T.C.
ONDOKUZ MAYIS ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BAYESCİ OPTİMİZASYON ALGORİTMASININ BESLENME PROBLEMİNDE KULLANIMI

YÜKSEK LİSANS TEZİ

Serpil GÜMÜŞTEKİN

İstatistik Anabilim Dalı

AĞUSTOS 2013
SAMSUN

T.C.
ONDOKUZ MAYIS ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

İSTATİSTİK ANABİLİM DALI

BAYESCİ OPTİMİZASYON ALGORİTMASININ BESLENME PROBLEMİNDE KULLANIMI

YÜKSEK LİSANS TEZİ

Serpil GÜMÜŞTEKİN
(11213008)

Tezin Savunma Tarihi : 02.09.2013

Tez Danışmanı: Yrd. Doç. Dr. Talat ŞENEL

Ondokuz Mayıs Üniversitesi Fen Bilimleri Enstitüsü

(İstatistik) Anabilim Dalında

(Serpil GÜMÜŞTEKİN) Tarafından Hazırlanan

BAYESCİ OPTİMİZASYON ALGORİTMASININ BESLENME PROBLEMİNDE KULLANIMI

başlıklı bu çalışma jürimiz tarafından 02/09/2013 tarihinde yapılan sınav ile YÜKSEK LİSANS tezi olarak kabul edilmiştir.

Başkan	**:**	Doç.Dr.Mehmet Ali CENGİZ Ondokuz Mayıs Üniversitesi	.....
Jüri Üyeleri	**:**	Yrd.Doç.Dr.Talat ŞENEL Ondokuz Mayıs Üniversitesi	.....
		Yrd.Doç.Dr.Naci MURAT Ondokuz Mayıs Üniversitesi	.....

02/09/2013

Prof. Dr. Recep TAPRAMAZ

Enstitü Müdürü

Gümüştekin Ailesine,

ÖNSÖZ

Çalışma konusu belirlememde ve çalışmalarım boyunca değerli yardım ve katkılarıyla beni yönlendiren, daima beni destekleyen danışmanım Yrd.Doç.Dr.Talat Şenel'e; tüm bilgisini hiç esirgemeden benimle paylaşan, çalışmalarımda yardımcı olan Doç.Dr.Mehmet Ali Cengiz'e ve Bayesci Optimizasyon Algoritması konusunda bilgi ve tecrübelerinden faydalandığım Prof.Dr.John Mc Call'a en içten teşekkürlerimi sunarım.

Ayrıca beni hiçbir zaman yalnız bırakmayan, her zaman destekleyen, her türlü desteklerini daima hissettiğim babam Ahat, annem Pakize ve kardeşim Serkan Gümüştekin'e sonsuz teşekkür ederim. Sizleri çok seviyorum.

Son olarak bu tez çalışmam boyunca emeği geçen tüm hocalarıma, arkadaşlarıma teşekkür ederim. İyi ki varsınız…

Ağustos 2013

Serpil GÜMÜŞTEKİN
Araştırma Görevlisi

İÇİNDEKİLER

Sayfa

ÇİZELGELER LİSTESİ

Sayfa

ŞEKİLLER LİSTESİ

KISALTMALAR

BMDA : Bivariate Marginal Distribution Algorithm
BOA : Bayesian Optimization Algorithm
cGA : compact Genetic Algorithm
COMIT : Combining Optimizers with Mutual Information Trees
DP : Doğrusal Programlama
EBNA : Estimation of Bayesian Network Algorithm
ECGA : Extended Compact Genetic Algorithm
EDA : Estimation of Distribution Algorithm
FDA : Factorised Distribution Algorithm
GA : Genetik Algoritma
g : Gram
jpdf : Joint Probability Distribution Function
kcal : Kilokalori
LFDA : Learning Factorised Distribution Algorithm
MDL : Minimum Description Length
mg : Miligram
MIMIC : Mutual Information Maximization for Input Clustering
PBIL : Population Based Incremental Learning
pdf : Probability Distribution Function
TL : Türk Lirası
UMDA : Univariate Marginal Distribution Algorithm

BAYESCİ OPTİMİZASYON ALGORİTMASININ BESLENME PROBLEMİNDE KULLANIMI

ÖZET

Son yıllarda karmaşık problemlerin çözümünde geleneksel yöntemlerin yetersiz kalması, yeni çözüm teknikleri arayışlarını gündeme getirmiştir. Genetik Algoritmalar (GA), bu arayışların sonucu olarak ortaya çıkan tekniklerdir. Genetik Algoritmalarda değişimi sağlamak amacıyla çaprazlama ve mutasyon operatörleri kullanılmaktadır. Ancak son yıllarda değişimi sağlamak için, çaprazlama ve mutasyon operatörlerinin yerine dağılımın tahmin edilip, örneklenmesini içeren olasılıksal model yaklaşımı önerilmiştir. Olasılıksal model yaklaşımında, populasyondan elde edilen çözümlerin olasılık dağılımı bulunur ve yeni nesiller bu olasılık dağılımı yardımıyla oluşturulur. Olasılıksal model yaklaşımını kullanan algoritmalara Dağılımın Tahmini Algoritmaları (EDA) adı verilmektedir. EDA'lar optimizasyon için güçlü bir teknik olarak kabul edilmektedir. Çoğu zaman GA'ların yetersiz kaldığı durumlarda problemleri başarılı bir şekilde çözmektedirler. EDA'nın performansı, olasılık dağılımının nasıl tahmin edildiği ve bu tahminler yardımıyla populasyondan örneklemin nasıl çekildiğine göre değişir. Çoğu EDA çalışmaları bunun üzerine yoğunlaşmıştır. Özellikle, yönlü grafiksel modeller olarak da bilinen Bayesci Ağlar bu alanda geniş olarak yer almaktadır ve dağılımın tahmin edilip, örneklenmesi sürecinde önemli bir role sahiptir. Bu çalışmada, dağılımın tahmin edilip örneklenebilmesinde, çok değişkenli EDA algoritmalarından biri olan ve aynı zamanda yönlü grafiksel modele sahip olan Bayesci Optimizasyon Algoritması (BOA) incelenmiştir. Daha sonra beslenme probleminin çözümünde Bayesci Optimizasyon Algoritması uygulanmıştır. Elde edilen sonuçlar Doğrusal Programlama (DP) ve GA çözüm sonuçları ile karşılaştırılmıştır. Karşılaştırmalar sonunda, zaman avantajı sağlaması ve maliyeti düşürmesi bakımından sabah kahvaltısı için beslenme probleminin çözümünde BOA'nın, DP ve GA'ya göre daha etkin sonuçlar verdiği görülmüştür.

Anahtar Kelimeler: BOA; GA; Doğrusal Programlama; Dağılım Algoritmalarının Tahmini; Beslenme; Bayesci Ağ.

THE USE OF BAYESIAN OPTIMIZATION ALGORITHM FOR FEEDING PROBLEM

ABSTRACT

Over the last few decades, the traditional methods are not enough to solve complex problems, to seek a new solution techniques have raised. Genetic Algorithms (GA), as a result of this search, emerging techniques. Crossover and mutation operators in order to exchange Genetic Algorithms are used. But in recent years to ensure that exchange, instead of crossover and mutation operators to predict and distribution, including sampling method is proposed probabilistic model. This way, the population is obtained from the probability distribution solutions and new generations are created with the help of this probability distribution. Estimation of Distribution Algorithms (EDA) that use this approach on algorithms is called. EDA is recognized as a powerful technique for optimization. Most of the time successfully solved problems in case of failure of GA. An EDA performance, how to anticipate the probability distribution, sampling will be directly related. Most of the studies that have focused on EDA. Specifically, versatile graphical models known as Bayesian Networks in this area is extensive and predict and distribution, has been a useful approach in the sampling process. In this study, Bayesian Optimization Algorithm (BOA) which is one of the a multivariate EDA algorithms with graphical model has been examined. BOA then applied to the problem of nutrition for breakfast. At the end of the comparisons, the problem of recommended diet for breakfast, BOA include more effective results compared to Genetic Algorithm and Linear Programming. BOA has reduced the time and the cost advantage provided.

Key Words: BOA; GA; Linear Programming; Estimation Distribution Algorithm; Feeding; Bayesian Network.

1. GİRİŞ

Günümüzde karmaşık problemlerin çözümünde geleneksel yöntemlerin yetersiz kalması, yeni çözüm teknikleri arayışlarını gündeme getirmiştir. Evrimsel hesaplama teknikleri, bu arayışların sonucu olarak ortaya çıkan tekniklerdir ve 19. yüzyılda Charles Darwin tarafından ileri sürülen evrim teorisinden esinlenerek geliştirilmiştir. Evrimsel hesaplama teknikleri, doğadaki evrimsel süreçleri model olarak kullanan bilgisayara dayalı problem çözme tekniklerdir. Geleneksel programlama teknikleriyle çözülmesi güç olan sınıflandırma, çizelgeleme ve çok boyutlu optimizasyon problemleri, evrimsel hesaplama teknikleri ile daha kolay ve hızlı olarak çözülebilmektedir.

Evrimin temel ilkesi, çevre şartlarına en iyi uyum sağlayan bireylerin hayatta kalması olarak ifade edilebilir. Bu ilkeden yola çıkarak oluşturulan arama yöntemlerinin tümüne Evrimsel Algoritmalar adı verilmektedir. Evrimsel Algoritmalar temel olarak; Genetik Algoritmalar (GA) (Holland,1975), Evrimsel Stratejiler (Rechenberg,1973) ve Evrimsel Programlama (Fogel,1962) şeklinde ayrılmaktadır. Genetik Algoritmalarda değişimi sağlamak amacıyla çaprazlama ve mutasyon operatörleri kullanılmaktadır. Ancak son yıllarda değişimi sağlamak için, çaprazlama ve mutasyon operatörlerinin yerine dağılımın tahmin edilip, örneklenmesini içeren olasılıksal model yaklaşımı önerilmiştir. Olasılıksal model yaklaşımında, populasyondan elde edilen çözümlerin olasılık dağılımı bulunur ve yeni nesiller bu olasılık dağılımı yardımıyla oluşturulur. Bu yaklaşımı kullanan algoritmalara Dağılım Algoritmalarının Tahmini (EDA) (Muhlenbein ve Paaß, 1996; Larranaga ve Lozano, 2002) adı verilmektedir. EDA'lar optimizasyon için güçlü bir teknik olarak kabul edilmektedir. Çoğu zaman GA'ların yetersiz kaldığı durumlarda EDA başarılı sonuçlar vermiştir (Pelikan ve Goldberg, 2003). Olasılık dağılımının nasıl tahmin edildiği ve bu tahminler yardımıyla populasyondan örneklemin nasıl çekildiğine göre EDA'nın performansı değişir. Çoğu EDA çalışmaları bunun üzerine yoğunlaşmıştır. Özellikle, yönlü grafiksel modeller olarak da bilinen Bayesci Ağlar bu alanda geniş olarak yer almaktadır ve dağılımın tahmin edilip, örneklenmesi sürecinde önemli bir yere sahiptir. Bu tezde, dağılımın tahmin edilip, örneklenebilmesinde çok değişkenli EDA algoritmalarından biri ve aynı zamanda yönlü grafiksel modele sahip olan Bayesci Optimizasyon Algoritması (BOA) incelenmektedir.

1.1 Tezin Amacı

Bu çalışmanın amacı, uygulamada karşılaşılan bazı zor problemlerin çözümünde aday çözümlerden yararlanarak problem ayrıştırmayı göstermek ve uzman bir optimizasyon algoritması olan Bayesci Optimizasyon Algoritmasını tanıtmaktır. Sadece çaprazlama ve mutasyon operatörleri kullanarak çözüm üretmek yerine populasyonun olasılıksal modelinin tahmin edilip, daha sonra bu olasılıksal model yardımı ile gelecek populasyonların oluşturulduğu yöntemi incelemek ve beslenme problemine uygulayarak bu algoritmayı test etmektir.

1.2 Literatür Araştırması

Ayrışabilir problemleri hızlı ve güvenilir bir şekilde çözmesi sebebiyle Bayesci Optimizasyon Algoritması (BOA) fen ve mühendislik alanlarında sıklıkla kullanılmaktadır. Bu algoritma ilk kez Pelikan ve diğ. (1999a,2000a,2000b), Pelikan ve Goldberg (2000) tarafından önerilmiştir. Literatüre baktığımızda BOA hakkında yayımlanmış çok fazla kitap bulunmamaktadır. Çalışmalar daha çok kitap bölümleri ve makaleler şeklindedir. Tek kitap olan "Hiyerarşik Bayesci Optimizasyon Algoritması" Martin Pelikan (2005) tarafından yayımlanmıştır. Bunun haricinde BOA ile ilgili yapılan bazı çalışmalara aşağıda değinilmiştir.

Pelikan ve diğ. (1999) "BOA: Bayesci Optimizasyon Algoritması" adlı çalışmada yeni aday çözümleri oluşturmak için umut verici çözümlerin ortak dağılımını kullanan genetik algoritma tabanlı Bayesci optimizasyon algoritmasını incelemişlerdir.

Ocenasek ve Schwarz (2000) "Paralel Bayesci Optimizasyon Algoriması" adlı çalışmada işlemci sayısına bağlı olarak optimizasyon zamanını düşüren paralel Bayesci optimizasyon algoritmalarını incelemişlerdir.

Pelikan ve diğ. (2002) "Bayesci Optimizasyon Algoritmasının Ölçülebilirliği" adlı çalışmada yeni aday çözümleri örneklemek ve umut verici çözümleri modellemek için Bayesci ağları kullanan Bayesci optimizasyon algoritması incelemişlerdir.

Li ve Aickelin (2003) "Hemşire Planlama Problemi için BOA" adlı çalışmada her hemşirenin atanmasını zamanlama kuralları doğrultusunda seçen, olasılıksal modelleri kullanma fikrine dayanan Bayesci optimizasyon algoritmasını incelemişlerdir.

Knowles ve Hughes (2005) "250 Değerlendirme Yapılan Bir Bütçe için Çok Amaçlı Optimizasyon" adlı çalışmada gelişmiş başlatma ve arama stratejilerini kullanan çok amaçlı optimizasyonu sağlayan iki farklı algoritma incelemişlerdir.

Pelikan ve diğ. (2007) “Az Parametreli Hiyerarşik BOA” adlı çalışmada hiyerarşik BOA’yı az parametreye sahip, sınırları dar olan bir algoritmayla, genetik algoritmayla ve paralel algoritma ile karşılaştırmışlardır.

Hauschild ve diğ. (2009) “Hiyerarşik BOA’da Olasılıksal Modellerin Analizi” adlı çalışmada test problemlerinin dört önemli sınıfı için hiyerarşik BOA’daki olasılıksal modelleri incelemişlerdir.

Hauschild ve Pelikan (2011) “EDA’lara Giriş ve Araştırma” adlı çalışmada farklı türdeki dağılım algoritmalarının avantaj ve dezavantajlarını incelemişlerdir.

Shengyao ve diğ. (2012) “Esnek İş Planlama Problemleri için EDA” adlı çalışmada maksimum tamamlama süresini azaltma kriteri ile esnek iş planlama problemini çözmede etkili bir dağılım algoritması önermişlerdir.

Kruse ve diğ. (2013) “Temel Evrimsel Algoritmalar” adlı çalışmada genel olarak evrimsel algoritmaları incelemişlerdir.

2. GENETİK ALGORİTMALARDAN DAĞILIM ALGORİTMALARININ TAHMİNİNE GEÇİŞ

Genetik algoritmalar (GA), doğal evrim ve genetikten ilham alan stokastik optimizasyon yöntemidir. Son yıllarda; iş yönetimi, mühendislik ve fen alanlarındaki problemlere başarıyla uygulanmaktadırlar. GA'lar geniş uygulanabilirliği ve işlemsel kolaylığından dolayı, sayısal optimizasyon alanında giderek daha önemli hale gelmiştir.

Olasılıksal model yapılı genetik algoritmalar (Pelikan ve diğ.,2002) arama uzayının genişlemesine yardımcı olmak amacıyla, genetik algoritmadaki çaprazlama ve mutasyon operatörleri yerine umut verici çözümlerden elde edilen olasılıksal modeli kullanırlar. Umut verici çözümlerin olasılık dağılımını tahmin etmenin pek çok yolu vardır ve her dağılım algoritması kendi elde ettiği tahmin ile ilgilenir.

2.1 Genetik Algoritmalar

Genetik algoritmalar, doğadaki canlıların geçirdiği süreci örnek alır ve iyi nesillerin kendi yaşamlarını korurken, kötü nesillerin yok olması ilkesine dayanır. Matematiksel modellemenin yapılamadığı veya kesin çözümün olmadığı problemlerde genetik algoritmalardan yararlanılır. Bu algoritma, ebeveynden (bir önceki nesil) doğan yeni bireylerin şartlara uyum sağlayıp yaşamlarını devam ettirmesine dayanır.

Goldberg (1989a)' in tanımına göre GA, rastlantısal arama tekniklerini kullanarak çözüm bulmaya çalışan, parametre kodlama esasına dayanan sezgisel bir arama tekniğidir.

2.1.1 Temel kavramlar

GA'ların çalışmasında ve başarılı çözüm değerlerine ulaşmada algoritma yapısında kullanılan kavramların ve bu kavram değerlerinin iyi belirlenmesi gerekmektedir. Aşağıda bu kavramlar açıklanmıştır.

Gen, Bir canlının (bireyin) kalıtsal özelliklerinden herhangi birini taşıyan parçadır. Her bir karar değişkeninin sayısal değeri, bir geni temsil eder. Bir problemde kaç adet karar değişkeni varsa o kadar da gen vardır (Koza,1995).

