve Baz 7 için de geçerlidir. Bu durum Şekil 3.7'de gösterilmiştir.



Şekil 3.7 Girdi referansı için mümkün üyelik fonk.tepkileri şeması

Yukarıdaki şeklin ilk değer G.A.'sı şöyledir:

Eğer $(\text{Ref.girdi} - y1)^2 < (\text{Ref.çıktı} - y2)^2$

Öyleyse b=y1 ya da b=y2

Üçgenler için uygun baz değerleri, G.A. projesinden sonra A nokta değerinde bulunur. Bilgi ve kurallar göze alındığında her birinin uzunluğu 6x10=60 parça olan 20 kromozomlu bir nüfusun 4.mutasyonundan sonraki optimum çözümü veren 24.nesil Çizelge 3.4'te verilmiştir.

Çizelge 3.4 Nesil 24

Baz 1	Baz 2	Baz 3	Baz 4	Baz 5	Baz 6	A2	Baz 7	Baz 8	b_1 (a=1)	b_2 (a=2)	b_3 (a=3)	b_4 (a=4)	b_5 (a=5)	Uygunluk 2449-$\sum$ hat
3.23	3.61	3.61	0.94	14.76	0.0	6.66	22.96	15.23	4.55	9.11	15.88	20.58	25.29	2447.81
3.61	1.30	2.0	1.07	16.19	2.53	5.71	6.55	22.85	3.76	8.94	15.88	20.58	25.29	2448.08
2.19	2.49	2.76	1.54	10.00	0.0	3.80	9.14	19.52	3.80	9.13	16.02	20.68	25.34	2448.04
4.28	0.37	2.95	2.75	19.52	0.0	3.33	25.39	22.85	4.55	9.11	15.6	20.4	25.2	2447.71
4.28	0.08	0.66	4.82	19.52	0.0	3.33	25.39	22.85	4.55	9.11	15.6	20.4	26.14	2448.33
5.90	3.13	3.23	0.21	25.71	1.58	16.66	7.83	26.66	4.35	8.70	15.76	20.50	25.25	2447.91
3.04	0.01	0.19	5.25	13.33	0.49	0.95	26.74	29.04	4.37	8.75	15.24	20.33	25.42	2447.78
3.61	1.30	2.0	1.07	16.19	2.53	5.71	6.55	22.85	3.76	8.94	15.88	20.58	25.29	2448.08
4.28	0.37	2.95	2.75	19.52	0.0	3.33	25.39	22.85	4.55	9.11	15.6	20.4	25.2	2447.71
4.28	0.08	0.66	4.82	19.52	0.0	3.33	25.39	22.85	4.55	9.11	15.6	20.4	26.14	2448.33
0.0	1.30	1.90	1.17	14.28	2.58	2.85	7.32	27.14	1.05	3.45	9.71	6.25	18.12	2118.97
4.28	0.08	0.66	4.82	19.52	0.0	3.33	25.39	22.85	4.55	9.11	15.6	20.4	26.14	2448.33
2.19	2.49	2.76	1.54	10.0	0.0	3.80	9.14	19.52	3.80	9.13	16.02	20.68	25.34	2448.04
0.0	1.30	1.90	1.49	14.28	2.58	2.85	7.32	27.14	1.05	3.32	8.22	15.0	22.5	2310.38
4.28	0.08	0.66	4.82	19.52	0.0	3.33	25.39	22.85	4.55	9.11	15.6	20.4	26.14	2448.33
4.85	0.64	3.71	1.99	23.80	7.46	12.38	1.39	18.57	4.90	9.80	16.70	19.60	25.56	2446.70
4.85	0.64	3.71	1.99	23.80	7.46	12.38	1.39	18.57	4.90	9.80	16.70	19.60	25.56	2446.70
4.28	0.08	0.66	4.82	19.52	0.0	3.33	25.39	22.85	4.55	9.11	15.6	20.4	26.14	2448.33
2.19	2.49	2.76	1.54	10.0	0.0	3.80	9.14	19.52	3.80	9.13	16.02	20.68	25.34	2448.04
0.0	1.30	1.90	1.17	14.28	2.58	2.85	7.32	27.14	1.05	3.45	9.71	6.25	18.12	2118.97

Sonuç olarak "Genetik Algoritma Kullanan Fuzzy Mantık Üyelik Fonksiyonlarının Saptanması" adlı bu araştırmada G.A., Fuzzy Mantıktaki üyelik fonksiyonlarına bakmadan optimum çözümü bulmuştur. Üçgensel üyelik fonksiyonları kullanıldığı halde, matematik modeli tüm üyelik fonksiyonları için geçerlidir. Yani G.A., ya optimum sonucu bulur ya da üyelik fonksiyonları en iyi sonuca dönüşür.

17. **Dawei, L., Li, W.** ve **Mengguang, W.** tarafından **"Genetic Algorithm for production lot planning of steel pipe (Çelik Borunun Üretim Lotunun Planlanmasında Genetik Algoritma)"** isimli çalışma yapılmıştır (Dawei vd., 1999).

1°) *Giriş*

Çelik borunun üretim lotunu planlama problemi, lotlardaki siparişleri kombine etmektir. Öyleki bunlar, temel üretim birimi ile yani lot ile üretilebilirler. Aynı lottaki siparişler aşağıdaki şartları sağlamalıdır.

1. Dış çapları eşit olmalı
2. Kalınlıkları eşit olmalı
3. Çelik derecesine yakın olmalı

Ham borunun uzunluğu, siparişlerinkinden daha büyük olduğundan ham boru siparişlere uygun olarak kesilmelidir. Ayrıca ham borunun uzunlukları, eşit değildir. Böylece leftover lar sık sık oluşur. Leftover'ların azalması daha iyidir. Bu nedenle, kesimin organizasyonu çok önemlidir. Ele alınan strateji, lotlarda sipariş kombinasyonudur. Öyleki bazı siparişler, leftover'ları üretme yerine overgrade olabilir. Böylece iki amaç dikkate alınmaktadır. Birincisi leftoverları minimize etmek, ikincisi bütün overgrade'leri minimize etmektir. Bu kesme problemi, geleneksel olan yani sadece bir amaca sahip olan problemlerden biraz farklıdır.

Bu makalede, Matematiksel Model 2. kısımda, Modelin Genetik Algoritması 3. kısımda, Deneysel Sonuçlar ve Daha İleri Araştırmalar da 4. kısımda verilmiştir.

2°) *Matematiksel Model*

i) *Matematiksel Model*

Bu problem aşağıdaki matematiksel model ile ifade edilebilir. Kullanılan notasyonlar; r_i, i. dış çap; h_i, i nin kalınlığı; l_i, çeliğin derecesi; n_i, uzunluk ve sayı; L_j, j. ham borunun uzunluğu dur.

$S = \{L_1, L_2, ..., L_k\}$ ham boruların uzunlukları kümesi

$$x_{ij} = \begin{cases} 1 & , \quad \textit{eğer } i = j \textit{ (i siparişi } j \textit{ lotunda üretiliyorsa)} \\ 0 & , \quad \textit{diğer durumlarda} \end{cases}$$

$A_j = \{i \mid x_{ij} = 1\}$, j=1,2,..,m

$z_u^{\ i}$: L_u uzunluğuna sahip olan u. ham borudan kesilen i. siparişin sayısı

m: lotunun üst sınırı

α_{ij} : leftover ın neden olduğu ceza değeri

β_{ij} : overgrade in neden olduğu ceza değeri

O halde çelik boru probleminin (PLP) matematiksel modeli:

$$\min \sum_{j=1}^{m} \sum_{i \in A_j} \alpha_{ij} x_{ij} \tag{3.6}$$

$$\min \sum_{j=1}^{m} \left\{ \min_{1 \le u \le k} \beta_{uj} \right\} \tag{3.7}$$

s.t.

$$\sum_{j=1}^{m} x_{ij} = 1 \tag{3.8}$$

$$\left[(r_s - r_t)^2 + (h_s - h_t)^2 \right] x_{sj} x_{tj} = 0 \qquad \text{s,t=1,...,n ; j=1,...,m} \tag{3.9}$$

$$z_u^{\,r} \cdot z_u^{\,s} \cdot z_u^{\,t} = 0, \qquad \text{r,s,t=1,..,n ; u=1,...,k} \tag{3.10}$$

$$l_s \cdot z_u^{\,s} + l_t \cdot z_u^{\,t} \le L_u, \quad \text{s,t=1,...,n ; u=1,...,k} \tag{3.11}$$

$$x_{ij} = 0,1 \qquad \text{i=1,...,n} \qquad \text{j=1,2,...,m} \tag{3.12}$$

olacak şekilde x_{ij} leri bulmaktır.

ii) *Modelin Analizi*

Yukarıdaki model iki amaçlı kombinatoryal optimizasyon problemidir. (3.6) amacı, toplam leftover ları minimize etmek; (3.7) amacı, tüm over-grade leri minimize etmektir. (3.8) kısıtı, her bir siparişin bir ve sadece bir lota ait olması; (3.9) kısıtı, aynı lottaki siparişlerin aynı dış çapa ve kalınlığa eşit olması gerektiğini ifade eder. (3.10) kısıtı, her bir çelik borunun en fazla iki siparişle dağıtılabileceğini; (3.11) kısıtı, çelik borudan kesilecek toplam uzunlukları çelik borunun uzunluğundan daha büyük olmayacağını gösterir. (3.12) kısıtı ise değişkenler ile ilgili kısıttır. Eğer n=30, m=3 ve k=3 ise, değişkenler 240, kısıtlar da 30 000 nin üzerindedir. Böyle bir büyük ölçekli problem, kişisel bilgisayarlar kullanarak mevcut şartlar altında, makul zamanda optimal çözümü elde edemez. Pratikte yaklaşık optimal çözümün daha çok anlamı vardır.

3°) *Modelin Çözüm Metodu*

Büyük ölçekli kombinatoryal optimizasyon problemleri için genellikle kullanılan iki metod vardır. Biri decompozisyon (ayrışım) programlamadır. Yani problem küçük problemlere ayrılarak çözülür; diğeri direkt metod, yani orjinal ve dual problemi direk olarak çözmedir. Fakat optimal çözüm değil, yaklaşık optimal çözüm bulunur. Kullanılan algoritmalar, Genetic

Algorithms (GA), Simulated Annealing (SA), Tabu Search (TS) ve özel problemleri çözmek üzere heuristic (sezgisel) yöntemlerdir.

G.A. aşağıdaki yöntemi izler:

```
begin
    t=0;
    P_t başlangıç yığınını oluştur
    P_t 'yi değerlendir
    while not (bitirme koşulu) do
    begin
    t=t+1
    P_{t-1}'den P_t'yi seç (Yeniden üretim operatörü)
    P_t 'yi değişime uğrat (Çaprazlama ve mutasyon operatörü)
    P_t 'yi değerlendir
    end
end
```

G.A.'nın özellikleri aşağıdaki 4 yön ile özetlenebilir:

1. Değişkenlerin kendilerine değil, şifrelerine uygulanır.
2. Geleneksel araştırma metodlarının yaptığı gibi, bir noktadan değil, çözüm uzayında biçok noktadan çözümü arar.
3. Türev ya da başka yardımcı bilgiyi değil, amaç fonksiyon bilgisini kullanır.
4. Belirli (deterministik) transfer etme kurallarını değil, olasılıksal (probabilistic) transfer kuralını uygular

G.A. pek çok sahaya, örneğin kombinatoryal optimizasyon, iş çizelgeleri (job scheduling), makine öğrenimi, identification, sistem kontrol, diğer algoritmalarla karışık olarak (Simulated Annealing (SA) ve Sinir Ağları), hibrit algoritmalar oluşturma gibi, geniş bir şekilde uygulanabilir.

Aşağıdaki noktalar G.A. kullanımı için anahtar noktalardır

- Kromozomun kuruluşu
- Çaprazlama ve mutasyon içeren genetik operatörler
- Uyumluluk hesaplama

- G.A. parametreleri belirleme. Örneğin, popülasyon büyüklüğü, çaprazlama ve mutasyon olasılıkları, maksimum jenerasyon gibi.

Bu çalışmada kullanılan G.A.'nın ana esası şöyledir:

i) *Kromozom*: Kromozom, iki alt kromozomdan ibarettir. Toplam uzunluk m+n dir. Önceki kromozomun uzunluğu n; i. pozisyondaki k_1 geni, i. sıranın k_1 lotunda olması demektir. Sonraki uzunluk m; j. pozisyondaki k_2 geninin anlamı, j. lottun çelik borusu k_2. dedir. Örneğin,

p=(3 2 2 4 1 1 4 3 / 4 1 2 3)

p kromozomunun uzunluğu 12 (n=8, m=4), "/" işaretinden önceki n=8, "/" işaretinden sonraki m=4 dür. 5 ve 6.sıra 1. lotta, 2ve 3. sıra 2. lotta, 1 ve 8. sıra 3. lotta, 4 ve 7. sıra 4. lottadırlar (parti büyüklüğü). 1., 2., 3. ve 4.; sırasıyla 1., 2. ve 3. çelik boruyu kullanır.

ii) *Genetik Operatörler*

Çaprazlama Operatörü: Basit çaprazlama ilk alt kromozoma; amacı genetik operatörlerle elde edilen yeni dizilerin uygun birer çözüm olmasını sağlamak olan, sayısal kodlamanın kullanıldığı kombinatoryal eniyileme problemlerinden biri olan gezgin satıcı problemi için, Oliver ve arkadaşlarının (1987) CX (cycle crossover) adını verdikleri çaprazlama operatörü de ikinci alt kromozoma uygulanmıştır. Örneğin ("^" işareti çaprazlama yerini göstermektedir)

p_1=(3 2 ^ 2 4 1 1 4 3 / 4 1 2 3)

p_2=(2 4 ^ 4 1 3 2 2 1 / 2 3 4 1)

Çaprazlamadan sonra iki döl elde edilir.

c_1=(3 2 4 1 3 2 2 1 / 4 3 2 1)

c_2=(2 4 2 4 1 1 4 3 / 2 1 4 3)

Mutasyon Operatörü: Mutasyon bir kromozomun her ilk kromozomu (alt kromozom) için uygulanır. İlk alt kromozoma ters çevirme yöntemi uygulanır. İkinci alt kromozoma da g geninin yerine (g+1)'in mod m 'e göre eşiti yazılır. Örneğin,

p=(3 2 2 4 1 1 4 3 / 4 1 2 3)

c=(3 4 1 1 4 2 2 3 / 1 2 3 4)

Uygunluk (Fitness) : p kromozomunun fitness fonksiyonu, $f(p)$ olarak tanımlansın. $f(p)=C_{max}-a*F_1-b*F_2$ burada C_{max}, büyük bir tamsayı; a ve b, ağırlık katsayıları; $0 \le a,b \le 1$ ve a+b=1; F_1 ve F_2, (1) ve (2) amaçlarının değeridir. Aşağıda F_2'nin hesaplanma yöntemi vardır.

begin

F_2 =0;

for (j=1;j<=1;j++)

p kromozomundan j. lota ait olan s siparişini seç; f_s leftoverini hesapla, sadece s siparişini kes; eğer j. lotta bir başka sipariş varsa f_{st}'yi hesapla ki bu iki sipariş s ve t'yi kesmenin neden olduğu en küçük leftover'ı temsil eder;

seçim: eğer $f_s < f_{st}$, ise sadece s siparişini kes; değilse s ve t siparişlerini birlikte kes; $F_2 = F_2 + \min(f_s, f_{st})$;

end

Aşağıdaki tamsayılı lineer programlama, f_{st}'nin hesaplanmasında kullanılır.

Amaç: $\min N\left(L - l_s z_j^{\,s} - l_t z_j^{\,t}\right)$ (3.13)

s.t.

Kısıtlar: $l_s z_j^{\,2} + l_t z_j^{\,t} \le L$ (3.14)

$0 \le N \cdot z_j^{\,s} \le n_s$ (3.15)

$0 \le N \cdot z_j^{\,t} \le n_t$ (3.16)

$z_j^{\,s}, z_j^{\,t} \in l$ (3.17)

L, çelik borunun uzunluğu; N, s ve t siparişlerini kesmede kullanılan çelik borunun sayısını gösterir. Alt sınır, $\left[\left(n_s l_s + n_t l_t\right)/L\right]$ yoluyla belirlenebilir. l, tamsayı kümesidir. Tamsayılı lineer programlamanın çözümü için dal ve sınır metodu uygulanır.

