BAŞKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GENELLEŞTİRİLMİŞ YAYILMA PROBLEMİ İÇİN KARMA ÇÖZÜM YÖNTEMİ

HYBRID METHOD FOR THE GENERALIZED MINIMUM SPANNING TREE PROBLEM

YİĞİT KORAY GENÇ

YÜKSEK LİSANS TEZİ
ANKARA
AĞUSTOS, 2007

GENELLEŞTİRİLMİŞ YAYILMA PROBLEMİ İÇİN KARMA ÇÖZÜM YÖNTEMİ

HYBRID METHOD FOR THE GENERALIZED MINIMUM SPANNING TREE PROBLEM

YİĞİT KORAY GENÇ

Başkent Üniversitesi
Lisansüstü Eğitim Öğretim ve Sınav Yönetmeliğinin
ENDÜSTRİ Mühendisliği Anabilim Dalı İçin Öngördüğü
YÜKSEK LİSANS TEZİ
olarak hazırlanmıştır.

2007

Fen Bilimleri Enstitüsü Müdürlüğü'ne,

Bu çalışma, jürimiz tarafından **ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI 'nda YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Başkan : Prof.Dr. Berna DENGİZ .....

Üye : Prof.Dr. İmdat KARA .....

Üye : Prof.Dr. Fulya ALTIPARMAK .....

ONAY

Bu tez 02/08/2007 tarihinde Enstitü Yönetim Kurulunca belirlenen yukarıdaki jüri üyeleri tarafından kabul edilmiştir.

...../...../.....

Prof.Dr. Emin AKATA

FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRÜ

TEŞEKKÜR

Bu çalışmada, değerli görüş ve katkılarıyla beni yönlendiren ve her zaman bana destek olan hocam Sayın Prof.Dr. Berna DENGİZ'e,

Karşılaştığım güçlüklerin aşılmasında bana yol gösterici olan hocam Sayın Prof.Dr. İmdat KARA'ya,

Çalışmam sırasında beni hoşgörü ve anlayışla karşılayan hocam Sayın Prof.Dr. Neş'e ÇELEBİ'ye

Her türlü problemimde yanımda olduklarını bildiğim arkadaşlarım Sayın Hande TEMELOĞLU'na, Sayın Arş.Gör. Hüseyin GÜDEN'e, Sayın Arş.Gör. Uğur BAÇ'a, Sayın Öğr.Gör. Yavuz Selim ÖZDEMİR'e ve değerli hocam Sayın Öğr.Gör. Arzu AKYÜZ'e,

Sunumumda bulunan değerli jüri üyelerine,

Ve benim için hiçbir fedakarlıktan kaçınmayan aileme sonsuz teşekkürlerimi sunarım.

ÖZ

GENELLEŞTİRİLMİŞ YAYILMA PROBLEMİ İÇİN KARMA ÇÖZÜM YÖNTEMİ

Yiğit Koray Genç

Başkent Üniversitesi Fen Bilimleri Enstitüsü

Endüstri Mühendisliği Anabilim Dalı

Bu çalışmada, günümüzde şebeke tasarım problemlerinde kullanılan Genelleştirilmiş Yayılma Problemi için yeni sezgisel çözüm yöntemleri geliştirilmiştir. Genetik Algoritma, Tavlama Benzetimi ve Kuş Sürüsü Algoritmasına dayalı yeni sezgisel çözüm yöntemleri geliştirilmiştir. Geliştirilen Algoritmalar, literatürde yer alan test problemleri üzerinde denenmiş ve performansları incelenmiştir.

Geliştirilen Kuş Sürüsü Algoritması, Genelleştirilmiş Yayılma Problemi için ilk kez bu çalışmada önerilmiştir. Geliştirilen Genetik Algoritmanın performansının iyileştirilmesi amacıyla, yerel arama algoritmalarından biri olan Tavlama Benzetimi ile birlikte kullanılabileceği bir melez algoritma önerilmiştir.

Geliştirilen algoritmaların parametre değerlerinin en iyi kombinasyonunun belirlenmesi için Cevap Yüzeyi metodu kullanılmıştır.

ANAHTAR SÖZCÜKLER: Genelleştirilmiş Yayılma Problemi, Genetik Algoritma, Kuş Sürüsü Algoritması.

Danışman: Prof.Dr. Berna DENGİZ, Başkent Üniversitesi, Endüstri Mühendisliği Bölümü.

ABSTRACT

HYBRID METHOD FOR THE GENERALIZED MINIMUM SPANNING TREE PROBLEM

Yiğit Koray Genç

Başkent University, Instıtute of Science

Department of Industrial Engineering

In this study, new heuristic methods based Genetic Algorithms, Simulated Annealing and Particle Swarm Optimization Algorithm are developed for Generalized Minimum Spanning Tree Problem. The performances of algorithms have been evaluated on the test problems given in the literature.

Particle Swarm Optimization Algorithm is firstly used for the Generalized Minimum Spanning Tree Problem in this study. In addition, a hybrid algorithm based on Genetic Algorithms and Simulated Annealing is proposed to improve the performance of the Genetic Algorithm.

To determine the best combination of the parameter values of the developed algorithms, response surface method is applied.

KEY WORDS: Generalized Minimum Spanning Tree Problem, Genetic Algorithm, Particle Swarm Optimization Algorithm

Adviser: Prof.Dr. Berna DENGİZ, Başkent University, Department of Industrial Engineering.

İÇİNDEKİLER LİSTESİ

Sayfa

TEŞEKKÜR..... i
ÖZ..... ii
ABSTRACT..... iii
İÇİNDEKİLER LİSTESİ..... iv
ŞEKİLLER LİSTESİ.....v
ÇİZELGELER LİSTESİ.....vi
SİMGELER VE KISALTMALAR LİSTESİ.....vii

1 GİRİŞ.....1

2 GENELLEŞTİRİLMİŞ YAYILMA PROBLEMİ.....3

2.1 Temel Yapısı ….....3
2.2 Uygulama Alanları 5
2.3 Enküçük Yayılan Ağaç ……6
2.3.1 Enküçük Yayılma Probleminin Matematiksel Modeli…..…....6
2.4 Enküçük Yayılma Probleminin Çözümünde Kullanılan Bazı……...7
Algoritmalar
2.4.1 Prim Algoritması …….....8
2.4.2 Kruskal Algoritması ……...9
2.5 Genelleştirilmiş Yayılma Problemi İçin Kaynak Taraması……...9
2.5.1 Matematiksel Modeller…….....10
2.5.2 Sezgisel Yöntemler……..…….....12