Kromozom, Genlerin bir dizi halinde sıralanması ile ortaya çıkan genler dizisidir. Yani, problemdeki karar değişkenlerinin her biri bir arada bulunmaktadır.

Popülasyon, Çözüm bilgilerini içeren kromozomların bir araya gelmesiyle oluşan olası çözüm yığınıdır. Fazla sayıdaki kromozom yığını, problemin çözüm süresini uzatırken az sayıdaki yığın çözüm değerlerine ulaşılamamasına sebep olabilir ya da sistemin belirli çözüm uzayında takılıp iyileşememesine neden olabilir. Popülasyon büyüklüğü yaygın olarak, 30 ile 100 adet arası kromozom içerecek şekilde düzenlenmektedir.

Kodlama, GA'ların çok önemli bir kısmını oluşturmaktadır. Probleme GA uygulanmadan önce, verinin uygun şekilde kodlanması gerekmektedir. Kurulan genetik modelin hızlı ve güvenilir çalışması için bu kodlamanın doğru yapılması önemlidir.

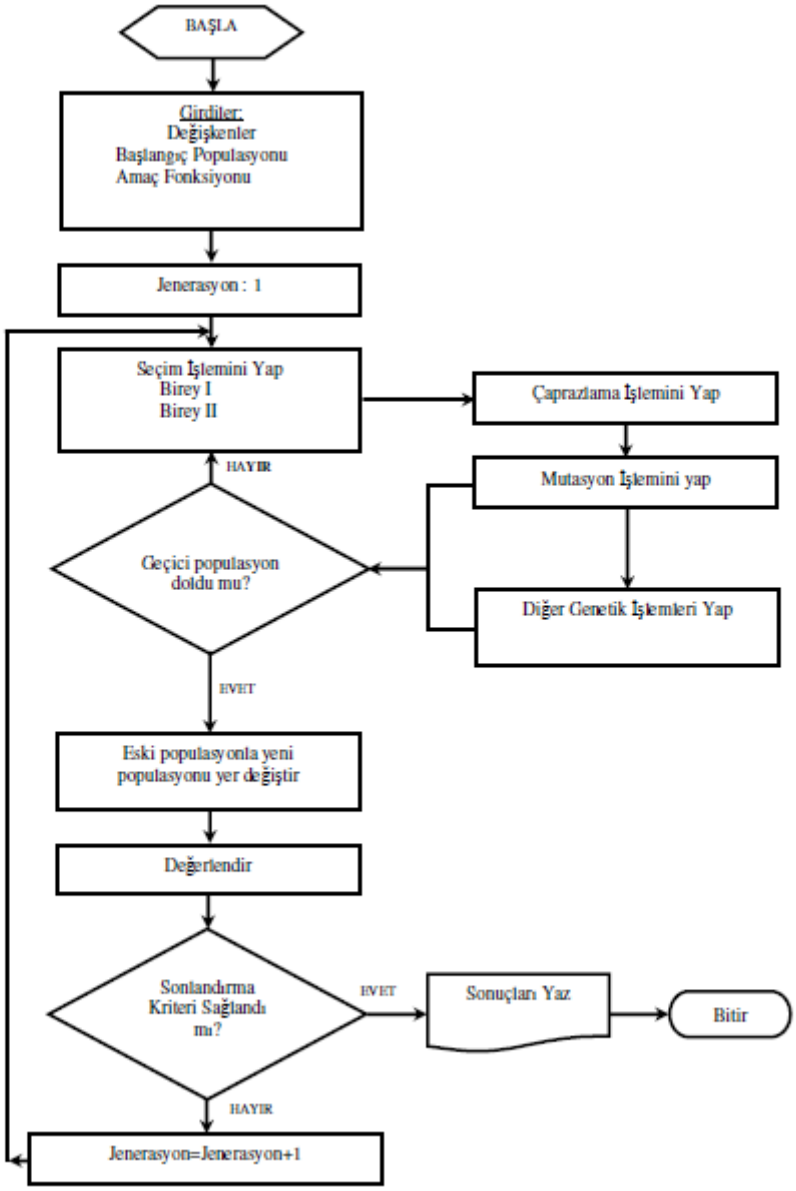
Uygunluk Fonksiyonu, Belirlenen çözümlerin uygunluk derecelerinin ölçülmesini sağlayan bir fonksiyondur. Her bir problem için ayrı bir uygunluk fonksiyonunun belirlenmesi gerekmektedir.

Uygunluk Değeri, GA' da uygunluk değerlendirmesi, bir uygunluk fonksiyonu sonucu elde edilen uygunluk değeri ile yapılmaktadır. Genetik algoritmalarda oluşturulan tüm dizilerin amaç fonksiyonu değeri ile ilişkili bir uygunluk değeri vardır. Dizi ile taşınan genetik bilgi, her nesilde uygunluk değeri daha iyi dizilerin seçilmesini sağlar. Dizinin uygunluk değeri ne kadar yüksekse, yaşama ve çoğalma şansı da o kadar yüksektir.

2.1.2 Genetik algoritmanın çalışması

Genetik algoritmaların temel prensibi, her adımda bir önceki nesilden yeni bireyler oluşturarak amaç fonksiyonunun uygunluk derecesini artırmak ve sonuç olarak belli kısıtlamaları sağlayacak şekilde amaç fonksiyonunu sağlayan en uygun değerini elde etmektir.

Arama uzayındaki tüm gerçekleşmesi muhtemel çözümler dizi olarak kodlanır. Genellikle gelişigüzel olarak bir çözüm kümesi seçilir ve başlangıç topluluğu olarak kabul edilir. Her bir dizi için bir uygunluk değeri hesaplanır, bulunan uygunluk değerleri dizilerin çözüm kalitesini gösterir. Bir grup dizi belirli bir olasılık değerine göre gelişigüzel olarak seçilip çoğalma işlemi gerçekleştirilir. Yeni bireylerin uygunluk değerleri hesaplanarak, çaprazlama ve mutasyon işlemlerine tabi tutulur. Önceden belirlenen kuşak (adım) sayısı boyunca yukarıdaki işlemler devam ettirilir. Yineleme (iterasyon), belirlenen kuşak sayısına ulaşınca işlem sona erdirilir. Uygunluk fonksiyonuna göre en uygun olan dizi seçilir. Şekil 2.1'te Genetik algoritma akış şeması gösterilmektedir.



Şekil 2.1: Genetik algoritma akış şeması

2.1.3 Kodlama

Bir problemin çözümü için genetik algoritmanın uygulanmasının ilk adımı, kodlamanın doğru bir şekilde gerçekleştirilmesidir. Çözümler kodlanarak, genetik algoritmanın kullanacağı şekle dönüştürülür. Tüm problem tipleri için geçerli olan bir kodlama şekli yoktur. Hatta benzer problemler bile bazen farklı kodlamalar gerektirebilir. Literatürde sıklıkla kullanılan kodlamalar aşağıda açıklanmaktadır.

İkili Kodlama, hem kolaylığı hem de pek çok probleme uygulanabilirliği açısından en yaygın kullanılan kodlama biçimidir. Problemin olası çözümleri 0 ve 1 değerleri kullanılarak kodlanır. Çizelge 2.1'te ikili kodlama biçimi gösterilmiştir.

Çizelge 2.1 : İkili kodlama gösterimi

Kromozom A	1 0 1 1 0 0 1 0 1
Kromozom B	1 1 1 0 0 1 1 1 1

Sıralı (Permutation) Kodlama : Sıralı kodlama genellikle birleşi optimizasyon problemlerinde kullanılmaktadır. Bu tip problemlerde dizinin uyum değeri, gen değerlerine ve ayrıca genlerin sırasına da bağlıdır. Gezgin satıcı problemi ve araç rotalama problemi sıralı kodlama yöntemine uygun problemlerdir. Çizelge 2.2'te sıralı kodlama biçimi gösterilmiştir.

Çizelge 2.2 : Sıralı kodlama gösterimi

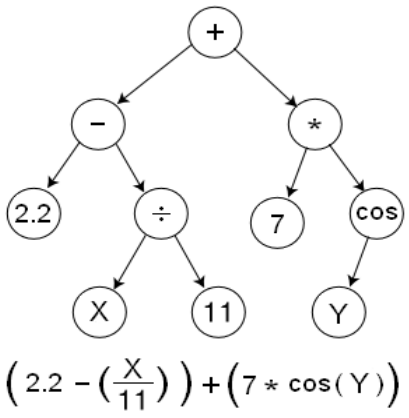
Kromozom A	1 5 3 2 6 4 7 9 8
Kromozom B	8 5 6 7 2 3 1 4 9

Değer Kodlaması : Değer kodlaması, karmaşık değerlerin kullanıldığı problemler için uygun bir kodlama yöntemidir. Değer kodlamasında her dizi, bir değerler dizisinden oluşmaktadır. Çizelge 2.3'te değer kodlama biçimi gösterilmiştir.

Çizelge 2.3 : Değer kodlama gösterimi

Kromozom A	1.235 5.323 0.454 2.321
Kromozom B	(sol), (sağ), (sol), (ileri)

Ağaç Kodlaması : Bu kodlama yöntemi, özellikle genetik programlamada kullanılmaktadır. Yöntemde her dizi, nesnelerden oluşan bir ağaç olarak ifade edilir. Şekil 2.2'te ağaç kodlama biçimi gösterilmiştir.



$$\left(2.2 - \left(\frac{X}{11}\right)\right) + \left(7 * \cos(Y)\right)$$

Şekil 2.2: Ağaç kodlama gösterimi

Genetik algoritmalarda kodlama yapılıp, rastgele ilk popülasyon oluşturularak, uygunluk değeri hesaplandıktan sonra yapılacak işlem, elde edilen bu popülasyona temel genetik işlemleri uygulamaktır. Genetik işlemler, ilerleyen nesillerde daha kaliteli ve daha çeşitli çözümler elde etmemizi sağlar.

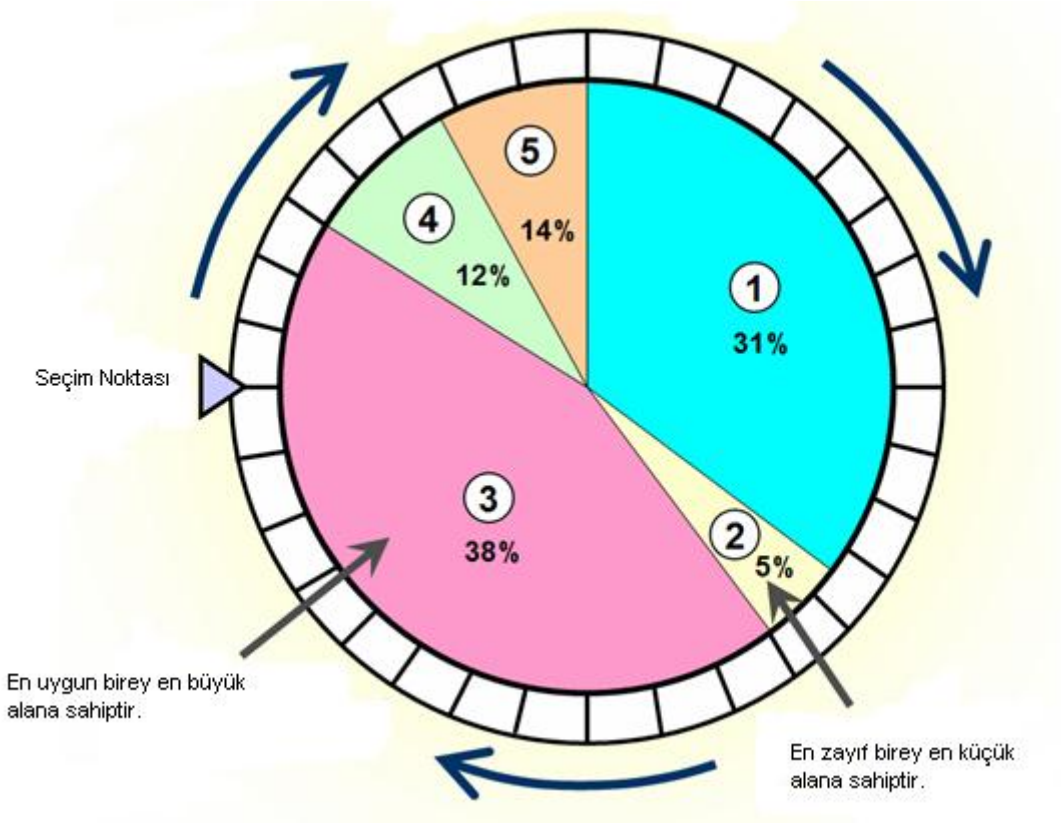
2.1.4 Seçim

Seçim işlemi, dizilerin uygunluk fonksiyonu değerine göre kopyalandığı bir süreçtir. Seçim işlemi; dizileri seçme, seçilmiş dizileri eşleme havuzuna kopyalama ve havuzda dizileri genelde çiftler halinde olmak üzere gruplara ayırma işleminden oluşmaktadır. Bu yöntemin amacı, uygunluk değeri ortalamanın üzerinde olan bireylere çoğalma fırsatı vermektir. Literatürde "çoğalma operatörü" olarak da isimlendirilmektedir ve çeşitli çoğalma operatörleri mevcuttur. Yaygın olarak kullanılan yöntemler arasında; Rulet Tekerleği (Roulette Wheel) ve Turnuva (Tournament) yöntemleri sayılabilir.

Rulet Tekerleği Yöntemi, ilk defa Holland (1975) tarafından ortaya çıkarılmış bir yöntemdir. Bu yöntemde, toplumdaki her bir bireyin uygunluk değerlerini gösteren f_i'ler hesaplanır. Bireylerin uygunluk değerleri toplanarak toplumun uygunluk değeri elde edilir. Her bireyin uygunluk değerleri, toplumun uygunluk değerine bölünerek, bireylerin seçilme olasılıkları elde edilir.

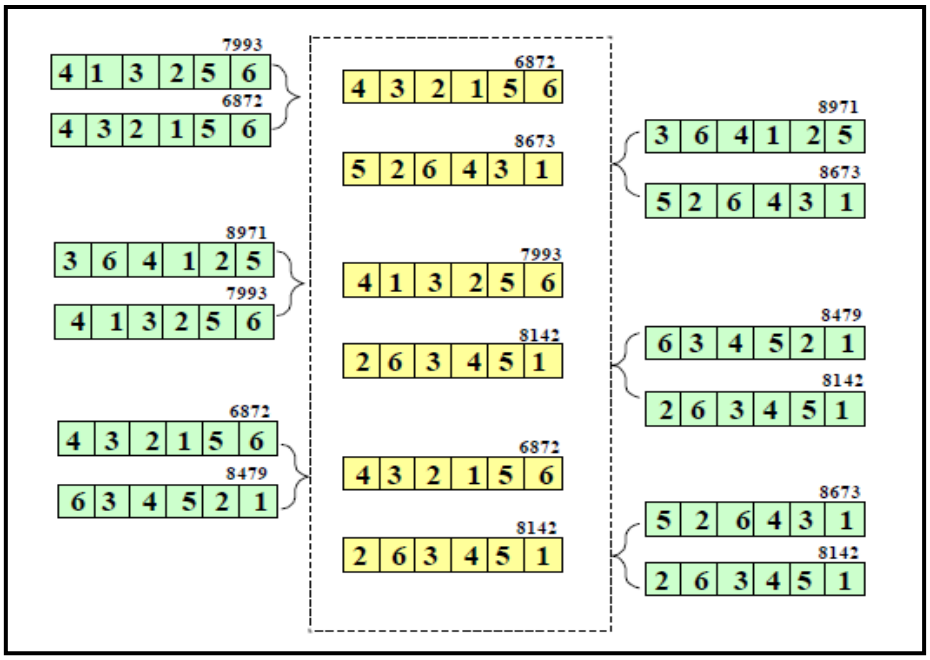
$$F_i = \frac{f_i}{\sum_{i=1}^{n} f_i} \qquad i = 1, \dots, N \qquad \textbf{(2.1)}$$

Bireyler seçilme olasılıklarına göre rulet çarkında yer alırlar. Bu durumda hangi bireyin olasılık değeri daha yüksekse o bireyin seçilme olasılığı daha fazla olacaktır. Şekil 2.3'te rulet tekerleği seçim yöntemi gösterilmiştir.



Şekil 2.3: Rulet tekerleği seçim yöntemi

Turnuva Yöntemi, turnuva seçim yönteminde, yeteneklilerin yani uyumluların yaşaması diğerlerinin ise yaşamaması ilkesi temel alınmıştır. Bu durumda rastgele seçilen iki kromozom uygunluk değerlerine göre turnuvaya sokulur ve iyi olan turnuvayı kazanarak hayatta kalmaya devam eder. Ortalama uygunluk değerine sahip olanlar ise en iyi kromozomların yarısı kadar bir sıklıkta seçilmiş olur. Bu şekilde populasyonun ortalama uygunluk değeri de giderek artar. Şekil 2.4' de turnuva seçim yöntemi gösterilmiştir.