4°) *Deneysel Sonuçlar*

Algoritmanın etkin olduğunu göstermek için k=4, n=12, 20 ve 30 olan denemeler PC 486'da test edilmiş ve 10 kez çalıştırılmıştır. Programlama Microsoft C kodundadır. Kullanılan

datalar, P.R. China'da Tianjin Çelik Boru şirketinden elde edilmiştir. Kullanılan G.A. parametreleri ve deneysel sonuçlar Çizelge 3.5' de verilmiştir.

Çizelge 3.5'deki çaprazlama ve mutasyon olasılıkları, programın pekçok kez çalıştırılarak hesaplanmış en iyi değerleridir. Çizelge 3.5'den aynı problem için, eğer lotlar eşit değil ise, küçük bir fark olduğu görülür. Böylece çelik borunun PLP problemi önemlidir. Ayrıca burada tartışılan problem lotların sayılarının verilmesidir. Daha ileri bir araştırmada, lotların sayısının değişken olduğu duruma konsantire olunabilir. Yani, G.A.'nın daha iyi kromozomlarını bulmada bir araştırma yönüdür.

Çizelge 3.5 Deneysel Sonuçlar

Siparişlerin Sayısı	12		20		30		
Lot'ların Sayısı	2	3	3	4	5	6	7
Popülasyon Boyutu	80	80	50	50	50	50	50
Maksimum Jenerasyon	100	100	100	100	100	100	100
Çaprazlama Olasılığı	0.8	0.8	0.8	0.8	0.8	0.8	0.8
Mutasyon Olasılığı	0.02	0.02	0.03	0.03	0.03	0.03	0.03
En iyi değer	248	236	463	444	847	831	843
En kötü değer	281	272	504	491	938	934	940
Ortalama değer	257.6	249.3	477.9	462.4	872.5	862.2	877.7

18. **Sakawa, M., Nishizaki, I.** ve **Hitaka, M.** tarafından hazırlanan "**Genetik Algoritmalar Yoluyla Bulanık Parametreli Çok Seviyeli 0-1 Programlama İçin Etkileşimli Bulanık Programlama (Interactive fuzzy programming for multi-level 0-1 programming problems with fuzzy parameters through genetic algorithms)**" isimli çalışmada (Skawa, vd., 2001) Genetik Algoritmalar yoluyla bulanık parametreli çok-seviyeli 0-1 programlama için etkileşimli bulanık programlama metodu sunulmaktadır. Metod, kendilerine ait amaç fonksiyonları (objective function) olan fakat kararlarını koordine edebilen, yani ortak çalışma durumunda olan karar vericilerin (decision makers) bulunduğu hiyerarşik karar problemlerine uygulanabilir. Karar vericilerin bütün seviyelerdeki bulanık hedefleri (goal) belirlendikten sonra, bütün seviyeler arasında tatmin edici (satisfactory) denge göz önüne alınır ve karar vericilerin tatmin seviyeleri güncellenerek verimli bir tatmin edici çözüme varılır. Önerilen metodun ne kadar iyi işlediğini göstermek için (feasibility) üç-seviyeli bir 0-1 programlama problemi için sayısal bir örnek de verilmiştir.

Sakawa, M., Nishizaki, I. ve Hitaka, M.'nin bu çalışmalarını aşağıda ayrıntılı olarak inceleyelim.

1°) *Giriş*

Çok-seviyeli programlama problemlerine bir çözüm fikri olarak Stackelberg çözümü kullanılagelmiştir. Stackelberg çözüm fikrini tarif edebilmek için iki-seviyeli bir problem düşünelim. İki adet karar verici (*KV*) vardır. Her bir *KV*, iki karar vericinin amaç fonksiyonlarını ve kısıtlarını (constraint) tamamen bilir. Üst seviyedeki *KV* (lider)(leader) önce bir karar verir ve akabinde alt seviyedeki *KV* (takip eden) (follower) , liderin kararını bütünüyle bilerek (full knowledge) amaç fonksiyonunu optimize edecek bir karar belirler. Kurala göre, lider de kararını altseviyedeki karar vericinin amaç fonksiyonunu optimize edecek şekilde belirler. Nihayetinde yukarıda bahsi geçen yöntemle bulunan çözüme Stackelberg çözümü denir.

Stackelberg çözümü uygulandığı zaman, eğer iki *KV* arasında hiç iletişim olmadığı veya bir iletişim olsa bile herhangi bir bağlayıcı sözleşme (binding agreement) olmadığı varsayılır. Ne var ki, aralarında işbirliği ilişkisi olduğu düşünüldüğü için üst yönetimin bir lider ve işlerin yürütüldüğü bölümün bir takipçi olarak ele alınacağı merkezi olmayan bir şirket için iki-seviyeli programlama problemi gibi durumlarda yukarıdaki kabul pek gerçekçi değildir. Stackelberg çözümünün işlemsel yönüne bakalım. *KV* 'lerin ikisini de amaç fonksiyonlarının ve ortak kısıt fonksiyonlarının lineer olduğu durumda bile, Stackelberg çözümüne ulaşma

probleminin özel yapılı bir dışbükey-olmayan (non-konveks) fonksiyon içerdiği bilinmektedir. Stackelberg çözümü elde etmek için bir çok algoritma önerilmişse de, çözüm bulma probleminin kuvvetli (strongly) NP-hard olduğu bilinmektedir (Shimizu, vd., 1997). Bu gibi güçlüklerden dolayı, çok-seviyeli programlama probleminin yapısını yansıtan ve hesaplaması kolay yeni çözüm metodu gereklidir.

Bu iki gereksinimi karşılayabilmek için, Lai (1973) ve Shih ve arkadaşları tarafından (1996) verilen çalışmadaki fikirleri takip edilerek, geliştirilmiş ve bulanık parametreli çok-seviyeli 0-1 programlama problemi için genişletilmiştir. Diğer bir deyişle, liderin, bütün seviyeler arasındaki genel tatmin dengesini göz önüne alarak, asgari bir tatmin seviyesini belirlediği bir çözüm talep ettiği ve takip edenin de liderin asgari tatmin seviyesini sağlayacak şekilde kendi amaç fonksiyonunu optimize edeceği ana fikrinden yola çıkılarak bir algoritma geliştirilmiştir.

Genel itibariyle, liderin amaç fonksiyonu ve ortak kısıtlar için probleme optimal çözüm ile takip edenin amaç fonksiyonları ve kısıtları için bulunan optimal çözüm örtüşmez. Bu yüzden, problemin yapısı ve amaç fonksiyonları göz önüne alınarak, ikisinin ortası (compromisable) bir çözüm üretilmek zorundadır.

Diğer yandan, Sakawa ve arkadaşları, uzmanların problemin formülasyonu prosesinde parametrelerin doğasının belirsizliği (imprecise) veya bulanık anlama (fuzzy understanding) bakış açılarından bulanık parametreli matematiksel programlama problemlerini formüle ettiler ve çok amaçlı (multiobjective) programlama problemleri için bir bulanık programlama metodu teklif ettiler (Skawa, vd., 1990).

Sakawa ve arkadaşlarının hazırladığı bu çalışmada, bulanık parametreli, çok-seviyeli, 0-1 programlama problemleri için etkileşimli bulanık programlama problemi sunulmuştur. Lai (1996) ve Shih, Lai ve Lee (1996)'nin metotlarındaki problemlerin üstesinden gelebilmek için karar değişkenleri (decision variables) için bulanık hedefli (fuzzy goal) çok-seviyeli 0-1 programlama problemi açıklanmaktadır. Verilen bu etkileşimli metotta, bütün seviyelerdeki bulanık amaçlar tespit edildikten sonra, birbirine komşu herhangi iki seviye arasındaki tatmin dengesi göz önüne alınarak *KV* 'lerin tatmin seviyeleri güncellenerek verimli bir tatmin edici çözüm elde edilmektedir. Önerilen metodun geçerliliğini göstermek üzere üç-seviyeli bir 0-1 programlama problemi için açıklayıcı bir sayısal örnek verilmektedir.

2°) *Çok-seviyeli 0-1 programlama problemi için etkileşimli bulanık programlama*

Sakawa ve arkadaşlarının bu çalışmasında, her seviyedeki her bir *KV* 'nin, bütün seviyeler

arasındaki genel (overall) tatmin dengesini göz önüne aldığı ve diğerlerinin tercihine ciddi şekilde dikkat ederek *KV*'nin (lider) amaç fonksiyonunu optimize ettiği bir durumda çok seviyeli 0-1 programlama problemi ele alınmaktadır. Böyle bir çok seviyeli 0-1 programlama problemi şu şekilde ifade edilebilir:

KV1 Amaç: $\min\ z_1(x_1,...,x_t) = c_{11}x_1 + ... + c_{1t}x_t$ (3.18a)

.

.

.

KV t Amaç: $\min\ z_t(x_1,...,x_t) = c_{t1}x_1 + ... + c_{tt}x_t$ (3.18b)

Kısıtlar: $A_1x_1 + ... + A_tx_t \le b$ (3.18c)

$x_1 \in \{0,1\}^{n_1},...,x_t \in \{0,1\}^{n_t}$ (3.18d)

Burada,

x_i, $i = 1,...,t$ n_i boyutlu bir 0-1 karar değişkeni;

c_{ij}, $i = 1,...,t$; $j = 1,...,t$; n_j boyutlu sabit bir satır vektörü,

b, m-boyutlu sabit bir sütun vektörü ve

A_i, i= 1,...,t ise $m \times n_i$ sabit matrisi.

dir. Sadeleştirmek açısından aşağıdaki notasyon kullanılsın.

$$x = (x_1,...,x_t)^T \in \{0,1\}^{n_1+...+n_t}$$

$$c = \begin{bmatrix} c_{11} & ... & c_{1t} \\ c_{t1} & ... & c_{tt} \end{bmatrix}$$

$$c_i = (c_{i1},...,c_{it})\ \ i = 1,..,t \quad A = [A_1...A_t]$$

Ayrıca *KV* i 'de i-inci seviyedeki *KV* 'yi belirtsin. Üstteki T transpozeyi ifade etmektedir. Çok-seviyeli 0-1 programlama probleminde (3.18), $z_i(x_1,x_2,...,x_t)$ i=1,...,t i-inci seviyedeki amaç fonksiyonunu gösterir ve *KV* i tarafından minimize edilir. x_i, i = 1,...,t ise *i* ninci seviyedeki bir 0-1 karar değişken vektörünü temsil eder.

Gerçek dünyadaki karar durumunu yakından tanımlayan ve temsil eden bir matematiksel programlama problemini tarif ederken, amaç fonksiyonların ve kısıtların tarifi esnasında gerçek dünyaya ait çeşitli etkenlerin yansıtılması icap eder. Doğal olarak, bu amaç fonksiyonlar ve kısıtlar, uzmanlar tarafından tayin edilebilecek muhtemel değerli birçok

parametre içerir. Alışılagelmiş metotlarda (conventional), bu parametrelerin problemin ifade edilme süreci esnasında uzmanların parametrelerin doğasını anlamaları vasıtasıyla deneysel ve/veya öznel bir tarzda bazı değerlere sabitlenmesi gerekmektedir.

Bu hususta, gerçek dünya durumlarının (real-world situation) çoğunda, genel itibariyle bu parametrelerin muhtemel değerlerinin uzmanlar tarafından ancak hatalı ya da belirsiz bir biçimde bilindiği gözlemlenmektedir. Bu gözlem akılda tutularak, uzmanların parametreleri anlayışını bulanık sayılar (fuzzy numbers) olarak bilinen reel doğrunun bulanık kümeleri (fuzzy sets) vasıtasıyla gösterilebilecek bulanık sayısal veri (fuzzy numerical data) olarak yorumlamak kuşkusuz daha uygun olacaktır. Bulanık parametreleri içererek meydana getirilen matematiksel programlama problemi, alışılagelmiş metodun daha gerçekçi bir versiyonu olarak görülmektedir (Skawa ve Yano, 1990; Skawa, 1993).

Bu bakış açısıyla çalışmada, çok-seviyeli 0-1 programlama probleminin amaç fonksiyonlarında ve kısıtlarında geçen parametrelerin bulanık sayılarla karakterize edildiği kabulü vardır. Sonuç olarak, bulanık parametreli bir çok-seviyeli 0-1 programlama problemi aşağıdaki biçimde ifade edilebilir:

KV1 , Amaç: $\min\ z_1(x_1,...,x_t) = \tilde{c}_{11}x_1 + ... + \tilde{c}_{1t}x_t$ (3.19a)

$\vdots$

KV t , Amaç: $\min\ z_t(x_1,...,x_t) = \tilde{c}_{t1}x_1 + ... + \tilde{c}_{tt}x_t$ (3.19b)

Kısıtlar: $\tilde{A}_1x_1 + ... + \tilde{A}_tx_t \le \tilde{b}$ (3.19c)

$x_1 \in \{0,1\}^{n_1}, ..., x_t \in \{0,1\}^{n_t}$ (3.19d)

Burada,

$\tilde{c}_{ij} = (\tilde{c}_{ij,1},...,\tilde{c}_{ij,n_j})$, $i = 1,...,t$; $j = 1,...,t$;

$\tilde{b} = (\tilde{b}_1,...,\tilde{b}_m)^T$

$\tilde{A}_i = (\tilde{a}_{i,jk})$, i = 1,...,t , $j = 1,...,m$, $k = 1,...,n_i$

ler bulanık parametrelerdir.

Sadeleştirmek açısından aşağıdaki notasyon kullanılsın.

$$\tilde{c} = \begin{bmatrix} \tilde{c}_{11} & \cdots & \tilde{c}_{1t} \\ \tilde{c}_{t1} & \cdots & \tilde{c}_{tt} \end{bmatrix}, \qquad \tilde{A} = [\tilde{A}_1 ... \tilde{A}_t]$$

$\tilde{c}, \tilde{b}, \tilde{A}$ bulanık parametrelerinin bulanık sayılarca karakterize edildiği varsayılarak , bunlara

karşılık gelen üyelik fonksiyonları (membership function) $\mu_{\tilde{c}_{ij,k}}(c_{ij,k})$, i, j = 1,...,t, $k = 1,...,n_j$, $\mu_{\tilde{b}_i}(b_i)$, i=1,...,m, $\mu_{\tilde{a}_{i,jk}}(a_{i,jk})$, i=1,...,t, j=1,...,m, $k = 1,...,n_i$ dir.

$\tilde{c},\tilde{b},\tilde{A}$ bulanık sayılarının α-seviye kümesi, üyelik fonksiyonlarının α seviyesini aştığı adi $(\tilde{c},\tilde{b},\tilde{A})_\alpha$ kümesi,

$$(\tilde{c},\tilde{b},\tilde{A})_\alpha = \left\{ \begin{array}{l} (c,b,A) \mid \mu_{\tilde{c}_{ij,k}}(c_{ij,k}) \geq \alpha,\ i,j = 1,...,t,\ k = 1,...,n_j,\ \mu_{\tilde{b}_i}(b_i) \geq \alpha, \\ i = 1,...,m,\ \mu_{\tilde{a}_{i,jk}}(a_{i,jk}) \geq \alpha,\ i = 1,....,t,\ j = 1,...,m,\ k = 1,...,n_i \end{array} \right\} \quad (3.20)$$

olarak tanımlansın.

Şimdi KV_1'nin 0-1 programlama probleminde geçen bulanık sayıların üyelik fonksiyonlarının tamamının derecesinin belli bir α değerinden büyük veya eşit olmasını göz önüne aldığın farzedelim. Bu durumda, böyle bir α değeri için, problem katsayı vektörü $(c,b,A) \in (\tilde{c},\tilde{b},\tilde{A})_\alpha$ ya bağlı olarak aşağıdaki bulanık olmayan çok-seviyeli 0-1 programlama problemi, aşağıdaki şekilde yorumlanabilir.

KV1 Amaç: $\min\ z_1(x_1,...,x_t) = c_{11}x_1 + ... + c_{1t}x_t$ (3.21a)

.

.

.