3 GENELLEŞTİRİLMİŞ YAYILMA PROBLEMİ İÇİN GELİŞTİRİLEN…….. 14 GENETİK ALGORİTMA, MELEZ ALGORİTMA VE KUŞ SÜRÜSÜ ALGORİTMASI

3.1 Genelleştirilmiş Yayılma Problemi İçin Geliştirilen Genetik Algoritma.. 14
3.1.1 Dizi Yapısı…….....17
3.1.2 Başlangıç Yığınının Oluşturulması…….....17
3.1.3 Değerlendirme Fonksiyonu…….....18
3.1.4 Yeniden Üretim Operatörü……18
3.1.4.1 Orantılı Yeniden Üretim Mekanizmaları…….....19

3.1.4.1.1 Rulet Çemberi Seçim Mekanizması…….. 19
3.1.4.1.2 Stokastik Artan Seçim Mekanizması……....20
3.1.4.1.3 Stokastik Üniversal Seçim Mekanizması…..20
3.1.4.2 Sıralı Seçim Mekanizması……..20
3.1.4.3 Turnuva Seçim Mekanizması……20
3.1.4.4 Denge Durumu Seçim Mekanizması……21
3.1.4.5 $(\mu + \lambda)$ Seçim mekanizması……....21
3.1.4.6 (μ, λ) Seçim mekanizması……21
3.1.5 Genetik Operatörler……... 21
3.1.5.1 Çaprazlama Operatörü……...22
3.1.5.2 Mutasyon Operatörü……24
3.2 Geliştirilen Genetik Algoritmanın Yapısı…….25
3.2.1 Dizi Gösterimi……..25
3.2.2 Başlangıç Yığınının Oluşturulması……....26
3.2.3 Kullanılan Amaç ve Uygunluk Fonksiyonu…….....27
3.2.4 Çaprazlama Operatörü…….....27
3.2.5 Mutasyon Operatörü……..28
3.2.6 Geliştirilen Genetik Algoritma …..…….....28
3.3 Genelleştirilmiş Yayılma Problemi İçin Geliştirilen Tavlama…….30
Benzetimi Algoritması
3.3.1 Geliştirilen Tavlama Benzetimi Algoritması……33
3.4 Genelleştirilmiş Yayılma Problemi İçin Bir Melez Yaklaşım…….. 34
3.4.1 Geliştirilen Melez Algoritma……...35
3.5 Genelleştirilmiş Yayılma Problemi İçin Kuş Sürüsü Algoritması……...36
3.5.1 Kuş Sürüsü Algoritması İle İlgili Temel Kavramlar……37
3.5.2 Geliştirilen Kuş Sürüsü Algoritması……..... 40

4 GELİŞTİRİLEN ALGORİTMALAR İÇİN UYGUN PARAMETRE……..…… 43 KÜMELERİNİN BELİRLENMESİ

4.1 Geliştirilen Genetik Algoritma İçin Eniyi Parametre……. 43
Kümesinin Belirlenmesi
4.2 Geliştirilen Melez Yöntem İçin Eniyi Parametre Kümesinin…… 46
Belirlenmesi

4.3 Geliştirilen Kuş Sürüsü Algoritması İçin Eniyi Parametre…….....50
Kümesinin Belirlenmesi

5 UYGULAMA VE TARTIŞMA……54

6 SONUÇ……..59

KAYNAKLAR LİSTESİ…….61
EK-1……65
EK-2……66

ŞEKİLLER LİSTESİ

Sayfa

Şekil 2.1 GYP serimi örneği..... 4
Şekil 2.2 GYP için bir uygun çözüm..... 4
Şekil 2.3 İlişkilendirilen düğümlerin oluşturduğu serim..... 5
Şekil 2.4 Prim Algoritması..... 8
Şekil 2.5 Kruskal Algoritması..... 9
Şekil 3.1 Standart bir GA'nın adımları..... 16
Şekil 3.2 Tek noktadan çaprazlama örneği..... 22
Şekil 3.3 İki noktadan çaprazlamaya örnek..... 23
Şekil 3.4 Uniform çaprazlamaya örnek..... 24
Şekil 3.5 Mutasyona örnek..... 24
Şekil 3.6 GYP örneği..... 25
Şekil 3.7 GA'da kullanılan dizi örneği..... 26
Şekil 3.8 Kullanılan çaprazlama operatörü..... 27
Şekil 3.9 Kullanılan mutasyon operatörü..... 28
Şekil 3.10 GYP için geliştirilen GA'nın adımları..... 29
Şekil 3.11 Standart bir Tavlama Benzetimi algoritması..... 32
Şekil 3.12 Tavlama Benzetimi algoritmasında komşu üretme..... 33
Şekil 3.13 GYP için geliştirilen Tavlama Benzetimi algoritması..... 34
Şekil 3.14 Geliştirilen melez yöntemin adımları..... 36
Şekil 3.15 Standart bir KSA'nın adımları..... 39
Şekil 3.16 Üç kümeli, sekiz düğümlü GYP örneği..... 40
Şekil 3.17 GYP için geliştirilen KSA..... 42
Şekil 4.1 Yığın genişliğinin algoritma performansı üzerine etkisi..... 45
Şekil 4.2 Genetik Algoritma için küp grafiği..... 45
Şekil 4.3 GA için elde edilen cevap yüzeyi..... 46
Şekil 4.4 Başlangıç sıcaklığının algoritma performansı üzerine etkisi..... 48
Şekil 4.5 Her sıcaklık algoritma performansı üzerine etkisi..... 48
Şekil 4.6 Tavlama Benzetimi için küp grafik..... 49
Şekil 4.7 Tavlama Benzetimi için etki yüzeyi..... 49
Şekil 4.8 c_1 faktörünün algoritma performansı üzerine etkisi..... 51
Şekil 4.9 Kuş Sürüsü Algoritması için küp grafiği..... 52

Şekil 4.10 Kuş Sürüsü Algoritması için cevap yüzeyi..... 52
Şekil 5.1 15 Test Problemi İçin Algoritma Sonuçları…… 54
Şekil 5.2 GA'nın 11eil51 Problemi İçin Yakınsama Grafiği…… 55
Şekil 5.3 GA'nın 20kroa100 Problemi İçin Yakınsama Grafiği…….. 55
Şekil 5.4 Melez Yöntemin 11eil51 problemi için yakınsama grafiği…….. 56
Şekil 5.5 Melez Yöntemin 20kroa100 problemi için yakınsama grafiği…… 56
Şekil 5.6 KSA'nın 11eil51 problemi için yakınsama grafiği…… 57
Şekil 5.7 KSA'nın 20kroa100 problemi için yakınsama grafiği…….. 57