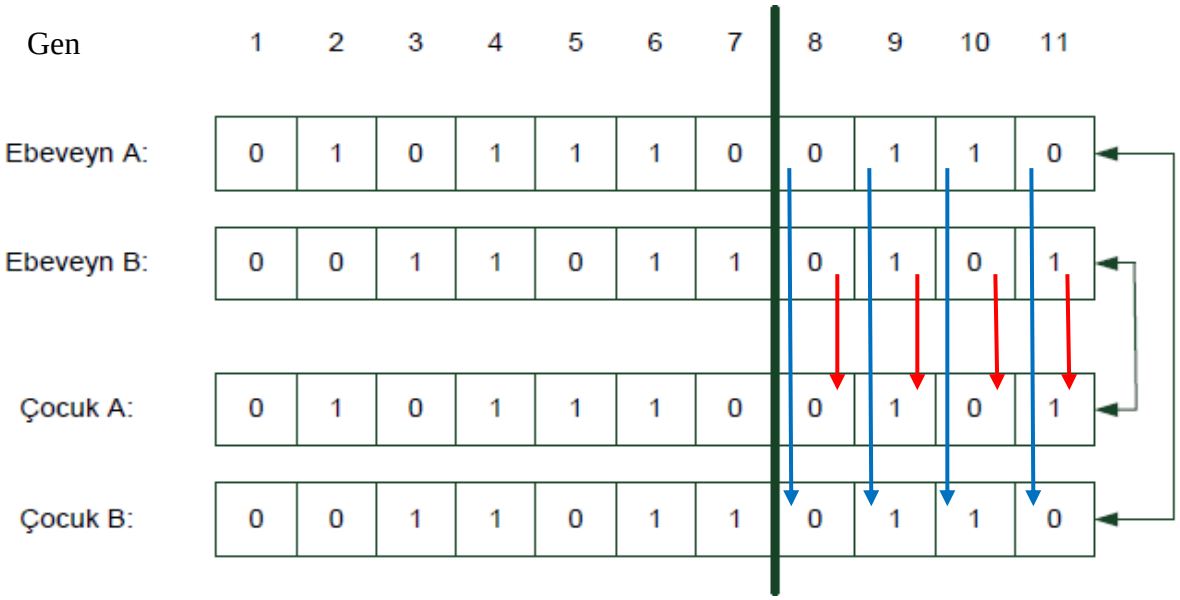
Şekil 2.4 : Turnuva seçim yöntemi

Şekil 2.4'te kromozomlar permütasyon kodlama ile kodlanmış olup, incelenen problem araç rotalama problemidir. Araç rotalama probleminde amaç, gidilecek tüm şehirlere en kısa mesafede ulaşmak olduğundan kromozomlar turnuvaya sokulduğunda daha az kilometreye sahip olan kromozom turnuvayı kazanarak yaşamaya devam etmektedir.

2.1.5 Çaprazlama

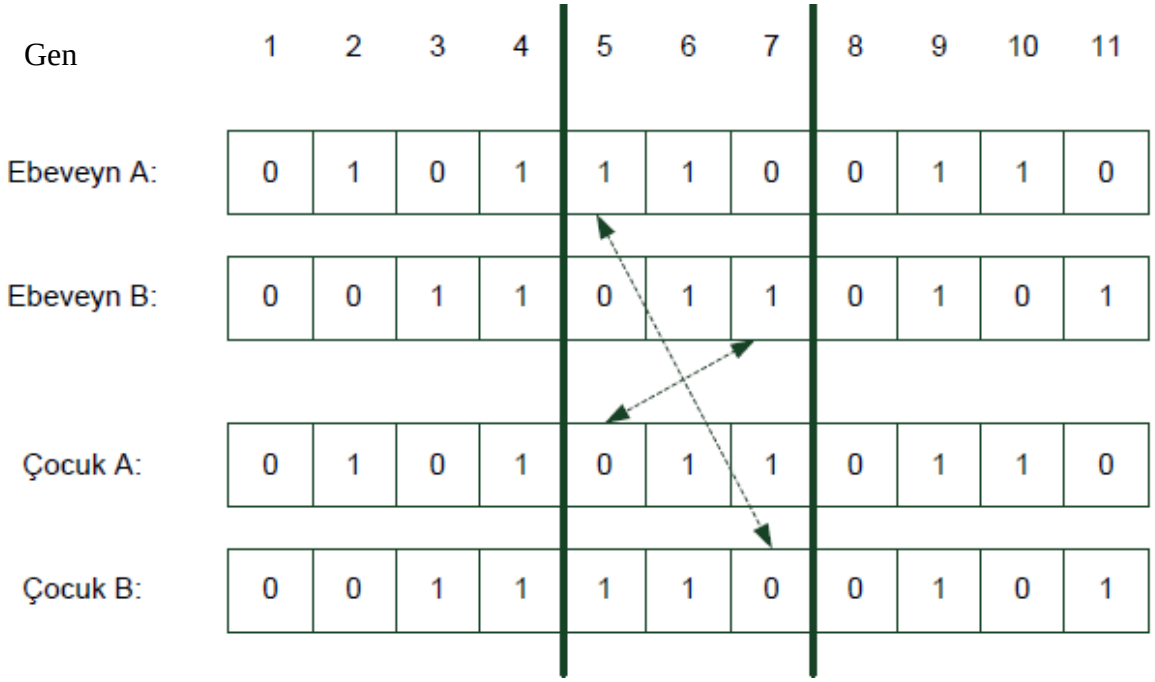
Çaprazlama, ebeveynlerde bulunan genlerin yeni oluşturulan bireylere aktarılmasıdır. İşlem, ebeveyn olarak seçilen kromozomlar üzerinde rastgele belirlenen konum/konumlarda, bilgilerin çapraz bir şekilde yer değiştirilmesi yolu ile gerçekleştirilmektedir. Çaprazlama işlemi, popülasyonda bulunan kromozomların belirli bir oranına uygulanmaktadır. Bu oran, algoritmanın başında ya da her yeni popülasyonu oluşturmadan önce belirlenmektedir. P_c çaprazlama oranı olmak üzere, popülasyonda N üye varsa, $N.P_c$ kadar üye çaprazlama için seçilmelidir. Literatürde en sık kullanılan çaprazlama türleri aşağıda açıklanmıştır.

Tek Noktalı Çaprazlama, rastgele seçilen kromozom çiftinde, çaprazlama yapılacak bölge rastgele seçilerek çaprazlama yapılmaktadır. İkili kodlama yapısına sahip 11 genden oluşan bir kromozom yapısı için gerçekleştirilmiş tek noktalı çaprazlama örneği Şekil 2.5'te gösterilmiştir.



Şekil 2.5 : Tek noktalı çaprazlama işlemi

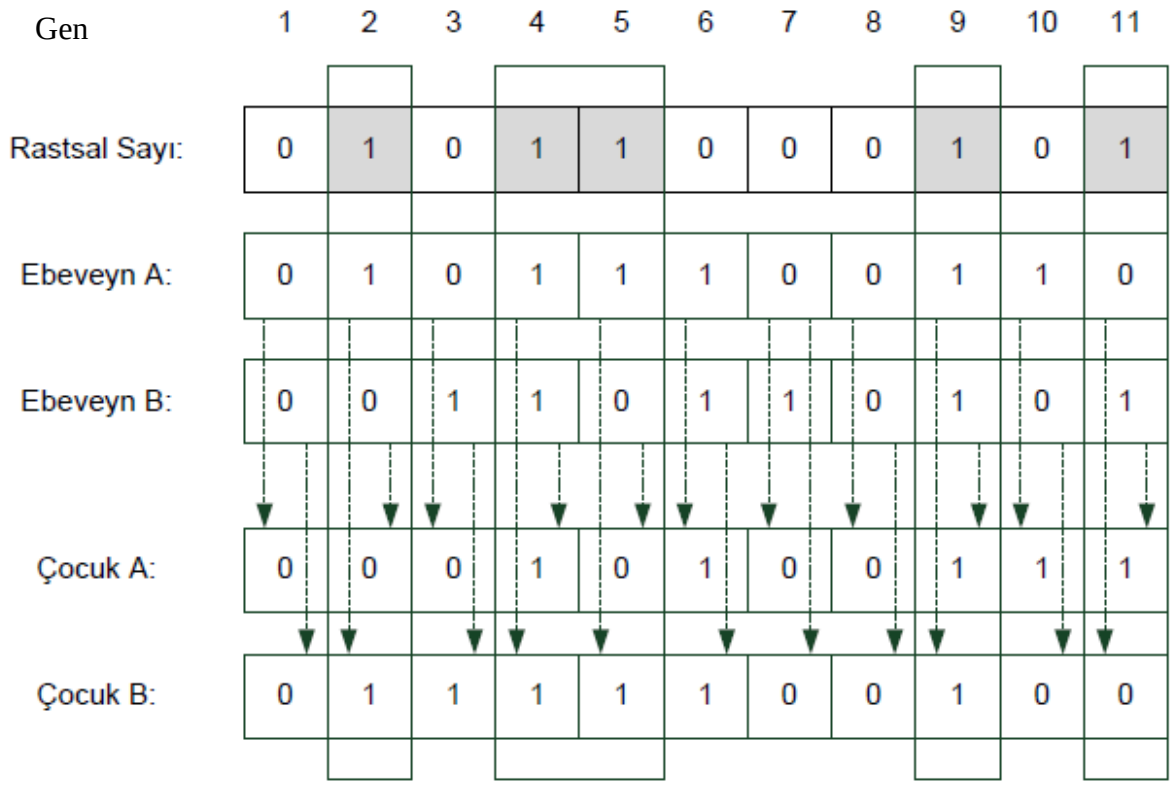
İki Noktalı Çaprazlama, kromozom eşleri iki farklı yerden kesilerek üç parçaya ayrılmaktadır. Parçalar karşılıklı olarak yer değiştirilerek çaprazlama yapılmaktadır. İkili kodlama yapısına sahip 11 genden oluşan bir kromozom yapısı için gerçekleştirilmiş iki noktalı çaprazlama örneği Şekil 2.6'da gösterilmiştir.



Şekil 2.6 : İki noktalı çaprazlama işlemi

Düzgün (Uniform) Çaprazlama, kromozomda rastgele seçilen bitlerin karşılıklı olarak yer değiştirmesi mantığına dayanmaktadır. Çaprazlanmak için seçilen iki kromozomda,

bir kromozomun her bir geninin diğer kromozomdaki karşılığı olan gen ile değiştirilmesi olasılığı 0,5 olarak belirlenmiştir (Syswerda 1989). Şekil 2.7'te gösterildiği gibi kromozomdaki gen sayısının uzunluğu kadar rastgele 0 veya 1'den oluşan dizideki 1'ler diğer kromozomdaki 1'ler ile 0'lar da 0'lar ile yer değiştirir.



Şekil 2.7 : Düzgün (Uniform) çaprazlama işlemi

2.1.6 Mutasyon

Kromozomların başkalaştırılması ya da farklılaştırılması için kullanılan bir operatördür. Çaprazlama işlemi ile elde edilemeyecek farklılıkları oluşturmak amacıyla yapılmaktadır. Mutasyon, kromozomdaki genlerin değişimidir ve problemin yapısına bağlı olarak aşağıdaki mutasyon operatörlerinden biri seçilebilir(Bolat ve diğ.,2004).

- Ters çevirme
- Yer değişikliği
- Ekleme
- Karşılıklı değişim

Mutasyon çeşitlerinin, ikili sistemde kodlanan bir kromozom üzerinde uygulanmasına ilişkin örnekler Çizelge 2.4'te verilmektedir.

Çizelge 2.4 : Mutasyon çeşitleri ve örnekleri

Mutasyon öncesi seçilen kromozom : [0 1 1 1 0 0 0 1 0 1]			
Mutasyon sonrası oluşan kromozom:			
Ters çevirme	**Yer değişikliği**	**Ekleme**	**Karşılıklı değişim**
[01**0011**0101]	[0101011**100**]	[01**0**1000101]	[0**0**110**1**0101]

2.1.7 Durdurma kriteri

Genetik algoritmalar her bir problem için belirlenen durdurma kriteri sağlanıncaya kadar hesaplama işlemlerini yapmaya devam ederler. GA'yı sonlandırmak amacıyla kullanılabilecek en iyi tercihler;

- seçilen en iyi kromozom, bilinen en iyi çözüme ulaştığında,
- jenerasyonlarda tekrarlı bir şekilde, en iyi kromozom aynı olduğunda,
- belirlenen istatistiksel değerlere erişildiğinde,
- belirlenen jenerasyon sayısı tamamlandığında,
- jenerasyon sonuçları açısından bir gelişme olmadığında,
- optimuma yakın bir değere erişildiğinde,

şeklindedir (Haupt ve Sue,2004).

Sıkça kullanılan durdurma kriterleri; iterasyon sayısı, yeni nesiller için oluşturulan çocuk (offspring) sayısı (Wang ve Ming,2002) ve algoritmanın çalışma süresidir (Naphade ve diğ.,1997, (Brucker ve diğ.,1999).

2.2 Dağılımın Tahmini ve Örneklenmesi

Dağılımın tahmini, veritabanındaki verinin olasılık dağılımını hesaplamayı amaçlar. GA'da dağılımın tahmini, sadece küçük bir alt kümesinde verilen çözümdeki iyi çözümlerin olasılık dağılımıdır. Olasılık dağılımı tahmin edildiğinde, iyi çözümlerin diğer kümelerini elde etmek için yeni nesiller oluşturulabilir.

2.2.1 Genel bilgiler

Bir olayın gerçekleşme ihtimali veya şansının ölçülmesine olasılık denir. X_i rastgele bir değişken ve x_i , X_i'nin alacağı değerler olsun. Bu durumda, $p(X_i = x_i)$, olasılık yoğunluk

fonksiyonunu (pdf) göstermektedir. $X = \{X_1, X_2, \dots, X_n\}$ n tane rastgele değişkenin bir vektörü ve $x = \{x_1, x_2, \dots, x_n\}$ de vektördeki her bir değişkenin aldığı değerler olmak üzere, $p(X = x)$, X'in ortak olasılık yoğunluk fonksiyonunu (jpdf) göstermektedir. Benzer şekilde, X_j verildiğinde X_i'nin koşullu olasılık yoğunluk fonksiyonu $p(X_i = x_i \setminus X_j = x_j) = p(X_i | X_j)$ şeklinde gösterilebilir.

Eğer problem kesikli yani her X_i, sınırlı bir değer kümesinde kesikli bir rastgele değişken ise $p(X_i = x_i) \equiv p(X_i = x_i)$, X_i'nin tek değişkenli marjinal dağılımını göstermektedir. Eğer X'deki tüm değişkenler kesikli ise $p(X = x) \equiv p(X = x)$ de ortak olasılık dağılımı olacaktır. Benzer şekilde, $X_j = x_j$ verildiğinde $X_i = x_i$'nin koşullu olasılık yoğunluk fonksiyonu $p(X_i = x_i \setminus X_j = x_j) = f(X_i = x_i \setminus X_j = x_j)$ şeklindedir.

X_S , X'in bir alt vektörü, x_S , X_S'nin alabileceği değerleri göstermek üzere; $p(X_S = x_S)$, X_S kümesinin marjinal dağılımıdır. Tek değişkenli marjinal dağılım, alt vektörler tek bir değişken içerdiğinde marjinal dağılımın basitleştirilmiş bir halidir.

X_A ve X_B , X_S 'nin parçalanmış alt vektörleri olsun. Bu durumda, $p(X_A = x_A \setminus X_B = x_B)$, $X_B = x_B$ verildiğinde $X_A = x_A$ 'nın koşullu olasılığı olarak (Eşitlik 2.2)'deki gibi tanımlanabilir. (Larranaga ve diğ., 1999)

$$p(x_A \setminus x_B) = \frac{p(x_A x_B)}{p(x_B)} \qquad \textbf{(2.2)}$$

Örneğin; $X = \{X_1, X_2, \dots, X_n\}$ rastgele değişkenlerin bir kümesi olmak üzere, $x = \{x_1, x_2, \dots, x_n\}$ 'in X kümesinden değerler alan bir çözüm olduğunu kabul edelim, $x_i \in \{0,1\}$. Sonra, dağılımı tahmin etmek için çözümlerin populasyonu olan P'den ortak olasılık yoğunluk fonksiyonu; $p(x) = p(x_1, x_2, \dots, x_n)$ yaklaşık olarak bulunur. Genelde $p(x)$'in hesaplanması, x'in tüm 2^n kombinasyonları için olasılıkların hesaplanmasını içerir ve bundan dolayı pek uygulanabilir bir yaklaşım değildir. Ancak, değişkenler arası bağıntıya bağlı olarak $p(x)$, etkileşimli değişkenlerin marjinal olasılıklarının hesaplanabilmesi için çarpanlarına ayrılabilir. Dolayısıyla $p(x)$'i tahmin etmek için, ilk olarak değişkenler arası bağıntıyı öğrenmeli daha sonra etkileşimli değişkenlerin marjinal veya koşullu olasılıklarını tahmin etmeliyiz.

Genel olarak çözüm kümesi verildiğinde, değişkenler arasındaki ilişkiyi bulmak için pek çok istatistiksel teknik kullanılabilir. Bizim örneğimiz için, ilişkinin önceden bilindiğini varsayalım. Her değişken sadece kendi ile ilişkili ve diğer değişkenler ile ilişkisiz olsun, $x_i \in X$. Böyle bağıntılar için, ortak olasılık dağılımı Eşitlik 2.3'te ifade edildiği gibi tek

değişkenli marjinal olasılıkların çarpımı şeklinde yazılabilir.

$$p(x) = \prod_{i=1}^{n} p(x_i) \qquad (\mathbf{2.3})$$

Olasılıksal modeli elde ettikten sonra amacımız ilişkili değişkenlerin olasılıklarını tahmin etmektir. Bunu yapmanın çeşitli yolları vardır. Bizim örneğimizde, populasyondan (P) elde edilen iyi çözümlerin bir kümesi (D) verildiğinde x_i'nin tek değişkenli marjinal olasılığı olan $p(x_i = 1)$, Eşitlik 2.4 yardımıyla hesaplanır.

$$p(x_i = 1) = \frac{1}{N_{x \in D, x_i = 1}} \qquad (\mathbf{2.4})$$

Burada, N; D'nin içindeki toplam çözüm sayısıdır. $p(x_i = 0)$ 'da benzer şekilde hesaplanabilir.