KVt Amaç: $\min\ z_t(x_1,...,x_t) = c_{t1}x_1 + ... + c_{tt}x_t$ (3.21b)

Kısıtlar: $A_1x_1 + ... + A_tx_t \leq b$ (3.21c)

$x_1 \in \{0,1\}^{n_1},...,x_t \in \{0,1\}^{n_t}$ (3.21d)

$(c,b,A) \in (\tilde{c},\tilde{b},\tilde{A})_\alpha$ katsayı vektörüne bağlı olarak böyle (3.21) gibi problem sonsuz sayıda mevcuttur ve (c,b,A) nın değerlerinin herhangi bir $(c,b,A) \in (\tilde{c},\tilde{b},\tilde{A})_\alpha$ için keyfi olduğuna dikkat ediniz. Şu anlamda ki; (3.21) problemindeki bulanık sayıların bütün üyelik fonksiyonlarının değerlerinin derecesi α seviyesini aşar. Bununla birlikte, eğer mümkün olursa, KV 'lerin verilen kısıtlar altında amaç fonksiyonları minimize edecek şekilde (3.21) probleminde $(c,b,A) \in (\tilde{c},\tilde{b},\tilde{A})_\alpha$ yı seçmesi arzu edilir. Bu açıdan bakıldığında, belli bir α değeri için, bulanık parametreli çok-seviyeli 0-1 programlama probleminin aşağıdaki bulanık-olmayan çok-seviyeli 0-1 programlama problemi şeklinde anlaşılması gayet doğal olarak

görülecektir.

KV1 Amaç: $\min_{x,c,b,A} z_1(x_1,...,x_t) = c_{11}x_1 + ... + c_{1t}x_t$ (3.22a)

.

.

.

KVt Amaç: $\min_{x,c,b,A} z_t(x_1,...,x_t) = c_{t1}x_1 + ... + c_{tt}x_t$ (3.22b)

Kısıtlar: $A_1x_1 + ... + A_tx_t \le b$ (3.22c)

$x_1 \in \{0,1\}^{n_1}, ..., x_t \in \{0,1\}^{n_t}$ (3.22d)

$(c,b,A) \in (\tilde{c},\tilde{b},\tilde{A})_\alpha$ (3.22e)

KV 1 'in bir α seviyesi seçtiği kabul edilir. (3.22) probleminde (c,b,A) parametrelerinin sabitten ziyade karar değişkenleri olarak ele alındığı kaydedilmelidir.

İnsan hükümlerinin bulanıklığı gözönüne alındığında *KV* 'lerin amaç fonksiyonları olarak bulanık amaçlara sahip olması doğaldır. (3.22) deki her bir $z_i(x,c_i)$, i= 1,....,t amaç fonksiyonları için, *KV* 'lerin "$z_i(x,c_i)$ amaç fonksiyonunun belli bir p_i değerinden doyurucu, küçük veya eşit olması gerekir" gibi bulanık hedeflere sahip olduğunu varsayalım.

α= 0 olsun, *KV* i $z_i(x,c_i)$ amaç fonksiyonları için bulanık hedef tayin eden bir üyelik fonksiyonu seçiminde, $z_i(x,c_i)$ amaç fonksiyonunun bireysel minimumu **(individual minimum)**,

$$z_i^{\min} = z_i(x^{io}, c_i^o) = \min\left\{ z_i(x,c_i) \mid A_1x_1 + ... + A_tx_t \le b, x_i \in \{0,1\}^{n_i}, i = 1,...,t, \atop (c,b,A) \in (\tilde{c},\tilde{b},\tilde{A})_o \right\} \quad (3.23)$$

olarak,

ve bireysel maksimumu**(individual maximum),**

$$z_i^{\max} = \max\left\{ z_i(x,c_i) \mid A_1x_1 + ... + A_tx_t \le b, x_i \in \{0,1\}^{n_i}, i = 1,...,t, (c,b,A) \in (\tilde{c},\tilde{b},\tilde{A})_o \right\} \quad (3.24)$$

olarak alınır.

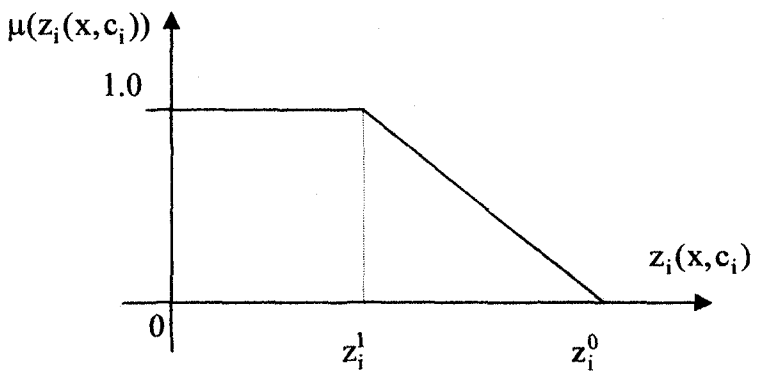
KV i , bireysel minimum (3.23) ve bireysel maksimum (3.24) arasındaki aralıkta tatmin derecesi değişim oranlarını gözönünde bulundurarak, $z_i(x,c_i)$ için kuvvetli monoton azalan $\mu_i(z_i(x,c_i))$ üyelik fonksiyonlarını tayin eder. Üyelik fonksiyonunun tanım kümesi $[z_i^{min}, z_i^{max}]$ aralığıdır ve KV i , amaç fonksiyonunun tatmin derecesinin 0 olduğu z_i^0 ve 1 olduğu z_i^1 değerlerini belirler. z_i^0 dan (daha büyük) istenmeyen değer için $\mu_i(z_i(x,c_i)) = 0$ olarak tanımlanır ve z_i^1 den (daha küçük) arzu edilen değerler için $\mu_i(z_i(x,c_i)) = 1$ olarak tanımlanır.

Sadeleştirmek amacıyla, bu makalede, bütün i=1,...,t için KV i 'nin fuzzy hedefini karakterize eden doğrusal bir üyelik fonksiyonu benimsenmiştir. Karşılık gelen $\mu_i(z_i)$ doğrusal üyelik fonksiyonu,

$$\mu_i(z_i(x,c_i)) = \left\{ \begin{array}{lll} 0 & , & z_i(x,c_i) > z_i^0 \\ \dfrac{z_i(x,c_i) - z_i^0}{z_i^1 - z_i^0} & , & z_i^1 < z_i(x,c_i) \le z_i^0 \\ 1 & , & z_i(x,c_i) \le z_i^1 \end{array} \right\} \qquad (3.25)$$

olarak tanımlanır.

Burada z_i^0 ve z_i^1 üyelik fonksiyon derecesi sırasıyla 0 ve 1 olacak şekilde $z_i(x,c_i)$ amaç fonksiyonunun değerlerini göstermektedir ve KV i 'nin subjektif olarak z_i^0 ve z_i^1 ı belirler. Üyelik fonksiyonu Şekil 3.8'de gösterilmiştir.



Şekil 3.8 Lineer Üyelik Fonksiyonu

Bütün i =1,...,t için Zimmermann (1978) tarafından önerilen yolun uyguladığını ve α=0 alınarak KV i 'nin z_i^0 ve z_i^1 değerlerini belirlediğini varsayalım. Yani,

$$z_i^m = \max(z_i(x^{1o}, c_i^o), ..., z_i(x^{i-1,o}, c_i^o), z_i(x^{i+1,o}, c_i^o), ..., z_i(x^{to}, c_i^o)) \quad (3.26)$$

ile birlikte (3.23) deki bireysel minimumu kullanarak, *KV* i , $z_i^1 = z_i^{min}$, $z_i^0 = z_i^m$ seçerek (3.25) deki gibi doğrusal üyelik fonksiyonu belirler. Bir üyelik fonksiyonunu belirledikten sonra, *KV* i i=1,...,t-1 üyelik fonksiyonu $\mu_i(z_i(x,c_i))$ için minimal bir tatmin seviyesi $\hat{\delta}_i \in [0,1]$'i kişisel olarak belirler. Sonra *KV* t 'de bir üyelik fonksiyonu belirler ve verilen kısıtlar altında üst seviyelerdeki *KV* 'lerin $\mu_1(z_1(x,c_1)),...,\mu_{t-1}(z_{t-1}(x,c_{t-1}))$ üyelik fonksiyonlarının $\tilde{\delta}_1,...,\tilde{\delta}_{t-1}$ lerden büyük ya da eşit olacak şekilde kendi üyelik fonksiyonunu maksimize eder, yani diğer bir deyişle, *KV* t aşağıdaki problemi çözer:

Amaç: $$\max_{x,c,b,A} \mu_t(z_t(x,c_t)) \quad (3.27a)$$

Kısıtlar: $$A_1x_1 + ... + A_tx_t \le b \quad (3.27b)$$

$$\mu_i(z_i(x,c_i)) \ge \tilde{\delta}_i \qquad i = 1,...,t-1 \quad (3.27c)$$

$$x_1 \in \{0,1\}^{n_1}, ..., x_t \in \{0,1\}^{n_t} \quad (3.27d)$$

$$(c,b,A) \in (\tilde{c},\tilde{b},\tilde{A})_\alpha \quad (3.27e)$$

(3.27) ifadesinde, *KV* 'lerin karar değişkenleri için bulanık hedefler üzerindeki kısıtlar Lai (1996) ve Shih ve arkadaşları (1996) formülasyonunda bulunduklarından elenmiştir. Yazarlar, bulanık hedeflerin üst seviyede hem bir amaç fonksiyonuna ve hem de karar değişkenlerine katıldığı zaman istenmeyen çözümlerin ortaya çıktığını belirtmişlerdir.

Eğer (3.27) probleminin optimal bir çözümü varsa, *KV* i 'ler i=1,...t , kendi belirledikleri minimal tatmin seviyesinden büyük eşit tatmin derecelerine sahip bir tatmin çözümü bulmuş demektir. Ne var ki, bazı *KV* 'ler minimal tatmin seviyelerinden daha büyük değerler belirlemişlerse, *KV* t 'nin tatmin derecesi oldukça küçük olacak veya hiç bir uygun çözüm (feasible solution) mevcut olmayacaktır. Sonuç olarak, bütün seviyeler arasında bütün (overall) bir tatmin dengesinin ayarlanamamasından korkulmaktadır.

Bütün seviyeler arasındaki genel tatmin dengesini hesaba katabilmek amacıyla, *KV* i 'nin kendisinden alt seviyelerdeki bütün i = 1, ...,t-1 olarak *KV* 'ler, onların minimal tatmin seviyeleri üzerine uzlaşma yapması gerekir. Bunun için, bütün seviyelerdeki *KV* 'lerin tatmin derecesi,

$$\lambda = \min\left(\mu_1(z_1(x,c_1)),\ldots,\mu_t(z_t(x,c_t))\right) \tag{3.28}$$

olarak tanımlanır ve (3.27) problemi yerine aşağıdaki problem,

Amaç: $\max_{x,z,c,b,A} \lambda$ (3.29a)

Kısıtlar: $A_1x_1 + ..+ A_tx_t \le b$ (3.29b)

$\mu_i(z_i(x,c_i)) \ge \hat{\delta}_i \ge \lambda \quad i = 1,\ldots,t-1$ (3.29c)

$\mu_t(z_t(x,c_t)) \ge \lambda$ (3.29d)

$0 \le \lambda \le 1$ (3.29e)

$x \in \{0,1\}^{n_1+\ldots+n_t}$ (3.29f)

$(c,b,A) \in (\tilde{c},\tilde{b},\tilde{A})_\alpha$ (3.29g)

ele alınır.

(3.29) problemi için aşağıdaki yardımcı problemi tanıtalım:

Amaç: $\max_{x,z,c,b,A} \lambda$ (3.30a)

Kısıtlar: $A_1x_1 + ..+ A_tx_t \le b$ (3.30b)

$\mu_i(z_i(x,c_i)) \ge \lambda \quad i = 1,\ldots,t$ (3.30c)

$0 \le \lambda \le 1$ (3.30d)

$x \in \{0,1\}^{n_1+\ldots+n_t}$ (3.30e)

$(c,b,A) \in (\tilde{c},\tilde{b},\tilde{A})_\alpha$ (3.30f)

(3.30) problemi çözülerek, *KV* 'ler arasında minimal tatmin derecesini maksimize edecek bir çözüm bulunur. Maalesef, (3.30) problemi bütün $\mu_i(z_i(x,c_i))$, i = 1,...,t üyelik fonksiyonları lineer olsalar bile, doğrusal 0-1 programlama problemi olmamaktadır. (3.30) problemini çözmek için, küme değerli $S_i(c_i)$ ve T_j fonksiyonlarını takdim ediyoruz:

$$\left.\begin{aligned} S_i(c_i) &= \{(x,\lambda) \mid \mu_i(z_i(x,c_i)) \ge \lambda\} \quad i = 1,\ldots,t \\ T_j(b_j, A_{1,j},\ldots,A_{t,j}) &= \{x \mid A_{1,j}x_1 + \ldots + A_{t,j}x_t \le b_j\} \qquad j = 1,\ldots,m \end{aligned}\right\} \tag{3.31}$$

Burada $A_{i,j}$, $m \times n_i$ lik A_i matrisinin j ninci satırına karşılık gelen bir satır vektörüdür. O halde $x \in \{0,1\}^{n_1+...+n_t}$ olduğunda, aşağıdaki bağıntıların $S_i(c_{i.})$ ve $T_j(b_j, A_{1,j},..., A_{t,j})$ için sağlandığı kolaylıkla gösterilebilir:

Önerme 1 :

(1) j=1,...,t için eğer $c_{ij}^1 \le c_{ij}^2 \Rightarrow S_i(...,c_{ij}^1,...) \supseteq S_i(...,c_{ij}^2,...)$

(2) Eğer $b_j^1 \le b_j^2 \Rightarrow T_j(b_j^1,...) \subseteq T_j(b_j^2,...)$

(3) i=1,..., t için eğer $A_{i,j}^1 \le A_{i,j}^2 \Rightarrow T_j(...,A_{i,j}^1,...) \supseteq T_j(...,A_{i,j}^2,...)$

$\tilde{c}_{ij}, \tilde{b}$ bulanık sayıları vektörleri ve $\tilde{A}_i$ bulanık sayı matrisinin vektörleri için α-değerli küme özelliklerinden, c_{ij}, b_j ve A_i için uygun **(feasible)** bölgelerin sırasıyla $\left[c_{ij}^L, c_{ij}^R\right]$, $\left[b_j^L, b_j^R\right]$ ve $\left[A_i^L, A_i^R\right]$ olarak gösterilebileceğini kaydedelim. Bu nedenle, Önerme 1 ışığı altında, (3.30) probleminin için optimal çözümü aşağıdaki 0-1 programlama problemi çözülerek bulunabilir.

Amaç: $$\max_{x,\lambda} \ \lambda \qquad (3.32a)$$

Kısıtlar: $$A_1^L x_1 + .. + A_t^L x_t \le b^R \qquad (3.32b)$$

$$\mu_i\left(z_i\left(x, c_i^L\right)\right) \ge \lambda \qquad i = 1,...,t \qquad (3.32c)$$

$$0 \le \lambda \le 1 \qquad (3.32d)$$

$$x \in \{0,1\}^{n_1+...+n_t} \qquad (3.32e)$$

(3.32) problemi çözülerek, bütün *KV* 'lerin minimal tatmin dereceleri arasında minimal tatmin derecesini maksimize eden bir çözüm bulunur. Eğer (3.30) probleminin $\left(x^*, \lambda^*, c^L, b^R, A^L\right)$ optimal çözümü, $\mu_i\left(z_i\left(x^*, c_{i.}^L\right)\right) \ge \hat{\delta}_i \quad i = 1,...,t-1$ koşulunu sağlıyorsa, *KV* i , i =1,...,t nin tatmin edici bir çözüm elde ettiği görülür. Ne var ki, $\left(x^*, \lambda^*, c^L, b^R, A^L\right)$ çözümü bu koşulları her zaman sağlamaz. Bu nedenle, Lai (1996) tarafından tanımlanan, komşu iki seviye arasında tatmin derecesi oranı Δ_i ,

$$\Delta_i = \frac{\mu_{i+1}\left(z_{i+1}\left(x^*, c_{i+1.}^L\right)\right)}{\mu_i\left(z_i\left(x^*, c_{i.}^L\right)\right)} \ , \ i = 1,....,t-1 \qquad (3.33)$$

oranı faydalıdır. Δ_{iL} ve Δ_{iU} sırasıyla *KV* i tarafından tanımlanmış Δ_i nin alt ve üst

değerlerini belirtsin. Eğer $\Delta_i > \Delta_{iU}$ yani $\mu_{i+1}(z_{i+1}(x^*), c_{1.}^L, c_{2.}^L > \Delta_{iU}\mu_i(z_i(x^*), c_{1.}^L, c_{2.}^L)$, o zaman *KV* i , minimal tatmin seviyesi $\hat{\delta}_i$ yi, $\hat{\delta}_{i.}$ kadar artırarak, günceller. Bu durumda *KV* i daha büyük bir tatmin derecesi elde eder ve *KV* (i+1) daha küçük bir tatmin derecesini kabul eder. Tam tersine, eğer $\Delta_i < \Delta_{iL}$ ise, yani $\mu_{i+1}(z_{i+1}(x^*), c_{1.}^L, c_{2.}^L) < \Delta_{iL}\mu_i(z_i(x^*, c_{1.}^L, c_{2.}^L))$ ise, o zaman da *KV* i , minimal tatmin seviyesi $\hat{\delta}_i$ yi, $\hat{\delta}_{i.}$ kadar azaltarak, güncelleştirir ve *KV* i , daha küçük bir tatmin derecesi kabul eder ve *KV* (i+1) daha büyük bir tatmin derecesi elde eder.