ÇİZELGELER LİSTESİ

Sayfa

Çizelge 3.1: Tavlamada adı geçen işlem ve durumların TB'deki karşılıkları..... 30
Çizelge 3.2: Örnek kuş gösterimi..... 41
Çizelge 4.1: Algoritmaların kullandığı parametreler..... 43
Çizelge 4.2: GA için deney tasarımı ve sonuçları..... 44
Çizelge 4.3: GA için elde edilen ANOVA çizelgesi.....44
Çizelge 4.4: TB için deney tasarımı ve sonuçları..... 47
Çizelge 4.5: Geliştirilen TB için elde edilen ANOVA çizelgesi.....47
Çizelge 4.6: KSA için deney tasarımı ve sonuçları..... 51
Çizelge 4.7: KSA için elde edilen ANOVA tablosu..... 51
Çizelge E1 : Algoritma Performanslarının Karşılaştırma Çizelgesi……EK-1
Çizelge E2 : Geliştirilen Melez Yöntemin Performansının Literatürdeki……EK-2
Algoritmaların Performansları İle Karşılaştırması

SİMGELER VE KISALTMALAR LİSTESİ

$G = (N, A)$: N düğümler kümesi, A ayrıtlar kümesi olmak üzere G serimi

p_c : Çaprazlama olasılığı

p_m : Mutasyon olasılığı

μ : Ebeveynler

λ : Çocuklar

r : Her sıcaklık derecesinde tekrar sayısı

t : Başlangıç sıcaklığı değeri

x_{ijs}^k : k.çevrimde i.kuşun j.kümesinin s.elemanının pozisyon değeri

x_{mak} : Herhangi bir kuşun pozisyon değerinin alabileceği en yüksek değer

$x_{\min}$: Herhangi bir kuşun pozisyon değerinin alabileceği en küçük değer

v_{ijs}^k : k.çevrimde i.kuşun j.kümesinin s.elemanının hız değeri

v_{mak} : Herhangi bir kuşun hız değerinin alabileceği en yüksek değer

$v_{\min}$: Herhangi bir kuşun hız değerinin alabileceği en küçük değer

pb_{ij}^k : k.çevrime kadar i.kuşun eniyi amaç fonksiyonu değerini aldığı durumda j.kümesinin ilk konum değeri

gb_j^k : k.çevrime kadar sürüdeki tüm kuşların içinden eniyi amaç fonksiyonu değerini alan kuşun j.kümesinin ilk konum değeri

w : Eylemsizlik faktörü (0.4-1.4)

c_1 : Öğrenme faktörü (1.5-2)

c_2 : Öğrenme faktörü (2-2.5)

k : Çevrim sayacı

k_{mak} : Algoritmanın çalıştırılacağı çevrim sayısı

r_1, r_2 : *U(0,1)* olan rassal sayılar

GA : Genetik Algoritma

KSA : Kuş Sürüsü Algoritması

TB : Tavlama Benzetimi

MY : Melez Yöntem

1. GİRİŞ

Bütünleşen Dünya'mızda küreselleşmeyle giderek artan ve büyük boyutlara ulaşan sanayi ve ticaret hacmine bağlı olarak, teknoloji de büyük bir hızla ilerleme göstermektedir. İletişim sistemleri, bilgisayar sistemleri, petrol-doğalgaz boru hatları, sulama kanalları, elektronik devreler, otoyollar, bilgi bankaları bu teknolojik gelişmelere bağlı olarak ortaya çıkmış sistemlerdir. Yüksek maliyetlerle kurulan bu sistemlerin tasarımı, günümüz koşullarında bir rekabet unsuru olarak ortaya çıkmaktadır. Bu tür sistemler genellikle serim problemi olarak modellenebilmektedir. Düşük maliyetle serimlerin oluşturulabilmesi için literatürde yapılmış çok sayıda çalışma bulunmaktadır. Ancak problemlerin türüne bağlı olarak ortaya çıkan bazı özel koşullar, yeni çalışmalar yapılmasını zorunlu kılmaktadır.

Bu nedenle bu çalışmada, yukarıda bahsedilen sistemlerin birçoğunun enküçük maliyetle oluşturulmasında kullanılabilen Genelleştirilmiş Yayılma Problemi için yeni çözüm yaklaşımları geliştirilmiştir. Bu algoritmalar son yıllarda yaygın olarak kullanılmakta olan Genetik Algoritmalara (GA) dayanan iki farklı algoritma ile, henüz birçok probleme uygulanmamış olan Kuş Sürüsü Algoritması (KSA) dır.

Birleşi eniyileme probleminde başarıyla uygulanmış olan GA ile genellikle sürekli eniyileme problemlerinde kullanılan KSA sezgisel yöntemler olup, her zaman eniyi sonucu veremeyebilirler. Her ikisi de evrimsel algoritmalar grubunda anılan GA ve KSA ile GYP'nin çözümü için geliştirilen algoritmalar literatürde yer alan test problemleri üzerinde denenerek çözüm kalitesi ve çözüm zamanı açısından karşılaştırılmıştır. Algoritmalar, GYP'nin Enküçük Yayılma Problemi (EYP) olarak modellenebilme özelliğinden yararlanılarak geliştirilmiştir. Buna göre, literatürde EYP için geliştirilmiş son derece etkin bir algoritma olan Prim Algoritmasının kullanılmasıyla GYP'nin çözümü elde edilmiştir.

Çalışmanın ikinci bölümünde, GYP tanımı, uygulama alanları, EYP ile ilişkisi, EYP için geliştirilmiş algoritmalar ve GYP için literatür taraması yer almaktadır.

Üçüncü bölümde, Genetik Algoritma (GA) ların tarihçesi, temel yapısı, GA'da kullanılan temel kavramlar, başlangıç yığınının oluşturulması, değerlendirme fonksiyonu, yeniden üretim operatörü ve genetik operatörler incelenmiş ve GYP için geliştirilen algoritma verilmektedir. Daha sonra Tavlama Benzetimi (TB) algoritmasının temel yapısı, GA'nın birlikte kullanılmasıyla geliştirilen melez algoritma yer almaktadır. Ayrıca Kuş Sürüsü Algoritması (KSA)'nın temel yapısı ve GYP için geliştirilen KSA yer almaktadır.