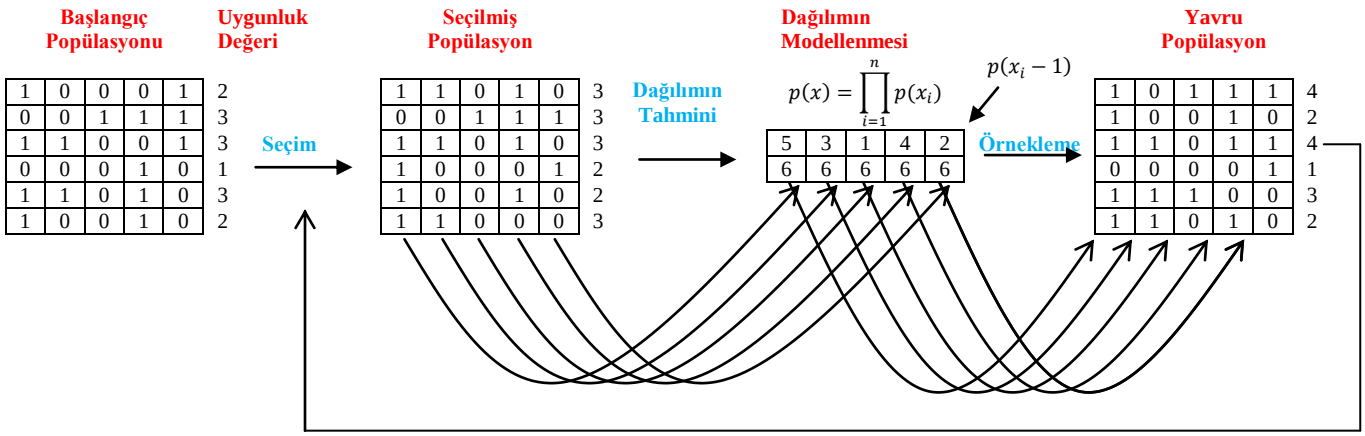
$p(x)$'i tahmin ettikten sonra, X'in diğer durumlarını oluşturmak için dağılımdan örneklem çekilir. Bizim örneğimizde, $p(x_i = 1)$'i tahmin ettiğimizde x_i'yi oluşturmak için

$$x_i = \begin{cases} 1, & rand(0,1) \leq p(x_i = 1) \\ 0, & Diğer\ Durumlarda \end{cases} \qquad \mathbf{(2.5)}$$

yazılabilir. Eşitlik 2.5'te, $rand(0,1)$; 0 ve 1 arasında rastgele sayılar üretir. Bunu tekrar ettirdikçe, yeni bireyler elde edilir ve bu şekilde bütün popülasyon elde edilinceye kadar süreç devam ettirilir. Daha sonra yeni elde ettiğimiz popülasyon ile eski popülasyon yer değiştirir. Onemax problemi, ikili gösterim ile kodlanmış dizideki 1'lerin toplamı olarak aşağıdaki gibi tanımlanır.

$$\mathrm{f_{onemax}(X)} = \sum_{\mathrm{i=0}}^{\mathrm{n-1}} \mathrm{X_i} \qquad (\mathbf{2.6})$$

$X = (X_0, X_1, \ldots, X_{n-1})$ n uzunluğundaki diziyi göstermektedir. Onemax fonksiyonu dizideki değerlerin tümü 1 olduğunda optimum olan doğrusal bir fonksiyondur. Onemax probleminde bitler birbirinden bağımsız olmak üzere, olasılıklar hesaplanır ve optimum değer bulunur. Onemax problemi için dağılımın tahmini örneği Şekil 2.8'te gösterilmektedir.



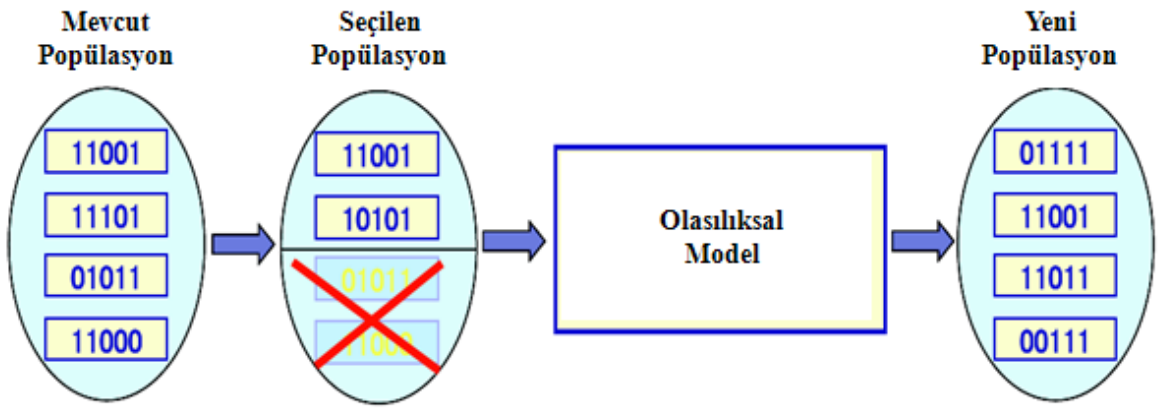
Şekil 2.8: Onemax probleminde dağılımın tahmin edilerek popülasyon üretilmesi

Şekil 2.8'te, seçimden sonra GA'da uygulanan çaprazlama uygulanmamıştır. Onun yerine, seçilmiş çözümlerin kümesinden tek değişkenli marjinal olasılıklar hesaplanmıştır. Daha sonra, yeni populasyon oluşturulmuştur.

3. DAĞILIM ALGORİTMALARININ TAHMİNİ (EDA)

Populasyondaki değişimin olasılıksal dağılım ve örnekleme yardımı ile yapıldığı algoritmaların sınıfına Dağılım Algoritmalarının Tahmini (EDA) denir (Mühlenbein & Paaß, 1996; Larranaga & Lozano, 2002). EDA, evrimsel algoritmalarda gelişmekte olan bir alandır. Olasılık dağılımını tahmin etme ve örneklemede farklı teknikler kullanan pek çok EDA vardır ve bu durum evrimsel algoritmalar arasında EDA'ları aktif araştırma konusu yapmaktadır.

EDA'lar yeni aday çözümler elde etmek için, umut verici çözümlerin olasılıksal modellerini kullanırlar. Şekil 3.1'te olasılıksal model kullanılarak oluşturulan yeni popülasyon gösterilmektedir.



Şekil 3.1 : Olasılıksal model kullanılarak oluşturulan yeni popülasyon

Aslında, EDA'lar genetik algoritmalara çok benzemektedirler. EDA'lar, olası çözüm uzayı üzerinden düzgün dağılım yardımıyla rastgele seçim yaparak başlangıç popülasyonunu oluştururlar. Her iterasyonda, umut verici çözümler aday çözümlerin mevcut popülasyonundan seçilirler. Seçilmiş çözümlerin olasılıksal modeli oluşturulur ve bu model yeni aday çözümleri elde etmede kullanılır. Yeni çözümler daha sonra orijinal popülasyona katılır ve eskilerden bazıları ile yer değiştirir. Bu süreç, durdurma kriteri sağlanıncaya kadar devam eder. Temel EDA prosedürü, Çizelge 3.1'te gösterilmiştir.

Çizelge 3.1 : Dağılım algoritmalarının çalışması

1.	P populasyonundan M büyüklüğünde başlangıç populasyonunu üret.
2.	P'den, N çözüm içeren D kümesini seç. N < = M
3.	D'den, p(x) olasılık dağılımını tahmin et.
4.	Yavru bireyleri oluşturmak için p(x)'den örneklem çek, ve anne baba bireylerle değiştir.
5.	Durdurma kriteri sağlanıncaya kadar Adım 2'ye geri git.

Tüm EDA'lar M tane çözüm içeren P tane popülasyonun üretilmesiyle başlar. Daha sonra, P'den N tane çözüm içeren D kümesi seçilir. D kümesi seçildikten sonra olasılık dağılımı tahmin edilerek, örneklem çekilir ve yavru bireyler için bu örneklem ile P populasyonu yer değiştirir. Durdurma kriteri sağlanana kadar algoritma bu şekilde adım 2'ye giderek tekrarlanır. Böylece genetik algoritmalar ve olasılıksal model yapılı algoritmalar arasındaki farkın, algoritmaların işleyiş süreçleri olduğu görülür. Bu durumda, EDA'lar genel olarak üç sürecin birleşimi olarak ifade edilebilir.

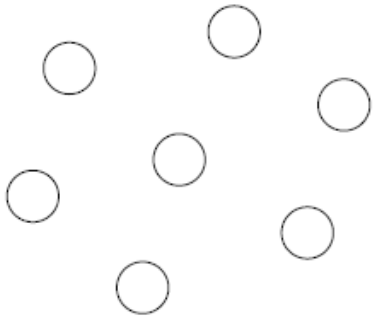
- İlişkinin tahmini,
- Olasılıkların tahmini,
- Dağılımın öneklenmesi.

3.1 Dağılım Algoritmaları

Dağılım algoritmalarının doğal bir sınıflandırması, kullandıkları modellerdeki ilişki sayısıdır. Düşük sayılı ilişkiden yüksek sayıdaki ilişkiye doğru değerlendirilirler. Bu bölüm ilk olarak, tüm değişkenler veya dizi konumlarının bağımsız olduğu durumdaki EDA'ları ele alarak başlayacaktır. Daha sonra; zincir, ağaç ve orman formundaki olasılıksal modeller kullanılarak bazı ikili etkileşime sahip EDA'lar incelenecektir ve son olarak, keyfi sayıda etkileşime sahip EDA'lar ele alınacaktır. Bu bölümde ikili dizi gösterimi kullanılmıştır ancak tüm ayrık EDA'lar sonlu sayıdaki dizi gösterimine genişletilebilir.

3.1.1 Tek değişkenli dağılım algoritmaları

EDA'lar her değişkenin bağımsız olduğu varsayımı üzerine kuruludur. Çözümdeki değişkenler arasında herhangi bir etkileşim yoktur. Bundan dolayı ortak olasılık fonksiyonu, tek değişkenli marjinal olasılık fonksiyonuna eşittir. Şekil 3.2'te ilişkinin grafiksel modeli gösterilmektedir.



Şekil 3.2 : Değişkenler arası etkileşimsiz ilişkinin grafiksel gösterimi

Kullanılan dağılım algoritması modelinin basitliğinden dolayı, bu kategorideki algoritmaların hesaplanması zor değildir, değişkenler arasında anlamlı ilişkiler olmayan problemlerde iyi çalışırlar. Ancak, yüksek etkileşime sahip problemlerde hata verme eğilimi gösterirler. Popülasyon Tabanlı Artımsal Öğrenme (PBIL) (Baluja, 1994), Tek Değişkenli Marjinal Dağılım Algoritması (UMDA) (Muhlenbein ve Paaβ, 1996), ve Kompakt Genetik Algoritma (cGA) (Harik ve diğ., 1997) olasılık dağılımının tek değişkenli modellerini kullanan algoritmalardır.

3.1.1.1 Popülasyon tabanlı artımsal öğrenme algoritması (PBIL)

PBIL algoritması, bütün olasılıkların eşit olduğu durum ile başlar. Olasılıkların kümesi, olasılık vektörü olarak adlandırılır. Bir nesilden diğer bir nesile geçmek için marjinal olasılıkları hesaplar. Her iterasyonda yeni çözümleri oluşturmak için olasılık vektörü güncellenir ve örneklenir. Algoritmanın çalışması Çizelge 3.2'te gösterilmiştir.

Çizelge 3.2 : PBIL algoritması

1. Başlangıçta bütün olasılıkların 0.5 olduğu olasılık vektörü (p) ile başla.
2. p'deki olasılıklar yardımıyla M çözümden, P popülasyonunu oluştur.
3. N tane umut verici çözüm içeren P'den D kümesini seç. N < = M.
4. Her x_i için $p(x_i)$ tek değişkenli marjinal olasılıklarını tahmin et.
5. Her i için eşitliğini kullanarak p'yi güncelle. $p_i = p_i + \lambda(p(x_i) - p_i)$
6. Her i için, eğer mutasyon istenmişse, yandaki eşitliği kullanarak mutasyon uygula. $p_i = p_i(1-\mu) + \text{random}(0 \text{ veya } 1)\mu$
7. Durdurma kriteri ile karşılaşıncaya kadar adım 2'yi tekrarla.

Burada, λ, öğrenme oranı ve μ, mutasyon aralığı olmak üzere, PBIL algoritmasının parametreleridir. Bir takım teorik bilgiler ve deneyler için (Baluja, 1994, 1995; Hohfeld ve

Rudolph, 1997; Berny, 2000; Gonzalez ve diğ., 2001), pek çok uygulama için (Galic ve Hohfeld, 1996; Inza ve diğ.,2001c; Petrovski ve diğ., 2006) kaynaklarına bakılabilir.

3.1.1.2 Tek değişkenli marjinal dağılım algoritması (UMDA)

UMDA algoritması, (Muhlenbein ve Paaß,1996) tarafından önerilmiştir. PBIL'den farklı olarak, olasılık vektörü veya öğrenme oranı parametresi kullanmadan basit bir şekilde, önceki nesillerden elde ettiği marjinal olasılıkları kullanarak gelecek nesilleri oluşturur. Aslında, UMDA, öğrenme oranı 1 alındığı takdirde PBIL'in özel bir halidir (Larranaga ve Lozano, 2002). Algoritmanın çalışması Çizelge 3.3'te gösterilmiştir.

Çizelge 3.3 : UMDA algoritması

1. M çözümden P populasyonu oluştur.
2. N tane umut verici çözüm içeren P'den D kümesini seç. N < = M.
3. Her x_i için $p(x_i)$ tek değişkenli marjinal olasılıklarını tahmin et.
4. $p(x_i)$'i örnekleyerek yeni bireyler elde etmek için M'yi oluştur ve P ile yer değiştir.
5. Durdurma kriteri ile karşılaşıncaya kadar adım 2'yi tekrarla.

3.1.1.3 Kompakt genetik algoritmalar (cGA)

Kompakt genetik algoritmaların gelişim sürecinde genetik algoritmanın, popülasyonun kendisini kullanarak olasılıksal bir model elde edebileceği gözlemlenmiştir (Harik ve diğ., 1997). PBIL'e benzer olarak, olasılık vektörü bu algoritmada da kullanılır ancak olasılık vektörü yardımı ile elde edilen sadece iki değişkenden olasılık vektörü güncellenir. Oluşturulan iki birey daha sonra turnuva seçim yöntemine benzer bir şekilde turnuvaya sokulur, uygunluk değerlerine göre olasılık vektörü tekrar güncellenir (Deb ve Goldberg, 1991). Olasılık vektöründeki değerler değişmez olana kadar bu şekilde algoritma çalışmaya devam eder. Ayrıca geniş popülasyonları depolama sorunu olmadığından yer açısından en etkili evrimsel algoritmalardan biri cGA'dır. Algoritmanın çalışması Çizelge 3.4'te gösterilmiştir.

Çizelge 3.4 : cGA algoritması

1. Başlangıçta bütün olasılıkların 0.5 olduğu olasılık vektörü ile başla.
2. p'deki örneklem olasılıklarına göre iki çözüm oluştur, çözümü daha uygun olan turnuvayı kazansın.
3. Her i için, aşağıdaki kuralı kullanarak p_i yi güncelle.
Eğer kazanan[i], kaybeden [i]'ye eşit değilken,
Eğer kazanan[i] = 1 ise; $p_i = p_i + \lambda$
değilse $p_i = p_i - \lambda$
4. Durdurma kriteri ile karşılaşıncaya kadar adım 2'yi tekrarla.

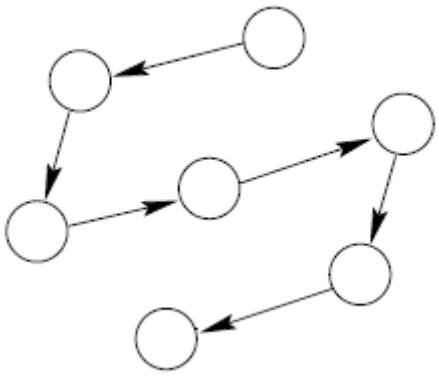
Harik ve diğ.(1997), λ parametresini 1/M olarak tanımlamışlardır. Burada M, algoritmanın parametresidir.

3.1.2 İki değişkenli dağılım algoritmaları

Değişkenler arası ikili etkileşim olduğu durumlarda bu algoritmalar, tek değişkenli EDA'lara göre daha iyi sonuç vermektedirler. Ancak birden çok değişken arasında etkileşim olduğunda başarısızdırlar. Girdi Kümelemesi için Karşılıklı Bilgi Maksimizasyonu (MIMIC) (de Bonet ve diğ., 1997), Karşılıklı Bilgi Ağaçları ile Optimize (COMIT) (Baluja ve Davies, 1997) ve İki Değişkenli Marjinal Dağılım Algoritması (BMDA) (Pelikan & Muhlenbein, 1999) olasılık dağılımının iki değişkenli modellerini kullanırlar.

3.1.2.1 Girdilerin kümelenmesi için karşılıklı bilginin maksimizasyonu algoritması (MIMIC)

MIMIC (de Bonet ve diğ., 1997) tek bir değişkenin bağımsız olduğu dağılım yapısı için zincir modelini kullanır. Zincir modeli; dizi pozisyonlarının düzenlenmesiyle, zincirin ilk pozisyonundaki 1'lerin olasılığıyla ve zincirdeki önceki pozisyondaki değerler verildiğinde diğer değerlerin şartlı olasılıklarıyla belirlenir. Zincirdeki ilk pozisyon haricindeki tüm pozisyonlar bir önceki pozisyona şartlı olarak bağımlıdır. Şekil 3.3'de ilişkinin grafiksel modeli gösterilmektedir.