Bir *l* iterasyonunda, $\mu_i(z_i^l)$, i= 1,...,t , *KV* i i=1,...,t nin tatmin derecelerini göstersin ve λ^l ise bütün seviyelerin tatmin derecesini göstersin. Ayrıca $\Delta_i^l = \mu_{i+1}(z_{i+1}^l)/\mu_i(z_i^l)$, i ninci ve (i+1). seviyelerinin tatmin dereceleri oranı olsun. x^l, bunlara karşılık gelen bir çözüm olsun. $\forall i = 1,...,t-1$ için *KV* i 'ye , *KV* (i+1) tarafından bir çözüm önerilir. Sonra t ninci seviye hariç bütün diğer seviyelerdeki *KV* 'ler tatmin çözümünü bulur ve etkileşimli (interactive) süreç, aşağıdaki iki koşul da aynı anda sağlanıyorsa sona erer.

Çok-seviyeli 0-1 programlama probleminin etkileşimli sürecinin sonlandırılma koşulları

(1) Her i = 1,...,t-1 için *KV* i 'lerin tatmin seviyeleri, *KV* i tarafından belirlenen $\hat{\delta}_i$ minimal tatmin seviyesinden büyüktür ya da eşittir, diğer bir deyişle, $\mu_i(z_i^l) \geq \hat{\delta}_i \quad i = 1,...,t-1$ dir.

(2) Her i =1, ...,t-1 için, Δ_i^l tatmin derecesi oranları *KV* i tarafından belirlenen üst ve alt sınır kapalı aralığındadır.

(1) koşulu, *KV* (i+1) tarafından önerilen çözümler için *KV* i 'nin sağlaması gerekli **(required)** koşullar demektir. (2) koşulu ile ise bütün seviyeler arasında umumi tatmin dengesinin korunmasını sağlamaktadır.

Bu koşullar aynı anda sağlanmazsa, *KV* i , i =1,...,t nin, $\hat{\delta}_{i.}$ minimal tatmin seviyesini güncellemesi gerekir. Farzedelim ki, (q+1). seviyeden t-1 e kadar olan seviyelerdeki *KV* 'ler, yani *KV* (q+1), *KV* (q+2), , *KV* (t-1) önerilen çözümü tatmin edici bulsun, fakat *KV* q tatmin edici bulmasın. Bu durumda, *KV* q, *KV* (q+1), , *KV* (t-1) kendi minimal tatmin seviyeleri $\hat{\delta}_i$ i = q,q+1,...,t-1 leri güncellemek zorundadırlar. Herhangi iki komşu seviye arasında, üst seviyedeki *KV* ciddi şekilde gözönüne alınarak, alt seviyedeki bir *KV* minimal tatmin seviyesini güncelleştirmelidir.

$\hat{\delta}_i$ *minimal tatmin seviyesini güncelleştirme prosedürü:*

1. Birinci sona erme koşulunu sağlamayan *KV*, diyelim *KV* i olsun, minimal tatmin seviyesi $\hat{\delta}_i$ yi düşürür.

2. Eğer Δ_i^l oranı kendi üst sınırını aşarsa, *KV* i minimal tatmin seviyesi $\hat{\delta}_i$ yi arttırır. Tam aksine, Δ_i^l oranı kendi alt sınırının altına düşerse, *KV* i minimal tatmin seviyesi $\hat{\delta}_i$ yi azaltır.

3. Eğer $\hat{\delta}_{i+1} / \hat{\delta}_i$ minimal tatmin seviyelerinin oranı Δ_i nin geçerli aralığında değilse, o zaman $\hat{\delta}_i$ minimal tatmin seviyesi, (2) deki güncelleştirmeye benzer şekilde güncelleştirilir.

$\hat{\delta}_i'$, i = q,...,t-1, güncelleştirilmiş minimal tatmin seviyesini belirtsin. *KV* t, güncelleştirilmiş minimal tatmin seviyeleri $\hat{\delta}_i'$ ler ile aşağıdaki optimizasyon problemini çözer:

Amaç: $$\max_{x,\lambda,c,b,A} \lambda \qquad (3.34a)$$

Kısıtlar: $$A_1x_1 + ... + A_tx_t \le b \qquad (3.34b)$$

$$\mu_i(z_i(x,c_i)) \ge \lambda \ , \ i = 1,...,q-1 \qquad (3.34c)$$

$$\mu_i(z_i(x,c_i)) \ge \hat{\delta}_i' \ , \ i = q,...,t-1 \qquad (3.34d)$$

$$\mu_t(z_t(x,c_t)) \ge \lambda \qquad (3.34e)$$

$$0 \le \lambda \le 1 \qquad (3.34f)$$

$$x \in \{0,1\}^{n_1+...+n_t} \qquad (3.34g)$$

$$(c,b,A) \in (\tilde{c},\tilde{b},\tilde{A})_\alpha \qquad (3.34h)$$

(3.34) probleminin bir optimal çözümü, (3.30) probleminden (3.32) problemine dönüşüme benzer bir dönüşüm yoluyla aşağıdaki problem çözülerek elde edilir.

Amaç: $\max_{x,\lambda} \lambda$ (3.35a)

Kısıtlar: $A_1^L x_1^L + ... + A_t^L x_t^L \leq b^R$ (3.35b)

$$\mu_i\left(z_i\left(x, c_{i.}^L\right)\right) \geq \lambda \ , \ i = 1,...,q-1 \quad (3.35c)$$

$$\mu_i\left(z_i\left(x, c_{i.}^L\right)\right) \geq \hat{\delta}_i' \ , \ i = q,...,t-1 \quad (3.35d)$$

$$\mu_t\left(z_t\left(x, c_{t.}^L\right)\right) \geq \lambda \quad (3.35e)$$

$$0 \leq \lambda \leq 1 \quad (3.35f)$$

$$x \in \{0,1\}^{n_1+...+n_t} \quad (3.35g)$$

Yukarıda anlatılan algoritma şu şekilde özetlenebilir :

Çok-seviyeli 0-1 programlama problemini çözmek için etkileşimli bulanık programlama algoritması

1. **Adım:** $l = 1$ al. *KV* 1, α-seviyesi kümesi $(\tilde{c}, \tilde{b}, \tilde{A})_\alpha$ nın bir α derecesini belirler. Bütün i = 1,...,t-1 için, *KV* i bir bulanık hedefin bir $\mu_i(z_i)$ üyelik fonksiyonunu belirler. Aynı zamanda, bir minimal tatmin seviyesi δ_i^l ve tatmin derecelerinin bir Δ_i oranı için bir alt ve üst sınır belirler.

2. **Adım:** *KV* t, bir bulanık hedef için üyelik fonksiyonu $\mu_t(z_t)$ yi seçer.

3. **Adım:** *KV* t, (3.30) yardımcı problemini (3.32) problemi aracılığıyla çözer. Bütün i = t, t-1, ... ,2 için, *KV* i, *KV* (i-1) 'e ardışık olarak x^l , $(z_1^l, ..., z_t^l)$, λ^l , $\mu_i(z_i^l)$ ve Δ_i^l çözümünü teklif eder.

4. **Adım:** Eğer çözüm bütün seviyelerin sonlandırma koşullarını sağlıyorsa, *KV* 'lerin bir tatmin edici bir çözüm elde etttiği anlaşılır ve algoritma durur. Aksi takdirde, $l = l + 1$ alınır.

5. **Adım:** Eğer *KV* (q+1), *KV* (q+2) ..., ve *KV* (t-1) önerilen çözümden memnunsa, fakat *KV* q memnun değilse, *KV* q, ..., *KV* (t-1), minimal tatmin seviyeleri $\hat{\delta}_q,...,\hat{\delta}_{t-1}$ leri, minimal tatmin seviyelerini güncelleştirme prosedürüne uygun olarak günceller.

6. **Adım:** *KV* t, (3.34) problemini (3.35) problemi yoluyla çözer. Bütün i = t, t-1,..., 2 için, *KV* i, *KV* (i-1) 'e (3.34) probleminin bir optimal çözümünü ardışık olarak önerir. 4. adıma geri dönülür. Eğer (3.34) probleminin herhangi bir uygun çözümü yoksa 7. adıma geçilir.

7. **Adım:** Bütün i = q, ..., t-1 için, *KV* i 'ler $\hat{\delta}_i$ minimal tatmin seviyelerini, $\hat{\delta}_i$ değerlerini düşürerek, güncelleştirir ve 6. adıma geri dönülür.

Örnek Problem:

Çok-seviyeli programlamanın kullanımını açıklamak amacıyla, özel becerilere sahip bir nakliyat şirketinin n tane iş (proje) aldığını hayal edelim. Bu şirketin, üst seviyelerdeki *KV* ler gibi düşünülebilecek merkezler, bazı siparişleri kabul ederler ve bazılarını da kaynak kısıtları yüzünden geri çevirmek zorunda kalırlar. Üstlenilen nakliyat projeleri için, alt seviyedeki bir *KV* olarak düşünülebilecek olan şirketin işlem (operation) bölümü, taşıma maliyetlerini minimize yapmak üzere bir plan yapar. Şu şartla ki; *l* projesi için, m_l depo olduğu ve o_l tane talep noktası vardır. Böyle bir problem iki-seviyeli 0-1 programlama problemi olarak aşağıdaki şekilde ifade edilebilir :

KV1 Amaç: $$\max \sum_{l=1}^{n} c_l x_l \qquad (3.36a)$$

KV2 Amaç: $$\min \sum_{l=1}^{m} \sum_{j=1}^{m_l} \sum_{k=1}^{o_l} d_{ljk} y_{ljk} x_l \qquad (3.36b)$$

Bu optimizasyon aşağıdaki koşullara bağlı olarak yapılır:

$$\sum_{l=1}^{n} a_{ls} x_l \le b_s \quad , \quad s = 1,...,r \qquad (3.36c)$$

$$\sum_{j=1}^{m_l} y_{ljk} = \beta_{lk} x_l \quad , \quad k = 1,...,o_l \quad , \quad l = 1,...,n \qquad (3.36d)$$

$$\sum_{k=1}^{o_l} y_{ljk} = \alpha_{lj} x_l \quad , \quad j = 1,...,m_l \quad , \quad l = 1,...,n \qquad (3.36e)$$

$$x_l \in \{0,1\} \quad , \quad l = 1,...n \qquad (3.36f)$$

$$y_{ljk} \in \{0,1\} \quad , \quad l = 1,...,n \quad , \quad j = 1,...,m_l \quad , \quad k = 1,...,o_l \qquad (3.36g)$$

Burada,

x taşıma projelerinin seçimi için 0-1 değişkeni

c taşıma projelerindeki kâr katsayıları

a taşıma projelerindeki teknoloji katsayıları

b taşıma şirketinin kaynakları

y_l *l* projesinin taşıma değişkenleri

d taşıma projelerindeki masraf katsayıları

α arz katsayıları

β talep katsayıları

(3.36) problemi, problem formülasyon prosesinde uzmanların, parametrelerin doğasını anlamalarını göz önüne alarak, (3.19) probleminde olduğu gibi bulanık parametreli bir problem olarak formüle edilebilir. Önerilen etkileşimli bulanık programlama algoritması kullanılarak, $\hat{\delta}_i$ parametresi, merkezlerin aldığı toplam kâr üzerinde bir tatmin derecesi olarak ve Δ_i parametresi ise, merkezlerin tatmin derecesinin işlem (operation) bölümünün harcadığı taşıma maliyetindeki tatmin derecesine oranı olarak yorumlanabilir.

3°) *Çift kromozom dizili (double strings) genetik algoritmalar*

(3.32) ve (3.35) problemlerini çözmek için, problemlerin kısıtları lineer ve bütün $[A]$ teknoloji katsayıları pozitif ise, Skawa ve arkadaşları (1996, 1997) tarafından önerilen çift kromozom dizili genetik algoritmalar uygulanabilir.

i) *Kodlama ve dekodlama*

(3.32) ve (3.35) problemlerini çözmek için, birey genellikle bir 0-1 binary string ile temsil edilir. Bu gösteriliş, bununla birlikte, genetik algoritmaların kabiliyetini zayıflatabilir. Çünkü fenotipi uygun olan bir birey, bu gösteriliş ile nadiren üretilir. Bu makalede, sadece uygun (**feasible**) çözümleri üretmek için bir mümkün yaklaşım olarak, Şekil 3.9'da gösterildiği gibi bir çift kromozom dizisi, bir bireyi temsil etmek üzere benimsenmiştir. Burada $s_{i(j)} \in \{1,0\}$, $i(j) \in \{1,...,n_1 + ... + n_t\}$ ve $j \neq j'$ için $i(j) \neq i(j')$ dür.

Bir çift kromozom dizisi içerisinde, i(j) ve $s_{i(j)}$, sırasıyla bir çözüm vektöründe bir elemanın indisi ve o elemanın değeri olmak üzere,bir S kromozom dizisi $x_{i(j)} = s_{i(j)}$, $j = 1,...,n_1+...+ n_t$ olarak bir $x = \left(x_{11},...,x_{1n_1},...,x_{t1},...,x_{tn_t}\right)^T$ çözümüne transfer edilebilir.

Ne yazık ki, bu eşleme (fonksiyon) (**mapping**) uygun olmayan çözümler üretebileceği için uygun olmayan çözümleri elemek için aşağıdaki dekodlama algoritması önerilmektedir. Burada şu da belirtmelidir ki aşağıdaki algoritma sadece (3.32) problemi için uygun çözümler üretir. Fakat (3.35) problemi için her zaman sadece uygun çözümler üretmez. Algoritmada

$n_1+...+n_t$	kromozom dizisi uzunluğunu (boyunu)
j	kromozom dizisi içerisindeki bir pozisyonu
i(j)	bir değişkenin indisini
$x_{i(j)}$	kromozom dizisinden dekod edilmiş (çözülmüş) i(j) indisli bir değişkenin 0-1 değerini
$a_{i(j)}$	$[A_1,...,A_t]$ katsayı matrislerinin i(j) ninci sütun vektörünü

temsil etsin.

1. Adım: $j = 1$, $\Sigma = 0$ al.
2. Adım: Eğer $s_{i(j)} = 1$ ise $j = j+1$ al ve 3. adıma git. Değilse, yani $s_{i(j)} = 0$ ise, $j = j+1$ al ve 4. adıma git.
3. Adım: Eğer $\sum + a_{s(i)} \le b$ ise $x_{s(i)} = 1$, $\Sigma = \Sigma + a_{i(j)}$ al ve 4. adıma git. Aksi takdirde, $x_{i(j)} = 0$ al ve 4. adıma git.
4. Adım: Eğer $j > n_1+...+n_t$ ise dur ve $x = \left(x_1,...,x_{n_1+...+n_t}\right)^T$ yı çift kromozom dizisi ile temsil edilmiş bireyin fenotipi olarak dikkate al. Aksi takdirde, 2. adıma geri dön.

$$\begin{array}{ll} \text{Değişkenin indisi} & : \\ \text{0-1 değeri} & : \end{array} \begin{pmatrix} i(1) & i(2) & \dots & i(n_1+\dots+n_t) \\ s_{i(1)} & s_{i(2)} & \cdots & s_{i(n_1+\dots+n_t)} \end{pmatrix}$$

Şekil 3.9 Çift kromozom dizisi

ii) *Uygunluk ve Ölçekleme (Fitness and scaling)*

Çift stringli genetik algoritmalarla yeniden üretilmiş bir birey (3.35d) kısıtından dolayı her zaman (3.35) probleminin bütün kısıtlarını sağlamayabilir. Bu güçlüğün üstesinden gelebilmek için, kısıtın bozulması durumunda bir ceza olarak -1 değerini alacak bir uygunluk fonksiyonu benimsenmektedir. Her bir S bireyinin uygunluk fonksiyonu,

$$f(S) = \min\left\{ \min_{i=1,...,t} \mu_i\left(z_i\left(x, c_i^L\right)\right), \chi(x) \right\} \tag{3.37}$$

$$\chi(x)=\begin{cases} 1, & \text{eğer } \mu_i \geq \hat{\delta}_i',\ \forall i \in \{q,...,t-1\} \\ -1, & \text{aksi takdirde} \end{cases} \qquad (3.38)$$

olsun.