Dördüncü bölümde, geliştirilen algoritmaların eniyi parametre kümelerinin belirlenmesi amacıyla kullanılan Cevap Yüzeyi metodu yer almaktadır.

Çalışmanın beşinci bölümünde ise algoritmaların, test problemleri üzerindeki performansları ve ardından sonuç bölümü yer almaktadır.

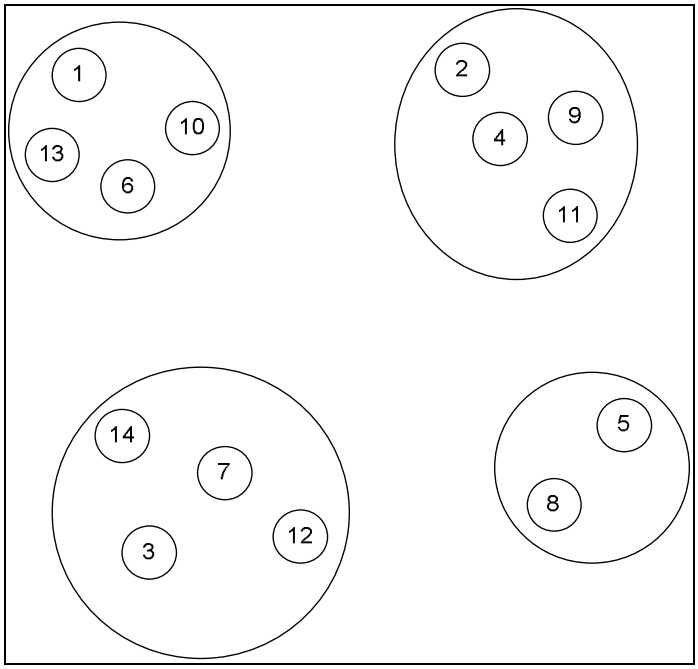
2. GENELLEŞTİRİLMİŞ YAYILMA PROBLEMİ

İletişim hattı tasarımı, elektrik şebekesi tasarımı, doğalgaz-petrol boru hattı tasarımı, yerel hatların birleştirilmesi gibi gerçek hayatta karşılaşılan problemlerin, EYP olarak çözümünün aranması bazen asıl problemi iyi ifade etmemektedir. Bu durum, serimde yer alan düğümlerin, her kümede en az bir düğüm olacak şekilde kümelere ayrılmasıyla ortaya çıkmaktadır. EYP'den farklı olarak tüm düğümlerin değil, tüm kümelerin birbiriyle enküçük maliyetle ilişkilendirilmesi istenmektedir. Örneğin, bir kampus içinde yerleşmiş bir üniversitenin fakültelerini, enstitülerini birbirine bağlayacak omurga ağın tasarımı problemini EYP olarak çözmek uygun olmayacaktır. Bu problem, her birimden bir noktayı birbirine bağlayacak EYP olarak çözülebilir.

Bu problem, Genelleştirilmiş Yayılma Problemi (GYP) adıyla 1995 yılında Myung, Lee ve Tcha tarafından tanımlanmıştır [21]. Aynı çalışmada bu problemin EYP'den farklı olarak NP-zor bir problem olduğu da ispatlanmıştır.

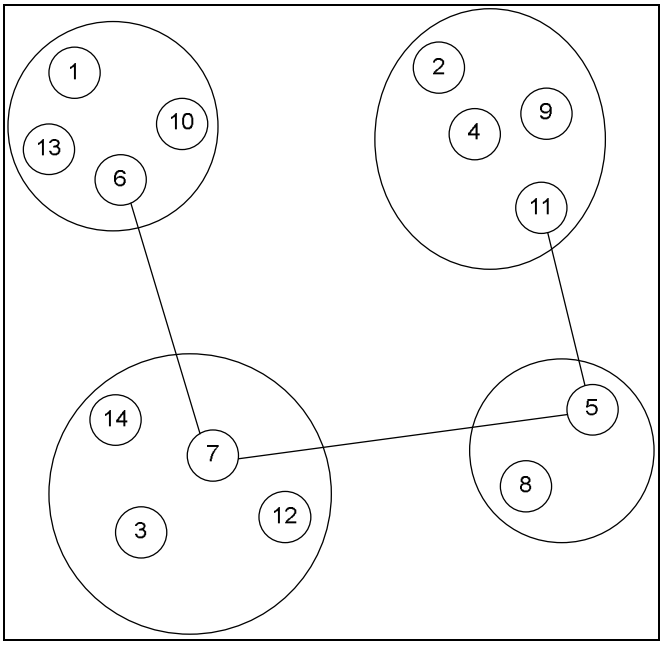
2.1 Temel Yapısı

GYP, EYP'de tanımlanan N düğümler kümesinin alt kümelere (cluster) ayrılması durumunun ele alınmasıyla ortaya çıkmıştır. GYP, EYP'nin genelleştirilmiş hali olarak ifade edilmektedir. GYP'de, serimde bulunan her düğüm kümesinde en az bir adet düğümün bulunması gerekir [17]. Ayrıca, kümeleri birbiriyle ilişkilendirme, her kümenin yalnızca bir düğümü üzerinden gerçekleştirilmelidir. GYP için örnek serim gösterimi Şekil 2.1'de görülmektedir.



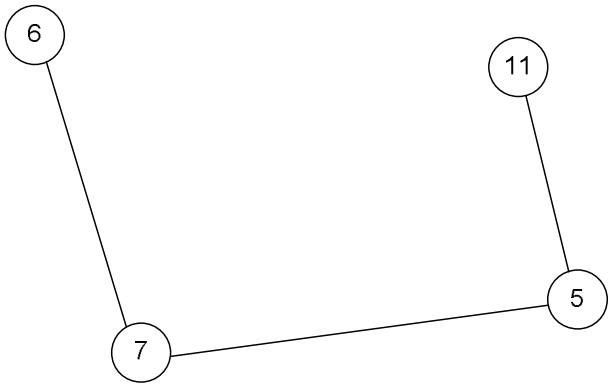
Şekil 2.1 GYP serimi örneği

Şekil 2.1'de gösterilen serim, dört küme içinde yer alan toplam on-dört adet düğümden oluşmaktadır. Bu kümelerin enküçük maliyetle/uzaklıkla ilişkilendirilmesi problemi GYP'dir. GYP için uygun bir çözüme örnek Şekil 2.2'de görülmektedir. Şekilden görüleceği gibi küme sayısının bir eksiği kadar ayrıt kullanılmış, her kümeden sadece bir düğüm kullanılarak bağlantı yapılmış ve ilişkilendirilmemiş küme kalmamıştır.