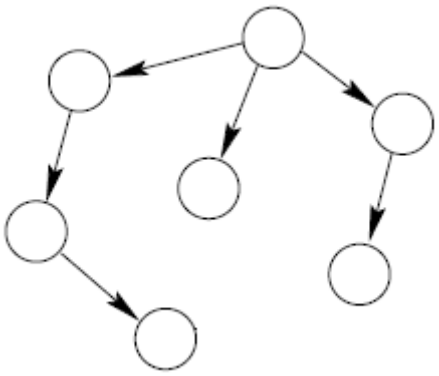


Şekil 3.3 : MIMIC grafiksel gösterim

3.1.2.2 Karşılıklı bilgi ağaçları ile optimize (COMIT)

COMIT (Baluja ve Davies 1997a, Baluja ve Davies 1997b) zincir yerine ağaç yapısında Bayesci Ağ modelini kullanan MIMIC algoritmasının gelişmiş halidir. Bağımlılık ağaçları olarak adlandırılan bu modelde, ağacın tepesindeki değişken bağımsızdır ve diğer her bir değişken kendi ailesi içinde üstten gelen bilgiye göre koşullanır. Bu algoritma, planlama ve

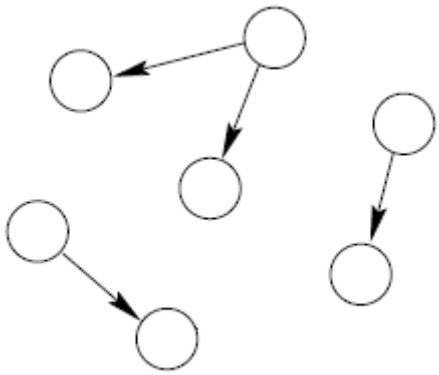
optimizasyon ile ilgili pek çok probleme başarılı bir şekilde uygulanabilmektedir. Şekil 3.4'te ilişkinin grafiksel modeli gösterilmektedir.



Şekil 3.4 : COMIT grafiksel gösterim

3.1.2.3 İki değişkenli marjinal dağılım algoritması (BMDA)

BMDA (Pelikan ve Muhlenbein 1999), daha önce açıklanan UMDA algoritmasının gelişmiş halidir. Karşılıklı bağımsızlık ağaçlarının bir kümesi olan orman modelini kullanır. MIMIC ve COMIT algoritmaları problemde sadece tek bir değişkenin bağımsız olduğu varsayımı üzerine kuruludur. Bu algoritmada ise yapı, pek çok değişkenin bağımsız olmasına izin verir yani hem tek değişkenin hem de iki değişkenin bağımsız olduğu problemleri modelleyebilir. Şekil 3.5'te ilişkinin grafiksel modeli gösterilmektedir.



Şekil 3.5 : BMDA grafiksel gösterim

3.1.3 Çok değişkenli dağılım algoritmaları

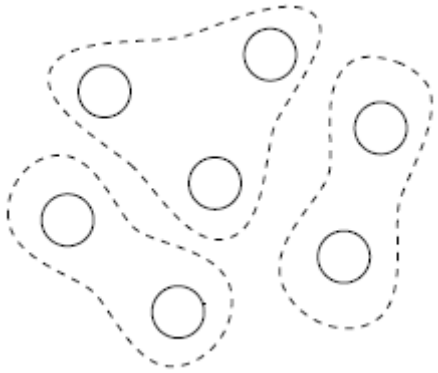
Bu bölümde, değişkenler arası etkileşimin ikiden fazla olduğu durumlar ele alınacaktır. Dolayısıyla olasılık dağılımının modeli de tek değişkenli ve iki değişkenli EDA'larda kullanılandan daha karmaşık hale gelecektir. Genişletilmiş Kompakt Genetik Algoritma (ECGA) (Harik ve diğ., 1997), Faktörleştirilmiş Dağılım Algoritması (FDA) (Muhlenbein ve

ark., 1999), Bayesci Optimizasyon Algoritması (BOA) (Pelikan ve diğ., 1999a), Faktörleştirilmiş Dağılım Algoritmasını Öğrenme (LFDA) (Muhlenbein & Mahnig, 1999b) ve Bayesci Ağların Tahmini Algoritması (EBNA) (Etxeberria ve Larranaga, 1999) olasılık dağılımında çok değişkenli modeli kullanan algoritmalardır. Son yıllarda Santana (2003a, 2005) tarafından bu gruba Markov Ağı kullanan iki algoritma daha dahil edilmiştir.

3.1.3.1 Genişletilmiş kompakt genetik algoritma (ECGA)

ECGA(Harik ve diğ., 1997), Kompakt Genetik Algoritmanın genişletilmiş halidir. ECGA'da kullanılan dağılımın modeli, Marjinal Ürün Modeli (MPM) olarak adlandırılır. MPM modeli, değişkenler arası etkileşimde çakışma olmadığını varsayar yani; etkileşime sahip değişkenlerin olduğu bir kümedeki bir değişken başka bir kümede olamaz.

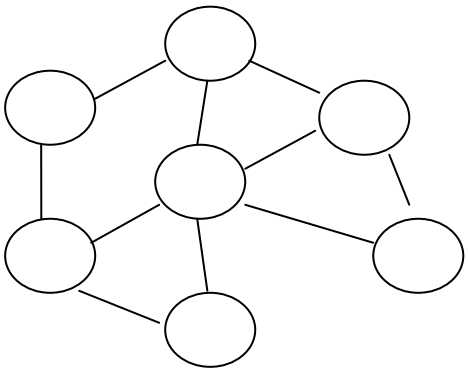
Eğer oluşturulan model doğruysa ve problemde hiç çakışan etkileşim yoksa, ECGA algoritması iyi çalışır. Fakat gerçek hayatta çok az problemde çakışmama durumu söz konusudur ve bu durumda ECGA algoritması başarısız olmaktadır. Sastry ve arkadaşlarının (Sastry ve Goldberg, 2004) son yıllardaki bazı çalışmaları bu konudaki etkinliği arttırmak üzerinedir. Şekil 3.6'da ilişkinin grafiksel modeli gösterilmektedir.



Şekil 3.6 : ECGA grafiksel gösterim

3.1.3.2 Faktörleştirilmiş dağılım algoritması (FDA)

Faktörleştirilmiş dağılım algoritması (FDA) (Muhlenbein ve diğ., 1999), UMDA algoritmasının genişletilmişi olarak önerilmiştir. FDA'da, birleşik olasılık dağılımı (jpd), etkileşimli değişkenlerin kümeleri arasında koşullu olasılıklara sahip olabilir. Genelde FDA, gerçek dünya problemlerinde pek mümkün olmayan önceden bağıntı bilgisini gerektirir. Ancak bu bilgi bilindiği takdirde FDA, GA'nın zorlandığı problemleri etkili bir şekilde çözümleyebilir. Şekil 3.7'de ilişkinin grafiksel modeli gösterilmektedir.



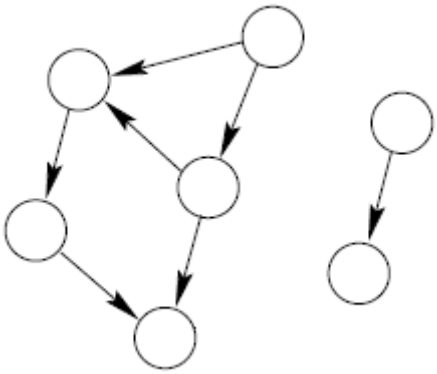
Şekil 3.7 : FDA grafiksel gösterim

3.1.3.3 Faktörleştirilmiş dağılım algoritmasını öğrenme algoritması (LFDA)

Faktörleştirilmiş dağılım algoritmasını öğrenme algoritması (LFDA), FDA'nın genişletilmişi olan LFDA algoritması Muhlenbein ve Mahnig (1999b) tarafından önerilmiştir. LFDA, bağıntı bilgisinin önceden bilinmesini gerektirmez. FDA'nın bazı teorik analizleri ve uygulamalar (Muhlenbein ve diğ., 1999; Muhlenbein ve Mahnig, 1999a,1999b; Zhang & Muhlenbein, 1999; Zhang ve Muhlenbein, 2004; Mahnig ve Muhlenbein, 2001a) da bulunabilir.

3.1.3.4 Bayesci optimizasyon algoritması (BOA)

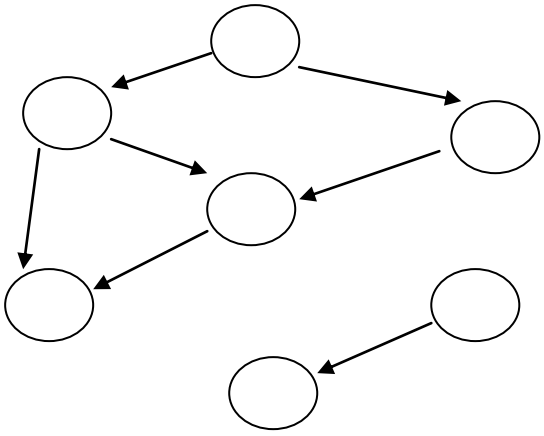
Bayesci Optimizasyon Algoritması (BOA), Pelikan ve diğ.(1999a) tarafından önerilmiştir. Problem yapısını modellemek için Bayesci Ağları, Bayesci ağın kalitesini ölçmek için Bayesci–Dirichlet metriği ve ağ uzayını araştırmak için açgözlü (greedy) algoritmasını kullanır. Her iterasyonda, seçilmiş bireylerin marjinal ve koşullu olasılıkları Bayesci ağda tutulur ve yeni bireyleri oluşturmak için kullanılır. BOA algoritması, Bölüm 4'te daha detaylı olarak incelenecektir. Şekil 3.8'de ilişkinin grafiksel modeli gösterilmektedir.



Şekil 3.8 : BOA grafiksel gösterim

3.1.3.5 Bayesci ağların tahmin algoritması (EBNA)

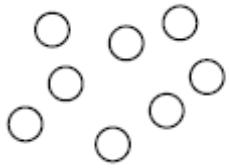
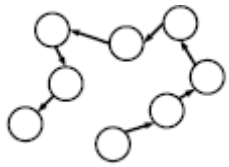
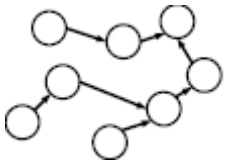
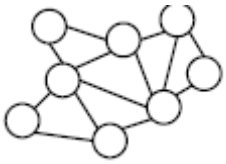
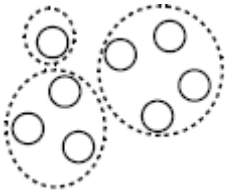
Bayesci Ağların Tahmini Algoritması (EBNA), (Etxeberria ve Larranaga, 1999; Larranaga ve diğ., 2000) tarafından önerilmiştir ve olasılık dağılımının modellenmesinde Bayesci ağları kullanır. EBNA'nın çalışma şekli BOA'ya benzemektedir. Farklı olan kısım EBNA'da Bayesci ağların kalitesini ölçen metrik kullanımında 3 farklı değişken vardır. Yani, EBNA'nın türüne göre metrik kullanımı da değişmektedir. Literatüre bakıldığında EBNA'nın optimizasyon problemlerine uygulanmış pek çok farklı çalışması bulunmaktadır. Şekil 3.9'da ilişkinin grafiksel modeli gösterilmektedir.



Şekil 3.9 : EBNA grafiksel gösterim

Yukarıda bahsedilen EDA'ları bir tablo halinde aşağıda göstermek, anlaşılmaları açısından kolaylık sağlayacaktır. Çizelge 3.5'te her bir EDA'nın adı, öneren kişiler, model karmaşıklığı ve ilişkinin grafiksel gösterimi bulunmaktadır.

Çizelge 3.5 : Kesikli EDA'ların tablo halinde gösterimi

Algoritma Adı ve Öneren Kişiler	**Model Karmaşıklığı**	**Grafiksel Gösterim**
PBIL (Baluja, Caruana), **cGA** (Harik, Lobo, Goldberg), **UMDA** (Mühlenbein and Paass)	Tek değişkenli yapıya sahiptirler. Olasılık vektörü veya marjinal olasılık kullanırlar.	
MIMIC (de Bonet, Davies)	İki değişkenli zincir modelini kullanır. Yönlü grafiksel modele sahiptir. Koşullu olasılıklar vardır.	
COMIT (de Bonet, Davies)	İki değişkenli ağaç modelini kullanır. Yönlü grafiksel modele sahiptir. Koşullu olasılıklar vardır.	
BMDA (de Bonet, Davies), **BOA** (Pelikan, Sastry, Goldberg), **EBNA** (Etxeberria, Larranaga)	İki değişkenli orman modelini kullanırlar. Yönlü Bayesci Ağ grafiksel modele sahiptirler. Koşullu olasılıklar vardır.	
FDA (Mühlenbein, Mahnig)	Yönlü olmayan modele sahiptir ancak yönlü model gibi modellenir.	
EcGA (Harik)	Yönlü olmayan Marjinal Ürün Modelini kullanır. Değişken gruplarının marjinal olasılıkları hesaplanır.	

4 BAYESCİ OPTİMİZASYON ALGORİTMASI

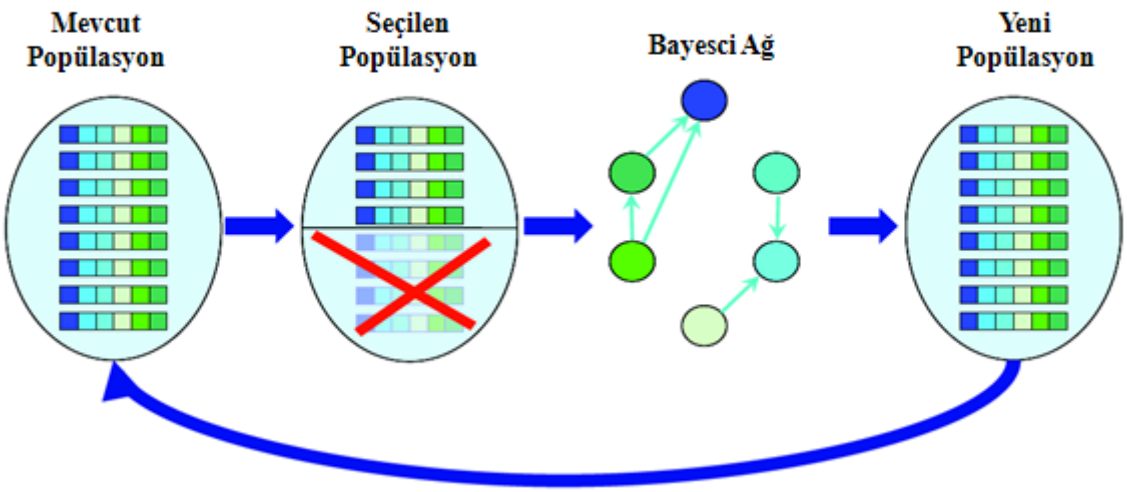
3.Bölümde olasılıksal model kullanımının, çok değişkenli etkileşime sahip zor modellerin daha kolaya indirgenip çözülebilmesinde güçlü bir yaklaşım olduğunu inceledik. Etkili ve verimli bir optimizasyon için, problemin modellenebilmesini sağlayacak uygun bir ayrıştırma yapılması gerekir. Bayesci Optimizasyon Algoritması (BOA), Bayesci ağları öğrenmek için metotları ve optimizasyon için olasılıksal modelleri birleştirme fikri üzerine kuruludur. BOA'nın genel çalışma algoritması Çizelge 4.1'te gösterilmektedir.

Çizelge 4.1 : BOA algoritması

1.	M çözümden P populasyonu oluştur.
2.	N tane umut verici çözüm içeren P'den D kümesini seç. N < = M.
3.	Seçilmiş çözümlerden Bayesci Ağı tahmin et.
4.	Bayesci ağı örnekleyerek M tane yeni birey oluştur ve P ile yer değiştir.
5.	Durdurma kriteri ile karşılaşıncaya kadar adım 2'yi tekrarla.

4.1 Bayesci Optimizasyon Algoritmasının Tanımı

Bayesci Optimizasyon Algoritması (BOA) (Pelikan ve diğ., 1998, 1999, 2000b), Bayesci ağları oluşturarak ve örnekleyerek, verilen bir problem için aday çözümlerin popülasyonunu değerlendirir. Başlangıç popülasyonunu rastgele oluşturur. Daha sonra bu popülasyon 4 adımda güncellenir. İlk adımda; GA seçme metoduna göre mevcut popülasyondan umut verici çözümler seçilir. İkinci adımda; daha sonrasında umut verici çözümleri tahmin edecek Bayesci ağ oluşturulur. Üçüncü adımda; Bayesci ağlardan örneklenerek yeni aday çözümler oluşturulur. Dördüncü adımda ise; yeni aday çözümler mevcut popülasyona karıştırılır ve uygunluk değerlendirmesine göre bazıları eskileriyle yer değiştirir. Bu adımlar durdurma kriteri ile karşılaşılıncaya kadar devam ettirilir. Şekil 4.1'te Bayesci Optimizasyon Algoritmasının çalışması gösterilmektedir.