Burada S, çift string tarafından temsil edilen bir bireyi, x ise S nin fenotipini temsil etsin. (3.32) probleminde, her x ler için $\chi(x)=1$ tanımlansın, çünkü çift string tarafından temsil edilen bir birey için kod çözme algoritması daima bir uygun çözüm verir.

iii) *Üreme (Seçme) (Reproduction)*

Literatürde (Goldberg tarafından 1989'da yazılmış Genetic Algorithms In Search, Optimization, and Machine Learning isimli kitabında ve Michalewicz tarafından 1996'da yazılmış Genetic Algorithms + Data Structures = Evolution Programs isimli kitabında) , çeşitli üreme metodları önerilmiş ve ele alınmıştır. Burada, bir üreme işlemcisi (**Reproduction operator**) olarak elitçi (en iyi, güzeli seçen) rulet çarkı seçimi benimsenmiştir (**elitist roulette wheel selection**). Elitçi rulet çarkı seçimi, aşağıda bahsedildiği gibi, elitçilik ve rulet çarkı seçimlerinin bir birleşimidir.

Elitism: Geçmiş popülasyonda bir stringin uyumluluğu, mevcut popülasyonda her stringin uyumluluğundan daha büyükse, bu string mevcut jenerasyonda korunur.

Rulet Çarkı Seçimi (Roulette Whell Selection): Rulet çarkı seçimi, en popüler seçim çeşitidir. Bu yeniden üretim, uyumluluk değerlerine göre ölçeklenmiş yarıkları(slot) olan bir rulet çarkını kullanarak döller tahsis eder. Yarığın (Slot'un) büyüklüğü $f(S_i)/\sum f(S_i)$ olasılığıyla verilir.

iv) *Crossover and mutation ((Gen) Çaprazlama ve mutasyon))*

Eğer çift string ile temsil edilen bireylere bir tek-nokta veya çok-nokta çaprazlaması uygulanıyorsa, bir dölde $i(j)$ indisi, $i(j')$ $(j \neq j')$ indisinin aldığı ile aynı sayıyı alabilir. Aynı ihlalin, genetik algoritmalar yoluyla gezgin satıcı problemlerini veya çizelgeleme problemlerini çözerken olduğunu hatırlayalım. Böyle bir ihlalin üstesinden gelmek için muhtemel bir yaklaşım, kısmi eşleştirmeli çaprazlama (partially matched crossover) (PMX) olacaktır. PMX ilk defa Goldberg ve Lingle tarafından (1985) kör gezgin satıcı problemi ile uğraşırlarken ortaya atılmıştır. Bu yöntem, sıralamalı (ordinal) gösterimin aksine çift string yapısını değiştirmeksizin istenilen nesilleri üretme imkanı sağlar. Ne var ki, çift string

yapısındaki her bir $s_{i(j)}$ elemanını etkin bir şekilde kullanabilmek için, yöntemin bazı noktalarını revize etmek gerekmektedir. Skawa ve arkadaşlarının üzerinde değişiklik yaptığı PMX yöntemi aşağıdaki gibi gösterilebilir:

1. Adım : Çift stringlerle temsil edilen iki S_1 ve S_2 bireyi için iki çaprazlama noktası seç.

2. Adım : PMX'e göre, S_1 ve S_2 nin üst stringleri ile birlikte karşılık gelen alt stringlerini tekrar sırala, ki bu S_1' ve S_2' yü oluşturur.

3. Adım : S_1' ve S_2' nün iki çaprazlama noktası arasındaki daha alt seviyedeki alt stringleri değiştir. Bu, çift stringler için revize edilmiş PMX'den sonra, S_1'' ve S_2'' döllerini meydana getirir.

Genetik Algoritmalarda bir mutasyon operatörünün (local random search) yerel rassal (tesadüfi) arama görevi gördüğü iyi bilinmektedir. Bu makalede, çift stringin daha alt stringi için bit-döndürme tipli mutasyon benimsenmiş ve başka bir genetik operatör, bir ters çevirme (inversion), PMX operatörü ile birlikte sunulmuştur. İnversiyon aşağıdaki gibi olmaktadır:

1. Adım: Bir S bireyi için, rasgele iki inversiyon noktası seç, yani

$$S=\begin{pmatrix} i(1) \ \dots \ | \ i(l) \ \ i(l+1) \ \dots \ i(m) \ | \ \dots \ i(n_1+\dots+n_t) \\ s_{i(1)} \ \dots \ | \ s_{i(l)} \ \ s_{i(l+1)} \ \ \dots \ s_{i(m)} \ | \ \dots \ s_{i(n_1+\dots+n_t)} \end{pmatrix}$$

2. Adım: İki inversiyon noktası arasında hem üstteki hem de alttaki altstringleri tersine çevir, yani

$$S'=\begin{pmatrix} i(1) \ \dots \ | \ i(m) \ \ i(m-1) \ \dots \ i(l) \ | \ \dots \ i(n_1+\dots+n_t) \\ s_{i(1)} \ \dots \ | \ s_{i(m)} \ \ s_{i(m-1)} \ \ \dots \ s_{i(l)} \ | \ \dots \ s_{i(n_1+\dots+n_t)} \end{pmatrix}$$

4°) Çok-seviyeli 0-1 programlama problemi için sayısal bir örnek:

Çok-seviyeli 0-1 programlama problemi için bir örnek olarak, aşağıdaki üç-seviyeli problemi göz önüne alalım:

KV 1, Amaç: $z_1 = \tilde{c}_{11}x_1 + \tilde{c}_{12}x_2 + \tilde{c}_{13}x_3$ (3.39a)

KV 2, Amaç: $z_2 = \tilde{c}_{21}x_1 + \tilde{c}_{22}x_2 + \tilde{c}_{23}x_3$ (3.39b)

KV 3, Amaç: $z_3 = \tilde{c}_{31}x_1 + \tilde{c}_{32}x_2 + \tilde{c}_{33}x_3$ (3.39c)

Kısıtlar: $\tilde{A}_1x_1 + \tilde{A}_2x_2 + \tilde{A}_3x_3 \le \tilde{b}$ (3.39d)

$x_j \in \{0,1\}$, $j = 1,2,\dots,30$ (3.39e)

Burada $x_1 = (x_1,...,x_{10})^T$, $x_2 = (x_{11},...,x_{20})^T$, $x_3 = (x_{21},...,x_{30})^T$ dir; 10-boyutlu satır sabit vektörlerinin c_{ij}, i,j =1,2,3 her bir elemanı ve 3×10 luk A_1, A_2, A_3 katsayı matrislerinin her bir elemanı, (0,100) aralığında rasgele değerlerdir ve $\tilde{c}_{22} = -\tilde{c}_{12}$, $\tilde{c}_{33} = -\tilde{c}_{13}$ dür. Sağ taraftaki b sabit sütun vektörünün her bir elemanı, A_1, A_2 ve A_3'ün karşılık gelen satır vektörlerinin elemanlarının toplanıp 0.6 ile çarpılmış hali; katsayıların %90'ı bulanık parametreler (sayılar) olarak tayin edilmiştir. Çizelge 3.6 ve 3.7' de gösterilen değerler katsayıları temsil eden bulanık sayılar cinsindendir ve bunların yayılımları (spread), [1.0,1.1] aralığında rasgele sayılar ile ortalama değerlerin çarpılması ile elde edilmiştir.

Çizelge 3.6 Üç-seviyeli problemin katsayıları

$\tilde{c}_{11}$	-67	-73	-93	-24	-84	-96	-78	-43	-67	-81
$\tilde{c}_{12}$	-16	-28	-14	-86	-75	-21	-14	-30	-80	-22
$\tilde{c}_{13}$	-56	-71	-20	-98	-25	-43	-75	-86	-89	-97
$\tilde{c}_{21}$	-40	-43	-13	-46	-24	-98	-65	-60	-24	-46
$\tilde{c}_{22}$	16	28	14	86	75	21	14	30	80	22
$\tilde{c}_{23}$	-79	-8	-48	-16	-25	-94	-61	-98	-48	-80
$\tilde{c}_{31}$	-74	-38	-48	-53	-10	-59	-35	-15	-78	-71
$\tilde{c}_{32}$	-45	-70	-10	-96	-55	-74	-58	-64	-78	-19
$\tilde{c}_{33}$	56	71	20	98	25	43	75	86	89	97
$\tilde{A}_1$	51	18	31	53	94	18	70	23	49	13
	93	87	86	67	76	58	39	36	20	82
	44	91	57	12	57	25	50	24	48	41
$\tilde{A}_2$	9	39	28	37	98	54	76	65	76	78
	42	46	97	13	22	95	74	41	78	76
	87	43	36	38	5	16	52	69	10	40
$\tilde{A}_3$	82	16	62	32	35	91	52	40	61	78
	95	3	32	75	25	59	5	95	32	6
	77	25	34	23	30	31	88	4	65	40
$\tilde{b}$	917		993		757					

Not: Altı çizili değerler bulanık olmayan sayılardır.

Çizelge 3.7 ($\alpha = 0.9$) bulanık parametrelerin ekstrem (uç) noktaları

c_{11}^L	-69	-75	-97	-25	-88	-104	-84	-45	-68	-86
c_{12}^L	-16	-28	-14	-87	-76	-21	-14	-31	-84	-23
c_{13}^L	-57	-75	-21	-105	-27	-46	-75	-87	-90	-105
c_{21}^L	-40	-43	-13	-48	-25	-98	-69	-63	-24	-49
c_{22}^L	15	25	13	83	73	19	12	27	78	21
c_{23}^L	-82	-8	-49	-17	-25	-94	-61	-107	-52	-80
c_{31}^L	-75	-38	-48	-53	-10	-59	-35	-15	-83	-72
c_{32}^L	-45	-75	-10	-102	-57	-74	-62	-65	-84	-20
c_{33}^L	51	68	19	97	23	42	74	81	80	91
A_1^L	46	17	31	51	89	16	64	22	47	12
	93	78	85	61	70	53	36	33	18	75
	43	90	53	11	56	24	47	21	43	40
A_2^L	8	37	26	37	92	52	76	64	74	73
	40	43	89	11	20	85	69	39	78	74
	86	42	33	36	4	14	50	66	9	38
A_3^L	75	15	61	29	35	85	50	40	56	72
	95	2	31	69	24	54	4	87	31	5
	74	22	31	21	28	29	83	3	60	39
b^R	920		1035		785					

Diyelim ki *KV* 1 ve *KV* 2 başlangıç minimal tatmin seviyelerini $\delta_1 = \delta_2 = 1.0$ olarak, Δ_1 ve Δ_2'nin alt ve üst sınırlarını [0.6,1.0] olarak belirlesin. Bulanık hedeflerin (3.25) deki üyelik fonksiyonları, (3.23) ve (3.26) değerleri kullanılarak tayin edilir. Kısıtların bütün katsayıları pozitif olduğundan, teker teker minimumlar ve buna karşılık gelen optimal çözümler, çift stringli genetik algoritmalar uygulanarak elde edilir ve bunlar Çizelge 3.8'de gösterilmektedir. Sonuç olarak $z_1^m = -972$, $z_2^m = -85$, $z_3^m = 166$ dır. Her bir probleme yaklaşık bir optimal çözümü elde ederken yapılan hesaplamalarda, çaprazlama olasılığı 0.7, mutasyon olasılığı 0.05, popülasyon büyüklüğü 100 ve nesil sayısı da 1000 olarak alınmıştır ve bu hesaplama denemeleri 10 kez yapılmıştır. Bulanık hedefler için üç üyelik fonksiyonu:

$$\mu_1(z_1(x,c_{1.})) \underset{=}{\Delta} (z_1(x,c_{1.}^L)+972)/(-1439+972) \tag{3.40}$$

$$\mu_2(z_2(x,c_{2.})) \underset{=}{\Delta} (z_2(x,c_{2.}^L)+85)/(-971+85) \tag{3.41}$$

$$\mu_3(z_3(x,c_{3.})) \underset{=}{\Delta} (z_3(x,c_{3.}^L)-166)/(-1011-166) \tag{3.42}$$

olarak alınmıştır.

$\alpha = 0.9$ olsun ve bu sayısal örnek için (3.32) problemi aşağıdaki gibi ifade edilebilir:

Amaç: $$\max_{x,\lambda} \min \left(\mu_1(z_1(x,c_1)), \mu_2(z_2(x,c_2)), \mu_3(z_3(x,c_3))\right) \tag{3.43a}$$

Kısıtlar: $$A_1^L x_1 + A_2^L x_2 + A_3^L x_3 \le b^R \tag{3.43b}$$

$$x_j \in \{0,1\} \ , \ j=1,2,...,30 \tag{3.43c}$$

Çizelge 3.8 Teker teker minimumlar ve karşılık gelen çözümler

z_1^{min}	-1439	x_1^{1o}	1	1	1	0	1	1	1	1	1	1
		x_2^{1o}	0	0	0	1	1	0	0	0	1	0
		x_3^{1o}	0	1	0	1	0	0	1	1	1	1
z_2^{min}	-971	x_1^{2o}	1	1	0	1	0	1	1	1	1	1
		x_2^{2o}	0	0	0	0	0	0	0	0	0	0
		x_3^{2o}	1	0	1	1	1	1	1	1	1	1
z_3^{min}	-1011	x_1^{3o}	1	1	1	1	0	1	1	1	1	1
		x_2^{3o}	1	1	0	1	1	1	1	1	1	0
		x_3^{3o}	0	0	0	0	0	0	0	0	0	0

Burada c_{ij}^L, A_i^L, b^R ler Çizelge 3.7 'de gösterilmiştir. Uygunluk fonksiyonu (3.37) olan ve $\chi(x) = 1$ olan çift stringli Genetik Algoritma uygulanmıştır. (3.43) probleminin yaklaşık bir optimal çözümünü içeren ilk iterasyonun verileri Çizelge 3.9 'da verilmiştir.

Çizelge 3.9 Üç-seviyeli problemin ilk iterasyonu

λ^1	0.537473									
x_1^1	1	1	1	1	1	1	1	1	1	1
x_2^1	0	1	0	1	1	1	0	0	0	0
x_3^1	0	0	1	0	1	0	1	1	0	1
z_1^1	-1223		$\mu_1(z_1^1)$		0.537473					
z_2^1	-564		$\mu_2(z_2^1)$		0.540632					
z_3^1	-476		$\mu_3(z_3^1)$		0.545455					
Δ_1^1	1.005877									
Δ_2^1	1.008920									

Etkileşim sürecinin birinci sona erme koşulu sağlanmamıştır çünkü *KV* 2'in tatmin derecesi $\mu_2^1 = 0.540632$, minimal tatmin derecesi $\hat{\delta}_2 = 1.0$'ı aşmamıştır. Buna bağlı olarak, *KV* 2 nin minimal tatmin derecesini $\hat{\delta}_2 = 1.0$'den $\hat{\delta}_2' = 0.9$'a güncelleştirdiği kabul edilsin. O zaman, (3.35) a karşılık gelen problem,

Amaç: $$\max_{x,\lambda} \min \left(\mu_1(z_1(x,c_1)), \mu_3(z_3(x,c_3))\right) \tag{3.44a}$$

Kısıtlar: $$\mu_2(z_2(x,c_2)) \geq 0.9 \tag{3.44b}$$

$$A_1^L x_1 + A_2^L x_2 + A_3^L x_3 \leq b^R \tag{3.44c}$$

$$x_j \in \{0,1\} \ , \quad j=1,2,...,30 \tag{3.44d}$$

şeklinde formüle edilebilir ve (3.37) ve (3.38) uygunluk fonksiyonları ile çift stringli olan genetik algoritmalar kullanılarak çözülür. (3.44) probleminin bir optimal çözümünü içeren ikinci iterasyonun verileri Çizelge 3.10 'da verilmiştir.