Şekil 2.2 GYP için bir uygun çözüm

Burada dikkat edilmesi gereken bir diğer nokta, her kümeden sadece bir düğümün seçilerek, bu düğümlerin birbiriyle enküçük maliyetle ilişkilendirilmesidir. Görüleceği üzere bu problem bir EYP'dir. Şekil 2.2'de gösterilen serimin GYP'nin en iyi çözümü olduğunu varsayarsak ve sadece ilgili düğümleri göz önüne alırsak, Şekil 2.3'te gösterilen serim ortaya çıkmaktadır.



Şekil 2.3 İlişkilendirilen düğümlerin oluşturduğu serim

GYP için eniyi çözüm olduğu bilinen bu serim, sadece ilgili düğümler dikkate alınırsa EYP'ye dönüşmektedir. Buna göre, Şekil 2.3'te gösterilen serim EYP'nin çözümü olmaktadır. Dolayısıyla, her kümeden birer düğüm seçilmesi ve bu düğümler için EYP'nin çözülmesiyle, GYP problemi için uygun çözümler elde edilebilmektedir.

EYP ise Bölüm 2.4'te verilen Prim veya Kruskal algoritması kullanılarak çözülebilmektedir. Bu algoritmalar kurucu sezgisel algoritmalar olup, EYP için en iyi çözümü vermektedirler [13].

2.2 Uygulama Alanları

GYP, sulama şebekesi tasarımı, iletişim ağı tasarımı, yerel iletişim ağlarının birleştirilmesi, petrol-doğalgaz boru hattı tasarımı, mağaza zincirleri için yer seçimi, dağıtım merkezi seçimi, otoyol tasarımı vb. durumlarda uygulanabilir bir problem türüdür [22].

Literatürde bu problemlerin bazıları için yapılmış çalışmalar bulunmaktadır. Örneğin, Dror, Haouari ve Chaouachi [7] bir tarımsal sulama şebekesi tasarımı problemini GYP olarak ele almışlardır. Bu çalışmada suyun kaynaktan, parsellere ayrılmış sulama alanındaki her parsele, toplam uzaklığı enküçük olan bir şebeke ile dağıtılması amaçlanmıştır.

2.3 Enküçük Yayılan Ağaç

Düğümleri birleştiren ayrıtların toplam uzunluklarının yada maliyetlerinin enküçüklenmesi amacını dikkate alarak oluşturulan yayılan ağaca Enküçük Yayılan Ağaç (EYA) denir [25, 27]. Bu probleme ise literatürde Enküçük Yayılma Problemi (EYP) denilmektedir. EYA, iletişim ağlarının, dağıtım sistemlerinin ve benzer serimlerin tasarımında kullanılmaktadır [5]. Bir serimde birden fazla EYA olabilir. Bu durum serimde aynı ağırlık (uzaklık) değerine sahip birden fazla ayrıt mevcut ise ortaya çıkmaktadır. Birden fazla EYA bulunan serimlerde herhangi bir EYA seçilerek kullanılabilir. Literatürde EYA problemin çözümü için geliştirilmiş matematiksel modeller ve etkin sezgisel algoritmalar bulunmaktadır.

2.3.1 Enküçük Yayılma Probleminin Matematiksel Modeli

EYP için geliştirilmiş matematiksel modelde kullanılan karar değişkenleri ve parametreler aşağıda verilmektedir.

Karar Değişkenleri

x_{ij} : *i.* düğüm ile *j.*düğüm arasında ayrıt varsa 1, değilse 0.

Parametreler

c_{ij} : i. düğüm ile j. düğüm arası uzaklık

n : serimdeki düğüm sayısı

s : Alt tur engelleme kısıt sayısı

Tanımlanan bu karar değişkenleri ve parametreler kullanılarak oluşturulan matematiksel model şöyledir;

$$\sum_{(i,j)\in A} x_{ij} = n-1 \qquad (1)$$

$$\sum_{(i,j)\in A(s)} x_{ij} \le |s|-1 \qquad (2)$$

$$x_{ij} \in \{0,1\} \qquad (3)$$

k.a.

$$Enk. \sum_{(i,j)\in A} c_{ij}.x_{ij} \qquad (4)$$

Bu modeldeki (1) numaralı kısıt, şebekede düğüm sayısından bir eksik sayıda bağlantı yapılmasını, (2) numaralı kısıt alt turların engellenmesini sağlamaktadır. (3) numaralı kısıt x_{ij} değerlerinin 0 yada 1 değeri almasını gerektirmektedir. Amaç fonksiyonu ise enküçük toplam uzaklığı ifade etmektedir.

2.4 Enküçük Yayılma Probleminin Çözümünde Kullanılan Bazı Algoritmalar

EYP'nin çözümünde kullanılan kurucu sezgisellerden Prim ve Kruskal algoritmaları en çok bilinen algoritmalardır [13]. Bu algoritmalar, her iterasyonda kalan seçenekler içinden en iyi seçimi yapmaktadır. Dolayısıyla her adımda yerel en iyinin seçilmesiyle çalışmaktadırlar. Ancak buna rağmen, bu algoritmalar EYP için en iyi sonucu polinom zamanda vermektedirler [13].

Prim algoritması her iterasyonda bir ağaç oluşturarak çalışmaktadır. Ancak Kruskal algoritmasının bazı iterasyonları ağaç olmayan, orman olarak tanımlanan yapıyı oluşturabilir. Ayrıca Prim algoritması serimdeki herhangi bir düğümün seçilmesiyle başlatılabilir [13]. Bu nedenle bu çalışmada, GYP'de her kümeden bir

düğümün seçilmesiyle ortaya çıkan EYP'nin çözümünde Prim algoritmasının kullanılması tercih edilmiştir.

2.4.1 Prim Algoritması

Prim algoritması Robert Prim tarafından 1957 yılında EYP için geliştirilmiştir. Bu algoritma, serimdeki düğümleri seçilmiş ve seçilmemiş düğümler kümesi olmak üzere iki kümeye ayırarak çalışmaktadır. Algoritmanın ilk adımında seçilmiş düğümler kümesinde (*P*) rassal olarak seçilen tek bir düğüm bulunmakta, kalan tüm düğümler ise seçilmemiş düğümler kümesinde yer almaktadır. Algoritmanın ilerleyen her adımında seçilmiş düğümler kümesine bir yeni düğüm eklenmekte, eklenen düğüm seçilmemiş düğümler kümesinden çıkarılmaktadır [13].