Şekil 4.1 : Bayesci optimizasyon algoritmasının çalışması

4.2 Bayesci Ağlar

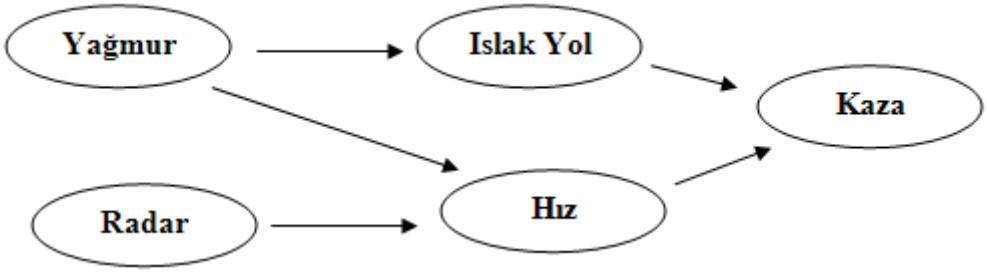
Bir Bayesci Ağ (Howard ve Matheson, 1981; Pearl, 1988) yapı ve parametreler olmak üzere iki bileşen ile tanımlanır.

Yapı, modellenmiş veri setindeki düğümlere karşılık gelen değişkenler ve kenarlara karşılık gelen koşullu olasılıkların yer aldığı yönlü grafiksel şekil ile gösterilir.

Parametreler, her bir değişkenin bağlı olduğu değişkenleri gösteren koşullu olasılıkların bulunduğu bir tabloda yer alırlar. Matematiksel olarak Bayesci ağlar,

$$p(X) = \prod_{i=0}^{n-1} p(X_i|\pi_i) \qquad (\mathbf{4.1})$$

olarak ifade edilen olasılık dağılımını kullanır. Burada, $X = (X_0, \dots, X_{n-1})$ problemdeki tüm değişkenlerden oluşan bir vektör; π_i, ağdaki X_i'nin ailelerinin kümesi ve $p(X_i|\pi_i)$, π_i ailesi verildiğinde X_i'nin koşullu olasılığını göstermektedir. Bayesci ağ yapısını göstermek amacıyla basit bir örnek Şekil 4.2'te ve örneğe ilişkin koşullu olasılıklar tablosu Çizelge 4.2'te gösterilmektedir.



Şekil 4.2 : Bayesci ağ yapısı

Çizelge 4.2 : Bayesci ağ koşullu olasılıklar tablosu

Kaza	Islak Yol	Hız	p(Kaza/Islak Yol, Hız)
Evet	Evet	Yüksek	0.18
Evet	Evet	Düşük	0.04
Evet	Hayır	Yüksek	0.06
Evet	Hayır	Düşük	0.01
Hayır	Evet	Yüksek	0.82
Hayır	Evet	Düşük	0.96
Hayır	Hayır	Yüksek	0.94
Hayır	Hayır	Düşük	0.99

Çizelge 4.2'te, düşük ve yüksek olmak üzere iki hız değerine sahip bir arabanın yolun ıslak olmasına bağlı olarak kaza yapma olasılıkları gösterilmiştir.

4.2.2 Bayesci ağları öğrenme

Başarılı bir BOA uygulaması için BOA'nın, problemi ayrıştıran bağımlılıkları ve bağımsızlıkları gösteren Bayesci Ağ modelinin içeriğini iyi bilmesi gerekir. Bir Bayesci ağı öğrenmenin iki tane amacı vardır.

Yapıyı öğrenme, ilk olarak ağın yapısı bilinmelidir. Yapı, ağ tarafından oluşturulan koşullu bağımlılıkları ve bağımsızlıkları tanımlar.

Koşullu olasılıkları öğrenme, yapı öğrenildikten sonra, son yapıya ilişkin koşullu olasılık değerleri öğrenilmelidir.

BOA'da yapı bilindiğinde parametreleri öğrenmek çok kolaydır. Çünkü umut verici çözümlerin populasyonundaki her bir değişkenin değeri belirlidir. Başka bir deyişle, veri tam olduğundan olasılıkları hesaplamak mümkündür. Yapıyı öğrenmek çok daha zor bir problemdir. Bayesci ağların yapısını öğrenmek için algoritmalar iki bileşene sahiptir.

Skor ölçütü, Bayesci ağ yapısının kalitesini ölçer. Genellikle skor ölçütü, yapının olasılık değerleri ile orantılıdır. Genetik Algoritmada ise skor ölçütü, yapının uygunluğunu belirler.

Arama uzayı, verilen skor ölçütüne ilişkin, tüm mümkün ağlar arasından en iyi ağ yapısını bulmak için araştırma yapılır.

4.2.2.1 Skor ölçütleri

Ağ yapılarının kalitesini ölçmek için Bayesci Ölçüt ve Minimum Uzaklık Ölçütü olmak üzere iki farklı yaklaşım vardır. Bayesci ölçüt (Cooper & Herskovits, 1992; Chickering ve diğ.,1994) eldeki veriye ve belirsizliklere ilişkin yapının marjinal olasılığını hesaplayarak her bir yapının kalitesini ölçer. Minimum uzaklık ölçütü (Rissanen, 1978,1989,1996) model tarafından izin verilen ölçüde veri sıkıştırma miktarı ile orantılı olan düzenlilik sayısına dayanır.

Bayesci ölçütler (Cooper ve Herskovits, 1992; Chickering ve diğ.,1994), marjinal olasılıkları hesaplayarak ağ yapısının kalitesini ölçmek için Bayes kuralını kullanır. Marjinal olasılıklar, modeldeki tüm koşullu olasılıklar üzerinden önsel dağılıma göre gözlemlenmiş veriden elde edilen modellerin olasılığının ortalaması ile hesaplanır. D, veri kümesini ve B'de parametreler hariç değerlendirilmiş Bayesci ağı göstermek üzere $p(B|D)$, D veri seti verildiğinde, B yapısının koşullu olasılığını göstermektedir.

$$p(B|D) = \int_{\theta} p(\theta|B)p(D|B,\theta)d\theta \qquad (\mathbf{4.2})$$

Burada θ 'nın her bir değeri, B ağındaki koşullu olasılıkları gösterir. Ayrıca $p(B)$, B ağ yapısının önsel olasılığıdır. $p(\theta|B)$, B ağı bilindiğinde θ parametrelerinin önsel olasılığıdır ve $p(D|B,\theta)$ ise B ağı ve θ parametreleri bilindiğinde D'nin olasılığını gösterir. D veri kümesi olmak üzere $p(D)$, tüm ağlarda aynı olduğundan yapı değerlendirmelerinde genellikle ihmal edilir.

Marjinal olasılığı hesaplamak için, her bir yapının parametreleri ile önsel olasılık dağılımı bilinmelidir.

Minimum uzaklık ölçütü (MDL) (Rissanen, 1978), model için gerekli olan alanı miktar olarak azaltan iyi modellere ve aynı zamanda model tarafından oluşturulan verideki düzenliliklerin sayısını arttıran modellere dayanarak ölçüm yapar. Bayesci ölçütlerin aksine, MDL ölçütü verinin içerdiği bilgiye karşı daha az duyarlıdır ve genellikle gereken tüm etkileşimleri elde edemeyen ağ modeli ile sonuçlanır(Pelikan ve diğ., 2002). Bu durumdan kurtulmak için, MDL'nin çok sayıda örneğe ihtiyacı vardır.

4.2.2.2 Arama uzayı

Yapının iyiliğini ölçecek skor ölçütü seçildiğinde, iyi bir model bulmak için mümkün modeller uzayı aranır. Aslında Bayesci ve Bayesci olmayan çoğu ölçüt için en iyi ağı bulmak zor bir problemdir (Chickering ve diğ., 1994). Dolayısıyla en iyi ağ yapısını bulmak için skor ölçütlerine ilişkin herhangi bir polinomial zaman algoritması yoktur. Ancak, Açgözlü (Greedy) algoritması (Chickering ve diğ., 1994) genellikle iyi ve başarılı bir şekilde ağ bulmayı gerçekleştirmektedir. Açgözlü algoritma, B Algoritması olarak da bilinir. Algoritma, genellikle kenarı olmayan bir yapı ile başlar. Daha sonra her adımda kaliteyi artıracak kenar eklenir. Bu durum hiçbir gelişme elde edilemeyinceye kadar devam eder. BOA'da başlangıç yapısı, önceki nesilde öğrenilen yapıyı düzenleyerek oluşturulabilir. Bu tezde sunulan deneylerde ağ, her nesilde boş bir ağdan elde edilmiştir.

Açgözlü algoritmanın kullandığı üç temel bileşen kenar ekleme, kenar kaldırma ve ters kenardır.

Kenar Ekleme, yeni bir bağımlılık elde etmek için ağa kenar eklenir.

Kenar Kaldırma, yeni bir bağımsızlık elde etmek için var olan bir bağımlılığa sahip kenar ağdan kaldırılabilir.

Ters Kenar, mevcut bir kenar, bağımlılığın yönünün değişmesi için ters çevrilir.

4.2.3 Bayesci ağların örneklenmesi

Bayesci ağın yapısı ve parametreleri öğrenildiğinde, yeni aday çözümler öğrenilen ağ tarafından elde edilen dağılıma göre oluşturulur. Örneklem, Bayesci ağların iki adımdan oluşan bir simülasyonu gibi düşünülebilir (Henrion, 1988). İlk adım, düğümlerin temel sıralamasının hesaplanmasıdır. Burada her düğüm kendi ailesinden önce gelmelidir. Temel fikir, belirli bir sırada çok değişken üretmektir. Dolayısıyla her değişkenin ailesinin değerleri, değişkenin kendi değeri oluşmadan önce oluşturulur. İkinci adımda ise, yeni aday çözümlerin tüm değişkenlerinin değerlerinin, hesaplanan sıraya göre oluşturulmasıdır. Algoritma bu şekilde oluşturduktan sonra, herhangi bir değişkenin değeri oluşturulmaya çalışıldığında ailesinin değeri zaten oluşmuş olacaktır. Dolayısıyla değişkenin ailesinin değerleri göz önüne alındığında, değişkenin dağılımında koşullu olasılıklar yer alacaktır.

5 UYGULAMA

Bu çalışmada sabah kahvaltısı için beslenme problemi ele alınmıştır. Minimum maliyetle, günlük besin öğesi ihtiyaçlarını karşılayacak şekilde hangi besinlerden ne kadar yenmesi gerektiği üzerinde durulmuştur. Bu amaçla gerçek veriler kullanılarak doğrusal programlama modeli oluşturulmuş, Matlab programlama dili, WinQSB paket programı ve R – Project ile çözümlenmiştir. Elde edilen sonuçlar, Bayesci Optimizasyon Algoritması ile zaman tasarrufu sağlandığını ve maliyetin daha az olduğunu göstermiştir.

5.1 Gereç ve Yöntem

Doğrusal programlama tekniği, değişkenlere ve kısıtlayıcılara bağlı kalarak amaç fonksiyonunu en uygun (maksimum veya minimum) kılmaya çalışır. Standart doğrusal programlama modeli (Eşitlik 5.1) ;

$$\max(\min)Z(x) = \sum_{j=1}^{n} c_j x_j$$

$$\sum_{j=1}^{n} a_{ij} x_j \ (\leq, =, \geq) \ b_i \quad i = 1,2,\dots,m \qquad \textbf{(5.1)}$$

$$x_j \geq 0 \qquad j = 1,2,\dots,n$$

şeklinde formüle edilmektedir. Burada,

x_j : karar vericinin denetimi altında olan ve bilinmeyeni gösteren karar değişkenlerini,

$Z(x)$: optimize edilecek amaç fonksiyonunu,

c_j : j. karar değişkeninin amaç fonksiyonundaki katkı katsayısını,

a_{ij} : j. karar değişkeninin i.kısıttaki katkı katsayısını (teknolojik katsayıları),

b_i : i. sınırlı kaynak miktarını yani i.kısıtın sağ taraf değerini

göstermektedir (Sarıaslan ve Karacabey, 2003).

Genetik Algoritma tekniği Bölüm 2'de ve Bayesci Optimizasyon Algoritması tekniği Bölüm 4'de açıklandığı için tekrar açıklanmamıştır.

5.2 Uygulama

Yeterli ve dengeli beslenmeyi sağlamak amacıyla kahvaltıda yenilen 20 adet yiyecek maddesi ve 10 adet besin öğesini kapsayan, günlük ortalama besin öğesi ihtiyacını karşılayacak şekilde gerçek veriler kullanılarak doğrusal programlama modeli kurulmuştur.

Karar değişkenleri olarak; Ekmek (x_1), Sucuk(x_2), Sosis(x_3), Salam(x_4), Süt(x_5), Beyaz Peynir(x_6), Kaşar Peynir(x_7), Tulum Peynir(x_8), Lor Peynir(x_9), Yumurta(x_{10}),

Tereyağ(x_{11}), Margarin(x_{12}), Domates(x_{13}), Salatalık(x_{14}), Siyah Zeytin(x_{15}), Yeşil Zeytin(x_{16}), Reçel(x_{17}), Bal(x_{18}), Tahin(x_{19}), Pekmez(x_{20}) alınmıştır.

Kahvaltıda yenilen yiyecek maddelerinin 100 gr'larına karşılık gelen fiyatlar (TÜİK, Tüketici Fiyat Endeksi, Nisan 2012 verileri) ve belirlenen yiyecek maddelerinin 100 gr'ları için besin bileşimleri (Baysal, 1995) Çizelge 5.1'te verilmiştir.

Çizelge 5.1 : Belirlenen yiyecek maddelerinin 100 gr için besin bileşimleri ve fiyatları

Yiyecek Maddeleri / Besin Bileşimleri	X_1 Ekmek	X_2 Sucuk	X_3 Sosis	X_4 Salam	X_5 Süt	X_6 Beyaz Peynir	X_7 Kaşar Peyniri	X_8 Tulum Peyniri	X_9 Krem Peynir	X_{10} Yumurta
Enerji (kcal)	276	179	322	450	50	289	404	215	72	158
Protein (g)	9.1	21	11.3	23.8	3.3	22.5	27	35	12.4	12.1
Yağ (g)	0.8	10.5	29.4	38.1	1.9	21.6	31.7	5.6	1	11.2
Karbonhidrat(g)	56.4	0	2.4	1.2	4.8	0	1.4	3.2	2.7	1.2
Kalsiyum (mg)	19	20	12	14	122	162	700	0	61	56
Demir (mg)	0,7	4.3	1.3	3.6	0.1	0.5	1	0	0.1	2.1
Potasyum(mg)	74	0	159	0	154	0	104	0	86	130
Sodyum (mg)	585	0	1	0	50	0	710	0	406	138
A vitamini	0	0	0	0	205	720	1000	1500	37	520
C vitamini	0	0	0	0	1	0	0	0	0	0
Fiyat (TL)	0.239	3.509	2.747	2.734	0.210	1.395	2.003	1.698	1.653	0.5

Çizelge 5.1 Devam : Belirlenen yiyecek maddelerinin 100 gr için besin bileşimleri ve fiyatları

Yiyecek Maddeleri / Besin Bileşimleri	X_{11} Tereyağı	X_{12} Margarin	X_{13} Domates	X_{14} Salatalık	X_{15} Siyah Zeytin	X_{16} Yeşil Zeytin	X_{17} Reçel	X_{18} Bal	X_{19} Tahin	X_{20} Pekmez
Enerji (kcal)	717	719	22	15	207	144	272	315	516	293
Protein (g)	0.9	0.9	1.1	0.9	1.8	1.5	0.6	0.3	10.5	0.6
Yağ (g)	81.1	80.5	0.2	0.1	21	13.5	0.1	0	28	0.1
Karbonhidrat(g)	0.1	0.9	4.7	3.4	1.1	2.8	70	78.4	53.5	70.6
Kalsiyum (mg)	24	30	13	25	77	90	20	15	91	400
Demir (mg)	0.2	0	0.5	1.1	1.6	2	1	0.8	9	10
Potasyum(mg)	26	43	244	160	0	0	88	0	0	0
Sodyum (mg)	826	943	3	6	0	0	12	0	0	0
A vitamini	3058	3307	900	250	60	300	10	0	0	0
C vitamini	0	0	23	11	0	0	2	4	0	0
Fiyat (TL)	2.162	0.618	0.236	0.186	1.158	1.040	0.910	2.435	1.117	1.002

Türkiye için önerilen günlük besin öğeleri tüketim standartlarından (Paker, 1996) elde edilen besin öğelerine ait günlük ağırlıklı ortalama ihtiyaç değerleri Çizelge 5.2'te verilmiştir.

Çizelge 5.2 : Besin öğelerinin günlük ağırlıklı ortalama ihtiyaç değerleri

Besin Öğeleri	Ağırlıklı Ortalama İhtiyaç
Enerji (kalori)	2246
Protein (g)	68
Yağ (g)	70
Karbonhidrat (g)	326
Kalsiyum (mg)	603
Demir (mg)	13
Potasyum (mg)	3500
Sodyum (mg)	2400
A vitamini	4485
C vitamini	61

Minimum maliyetle, günlük besin öğesi ihtiyaçlarını karşılayacak şekilde hangi besinlerden ne kadar yenmesi gerektiğiyle ilgili sabah kahvaltısı beslenme probleminin Doğrusal Programlama formülasyonu aşağıdaki gibidir. Modelimiz sabah kahvaltısı için kurulduğundan, Çizelge 5.2'de verilen ağırlıklı ortalamaların (bir günde üç öğün olduğu düşünülerek) 1/3'ü alınarak sağ taraf sabitleri belirlenmiştir. Amaç fonksiyonu ve kısıtlardaki katsayılar 1 grama karşılık gelen katsayılar olup Çizelge 5.1'de fiyatlar gösterilirken noktadan sonraki ilk üç rakam yazılmıştır.