Çizelge 3.10 Üç-seviyeli problemin ikinci iterasyonu

z_1^2	1095.9	$\mu_1(z_1^2)$	0.263383
z_2^2	884.7	$\mu_2(z_2^2)$	0.901806
z_3^2	99.9	$\mu_3(z_3^2)$	0.225149
Δ_1^2	3.423930		
Δ_2^2	0.249664		

Etkileşim sürecinin birinci sona erme koşulu sağlanmamıştır. Çünkü *KV* 1'in tatmin derecesi $\mu_1^2 = 0.263383$, minimal tatmin seviyesi $\hat{\delta}_1 = 1.0$'ı aşmamıştır. Sonuç olarak, *KV* 1'in minimal tatmin seviyesi $\hat{\delta}_1 = 1.0$'dan $\hat{\delta}_1' = 0.9$'a değiştirdiği kabul edilsin. O zaman, aşağıdaki problem formüle edilebilir.

Amaç: $$\max_{x,\lambda} \mu_3(z_3(x,c_3)) \quad (3.45a)$$

Kısıtlar: $$\mu_1(z_1(x,c_1)) \geq 0.9 \quad (3.45b)$$

$$\mu_2(z_2(x,c_2)) \geq 0.9 \quad (3.45c)$$

$$A_1^L x_1 + A_2^L x_2 + A_3^L x_3 \leq b^R \quad (3.45d)$$

$$x_j \in \{0,1\} \ , \ j=1,2,...,30 \quad (3.45e)$$

(3.45) probleminin uygun bir çözümü olmadığından, minimal tatmin seviyeleri $\hat{\delta}_1'$ ve $\hat{\delta}_2'$ lerin 0.6 olarak güncellendiğini varsayalım. O zaman karşımıza çıkan problemi şu şekilde ifade edebiliriz:

Amaç: $$\max_{x,\lambda} \mu_3(z_3(x,c_3)) \quad (3.46a)$$

Kısıtlar: $$\mu_1(z_1(x,c_1)) \geq 0.6 \quad (3.46b)$$

$$\mu_2(z_2(x,c_2)) \geq 0.6 \quad (3.46c)$$

$$A_1^L x_1 + A_2^L x_2 + A_3^L x_3 \leq b^R \quad (3.46d)$$

$$x_j \in \{0,1\} \ , \ j=1,2,...,30 \quad (3.46e)$$

Çizelge 3.11 'de (3.46) probleminin optimal çözümünü içeren üçüncü iterasyonun dataları verilmiştir.

Çizelge 3.11 Üç seviyeli problemin üçüncü iterasyonu

z_1^3	1253.9	$\mu_1(z_1^3)$	0.601713
z_2^3	621.4	$\mu_2(z_2^3)$	0.604966
z_3^3	233.2	$\mu_3(z_3^3)$	0.338997
Δ_1^3	1.005406		
Δ_2^3	0.560358		

Minimal tatmin dereceleri oranı $\Delta_2^3 = 0.560358$ ve $\Delta_1^3 = 1.005406$, Δ_1 ve Δ_2 nin [0.6,1.0] geçerli aralığında olmadığından, minimal tatmin seviyelerinin $\hat{\delta}_1' = 0.7$ ve $\hat{\delta}_2' = 0.5$ olarak güncellendiklerini varsayalım.

O zaman aşağıdaki problem formüle edilebilir:

Amaç: $$\max_{x,\lambda} \mu_3(z_3(x,c_{3.})) \quad (3.47a)$$

Kısıtlar: $$\mu_1(z_1(x,c_{1.})) \geq 0.7 \quad (3.47b)$$

$$\mu_2(z_2(x,c_{2.})) \geq 0.5 \quad (3.47c)$$

$$A_1^L x_1 + A_2^L x_2 + A_3^L x_3 \leq b^R \quad (3.47d)$$

$$x_j \in \{0,1\} \ , \ j=1,2,...,30 \quad (3.47e)$$

Çizelge 3.12'de (3.47) probleminin yaklaşık optimal çözümünü içeren dördüncü iterasyonun verileri verilmiştir.

Çizelge 3.12 Üç seviyeli problemin optimal çözümü

x_1^4	1	1	1	0	1	1	1	1	1	1
x_2^4	0	1	0	1	0	1	0	0	1	0
x_3^4	0	1	0	0	1	1	1	1	0	1
z_1^4	-1299.7		$\mu_1(z_1^4)$		0.700214					
z_2^4	-567.0		$\mu_2(z_2^4)$		0.542889					
z_3^4	-353.2		$\mu_3(z_3^4)$		0.440952					
Δ_1^4	0.775319									
Δ_2^4	0.812231									

Dördüncü adımda, *KV* 1 in tatmin derecesi $\mu_1(z_1^4) = 0.700214$, minimal tatmin seviyesi $\hat{\delta}_1' = 0.7$ den daha büyüktür ve *KV* 2 nin tatmin derecesi $\mu_2(z_2^4) = 0.542889$ minimal tatmin seviyesi $\hat{\delta}_2' = 0.5$ daha büyüktür. Tatmin olma derecelerinin $\Delta_1^4 = 0.775319$ ve $\Delta_2^4 = 0.812231$ oranları Δ_1 ve Δ_2 oranlarının geçerli [0.6,1.0] aralığındadır. Bu yüzden çözüm, etkileşimli sürecin bitme koşullarını sağlamaktadır ve bu çözüm bütün KVler için tatmin edici bir çözüm olur.

Sonuç:

Bu çalışmada, bulanık parametreli çok-seviyeli 0-1 programlama problemi için etkileşimli bulanık programlama sunulmuştur. Etkileşimli metotta, bütün seviyelerdeki *KV* 'lerin bulanık hedefleri belirlendikten sonra, genel tatmin dengesi göz önüne alınarak minimal tatmin seviyelerinin güncellenmesi vasıtasıyla tatmin edici çözüm, etkin bir şekilde bulunmuştur. Önerilen metodun kullanışlılığını gösterebilmek amacıyla üç seviyeli 0-1 programlama problemi olarak sayısal bir örnek de verilmiştir.

19. **Bauer, R.J.**'nin **Genetic Algorithms And Investment Strategies** (1994,s.55-72) adlı kitabının ikinci bölümünde, basit bir örnek problem kullanılarak G.A.'nın temel unsurları açıklanmıştır.

Örnek Problem

Bu örnek problemde, bir kitap yayıncı kuruluşun, JWI, maliyetleri en aza indirme problemi ele alınmıştır. "Türlerin Yokolmasının Önlenmesi" başlıklı kitabın yıllık satışı 20 000 kopya olarak tahmin edilmektedir. Ayrıca satış oranının, belirli bir mevsimsel değişim olmaksızın yıl boyunca aynı seviyede olacağı beklenmektedir. JWI, geçmişte, benzer durumlarda dört baskı yapmıştır. Öyleyse birinci alternatif, 5000 kopyalık dört baskı yapmaktır. Fakat yönetimin baskı ve dağıtım maliyetleri üzerinde kaygıları vardır.

JWI'ın maliyetlerinin sabit ve değişken unsurları vardır. Her bir baskı yeni bir ekipman kurulumu gerektirir. Bu bir önceki baskının temizlenip, yeni baskı için dizin oluşturulmasıdır. Her bir baskı kurulumu maliyeti 6000 dolardır. Bu her baskı için sabit giderdir. Basıldıktan sonra kitaplar, dağıtılmadan önce depolanmalıdır. Daha büyük baskılar, baskı sayısını azaltır ama depolama alanını büyütür. Depolama ve dağıtım maliyetleri ortalama envanter seviyesiyle kabaca orantılıdır, ortalama olarak her baskının yarısı kadar olacaktır. Yönetim, her baskıda kaç kitap basılarak maliyetlerin en aza, kârın en yükseğe çekilebileceğini bilmek istemektedir.

Böyle bir optimizasyon problemini çözmek için birçok metod vardır. Bu problemlere klasik ekonomik sipariş sayısı problemi (Economic Order Quantity) (EOQ) denir. Böylece bir çözüm metodu, uygun EOQ formülünü kullanarak, her bir baskının optimal büyüklüğünü doğrudan hesaplamaktır. Mümkün olabilecek ikinci bir çözüm metodu, "Deneme Yanılma" (trial-and-error) dır. Toplam maliyetler, optimal baskı büyüklüğünün değişik tahminleri ile hesaplanabilir. Daha büyük baskılar toplam baskı kurulum maliyetlerini azaltır, fakat depolama ve dağıtım maliyetlerini arttırır. Bu arada daha küçük baskılar tam tersini yapar. Başarılı deneme yanılma tahminleri en iyi seviyeye odaklanmak için kullanılabilir. Bu da, bu iki maliyet unsurunu dengeler. Üçüncü bir yaklaşım ise bir Genetik Algoritma'nın iyi bir sonuç bulmasına izin vermektir. "İyi bir sonuç" deyimi bilerek kullanılmıştır. G.A.'lar en iyiye yakın çözümleri bulacaktır.

Yukarıda anlatılan üç çözüm metodundan, EOQ formülü açıkça en iyisidir. Deneme-Yanılma yaklaşımı da oldukça iyi çalışır. Özellikle de Spreadsheet programı (satışlar, vergiler, net kazanç hesaplamaları yapmak için rakamları ekranda gruplar halinde gösteren bir bilgisayar

programı) kullanılırsa. Bu metodlar içinde G.A. kullanımı en kötüsüdür. Çünkü G.A.'ların genellikle daha büyük, daha karmaşık problemlere uygulanması daha uygundur. Fakat bilinen bir çözümü olan küçük bir problem kullanarak G.A. hakkında bilgi sahibi olmak daha kolaydır.

Örnek problem için EOQ formülü şöyledir:

$$Q^* = \sqrt{\frac{2FD}{V}} \tag{3.48}$$

Burada,

F: sabit kurulum maliyeti (6000$)

V: değişken maliyeti (6$ birim başına)

D: yıllık talep (20 000 birim)

dir.

Değişken maliyet, basılmış kitapların depolama maliyetidir. Çözüm her baskı için 6325 kitaptır.

"Türlerin yok olmasının önlenmesi" kitabının satış fiyatı 30$ dır. Yukarıda açıklanan maliyetlere ek olarak, JWI, ofis harcamaları, mürekkep, kağıt ve telif hakkı maliyetlerini de gözönüne almalıdır. Bunlar yıllık 350.000$ tutmaktadır. Bu ek bilgileride kullanarak Deneme-Yanılma Metodu ile, karın 6200 ile 6400 kitap aralığında en üst seviyeye çıktığı görülmüştür. Küçük baskılar sabit kurulum maliyetlerini arttırırken, büyük baskılar depolama maliyetlerini arttırır. Toplam sabit kurulum ve değişken depolama maliyetleri bir baskıda 6300 kitap basımıyla en aza iner. Önceki hesaplamalardan en uygun baskı sayısının 6325 kitap olduğu bilinmektedir.

Genetik Algoritma Çözümü

Burada, belirli bir G.A. çözümünden değil, bir G.A. çözümünden bahsediyoruz. Bunun iki nedeni vardır. İlki kullanılabilecek birçok G.A. Metodoloji varyasyonu vardır. İkincisi, verilen örnek problemi çözmeye girişmek için, kullanılabilecek diğer problem sunumları vardır.

Aşağıdaki adımlar bir "Basit" bir G.A.'yı tarif eder. Bunun temel özellikleri diğer birçok G.A.'da da bulunur.

1. *Problem Sunumunu Seçme:* G.A. kullanımındaki aşılacak ilk engel problem sunumudur. Problemimizi G.A.'nın ele alabileceği, uygun bir şekilde tanıtmalıyız. Çünkü G.A., yapısı kromozomları andıran sembol şeritleri (string) ile çalışır. Sunum çoğu zaman ikili kodlama

şeklinde olur. Tipik bir örnek şöyle görülür: 0010101011. 0 ya da 1 kullanan 10 parçalı parçalar. Şerit (string), olası bir problem çözümünü ifade eder.

G.A.'lar, problem için herbiri potansiyel bir çözüm ifade eden ve birbirleri arasında yarışan şeritlerin bir topluluğuyla (nüfus) çalışır. Topluluktaki birey şeritler, biyolojik bazlı operasyonlar kullanılarak yavaş yavaş dönüştürülür. Örneğin, iki şerit çiftleşebilir ve ebeveynlerinin en iyi özelliklerini birleştiren bir yavru üretebilir. En iyinin yaşaması kuralı ile nüfusun en iyi performanslı bireyi nüfusu yönetir.

Örnek problemde, kitap baskısının büyüklüğü optimize edilmeye çalışılmıştır. Bu değişken S olarak alınmıştır. Amaç, toplam kurulum ve depolama maliyetlerini en aza indirecek ve bu yolla karı en üst seviyeye çıkaracak bir S değerini bulmaktır. Yıllık S değeri maksimum 20000 dir. Bu 20000 kitabı bir tek seferde basmak demektir. Optimizasyon probleminde S tek değişkendir.

Bu problemin en doğrudan ikili sayı kodlaması, S değerinin bir ikili sayı şeridi olarak sunulmasıdır. Örneğin ikili sayı 110'un ondalık bir dengi vardır. (1x2^2+^1x2^1+0x2^0). Maksimum S değeri 20000 olduğu için ikili sayı sunumu birçok parça gerektirir. Eğer 14 bit kullanılacaksa maksimum ikili sayı 14 birden oluşan bir şerit olacaktır: 11111111111111. Bu ikili sayının dengini bulmak için, 2 nin 14. kuvveti alınıp 1 çıkartılır. Bu 16384 e tekabül etmektedir. 15 bit ile maksimum ondalıklı eşdeğeri, 2'nin 15. kuvveti eksi 1 dir. Yani 32768 dir. Burada bir seçim yapılması gerekmektedir. Eğer 14 bit kullanılırsa, S değerlerinin tüm aralığına ulaşılamamakta, fakat eğer 15 bit kullanılırsa maksimum değerin üstüne çıkılmaktadır. İşlemleri daha kolay ele alınabilir hale getirmek için 14 kullanılmıştır. (En iyi değer 6325 olarak hesaplandığı halde bu bilinmiyormuş gibi davranılmıştır) Ayrıca en iyi değerin 1 ile 16384 aralığında olduğunu kabul etmek büyük bir ihtimalle akıllıca olacaktır.

2. Popülasyonu Başlatma: Nüfus büyüklüğünü belirlemede katı kurallar yoktur. G.A. araştırmasında 100-200 aralığındaki nüfus büyüklükleri yaygındır. Aşağıda açıklanan adımlarla nüfus ortak bir noktaya yakınsar. "Yakınsaklık", nüfustaki çoğu ya da bütün şeritlerin birbirinin aynı olması demektir. Yakınsaklık tanımları, bu örneğin ele alındığı kaynakta tartışılmıştır. Kullanıcılar, nüfus büyüklüğü ile ilgili farklı değerlerin yakınsaklığı ve sonuç kalitesini nasıl etkilediğini görmek için sık sık nüfus büyüklüğü denemeleri yaparlar.

Nüfus büyüdükçe, daha büyük çeşitlilik sağlanır. Fakat daha fazla bilgisayar kaynağı gerektirir. Bu örnekte 20'lik bir nüfus büyüklüğü kullanılmıştır. Bu çok küçüktür. Fakat bir nesilden bir sonrakine G.A. performansını incelemeyi kolaylaştırmaya yardım edecektir.

Nüfus büyüklüğü seçildikten sonra, başlangıç nüfusu rastgele üretilir. G.A.'nın değişik adımları için gereken rastgele seçimler bilgisayarlar kullanılarak kolaylıkla yapılır. Çünkü nüfusta 14 bitlik 20 şerit vardır, 280 bit, ya 0 ya da 1 olmalıdır. Bilgisayar 280 bit pozisyon değerini, Yazı-Tura ile belirler. Yaklaşık olarak 140 bitin 1, 140'ında 0 olacağı umulmaktadır. Böyle bir sürecin gerçek sonuçları Çizelge 3.13'de dir.

Çizelge 3.13 Başlangıç Nüfusu

Şerit Sayısı	Şerit
1	00100010011010
2	00101011001000
3	00000010100011
4	00110001010100
5	00111100110101
6	10111001101001
7	10011011001111
8	01100101001100
9	11001010111000
10	11001111101100
11	00010011000001
12	11011010101010
13	10010011010010
14	10001100000000
15	11011001000111
16	11010001011111
17	00001000111011
18	11100011011000
19	11001111110010
20	11000011111011

Başlangıç nüfusu genellikle tarif ettiğimiz gibi oluşturulmuştur. Fakat bazı istisnalar vardır. Bazen başlangıç nüfusunu "tohumlamak" istenebilir; bu G.A. varyasyonu bu örnekte ele alınmayacaktır. Başlatma adımında problem kısıtları da belirlenebilir. Örneğin, 15 bitlik bir sunum seçilseydi, bazı şeritlerin ondalık dengi yıllık talep olan 20000'i aşabilirdi. Bu değerler bir problem yaratmazdı. Çünkü G.A, iyi olmayan çözümleri çok çabuk bir şekilde dışarı atar. Buna rağmen şerit değerleri başlatma adımında sınırlandırılabilirdi. Ondalık denkleri 20000'in üzerindeki her şerit atılabilir ve kısıtları karşılayan rasgele seçilmiş herhangi bir şeritle değiştirilebilirdi.