$G = (N, A)$ n düğümlü yönsüz ve bağlı bir serim, T ise enküçük ağırlıklı (maliyetli) ayrıt olsun. Buna göre Prim algoritmasının adımları Şekil 2.4'te gösterilmiştir.

Adım 1: Sayacı $i = 1$ olarak başlat. Serimin herhangi bir düğümünü ($n_1 \in N$) seç, bu düğümü *P* seçilmiş düğümler kümesine al, seçilmemiş düğümler kümesini $S = G - \{n_1\}$ olarak değiştir.

Adım 2: *T* ağacına *G* seriminde bulunan ve *P* kümesindeki herhangi *x* düğümünü *S* kümesinde bulunan herhangi bir $y(= v_i + 1)$ düğümü ile birleştiren en küçük ağırlık (uzaklık) değerine sahip ayrıtı ekle, y düğümünü *S* kümesinden silerek *P* kümesine al: $1 \le i \le n-1$, $|N| = n$, $P = \{n_1, n_2, .., n_i\}$, $T = \{a_1, a_2, .., a_{i-1}\}$ ve $S = N - P$.

Adım 3: Sayacı $i = i + 1$ olarak güncelle. Eğer $i = n$ ise *G* seriminin alt serimi $e_1, e_2, .., e_{n-1}$ olmak üzere toplam $n-1$ adet ayrıt ve *n* adet düğüm kullanılarak bağlanmış ve *G* seriminin optimal ağacını elde et. Eğer $i < n$ ise Adım 2'ye dön

Şekil 2.4 Prim Algoritması

2.4.2 Kruskal Algoritması

EYP'de kullanılan bir diğer kurucu algoritma olan Kruskal algoritması 1956 yılında Joseph Kruskal tarafından geliştirilmiştir. $G = (N, A)$ serimi yönsüz ve bağlı bir serim olmak üzere Kruskal algoritmasının adımları Şekil 2.5'te sunulmaktadır [13,14].

Adım 1: Sayacı $i = 1$ olarak başlat. *G* seriminde bulunan ayrıtlardan mümkün olduğunca küçük ağırlıklı (maliyetli) bir ayrıt seç ($a_1 \in A$).

Adım 2: $1 \le i \le n-2$ olmak üzere eğer $a_1, a_2, .., a_i$ seçilmiş ise *G* seriminin kalan ayrıtları arasından

a) ağırlık değeri mümkün olduğunca küçük bir ayrıt olmak ve

b) $a_1, a_2, .., a_{i+1}$ ayrıtlarıyla (ve ilgili düğümlerle) tanımlı *G* seriminin alt serimi döngü oluşturmamak üzere a_{i+1} ayrıtını seç

Adım 3: Sayacı $i = i + 1$ olarak güncelle. Eğer $i = n - 1$ ise *G* şebekesinin alt şebekesi $a_1, a_2, .., a_{n-1}$ olmak üzere toplam *(n-1)* adet ayrıt ve *n* adet düğüm kullanılarak bağlanmış ve *G* şebekesinin optimal ağacını elde et. Eğer *i<n-1* ise Adım 2'ye dön.

Şekil 2.5 Kruskal Algoritması

2.5 Genelleştirilmiş Yayılma Problemi İçin Kaynak Taraması

GYP için literatürde geliştirilmiş matematiksel modeller [17] ve sezgisel yöntemler bulunmaktadır. Problemin NP-zor olması nedeniyle araştırmacılar, matematiksel modellerin yanı sıra, sezgisel yöntemler üzerinde de yoğunlaşmışlardır. GYP'nin polinom zamanda çözülebileceği üç özel durum bulunmaktadır [7]. Bunlar;

1) Eğer $|N_k| = 1, \ \forall k = 1,2,..,K$ ise problem EYP olarak Prim, Kruskal vb. algoritmalarla çözülebilir.

2) Eğer $K = 1$ ise çözüm tek düğümden oluşur.

3) Eğer $K = 2$ ise çözüm bu iki kümeyi enküçük uzaklıkla ilişkilendiren düğümlerden oluşmaktadır.

2.5.1 Matematiksel Modeller

GYP, ilk olarak 1995 yılında tanımlanmış bir problem olduğundan, literatürde GYP ile ilgili çalışma yapan çok sayıda araştırmacı bulunmamaktadır. GYP üzerindeki ilk çalışmayı yapan Myung, Lee ve Tcha [21], çok ürünlü akış problemi tabanlı bir dual-artan prosedür geliştirerek dal-sınır algoritması ile 100 düğümlü ve 4500 ayrıtlı problemleri çözebilmişlerdir.

Feremans, Labbe ve Laporte [8] GYP için tamsayılı ve karışık tamsayılı sekiz model geliştirmişlerdir. Lineer gevşetmelere göre, sekiz modelden dördünün, diğer dört modele baskın olduğunu belirtmişlerdir. Feremans [9] başka bir çalışmasında bazı yeni eşitsizlikler geliştirmiş ve dallandır ve kes algoritması kullanarak 160 düğüme kadar olan GYP'lerde uygulamıştır.

Raghavan [23] GYP'nin, bazı düğümleri derece kısıtlı Steiner Ağacı problemi olarak modellenebileceğini göstermiştir. Ayrıca geliştirilen modelin, lineer gevşetmelere göre Feremans'ın geliştirdiği dört modele denk olduğunu belirtmiştir.

2000 yılında Pop, Kern ve Still [22] GYP için, polinom boyutlu bir karışık tamsayılı model geliştirmiştir. Geliştirilen model ile 100 düğüme kadar olan test problemleri çözülmüştür. Bu çalışmada küme genişliğinin sabit bir değer aldığı GYP'lerde sabit yakınsama faktörü kullanılabildiğini göstermişler ve 2ρ-yakınsama algoritması adını verdikleri algoritmayı sunmuşlardır.