Amaç Fonksiyonu :

$$\begin{aligned}\text{Min } z = {} & 0.00239020x_1 + 0.0350945x_2 + 0.0274729x_3 + 0.0273454x_4 + 0.00210370x_5 \\ & + 0.0139591x_6 + 0.0200363x_7 + 0.0169805x_8 + 0.0165302x_9 \\ & + 0.005x_{10} + 0.0216273x_{11} + 0.00618750x_{12} + 0.00236850x_{13} \\ & + 0.00186500x_{14} + 0.0115866x_{15} + 0.0104020x_{16} + 0.00910460x_{17} \\ & + 0.0243544x_{18} + 0.0111757x_{19} + 0.0100229x_{20}\end{aligned}$$

Kısıtlar;

Enerji Kısıtı :

$2.76x_1 + 1.79x_2 + 3.22x_3 + 4.50x_4 + 0.50x_5 + 2.89x_6 + 4.04x_7 + 2.15x_8 + 0.72x_9 + 1.58x_{10} + 7.17x_{11} + 7.19x_{12} + 0.22x_{13} + 0.15x_{14} + 2.07x_{15} + 1.44x_{16} + 2.72x_{17} + 3.15x_{18} + 5.16x_{19} + 2.93x_{20} \geq 750$

Protein Kısıtı :

$0.091x_1 + 0.21x_2 + 0.113x_3 + 0.238x_4 + 0.033x_5 + 0.225x_6 + 0.27x_7 + 0.35x_8 + 0.124x_9 + 0.121x_{10} + 0.009x_{11} + 0.009x_{12} + 0.011x_{13} + 0.009x_{14} + 0.018x_{15} + 0.015x_{16} + 0.006x_{17} + 0.003x_{18} + 0.105x_{19} + 0.006x_{20} \geq 23$

Kalsiyum Kısıtı :

$0.19x_1 + 0.20x_2 + 0.12x_3 + 0.14x_4 + 1.22x_5 + 1.62x_6 + 7x_7 + 0.61x_9 + 0.56x_{10} + 0.24x_{11} + 0.30x_{12} + 0.13x_{13} + 0.25x_{14} + 0.77x_{15} + 0.90x_{16} + 0.20x_{17} + 0.15x_{18} + 0.91x_{19} + 4x_{20} \geq 201$

Demir Kısıtı :

$0.007x_1 + 0.043x_2 + 0.013x_3 + 0.036x_4 + 0.001x_5 + 0.005x_6 + 0.01x_7 + 0.001x_9 + 0.021x_{10} + 0.002x_{11} + 0.005x_{13} + 0.011x_{14} + 0.016x_{15} + 0.02x_{16} + 0.01x_{17} + 0.008x_{18} + 0.09x_{19} + 0.10x_{20} \geq 5$

A vitamini Kısıtı :

$2.05x_5 + 7.20x_6 + 10x_7 + 15x_8 + 0.37x_9 + 5.20x_{10} + 30.58x_{11} + 33.07x_{12} + 9x_{13} + 2.50x_{14} + 0.6x_{15} + 3x_{16} + 0.10x_{17} \leq 1495$

Yağ Kısıtı :

$0.008x_1 + 0.105x_2 + 0.294x_3 + 0.381x_4 + 0.019x_5 + 0.216x_6 + 0.317x_7 + 0.056x_8 + 0.01x_9 + 0.112x_{10} + 0.811x_{11} + 0.805x_{12} + 0.002x_{13} + 0.001x_{14} + 0.21x_{15} + 0.135x_{16} + 0.001x_{17} + 0.28x_{19} + 0.001x_{20} \geq 24$

Karbonhidrat Kısıtı :

$0.564x_1 + 0.024x_3 + 0.012x_4 + 0.048x_5 + 0.014x_7 + 0.032x_8 + 0.027x_9 + 0.012x_{10} + 0.001x_{11} + 0.009x_{12} + 0.047x_{13} + 0.034x_{14} + 0.011x_{15} + 0.028x_{16} + 0.70x_{17} + 0.784x_{18} + 0.535x_{19} + 0.706x_{20} \geq 109$

Potasyum Kısıtı :

$0.74x_1 + 1.59x_3 + 1.54x_5 + 1.04x_7 + 0.86x_9 + 1.30x_{10} + 0.26x_{11} + 0.43x_{12} + 2.44x_{13} + 1.60x_{14} + 0.88x_{17} \leq 1167$

Sodyum Kısıtı :

$5.85x_1 + 0.01x_3 + 0.50x_5 + 7.10x_7 + 4.06x_9 + 1.38x_{10} + 8.26x_{11} + 9.43x_{12} + 0.03x_{13} + 0.06x_{14} + 0.12x_{17} \leq 800$

C Vitamini Kısıtı :

$0.01x_5 + 0.23x_{13} + 0.11x_{14} + 0.02x_{17} + 0.04x_{18} \geq 21$

Diğer Kısıtlar :

$x_1 \leq 150$, $x_2 \leq 100$, $x_3 \leq 100$, $x_4 \leq 100$, $x_5 \leq 100$, $x_6 \leq 100$, $x_7 \leq 100$, $x_8 \leq 100$, $x_9 \leq 100$, $x_{10} \leq 60$, $x_{11} \leq 100$, $x_{12} \leq 100$, $x_{13} \leq 100$, $x_{14} \leq 100$, $x_{15} \leq 100$, $x_{16} \leq 100$, $x_{17} \leq 100$, $x_{18} \leq 100$, $x_{19} \leq 100$, $x_{20} \leq 100$

$x_j \geq 0 \qquad j = 1,2, \dots, n$

WinQSB paket programı yardımıyla elde edilen doğrusal programlama çözüm sonuçları: Ekmek $(x_1) = 120.7768$, Süt $(x_5) = 83.1758$, Yumurta $(x_{10}) = 15.5074$, Margarin $(x_{12}) = 3.2020$, Domates $(x_{13}) = 90.9427$, Tahin $(x_{19}) = 60.5612$ ve amaç fonksiyonu (min.) değeri 1.4532 olarak bulunmuştur.

Problemin Genetik Algoritma çözümü ise Matlab (R2009b) programı Genetik Algoritma m – file dosyasına doğrusal programlama probleminin amaç fonksiyonu Şekil 5.1'deki gibi yazılarak elde edilmiştir.

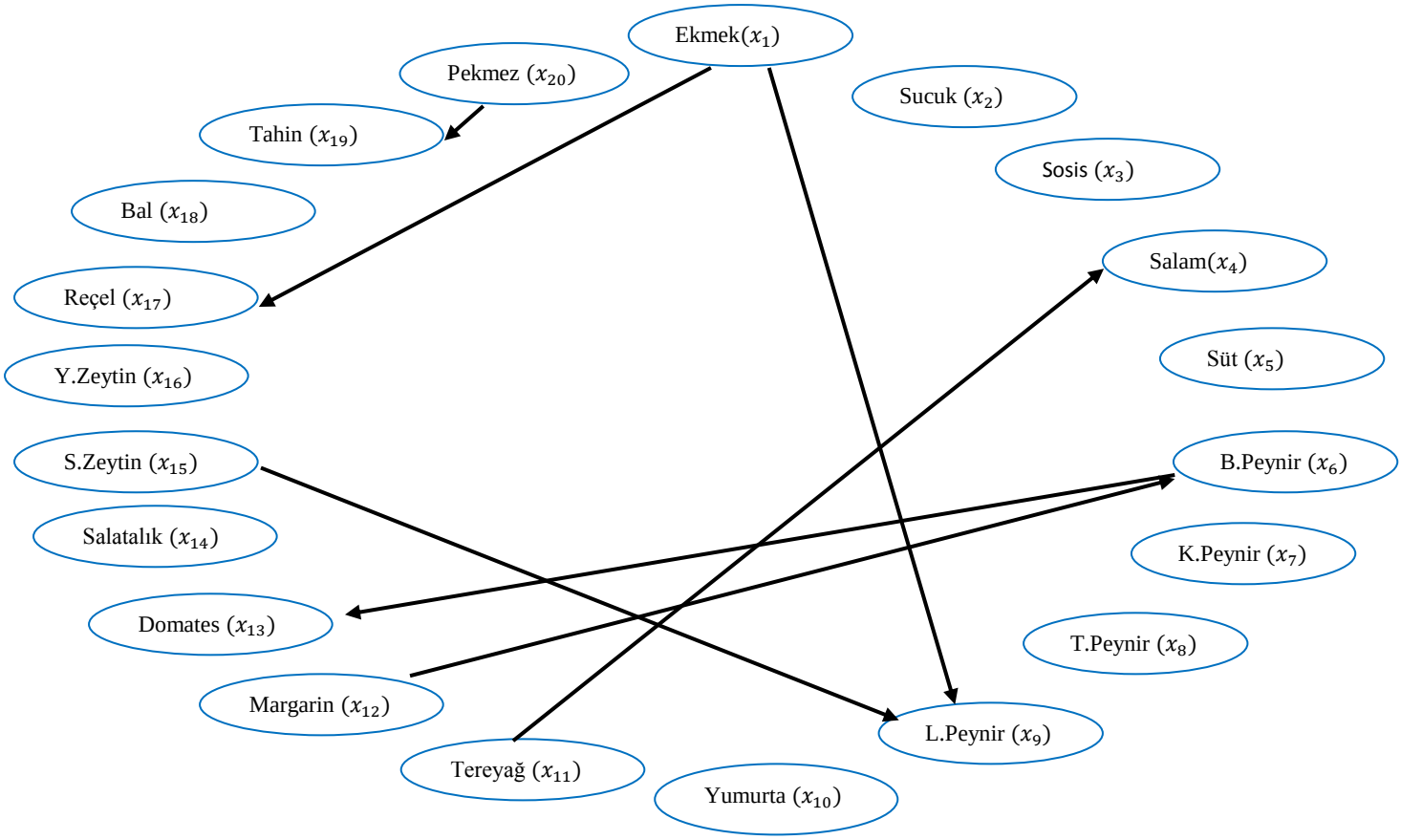
```
%function to minimize a linear equation
 function z = objfun(x)
 z = (0.00239020*x(1)+0.0350945*x(2)+0.0274729*x(3)+0.0273454*x(4)+0.00210370*x(5)
     +0.0139591*x(6)+0.0200363*x(7)+0.0169805*x(8)+0.0165302*x(9)+0.005*x(10)
     +0.0216273*x(11)+0.00618750*x(12)+0.00236850*x(13)+0.00186500*x(14)
     +0.0115866*x(15)+0.0104020*x(16)+0.00910460*x(17)+0.0243544*x(18)
     +0.0111757*x(19)+0.0100229*x(20));
```

Şekil 5.1 : Genetik algoritma m–file dosyası

Üreme fonksiyonu olarak rastgele bir seçim yapılarak "Uniform", seçim işlemi olarak problemin yapısına uygunluğundan dolayı "turnuva" yöntemi, çaprazlama işlemi için çeşitliliğin daha da artmasını sağlayacak "iki nokta" çaprazlama seçilmiştir. Çaprazlama ile çeşitlilik artacağından mutasyon için herhangi bir seçim yapılmayarak kısıtlara bağlı kalınmıştır. Ayrıca daha iyi sonuçlar elde etmek için "hybrid" kısmından "patternsearch" seçeneği işaretlenmiştir.

Modelin Genetik Algoritma çözüm sonuçları : Ekmek $(x_1) = 101.954$, Sosis $(x_3) = 0.125$, Salam$(x_4) = 0.125$, Süt$(x_5) = 99.998$, Beyaz Peynir$(x_6) = 1.515$, Kaşar Peynir$(x_7) = 0.291$, Tulum Peynir$(x_8) = 1.977$, Yumurta$(x_{10}) = 38.507$, Tereyağ$(x_{11}) = 0.001$, Margarin$(x_{12}) = 9.743$, Domates$(x_{13}) = 54.963$, Salatalık$(x_{14}) = 65.54$, Siyah Zeytin$(x_{15}) = 0.044$, Yeşil Zeytin$(x_{16}) = 0.028$, Reçel$(x_{17}) = 7.452$,Tahin$(x_{19}) = 29.598$, Pekmez$(x_{20}) = 28.629$ ve amaç fonksiyonu (min.) değeri 1.7131 olarak bulunmuştur.

Model Bayesci Optimizasyon Algoritması ile çözümlenmek istendiğinde ilk olarak Bayesci ağın oluşturulması gerekmektedir. Bayesci ağın oluşturulması R – Project programı yardımı ile yapılmıştır. Bayesci Ağ Şekil 5.2'teki gibi oluşturulmuştur.



Şekil 5.2 : Beslenme Problemi için Bayesci Ağ Yapısı

Daha sonra ağ yardımıyla her bir değişkenin birbiri ile ilişkilerinden yararlanarak koşullu olasılıklar yani Bayesci Ağ parametreleri Çizelge 5.3'teki gibi hesaplanmıştır.

Çizelge 5.3 : Beslenme Problemi için Bayesci Ağ Parametreleri

Karar Değişkeninin Olasılığı	**Olasılık Değeri**
$P(X_1)$	0.4949495
$P(X_2)$	0.4848485
$P(X_3)$	0.4848485
$P(X_4) = P(X_4\|X_{11})$	0.4
$P(X_5)$	0.4949495
$P(X_6) = P(X_6\|X_{12})$	0.6538462
$P(X_7) = P(X_7\|X_4)$	0.5740741
$P(X_8) = P(X_8\|X_7)$	0.4
$P(X_9) = P(X_9\|X_1, X_{15})$	0.6666667
$P(X_{10})$	0.5151515
$P(X_{11})$	0.5555556
$P(X_{12})$	0.5252525
$P(X_{13}) = P(X_{13}\|X_6)$	0.4897959
$P(X_{14})$	0.4949495
$P(X_{15})$	0.4747475
$P(X_{16})$	0.5454545
$P(X_{17}) = P(X_{17}\|X_1)$	0.4081633
$P(X_{18})$	0.4545455
$P(X_{19}) = P(X_{19}\|X_{20})$	0.3658537
$P(X_{20})$	0.4141414

Elde edilen parametre değerleri yardımıyla BOA'nın matlab programında çözümlenebilmesi için her bir parametre değerinin olasılıksal karşılığının amaç fonksiyonuna maliyet olarak yansıtılması gerekmektedir. Bu durumda, problemdeki her bir karar değişkeninin sahip olduğu olasılıksal değer ile 1 gramına karşılık gelen fiyat değeri çarpılarak amaç fonksiyonu katsayıları elde edilmiştir. Model Bayesci Optimizasyon Algoritması ile çözümlendiğinde; Ekmek$(x_1) = 0.413$, Sucuk$(x_2) = 0.166$, Beyaz Peynir$(x_6) = 0.257$, Kaşar Peynir$(x_7) = 0.233$, Tulum Peynir$(x_8) = 0.259$, Yumurta$(x_{10}) = 0.009$, Margarin$(x_{12}) = 0.042$, Reçel$(x_{17}) = 0.121$, Bal$(x_{18}) = 0.288$, Tahin$(x_{19}) = 0.206$, Pekmez$(x_{20}) = 0.107$ ve amaç fonksiyonu(min.) değeri 1.017 olarak bulunmuştur.

6 SONUÇ VE DEĞERLENDİRME

Bu çalışmada sabah kahvaltısı için beslenme problemi ele alınmıştır. Minimum maliyetle, günlük besin öğesi ihtiyaçlarını karşılayacak şekilde hangi besinlerden ne kadar yenmesi gerektiği üzerinde durulmuştur. Bu amaçla gerçek veriler kullanılarak doğrusal programlama modeli oluşturulmuş, Matlab programlama dili, WinQSB paket programı ve R – Project ile çözümlenmiştir.

Uygulamamızda elde edilen sonuçlara göre, sabah kahvaltısı beslenme problemi için doğrusal programlama yöntemiyle elde ettiğimiz çözümde yenmesi gereken maddeler ekmek(x_1), süt(x_5), yumurta(x_{10}), margarin(x_{12}), domates(x_{13}), tahin(x_{19}), olup, maliyeti 1.45 TL'dir. Genetik algoritma yöntemiyle elde ettiğimiz çözümde yenmesi gereken maddeler ekmek(x_1), sosis(x_3), salam(x_4), süt(x_5), beyaz peynir(x_6), kaşar peynir(x_7), tulum peynir(x_8), yumurta(x_{10}), tereyağ(x_{11}), margarin(x_{12}), domates(x_{13}), salatalık(x_{14}), siyah zeytin(x_{15}), yeşil zeytin(x_{16}), reçel(x_{17}), tahin(x_{19}), pekmez(x_{20}) olup, maliyeti 1.71 TL'dir. Bayesci Optimizasyon Algoritması yöntemiyle elde ettiğimiz çözümde yenmesi gereken maddeler ekmek(x_1), sucuk(x_2), beyaz peynir(x_6), kaşar peynir(x_7), tulum peynir(x_8), yumurta(x_{10}), margarin(x_{12}), reçel(x_{17}), bal(x_{18}), tahin(x_{19}) ve pekmez(x_{20}) olup, maliyeti 1.017 TL'dir. Maliyetin genetik algoritma sonucunda daha yüksek çıkması normaldir, çünkü yenmesi gereken besin sayısı da artmıştır. Başka bir açıdan bakarsak, Çizelge 5.4'de görüldüğü üzere, genetik algoritmanın 15sn'de çözdüğü beslenme problemini Bayesci Optimizasyon Algoritması 11sn'de, doğrusal programlama ise 35 sn'de çözmüştür. Yani, BOA, GA'ya ve WinQSB'ye göre çok daha hızlı bir sürede problemimize çözüm sağlamıştır. Günlük besin öğesi ihtiyaçlarından sapmalara baktığımızda ise sapma oranları GA ve BOA için aynı çıkarken doğrusal programlama da daha fazla çıkmıştır. Elde edilen çözüm sonuçları Çizelge 6.1'te ve Çizelge 6.2'te verilmiştir.