3. Uygunluğu Hesaplama: Şimdi potansiyel problem çözüm nüfusumuz var ve onların ne kadar iyi olduğunu görmemiz gerekmektedir. Öyleyse herbir şerit için bir uygunluk veya performans hesaplanır. Bu, örnek problemimizdeki gibi çabuk ve basit olabilir; ya da kompleks ve zaman alan bir işlem olabilir.

Basım maliyeti problemimiz için, uygunluk hesaplaması tekdüzedir. Her bir şerit, ondalık dengine ters kodlanmış (decode edilmiş), bu bize baskı işinin optimal büyüklüğü için aday değer verir. Sonra daha önce tartışılan değerler kullanılarak, toplam maliyet ve elde edilecek kâr hesaplanmıştır. Kâr değeri, gözönüne alınan şeridin uygunluk değeridir. Başlangıç nüfusu için sonuçlar Çizelge 3.14'te gösterilmiştir. Nüfusun daha sonraki jenerasyonlarda (nesillerde) uygunluk hesaplamasına geri döneceğiz.

Çizelge 3.14 Başlangıç Nüfusu (Uygunluk Hesabı İle)

Şerit Sayısı	Şerit	Ondalık Değer	Uygunluk
1	00100010011010	2.202	188.898
2	00101011001000	2.760	198.242
3	00000010100011	163	-486.685
4	00110001010100	3.156	202.509
5	00111100110101	3.893	207.496
6	10111001101001	11.881	204.257
7	10011011001111	9.935	208.116
8	01100101001100	6.476	212.042
9	11001010111000	12.984	201.806
10	11001111101100	13.292	201.096
11	00010011000001	1.217	147.746
12	11011010101010	13.994	199.443
13	10010011010010	9.426	208.991
14	10001100000000	8.960	209.727
15	11011001000111	13.895	199.679
16	11010001011111	13.407	200.828
17	00001000111011	571	38.129
18	11100011011000	14.552	198.098
19	11001111110010	13.298	201.082
20	11000011111011	12.539	202. 813

4. *Seçim Yapma:* G.A.'lar başarılı nesiller üzerinden evrim geçiren, yarışan problem çözümleri nüfusu ile çalışır. En uygunun yaşaması, nüfusun en iyi performans gösteren bireyinin uzun vadede yaşaması demektir. Kısa vadede eşitsizlikleri (odds), nadiren daha iyi

performans gösterenlerin yararına kullanırız; ama zayıf performans gösterenlerin tümünü elimine etmeyiz. Bu birçok şekilde yapılabilir.

Basit bir seçim şeması kullanalım. İlk adım, bir önceki adımda hesaplanan uygunluklarına göre, yarışan şeritlere, bir sıralama (rütbeleme) (ranking) üretmektir. Böylece en kötü performans gösteren şeriti, yani 20'nin en kötüsünü, en iyinin bir kopyasıyla değiştirilir. Böylece şimdi bir önceki neslin bir şerit dışında aynısı olan bir nesille kalınır; böylece 20'lik bir nüfusta en iyi şeridin iki kopyası oluşur.

Başlangıç nüfusunun yaratılması sırasında, rastgele başlangıcın şanslı olduğu ve optimal çözüme karşılık gelen bir string'i (şeriti) ürettiği varsayılmıştır. Bu optimal string'e "Optie" diyelim. Takip eden adımlarda Optie çaprazlama ve mutasyonla değiştirilecektir. Fakat eğer Optie hiç değişmeden yaşamaya devam ederse, her nesilde en kötü şeridin yerine geçecek şekilde kopyalanır. Çünkü her nesilde bir şerit değiştirilir ve biliyoruz ki nüfusun yakınsaklığı için en azından 20 nesil (jenerasyon) gerekir.

Olması daha muhtemel bir senaryo yakınsaklığın 20'den daha çok nesilde olmasıdır. Örneğin bir şerit, 10 jenerasyon için nüfusa hakim olabilir ve daha sonra başka bir şerit tarafından yerinden edilebilir. Nihai kazanan şerit, nüfusun tüm diğer üyelerini değiştirmeden önce en azından 20 nesil yaşamalıdır.

Eğer seçim sakar ve dikkatsiz bir şekilde yapılırsa, nüfus çabucak yakınsar. En kötü beş şeridi seçip, en iyinin beş kopyasıyla değiştirilebilir. Bu yaklaşımla ve Optie olan bir başlangıç nüfusu ile dört nesilde bir yakınsamaya ulaşılabilir ve eğer yakınsama çabuk olursa çözüm kalitesinden vazgeçmemiz gerekebilir. Bu prematüre yakınsaklık problemi olarak bilinmektedir. Çabuk yakınsaklıkta, G.A. uzayı keşfetmek için yeterli zaman bulamayabilir.

5. Çaprazlama Yapma: Çaprazlama (Geçişme) (Crossover), G.A.'ya güç veren adımdır. Araştırmanın çok iyi sonuçlar bulmak amacıyla farklı yönlerde dağılmasına yol açar ve iki şeritin çiftleşmesine izin verir. Bu ana babadan daha uyumlu olan bir yavru (döl) ile sonuçlanır.

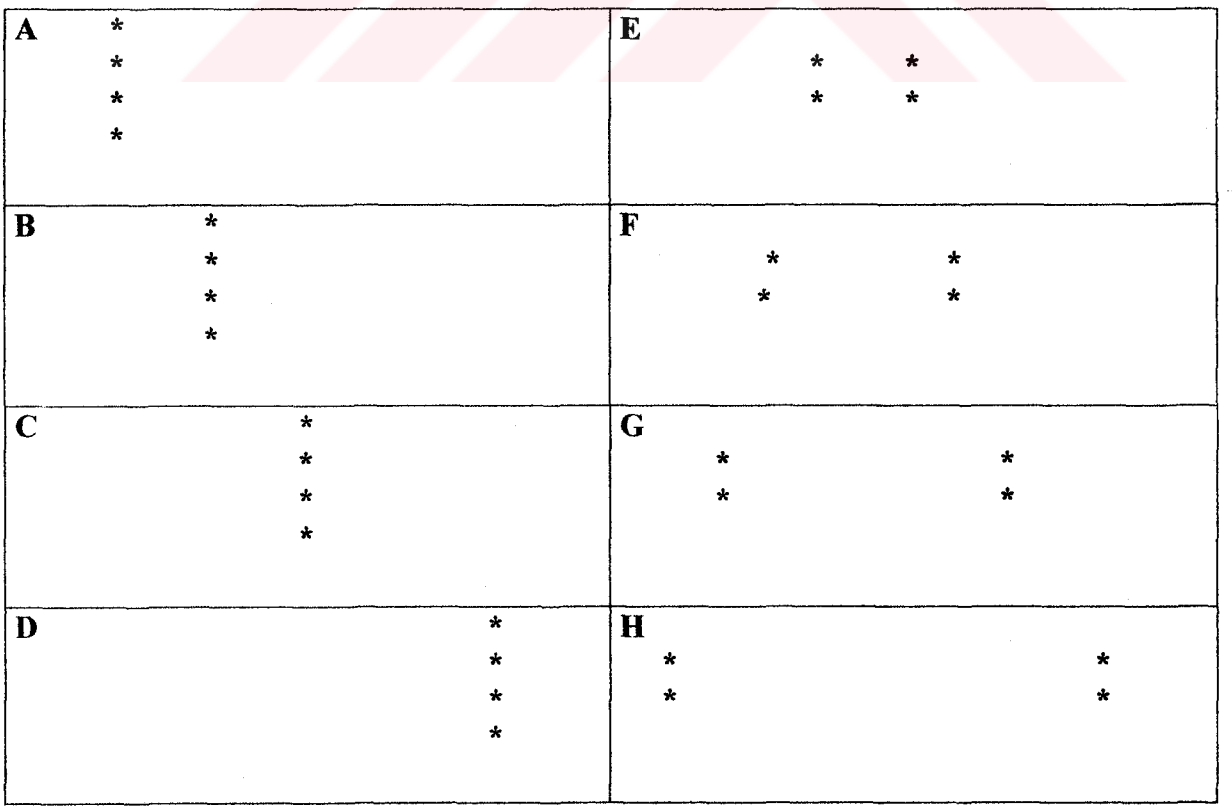
Çaprazlama dört küçük adımda gerçekleşir. İlk olarak, iki potansiyel ebeveyn, rastgele nüfus içinden seçilir. İkinci olarak, ağırlıklı ve kompüterize bir yazı-tura atılır ve geçişmenin olup olmayacağı belirlenir. Yazı-tura 10 da 6 ihtimalle "Evet geçişmeyi başlatın" diyecek şekilde hükümlendirilmiştir. Üçüncü olarak, cevabın evet olduğunu varsayarsak, rastgele bir ek noktası seçilir. Dördüncü olarak da, her bir şerit ek yeri noktasından kesilir. Örneğin 6. ve 7. bit arasından keserek, her bir şerit için iki parça üretilir. Birinci parça 1'den 6'ya, ikinci parça

7'den 14'e kadarki bitlerden oluşur. Son olarak şeritlerin iki ucunu değiştirerek yeniden birleştiririz. A şeridi B'nin ucuna, B de A'nınkine sahip olur.

Daha önceki araştırmalarda 0.6'lık bir geçişme (çaprazlama) oranının etkili olduğu görülmüştür, fakat bunun mümkün olan tüm G.A. uygulamalarında en iyi olduğuna dair bir kanıt yoktur.

Geçişme G.A.'nın arkasındaki gerçek güçtür. Çünkü uzayın araştırılmasında çok etkili bir tarama sağlar. Bunu resimlemek güç olabilir. Aşağıdaki kıyaslamanın yardımı olabilir.

Bir yer altı petrol rezervini kapladığından neredeyse emin olduğumuz çok geniş bir arazi düşünün. Bu alandaki herhangibir noktada keşif amaçlı bir kuyu delebiliriz. Başlangıçta tüm delme noktaları aynı ölçüde çekicidir. Şimdi de aynı anda dört kuyu delen keşif amaçlı bir kuyu delme programı düşünün. Mümkün olan sonsuz sayıda dört kuyu seçeneği vardır. Eğer dört adet dörtlü kuyu deleceksek, Şekil 3.10'da görüldüğü ABCD ve EFGH şeklinde ilerleyebiliriz. Birçoğumuz akıllı bir yolda ilerler ve sayılamayacak kadar başka ihtimaller vardır. Geçişme ise çözümü, farklı mantıklı yollarda ileri geri götürür. Petrol kuyusu örneğinde G.A. geçişme yaklaşımı BEHD ve ABEG sınırlamasını seçebilir. Bu mantıksız ve rastgele olarak görülebilir. Fakat farklı yollarla deneyler yapmanın meyve verebileceği ihtimali çok yüksektir.



Şekil 3.10 (Petrol Kuyusu Delme Şekilleri)

Geçişme, gücünün büyük bir bölümünü seçme prosedüründen alır. Dördüncü adımda anlatılan seçim nüfusun daha iyi performanslı üyeleri yararına çalışan yaşam faktörlerini itekler. Belirli bir problemde 5'den 8'e parça pozisyonundaki 1011 yolunun çok etkili çözümlere yol açtığını söyleyelim. Aynı zamanda 1'den 4'e kadar parça pozisyonundaki 1100 yolununda etkili olduğunu varsayalım. Eğer belirli bir şerit, y şeridi , 5'ten 8'e 1011 yoluna sahipse, yüksek dereceli bir uygunluğa sahiptir ve üreme şansıda o derece yüksektir. Eğer iki şeridi 1'den 4'e 1100 yoluna sahipse, benzer sonuçlar ona da uygulanır. Eğer bu iki şerit de üreyecekse, y ve 2'nin geçişme ile yaratılan çocuğunun 10111100 yoluna 1'den 8. parçada sahip olma şansı oldukça artar (Burada parça pozisyonu sağdan sola numaralanır). Ebeveynlerden iyi karakteristikler alan bu yavru süper-uygundur.

6. Mutasyonu Sağlama: Mutasyon nüfusa rastgele sapmaları tanıtır. Mutasyon 0'ı 1'e; 1'i 0'a çevirir. Nüfustaki her üyenin her bit (parça) pozisyonu kontrol edilir. Mutasyonun olup olmayacağına bilgisayar rastgele karar verir. Mutasyon genellikle düşük olabilirlikle uygulanır. Tersi olursa, seçim ve geçişme ile üretilen sıralama yapısını bozar. Mutasyon nüfusu daha iyi bir yola iter.

Örnek problemde 0.001 olabilirlikle mutasyon uygulanmıştır. Bazı ileri G.A. varyasyonlarında, mutasyon oranı başarılı nesiller üzerinde değişmektedir.

7. Yakınsaklığı Kontrol Etme: Yakınsaklık sık sık "eğilim" (bias) kavramı kullanılarak ölçülür ve popülasyonlar arasında bir anlaşma ölçüsü olarak tanımlanır. Eğilim %100 ve %50 arasındaki değerler olarak kabul edilir. Örnek olarak nüfustaki tüm üyelerin parça pozisyonunun 9 olduğunu düşünülmüştür. Eğer üyelerden onunda 9. pozisyon 1 ise diğer onunda 0'dır. Böylece oran 50-50 ve "bit eğilimi" %50; oran 75-25 ise (15 adet için 1; 5 adet için 0) eğilim %75; oran 25-75 ise eğilim yine %75 tir. Bit eğiliminden, şerit eğilimine geçeriz. Örnekte, her şeritte 14 bit vardır. Şerit eğilimi, 14 bitlik eğilimi değerinin ortalamasıdır. Bundan sonra eğilime ait tüm referanslar, ortalama string eğilimi olacaktır ve bu da tüm popülasyonun yakınsaklığının bir özet ölçüsü olarak hizmet görecektir.

Yakınsaklık oranı seçim prosedüründen etkilenir. Seçim eğer sakar ve dikkatsizce yapılmışsa nüfus çabucak yakınsar. Birçok G.A. varyasyonları için, yakınsaklık oranı, bir nesilden diğerine derece derece azalır. Genelde G.A. araştırmasının azalan getirileri (diminishing returns) vardır.

Yakınsaklık, bir şekilde bakana göre değişir. %95 eğilimli bir nesil oldukça üniformdur. Mutasyon, varolan sıralamayı rahatsız ettiği için yüksek mutasyon oranı, nüfusun tam

anlamıyla yakınsaklığını zorlaştırır. Farklı kullanıcılar, yakınsaklığı neyin oluşturduğu hakkında farklı fikirlere sahip olmaktadırlar.

Şimdi bu temel adımların sonuna geldik. Yakınsaklık ölçüsünü hesapladıktan sonra, eğilim gibi, iki mümkün uygulama yolumuz vardır. G.A.'yı durdurabiliriz ya da üçüncü adıma dönüp devam edebiliriz. Eğer yakınsaklığa ulaştıysak nüfusu inceleyebiliriz. Bu da genel görüşe göre problemimizin çözümüdür. Eğer yakınsaklığa ulaşmadıysak üçüncü adıma dönerek nüfusun evrime devam etmesine izin veririz.

Eğer G.A.'nın yakınsaması sonsuza kadar sürecek gibi görünüyorsa ne olur?. Alternatif bir sonlandırma kriteri de maksimum jenerasyon (nesil) sayısıdır. Örneğin, eğer eğilim %95'in üzerinde ise ya da eğer G.A. 1000 nesilden fazla çalışıyorsa G.A. durdurulabilir. Eğer yakınsaklık çok yavaşsa seçim prosedürünü, geçişme oranını, mutasyon oranını ya da bu üçünün herhangi bir kombinasyonunu ayarlamamız gerekebilir.

Prematüre yakınsamanın maliyeti optimalden uzak bir çözüm olabilir. Büyük zor problemler genellikle bazı uzlaşmalar gerektirebilir. Çabuk cevap isteyen bir kullanıcının optimal olmayan bir sonuçla yetinmesi gerekebilir; daha yüksek kaliteli bir sonuç isteyen kullanıcı ise daha sabırlı olmak zorundadır.