2004 yılında GYP için polinom sayıda kısıttan oluşan yeni bir matematiksel model geliştirilmiştir [17]. Geliştirilen matematiksel modelin, probleme eklenebilecek özel durumlar için uygun olduğu belirtilmiştir. $V=\{1,2,..,n\}$ düğümler ve $A=\{(i,j): i\neq j\in V\}$ ayrıtlar kümeleri, V_0 başlangıç bölge, $K=\{1,2,..,k\}$ bölgeler kümesi, $V_i\cap V_j=\varnothing$, $i\neq j$, $i,j\in K$ ve $K_i\cup V_i=V$ olmak üzere tanımlanan karar değişkenleri ve parametreler aşağıdaki gibi tanımlanmıştır.

x_{ij} : *i*.düğüm *j*.düğüm ile bağlanırsa 1, değilse 0,

u_p : Kök bölgeden itibaren *p*. bölgeye gelene kadar yapılan bağlantı sayısı ($p\in K$)

c_{ij} : *i* ve *j* düğümlerinin bağlantı uzaklığı

Buna göre GYP için geliştirilen karar modeli şöyledir [17];

$$\sum_{(i,j)\in A} x_{ij} = k \qquad (1)$$

$$\sum_{i\in V/V_p}\sum_{j\in V_p} x_{ij} = 1, \qquad p=1,2,..,k \qquad (2)$$

$$\sum_{i\in V/V_p} x_{ij} - \sum_{h\in V_q} x_{jh} \geq 0,\ j\in V_p, p\neq q,\ \mathrm{p,q}=1,2,..,\mathrm{k} \qquad (3)$$

$$u_p - u_q + k\sum_{i\in V_p}\sum_{j\in V_q} x_{ij} + (k-2)\sum_{i\in V_q}\sum_{j\in V_p} x_{ij} \leq k-1, \qquad p\neq q,\ \mathrm{p,q}=1,2,..,\mathrm{k} \qquad (4)$$

$$u_p + (k-1)\sum_{i\in V_0}\sum_{j\in V_p} x_{ij} \leq k, \qquad p=1,2,..,k \qquad (5)$$

$$u_p + \sum_{i\in V_0}\sum_{j\in V_p} x_{ij} \geq 2, \qquad p=1,2,..,k \qquad (6)$$

$$x_{ij}\in\{0,1\}, \qquad (i,j)\in A \qquad (7)$$

k.a.

$$Enk.\sum_{(i,j)\in A} c_{ij}.x_{ij}$$

Bu modelin (1) numaralı kısıtı, kök bölge dahil olmak üzere toplam bölge sayısından bir eksik sayıda ayrıt arasında bağlantı olmasını, (2) numaralı kısıt ara bölgelerin her birine başka bir bölgeden bir giriş olmasını sağlamaktadır. (3) numaralı kısıt, yayılmanın sürekliliğini sağlamakta, (4) numaralı kısıt alt tur oluşmasını engellemektedir. (5) ve (6) numaralı kısıtlar ise, kökten ilk bağlantı yapılan bölgenin karşı gelen u değişkenine 1 değerinin atanmasını ve u_p değişkenlerinin alabileceği en fazla değerin k olmasını sağlamaktadır.

2.5.2 Sezgisel Yöntemler

2000 yılında Dror, Haouari ve Chauoachi [7], GYP üzerinde dört basit sezgisel yöntem ile Genetik Algoritmayı, rassal olarak oluşturulan problemler üzerinde test etmişler ve sonuçlarını karşılaştırmışlardır. GA'nın tüm test problemlerinde, diğer dört sezgiselden çözüm kalitesine göre daha iyi sonuçlar verdiğini göstermişlerdir. Ancak çözüm süreleri açısından bakıldığında, GA'nın büyük boyutlu problemlerde uzun süreler gerektirdiğini vurgulamışlardır.

Feremans [9] 2001 yılında GYP üzerinde, eniyi çözümü bilinen test problemlerinin çözüm değerleri ile GA sonuçlarını karşılaştırmış ve GA sonuçlarının eniyi çözümden ortalama %6.53 oranında saptığını göstermiştir.

2003 yılında Ghosh [10], geliştirdiği Tabu Arama ve Değişken Komşuluk Arama Algoritmaları ile Feremans'ın [9] geliştirdiği Üst Sınır Algoritmasını 226 düğüme kadar olan test problemleri üzerinde karşılaştırmıştır. Üst Sınır Algoritmasına göre geliştirdiği Tabu Arama Algoritmanın yaklaşık olarak %2, Değişken Komşuluk Arama Algoritmalarının ise yaklaşık olarak %1.5 oranında daha iyi sonuçlar verdiğini belirtmiştir.

Golden, Raghavan ve Stanojevic [12] 2005 yılında GYP üzerinde bazıları TSPLIB'den seçilen ve bazıları rassal olarak üretilen 226 düğüme kadar olan test problemleri üzerinde yerel arama ile GA performanslarını karşılaştırmışlardır. Sonuç

olarak her iki yöntemin de iyi sonuçlar verdiğini, algoritmaların birbirine baskın olmadıklarını belirtmişlerdir.

2005 yılında Hu, Leitner ve Raidl [15], geliştirdikleri Değişken Komşuluk Arama Algoritması ile Pop'un 2002 yılında geliştirdiği Tavlama Benzetimi Algoritması ve Ghosh'un [10] geliştirdiği Tabu Arama Algoritmalarıyla karşılaştırmış ve bazı test problemlerinde bunlardan daha iyi sonuçlar bulduğunu belirtmişlerdir.

İzleyen bölümde, bu tezde Genelleştirilmiş Yayılma Problemi için geliştirilen algoritmalar incelenmektedir.

3. GENELLEŞTİRİLMİŞ YAYILMA PROBLEMİ İÇİN GELİŞTİRİLEN GENETİK ALGORİTMA, MELEZ ALGORİTMA VE KUŞ SÜRÜSÜ ALGORİTMASI

Bu tezde, Genelleştirilmiş Yayılma Problemi için Genetik Algoritma, Genetik Algoritma ve Tavlama Benzetimi Algoritmasına dayalı bir melez yöntem ve Kuş Sürüsü Algoritması geliştirilmiştir. Bu bölümün devamında adı geçen algoritmalar açıklanmakta ve Genelleştirilmiş Yayılma Problemi için kullanılan yapılar yer almaktadır.

3.1 Genelleştirilmiş Yayılma Problemi İçin Geliştirilen Genetik Algoritma

GA, ilk olarak Holland ve arkadaşları tarafından 1960'lı ve 1970'li yıllarda ortaya atılmış, ilk sistematik ve teorik açıklamalar ise 1975 yılında Holland'ın "Adaptation in Natural and Artificial Systems (Doğal ve Yapay Sistemlerde Adaptasyon)" adlı kitabında yer almıştır [24]. GA, Charles Darwin'in "en iyi olan yaşar" prensibinin karmaşık problemlerin çözümünde kullanılması düşüncesi üzerine geliştirilmiştir.