Çizelge 6.1 : Doğrusal programlama ve genetik algoritma çözüm sonuçları

Besin Öğesi	Tavsiye Edilen Miktar	Doğrusal Programlama		Genetik Algoritma		Bayesci Optimizasyon Algoritması	
		Bulunan Miktar	Sapma(%)	Bulunan Miktar	Sapma (%)	Bulunan Miktar	Sapma (%)
Enerji (kalori)	750	754.959	0.04	754.959	0.04	751.99	0.0199
Protein (gr)	23	22.999	0.00001	22.999	0.00001	23	0
Yağ (gr)	24	16.95	0.07	16.95	0.07	28.71	0.04
Karbonhidrat (gr)	109	109	0	109	0	108.99	0.000001
Kalsiyum (mg)	201	201	0	201	0	336.909	1.359
Demir (mg)	5	7.159	0.02	7.159	0.02	5.60	0.006
Potasyum (mg)	1167	460.90	7.061	460.90	7.061	529.72	6.37
Sodyum (mg)	800	802.45	0.0245	802.45	0.0245	799.99	0.00002
A vitamini	1495	1175.52	3.19	1175.52	3.19	1430.30	0.646
C vitamini	21	21.74	0.0074	21.74	0.0074	20.99	0.0000009
Beslenme Maliyeti		1.45		1.71		1.017	
Toplam Sapma		%10		8%		8%	
Çözüm Zamanı		35sn		15sn		11sn	

Çizelge 6.2 : Elde edilen besin öğeleri ve kahvaltı maliyeti

Besin Öğeleri	Doğrusal Programlama	Genetik Algoritma	Bayesci Optimizasyon Algoritması
Ekmek	✓	✓	✓
Sucuk			✓
Sosis		✓	
Salam		✓	
Süt	✓	✓	
Beyaz Peynir		✓	✓
Kaşar Peynir		✓	✓
Tulum Peynir		✓	✓
Lor Peyniri			
Yumurta	✓	✓	✓
Tereyağ		✓	
Margarin	✓	✓	✓
Domates	✓	✓	
Salatalık		✓	
Siyah Zeytin		✓	
Yeşil Zeytin		✓	
Reçel		✓	✓
Bal			✓
Tahin	✓	✓	✓
Pekmez		✓	✓
Fiyat	**1.45 TL**	**1.71 TL**	**1.01 TL**

Sonuç olarak besin öğesi ihtiyaçlarından sapmalar aynı çıksa da zaman avantajı sağlaması ve maliyeti azaltması bakımından sabah kahvaltısı için beslenme probleminin çözümünde Bayesci optimizasyon algoritması etkin bir biçimde kullanılabilir.

Son olarak, gelecek çalışmalarda çeşitli optimizasyon problemlerine Bayesci optimizasyon algoritması uygulanabilir. Bayesci optimizasyon algoritması geliştirilerek hiyerarşik Bayesci optimizasyon algoritması ele alınabilir. Bu tezde yer alan genetik algoritmalar ve BOA çeşitli problemlere uygulanarak problemlerin etkinlikleri incelenebilir.

KAYNAKLAR

Baluja S., 1994. Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning, Tech. Rep. CMU-CS-94- 163, Pittsburgh, PA.

Baluja S., 1995. An empirical comparison of seven iterative and evolutionary function optimization heuristics, Tech. Rep. CMU-CS-95-193, Carnegie Mellon University.

Baluja S., Davies S., 1997. Using optimal dependency-trees for combinatorial optimization: Learning the structure of the search space, In Proceedings of the 1997 International Conference on Machine Learning.

Baluja S., Davies S., 1997a. Combining multiple optimization runs with optimal dependency trees, *Technical Report CMU-CS-97-157*, School of Computer Science, Carnegie Mellon University.

Baluja S., Davies S., 1997b. Using optimal dependency-trees for combinational optimization, *ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning*, Morgan Kaufmann Publishers Inc, pp. 30–38.

Baysal A., 1995, Genel Beslenme, Hatipoğlu Yayınevi,Ankara.

Berny A., 2000. An adaptive scheme for real function optimization acting as a selection operator. In X. Yao, ed., First IEEE Symposium on Combinations of Evolutionary Computation and Neural Networks.

Bolat B., Osman K.E., Erdem C.İ., 2004. *Mühendislik uygulamalarında genetik algoritma ve operatörlerin işlevleri*, Mühendislik ve Fen Bilimleri Dergisi, Sigma 2004/4, ss.264-271.

Brucker P., Andreas D., Rolf M., Klaus N., Erwin P., 1999. Resource-constrained project scheduling: Notation, classification, models, and methods, *European Journal of Operational Research*, 112 (1), 3-41.

Cooper G.F., Herskovits E.A. 1992. A Bayesian method for the induction of probabilistic networks from data, *Machine Learning*, 9, 309–347.

Chickering D.M., Geiger D., Heckerman D., 1994. Learning Bayesian networks is NP-Hard, *Technical Report MSR-TR-94-17*, Microsoft, URL: *citeseer.ist.psu.edu/chickering94learning.html.*

de Bonet J.S., Isbell C.L., Jr. Viola, P., 1997. MIMIC: Finding optima by estimating probability densities. In M.C. Mozer, M.I. Jordan & T. Petsche, eds., Advances in Neural Information Processing Systems, vol. 9, The MIT Press.

Deb K., Goldberg D.E., 1991. *Analyzing deception in trap functions* (IlliGAL Report No. 91009). Urbana, IL: University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory.

Etxeberria R., Larra˜naga P., 1999. Global optimization with Bayesian networks, *II Symposium on Artificial Intelligence, CIMAF99*.

Fogel L.J., 1962. Autonomous automata, *Industrial Research* 4: 14–19.

Gali´c E., H¨ohfeld M., 1996. Improving the generalization performance of multi-layerperceptrons with population-based incremental learning. In Parallel Problem Solving from Nature. PPSN-IV , 740–750.

Goldberg, D. E. (1989a). Genetic algorithms in search, optimization, and machine learning. Reading, MA : Addison – Wesley.

Gonz´alez C., Lozano J., Larra˜naga P., 2001. Analyzing the PBIL algorithm by means of discrete dynamical systems. Complex Systems, 12.

Harik G.R., Lobo F.G., Goldberg D.E., 1997. The compact genetic algorithm, *IEEEEC*, 3, 287.

Haupt R.L., Sue Ellen H., 2004. *Practical Genetic Algorithms*, New Jersey: Second edition, Jhon Wiley & Sons Inc.

Hauschild M.W., Pelikan M., 2011. An introduction and survey of estimation of distribution algorithms, *Swarm and Evolutionary Computation*, 1(3), 111–128.

Hauschild M.W., Pelikan M., Sastry K., Lima C.F., 2009. Analyzing probabilistic models in hierarchical BOA, *IEEE Transactions on Evolutionary Computation*, 13(6), 1199–1217.

Henrion, M. (1988). Propagation of uncertainty in Bayesian Networks by logic sampling. In Lemmer, J.F., ve Kanal, L. N. (Eds.), Uncertainty in Artificial Intelligence (pp. 149 – 163). Amsterdam, London, New York : Elsevier.

Holland J. H., 1975. Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control, and artificial intelligence, University of Michigan Press, Ann Arbor. by John H. Holland.; Includes index.; Bibliography: p.175-177.

Hohfeld M., Rudolph G., 1997. Towards a theory of population-based incremental learning, In Proceedings of the 4th International Conference on Evolutionary Computation, 1–5, IEEE Press.

Howard R.A., Matheson J.E., 1981. Influence diagrams. In Howard, R. A., & Matheson, J. E. (Eds.), *Readings on the principles and applications of decision analysis*, Volume II (pp. 721–762). Menlo Park, CA: Strategic Decisions Group.

Inza I., Merino M., Larra˜naga P., Quiroga J., Sierra B., Girala M. 2001c. Feature subset selection by population-based incremental learning, A case study in the survival of cirrhotic patients with TIPS. Artificial Intelligence in Medicine.

Koza J.R., 1995. Two ways of discovering the size and shape of a computer program to solve a problem, *Proceedings of the Sixth International Conference on Genetic Algorithm*, ss.287-294.

Kruse R., Borgelt C., Klawonn F., Moewes C., Ruß G., Steinbrecher M., Held P., 2013. *Computational Intelligence-A Methodological Introduction*, Springer-Verlag, Berlin/ Heidelberg, Germany, ISBN 978-1-4471-5012-1.

Larranaga P., Lozano J.A., 2002. Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation, Kluwer Academic Publishers, Boston.

Larranaga P., Etxeberria R., Lozano J., Pe˜na J., 1999. Optimization by learning and simulation of bayesian and gaussian Networks, Tech. Rep. EHU-KZAA-IK-4/99, University of the Basque Country.

Li J., Aickelin U., 2003. *A Bayesian optimization algorithm for the nurse scheduling problem*. In Proceedings of Congress on Evolutionary Computation. IEEE Press, pp 2149-2156.

Lima C.F., Lobo F.G., Pelikan M., 2008. From mating pool distributions to model overfitting, In GECCO ’08: Proc. of the 10th annual conference on Genetic and evolutionary computation (pp. 431–438). New York, NY, USA: ACM.

Larranaga P., Etxeberria R., Lozano J.A., Pe˜na J.M., 2000. Combinatorial optimization by learning and simulation of Bayesian Networks, In Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence, 343–352, Stanford.

Mahnig T., M¨uhlenbein H., 2001a. Optimal mutation rate using Bayesian priors for estimation of distribution algorithms, *SAGA '01: Proceedings of the International Symposium on Stochastic Algorithms*, Springer-Verlag, London, UK, pp. 33–48.

Muhlenbein H., Mahnig T., 1999b. FDA - A scalable evolutionary algorithm for the optimization of additively decomposed functions, *Evolutionary Computation*, 7, 353 – 376.

Muhlenbein H., Paaß G., 1996. From recombination of genes to the estimation of distributions I. binary parameters, *PPSN IV: Proceedings of the 4th International Conference on Parallel Problem Solving from Nature*, Springer-Verlag, pp. 178–187.

Muhlenbein H., Mahnig T., Ochoa A.R., 1999. Schemata, distributions and graphical models in evolutionary optimization, *Journal of Heuristics*, 5, 215–247.

Muhlenbein H., Mahnig T., 1999a. Convergence theory and application of the factorized distribution algorithm, *Journal of Computing and Information Technology*, 7, 19–32.

Naphade K.S., David Wu S., Storer R.H., 1997. Problem space search algorithms for resource constrained project scheduling, *Annals of Operations Research*, 70, 307-326.

Ocenasek J., Schwarz J., 2000. *The parallel Bayesian optimization algorithm*, In Proceedings of the European Symposium on Computational Inteligence, Physica-Verlag, 61–67, Kosice, Slovak Republic.

Paker H.S., 1996. Besinlerin yenebilen 100 gramlarının enerji ve besin öğeleri değerleri, sporda beslenme, Gen Matbaacılık ve Reklamcılık, Ankara.

Pearl J., 1988. *Probabilistic Reasoning in Intelligent Systems*, Morgan Kaufman Publishers, Palo Alto, CA.

Pelikan M., 2005. Hierarchical Bayesian optimization algorithm – toward a new generation of evolutionary algorithms, Springer, Berlin Heidelberg New York.

Pelikan M., Sastry K., Goldberg D.E., 2002. Scalability of the Bayesian optimization algorithm, *International Journal of Approximate Reasoning* **,** 31(3), 221–258.

Pelikan M., Goldberg D.E., Cant´u-Paz E., 1998. *Linkage problem, distribution estimation, and Bayesian networks* (IlliGAL Report No. 98013). Urbana, IL: University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory.

Pelikan M., Goldberg D.E., Cant´u-Paz E., 1999. BOA: The Bayesian optimization algorithm. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99), I* , 525–532. Also IlliGAL Report No. 99003.

Pelikan M., Goldberg D.E., Cant'u-Paz E., 1999a. BOA: The Bayesian Optimization Algorithm. In W. Banzhaf et al., ed., Proceedings of the Genetic and Evolutionary Computation Conference GECCO99, vol. I, 525–532, Morgan Kaufmann Publishers, San Fransisco, CA.

Pelikan M., Goldberg D.E., Cant´u-Paz E., 2000a. Bayesian optimization algorithm, population sizing, and time to convergence. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000)*, 275–282. Also IlliGAL Report No. 2000001.

Pelikan M., Goldberg D.E., Lobo F., 2002. A survey of optimization by building and using probabilistic models, *Computational Optimization and Applications*, *21* (1), 5–20. Also IlliGAL Report No. 99018.

Pelikan M., Goldberg D.E., Cant´u-Paz E., 2000b. Linkage problem, distribution estimation, and Bayesian networks. *Evolutionary Computation*, *8* (3), 311–341. Also IlliGAL Report No. 98013.

Pelikan M., Goldberg D., 2003. Hierarchical BOA solves Ising spin glasses and MAXSAT, *Proceedings of the Genetic and Evolutionary Computation COnference (GECCO 2003)*, Springer-Verlag, pp. 1271–1282.

Pelikan M., Goldberg D.E., 2000. Hierarchical problem solving by the Bayesian optimization algorithm, *in* D. Whitley, D. Goldberg, E. Cant´u-Paz, L. Spector, I. Parmee & H.-G. Beyer (eds), *Proceedings of the Genetic and Evolutionary Computation COnference (GECCO 2000)*, Morgan Kaufmann, pp. 267–274.

Pelikan M., 2005. Hierarchical Bayesian Optimization Algorithms., Springer Verlag.

Pelikan M., Goldberg D.E., Cant´u-Paz E., 1999. BOA: The Bayesian optimization algorithm, *in* W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela & R. E. Smith (eds), *Proceedings of the Genetic and Evolutionary Computation COnference (GECCO 1999)*, Morgan Kaufmann Publishers, San Francisco, CA, p. 532.

Pelikan M., Kalapala R., Hartmann A.K., 2007. Hybrid evolutionary algorithms on minimum vertex cover for random graphs. Genetic and Evolutionary Computation Conference (GECCO-2007), 547–554.

Pelikan M., Muhlenbein H., 1999. The bivariate marginal distribution algorithm. In R. Roy, T. Furuhashi & P.K. Chawdhry, eds., Advances in Soft Computing – Engineering Design and Manufacturing, 521–535, Springer-Verlag, London.

Petrovski A., Shakya S., McCall J., 2006. Optimising cancer chemotherapy using an estimation of distribution algorithm and genetic algorithms. In proceedings of Genetic and Evolutionary Computation COnference (GECCO 2006) (in press), ACM, seattle, USA.

Rechenberg I., 1973. Evolutionsstrategie – Optimierung technisher Systeme nach Prinzipien der biologischen Evolution, Frommann-Holzboog, Stuttgart, GER.

Rissanen J.J., 1978. Modeling by shortest data description, *Automatica* 14: 465–471.

Rissanen J.J., 1989. *Stochastic complexity in statistical inquiry*, Singapore: World Scientific Publishing Co.

Rissanen J.J., 1996. Fisher information and stochastic complexity, *IEEE Transactions on Information Theory*, *42* (1), 40–47.

Santana R., 2003a. A Markov network based factorized distribution algorithm for optimization, *Proceedings of the 14th European Conference on Machine Learning (ECMLPKDD 2003); Lecture Notes in Artificial Intelligence*, Vol. 2837, Springer-Verlag, Berlin, pp. 337–348.

Santana R., 2005. Estimation of distribution algorithms with Kikuchi approximations, *Evolutionary Computation*, 13(1), 67–97.

Sarıaslan H., Karacabey A.A., 2003. *İşletmelerde Sayısal Analizler*, Ankara; Turhan Kitabevi.

Saruhan H., "An Evolutionary Algorithm in Mechanical Design Problems",*4th International Advanced Technologies Symposium*, September 28-30, Konya, Türkiye, 1-10 (2005).

Sastry K., Goldberg D.E., 2004. Designing competent mutation operators via probabilistic model building of neighborhoods, Tech. Rep. 2004006, IlliGAL.

Shengyao W., Ling W., Gang Z., Ye X.. 2012. *An Estimation of Distribution Algorithm for the Flexible Job-Shop Scheduling Problem*, Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence Lecture Notes in Computer Science Volume 6839, 2012, pp 9-16

Syswerda G., 1989. Uniform crossover in genetic algorithms, Proc. of the Int. Conf. on Genetic Algorithms (ICGA-89), 2–9.

TÜİK, 2012. Tüketici Fiyat Endeksi, http://www.tuik.gov.tr, (Ziyaret tarihi: 03 Haziran 2013).

Wang P.Y., Ming L., 2002, *Genetic algorithm optimized resource activity critical path method*, Proceedings of the First Conference on Machine Learning and Cybernetics, 4, 4-5 November, ss.1978-1982.

Zhang Q., Muehlenbein H., 2004. On the convergence of a class of estimation of distribution algorithms, *IEEE Trans. on Evolutionary Computation*, 8,2.

Zhang Q., M¨uhlenbein H., 1999. On global convergence of FDA with proportionate selection. In Second Symposium on Artificial Intelligence. Adaptive Systems. CIMAF 99 , 340–343, la Habana.

ÖZGEÇMİŞ

Ad – Soyad : Serpil Gümüştekin
Doğum yeri ve tarihi : Ankara, 09.06.1988
Adres : Ondokuz Mayıs Üniversitesi, Fen Edebiyat Fakültesi, İstatistik Bölümü
Telefon : 03623121919/5217
Lisans : Karadeniz Teknik Üniversitesi / İstatistik ve Bilgisayar Bilimleri
Yabancı Dil : İngilizce