Örnek Problem İçin G.A. Sonuçları

Yukarıda açıklanan G.A.'yı çalıştırdığımızda; Çizelge 3.14, her şerit değerinin ondalık dengi ile başlangıç nüfusunu gösterir. Bu S optimal değerinin tahminini simgelemektedir. Uygunluk değeri her S değeri için kar miktarıdır. Başlangıç nüfusu 163'ten 14552'ye kadar giden tahminlerden oluşur. 8. şerit, 6476 değeri ile en iyi tahmindir.

Çizelge 3.15 , 1. nesilden 2. nesile geçişin sonuçlarını gösterir. Nüfustaki yavaş değişim, bazı detaylarda görülebilir. Bu nesildeki tahminler 560'tan 15943'e kadar gider. 1. nesildeki 8. şerit kopyalanmıştır ve iki kere görülmüştür. 13. ve 14. şeritler yenidir. Öyleyse bu şekil nüfus üzerindeki artan etkiyi ortaya koymak için hazır bulunur. 1. neslin, 11. ve 16. şeritleri sırasıyla 1217 ve 13407 tahmin rakamını temsil eder. 2. nesil de geçişerek 15. ve 16. şeritleri üretir. Sonuçtaki şerit şekilleri 5215 ve 9409 tahminlerini sunar ki bunların ikisi de optimal 6325 değerine daha yakındır.

Çizelge 3.15 Nüfus-2. Jenerasyon

Şerit Sayısı	Ebeveynler		Geçişme yeri	Şerit	Ondalık Değer	Uygunluk
1	14	17	4	00001000110000	560	34.034
2	14	17	4	10001100001011	8.971	209.711
3	18	15	11	11000011011000	12.504	202.891
4	18	15	11	11111001000111	15.943	194.644
5	7	10	1	11001111101101	13.293	201.094
6	7	10	1	10011011001110	9.934	208.118
7	20	5	9	00111011110011	3.827	207.163
8	20	5	9	11000100110101	12.597	202.683
9	12	9	4	11001010111010	12.986	201.801
10	12	9	4	11011010101000	13.992	199.448
11	19	2	14	11001111110010	13.298	201.082
12	19	2	14	00101011001000	2.760	198.242
13	8	8	6	01100101001100	6.476	212.042
14	8	8	6	01100101001100	6.476	212.042
15	16	11	13	01010001011111	5.215	211.344
16	16	11	13	10010011000001	9.409	209.019
17	6	4	2	00110001010101	3.157	202.518
18	6	4	2	10111001101000	11.880	204.259
19	13	1	14	10010011010010	9.426	208.991
20	13	1	14	00100010011010	2.202	188.898

Bu arada çaprazlama (geçişme) nüfusun bir kısmını yanlış yöne araştırma yapmaya yollayabilir. Örneğin, birinci neslin, 5. ve 20. şeridi geçişir. İkisi de optimum değeri aşar. 5. şerit 3893 olur, 20. şerit ise 12539 ile 12597 arasında olur. Bu detay seviyesinde G.A. çalışmasını incelersek, bunun kötü değil iyi olduğunu hatırlamamız gerekir. Biliyoruz ki optimal değer 6325'tir. Fakat G.A. bunu bilmez, en iyi değeri arar, çeşitli yönlere ağını atar ve büyük balığı yakalamaya çalışır. Büyük ihtimalle başarır; ama bu biraz zaman alır

Anlattığımız bu sonucu bilinen araştırma en iyi sonucu, S değerini bulamamış, fakat çok yaklaşmıştır. 24. nesilde oldukça iyi bir tahmin olan 6328'e ulaşmıştır. 50 nesil sonra nüfus eğilimi %95, 74 nesil sonra nüfus %99.5 eğilimle yakınsamıştır. Bir denemenin sonucu şanslı, şanssız ya da arada olabilir. Birçok denemenin ortalama performansı, sadece bir denemeninkinden daha farklı olur.

Çizelge 3.16 50 Deneme Özeti

Ortalama Çözüm	6.297
Maksimum değer	6.656
Minimum değer	6.111
Ortalama nesil	56
Bulunan en iyi çözüm sayısı	29

Çizelge 3.16, 50 denemenin sonuçlarını özetler. Bunlardan 29'unda G.A. en iyi sonuca ulaşmıştır. Ortalama sonuç değeri 6297, en iyi değere oldukça yakındır. G.A. aynı zamanda çabuk çalışmıştır. 56 denemede %99.5 (ortalama) yakınsamaya ulaşmıştır. Kısaca G.A. oldukça iyi bir performans göstermiştir. Daha büyük bir nüfusta daha iyi sonuçlara ulaşabilirdi.

Optimale yakın uygunluk değerine çok çabuk ulaşılmıştır. Tabiki bu optimal nokta civarındaki eğrinin sığ doğasından kaynaklanmaktadır. Optimal S değerinden küçük ya da orta büyüklükteki sapmalar toplam kara çok etki etmemektedir. Birçok olayda nüfus eğilimi 40. nesilde %90'ın üzerindedir.

Örnek problem kâr fonksiyonunun optimizasyonunu kapsar; bu örnekte kâr sadece bir parametreye ya da değişkene bağlıdır. Birçok benzer iş problemi birden çok parametreye gereksinim duyabilir. Bu da G.A. yaklaşımında bulunan bir olasılıktır.

4. SONUÇ

Dört bölümden oluşan bu çalışmanın ilk bölümünde, G.A.'nın tanımı, tarihçesi ve kullanılma nedenleri verilmiştir. Doğadaki evrim mekanizmasına dayanan G.A.'nın işleyişinden, ikinci bölümde bahsedilmiştir. Üçüncü bölümde ise, Genetik Algoritma'yı kullanarak yapılan çalışmalar ele alınmıştır.

Dördüncü bölüm olan bu bölümde ise, genellikle optimizasyon problemlerinin çözümünde kullanılan, yaklaşık optimal çözümü veren ve bir araştırma yöntemi olan Genetik Algoritma ile ilgili aşağıdaki sonuçlar verilmiştir.

1. **G.A.'nın temel adımları basittir**. Genetik operasyonların bilgisayar sistemleri için bir model olarak kullanılması fikri karmaşık gelebilir. Fakat prosedür, birçok programlama dilinde kullanılabilecek bir seri basit adıma bölüştürülebilir.

2. **Problem spesifiği olan yegane G.A. adımı performans hesaplamasıdır**. Geçişme, mutasyon ve seçim, soruşturma altındaki her problem için aynı şekilde; fakat string operasyonu üzerinde, kör bir şekilde çalışır. Bunlar, şeritlerin neyi tanıttığına dikkat etmezler. G.A. ve problem arasındaki yegane bağlantı ise, performans hesaplanmasıdır ki bu G.A. yaklaşımının en büyük avantajlarından biridir. Eğer problemdeki önemli yönleri şeritlerle ifade ediliyor ve her tam şeride bir performans tahsis eden bir metod görevlendiriliyorsa, G.A. yaklaşımı çok uygun olabilir. Performans hesaplaması dışındaki tüm adımlar, geniş kapsamlıdır (generic).

3. **G.A. optimale en yakın çözümlere kendinden emin gitmektedir**. Eğer G.A. kullanılacaksa, en iyiye yakın bir çözüm elde edilecektir. Birçok problem, özellikle iş sproblemleri için bu kolaylıkla kabul edilir. G.A. arama prosedürü tüm uzayı tarayarak en iyi çözümleri arar ve en kısa zamanda olası çözümlere yaklaşır.

4. **G.A.'ların kullanılması, özellikle optimizasyon problemlerinde uygundur.**

SÖZLÜK

Alel (Allelle): Kromozom üzerindeki yerlerde bulunan değişkenlerin olası değerleri.

Çaprazlama (Crossover): Çaprazlama, farklı çözümler arasında bilgi değişimini sağlayarak arama uzayının benzer, ancak araştırılmamış bölgelerine ulaşmayı sağlayan bir arama operatörü.

Gen (Gene): Kromozom üzerindeki yerlerde bulunan değişkenler.

Genetik Algoritma (Genetic Algorithm): İlk defa Charles Darwin tarafından ortaya atılan, "doğal sistemde güçlülerin hayatta kalması" (en iyinin hayatta kalması) diye özetleyeceğimiz doğal seleksiyon ve doğal genetik mekanizmasına (genetiğin evrim tezine dayandırılan) dayanan araştırma algoritmaları.

Genetik Operatörler (Genetic Operators): Yeni kuşağın atalarını oluşturan seçilen bireyler yani gelişmiş yeni çözümlerin elde edilmesini sağlayan operatörler.

Kodlama (Encoding): Problemin tipine bağlı olarak çözüm adaylarını temsil eden yapılar.

Kromozom (Chromosome): Çözüm adaylarının bulunduğu çözüm kümesi veya aynı sayıda elemandan oluşan bir bit dizisi olarak gösterilen her çözüm kümesi.

Mutasyon (Mutation): Bir dizi içindeki bir veya birkaç değeri rassal olarak değiştirerek, yığında yeni dizilerin (yani arama uzayında yeni çözüm noktalarının) elde edilmesini sağlayan operatör.

Popülasyon (Population): Aday (rasgele) çözümlerin oluşturulmasıyla oluşan çözüm kümesi.

Seçim (Selection): Bireylerin (çözüm adaylarının) üreme için seçildiği bir işlem.

Uygunluk (Fitness) Fonksiyonu: Kromozomları problemin parametreleri haline getirerek onların bir bakıma şifresini çözen (decoding), sonra bu parametrelere göre hesaplama yapan kromozomların uygunluğunu bulan fonksiyon.

KAYNAKLAR

Aksoy, B., (2001), Döviz Piyasalarında Teknik Analiz Temelli Bir Alım/Satım Stratejisi ve Genetik Algoritmalar İle Optimizasyonu, Yüksek Lisans Tezi, Boğaziçi Üniversitesi Fen Bilimleri Enstitüsü.

Arslan, A. ve Kaya, M., (2001), "Determination of fuzzy logic membership functions using genetic algorithms", Fuzzy Sets and Systems, 117: 297-306

Bazaraa, M.S., (1975), Computerised layout design: a branch and Bound approach, AIIE Transactions, 7 (4): 432-438.

Bauer, R.J., (1994), Genetic Algorithms and Investment Strategies, John Wiley&Sons, Inc., Newyork.

Bayraktar, B., (1996), Neural Network Topology Optimization Genetic Algorithms,Yüksek Lisans Tezi, Boğaziçi Üniversitesi Fen Bilimleri Enstitüsü.

Biegel, J.E. ve Davern, J.J., (1990), "Genetic Algorithms and Job Shop Scheduling", Computers and Industrial Engineering, 19(1-4): 81-91.

Bodnovich, T., (1995), "Genetic Algorithms in Business and Their Supportive Role in Decision Making", College of Business Administration Kent State Üniversity.

Cantoni, M., Zio, E., ve Marseguerra, M., (2000), "Genetik Algoritmalar ve Optimal İşletme Tasarımı İçim Monte Carlo Modeli", Reliability Engineering and System Safety, 68: 29-38.

Chang, N.B. ve Wei, Y.L., (2000), "Siting recycling drop-off stations in urban area by genetic algorithm-based fuzzy multi objective nonlinear integer programming modeling", Fuzzy Sets and Systems, 114: 133-149.

Davis, L., (1985), "Job Shop Scheduling with Genetic Algorithms", (Ed: Grefenstette, J.J.), Proceedings of the First International Cenference on Genetic Algorithms, Lawrence Erlbaum Associates, Hillsdale, NJ, 136-140.

Davis, L., (1991), Handbook of Genetic Algorithms, Van Nostrand, Reinhold, New-York.

Dawei, L. ve Li, W., (1999), "Genetic algorithm for production lot planning of steel pipe", Mengguang Production Planning&Control, 10 (1): 54-57.

Dengiz, B. ve Altıparmak F., (1998), "Genetik Algoritmalara Genel Bakış", Gazi Üniversitesi Mühendislik Mimarlık Fakültesi, Endüstri Mühendisliği, 9 (3).

Dengiz, B. ve Altıparmak F., Smith, A.E., (1997b) "Local Search Genetic Algorithm for Optimization of Highly Reliable Communications Networks", (Ed: Back, T. Proceedings of the Seventh International Conference on Genetic Algorithms, Morgan Kaufmann Publishers, Los-Altos, CA, 650-657.

Eren, A. M., (1999), Optimization of Virtual Paths In Atm Networks Using A Parallel Annealed Genetik Argorithm, Yüksek Lisans Tezi, Boğaziçi Üniversitesi Fen Bilimleri Enstitüsü.

Grefenstette, J. J., (1986), Optimization of Control Parameters for Genetik Algorithms, IEEE Transactions on Systems, Man and Cybernetics.

Iris, E.S. ve Moghaddain, R.T., (1998), "Facilities Layout Design By Genetic Algorithms", Computers and Engn, 35 (3-4): 527-530.

Kuo, R.J., Chen, C.H. ve Hwang, Y.C., (2001), "An intelligent stock trading decision support system through integration of genetic algorithm based fuzzy neural network and artificial neural network", Fuzzy Sets and Systems, 118: 21-45.

Oğuz, F., (1998), Havadan Havaya Bir Füze Otopilot Denetleyicisinde G.A.'lar İle Başarım Artırımı, Yüksek Lisans Tezi, Boğaziçi Üniversitesi Fen Bilimleri Enstitüsü.

Özuysal, M. ve Soysal, O., (1997), Çok Değişkenli Fonksiyonların En İyilemelerinde Genetik Algoritmaların Davranışları, Fen Lisesi Bitirme Ödevi, İzmir Fen Lisesi.

Paşaoğlu, G., (1999), A GeneticAlgorithm Application for the Cutting Wrapping Problem İn Textile Industry, Yüksek Lisans Tezi, Boğaziçi Üniversitesi Fen Bilimleri Enstitüsü.

Periaux, J., Winter, G. ve Galan, M., (1995), Genetic Algorithms in Engineering and Computer Science, JOHN WILEY & SON Ltd.

Skawa, M., (1993), Fuzzy Sets and Interactive Multiobjective Optimization, Plenum Press, Newyork.

Skawa, M. Ve Yano, H., (1990), An Interactive fuzzy satisficing method for generalized multiobjective linear programming problems with fuzzy parameters, Fuzzy Sets and Systems 35: 125-142.

Skawa, M., Nishizaki, I.. ve Hitaka, M., (1998), "Interactive fuzzy programming for multi-level 0-1 programming problems with fuzzy parameters through genetic algorithms", Fuzzy Sets and Systems, 117:95-111.

Tanrıseven, D., (2000), Genetik Algoritmalar Kullanarak Peptidlerde Motif Bulma, Yüksek Lisans Tezi, Boğaziçi Üniversitesi Fen Bilimleri Enstitüsü.

Valenzuela, C.L., (1995), Evolutionary Divide and Conquer: a novel genetic approach to the TSP, Imperial College.

Whitley, L.D. ve Vose, M.D., (1995), Foundatiions of Genetic Algorithms Volume 3, Morgan Kaufmann Publishers, İnc.

Wilf, H.S., (1986), Algorithms and Complexity,Prentice-Hall, Inc. Obtained: Central Library of Imperial College (3 Computing 5.25 WIL).

Yılmazbayan, A., (1999), Genetik Algoritma ile Nükleer Yakıt Yöntemi Optimizasyonu (Nuclear Fuel Management Optimization Using Genetic Algorithm, Yüksek Lisans Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü.

Yeniay, M.Ö., (1999), Taguchi Deney Tasarımı Problemlerine Genetik Algoritma Yaklaşımı", Doktora Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü.

Zhao, C. ve Wu, Z., (2000), "A genetic algorithm for manufacturing cell formation with multiple objectives", 38 (2): 385-395.

ÖZGEÇMİŞ

Doğum Tarihi	07.10.1977	
Doğum Yeri	ISPARTA	
Lise	1991-1992	Naci Gökçe Lisesi
	1992-1994	Tınaztepe Lisesi
Lisans	1994-1998	Ankara Üniversitesi Fen Fakültesi Matematik Bölümü
Y.Lisans	1999-2002	Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Matematik Anabilim Dalı Matematik Programı
Çalıştığı Kurum	1999-Devam Ediyor	Deniz Lisesi Komutanlığı'nda Matematik Öğretmeni