GA, bilinen çözüm yöntemleriyle çözülemeyen veya çözüm süresi problemin büyüklüğü ile üstel olarak artan problemlerde eniyi çözüme yakın çözümler üretmektedir. GA biyolojik sistemlerin doğal evrim mekanizmasının benzetimini yapan stokastik bir arama yöntemidir [3]. Başlangıçta birleşi eniyileme problemlerinin çözümünde başarıyla kullanılan GA, günümüzde sürekli eniyileme problemlerine de başarılı bir şekilde uygulanabilmektedir [24].

GA, doğadaki evrim sürecini taklit eden bir yaklaşım olduğundan, problemin yapısının ve çözümlerinin tarifinde özel terimler kullanılır. Arama uzayını oluşturacak

çözümlerin kodları dizi (kromozom) olarak adlandırılmıştır. Dizilerde yer alan değişkenler gen, dizilerin oluşturduğu topluluk yığın olarak isimlendirilir. GA'nın her bir adımındaki mevcut yığın nesil (jenerasyon) olarak isimlendirilmektedir. Yığında yer alan diziler, problemin türüne göre belirlenen uygunluk fonksiyonu ile değerlendirilir.

Standart bir GA temel olarak aşağıdaki üç ana adımdan oluşmaktadır.

i. Yeniden üretim işlemi
ii. Genetik operatörlerle yeni dizilerin elde edilmesi
iii. Uygunluk değerinin hesaplanması

Yeniden üretim işlemi, bir nesilden bir sonraki nesile geçerken kopyalanacak dizilerin seçilmesini ifade eder.

Genetik operatörler, çaprazlama ve mutasyon adı verilen operatörlerdir. Çaprazlama operatörü, ebeveyn olarak isimlendirilmek üzere seçilen farklı diziler arasında bilgi değişimi sağlayarak çocuklar adını alan yeni çözümler elde eder. Mutasyon operatörü ise dizilerin mevcut bazı özelliklerini değiştirerek yerel aramayı sağlar ve böylece arama işlemi yerel eniyi çözüme takılmaktan kurtulur.

Uygunluk değeri genellikle dizilerin amaç fonksiyonu veya tanımlı bir fonksiyonunun değerine eşittir. Bu değerler göz önüne alınarak, dizinin bir nesilden diğer nesile taşınması işlemi gerçekleştirilir.

Standart bir GA kısaca Şekil 3.1'de gösterildiği gibi çalışır.

Adım 1 : Başlangıç yığınını oluştur.
Adım 2 : Yığındaki her diziyi belli bir kritere göre değerlendir.
Adım 3 : Mevcut dizilere genetik operatörlerini uygula.
Adım 4 : Yığında, elde edilen çocuklara yer vermek için ebeveynleri sil, veya yer değiştir.
Adım 5 : Dizileri belli bir kritere göre değerlendirerek yığına dahil et.
Adım 6 : Durdurma koşulu sağlanıyorsa dur ve en iyi diziyi göster, sağlanmıyorsa Adım 3'e git.

Şekil 3.1 Standart bir GA'nın adımları

GA'nın uygulanmasında başlangıç yığınının oluşturulması, yığın genişliği, seçim mekanizmaları, uygunluk değerinin bulunması, çaprazlama ve mutasyon operatörlerinin türleri gibi birçok parametre algoritmanın performansında önemli rol oynamaktadır. Bu nedenle geliştirilen algoritmalar bu parametrelerin ince ayarları yapıldıktan sonra kullanılmalıdır. Ayrıca ele alınan problemi iyi ifade edecek dizi yapısının kullanılması gerekir.

GA'yı diğer arama yöntemlerden ayıran dört temel özellik aşağıda verilmiştir [11].

1. GA parametrelerin kendini değil, parametrelerin kodlarını kullanır.
2. GA, bir çözümden diğer bir çözüme geçerek değil, çözümlerden oluşan bir yığınla aramaya başlar.
3. GA amaç fonksiyonu değerlerini kullanır, türevlerini veya diğer yardımcı bilgileri kullanmaz.
4. Belirli seçim kuralları değil, olasılıklı seçim kuralları kullanılır.

3.1.1 Dizi Yapısı

Ele alınan problemin mümkün çözümlerinin dizi olarak kullanabilmesi için uygun çözümlerin GA'da kullanılabilecek bir gösterim şekline sahip olması gerekir. Bu amaçla probleme uygun özel kodlama yapısı seçilir. Literatürde kullanılan çok çeşitli dizi gösterim yapıları vardır. Bunlardan ikili düzende gösterim yaygın olarak kullanılmaktadır. Buna göre dizide yer alan her değişken 0 yada 1 değeri alabilmektedir. (0-1-1-0-0-1) ikili düzen dizi yapısına örnektir. Ancak bu yapı büyük boyutlu problemlerde dizinin çok uzun olmasına neden olabilmektedir.

Dizi yapısı farklı tamsayılardan da oluşabilir. (5-6-2-3-1-4) dizisi buna örnektir. Gezgin satıcı problemi, GYP gibi birleşi eniyileme problemlerinde bu yapının kullanılması, dizi uzunluğunun küçük boyutlarda kalmasını sağlamaktadır.

3.1.2 Başlangıç Yığınının Oluşturulması

Başlangıç yığını, daha önce de belirtildiği gibi GA'ya özgü bir yaklaşım olup, her nesilde aramanın yapıldığı diziler topluluğudur. Başlangıç yığının genişliği algoritma performansı üzerinde son derece etkilidir. Bölüm 4'te, bu çalışmada geliştirilen algoritmaların parametrelerinin belirlenmesi için uygulanan deney tasarımı kısmında yığın genişliğinin algoritma performansı üzerindeki etkileri daha açık görülebilir.

Başlangıç yığını rassal olarak oluşturulabildiği gibi, diğer sezgisel yöntemlerin kullanılmasıyla da oluşturulabilir. Örneğin Bouhmala [6], gezgin satıcı problemi için geliştirdiği GA'da yerel arama algoritmasını başlangıç yığınını oluşturmada kullanmıştır.

Dizi yapısı ile rassal çözümler üretilirken, ele alınan problem için uygun olmayan çözümler ortaya çıkabilir. Bu duruma engel olmak için düzeltici algoritmalar kullanılabileceği gibi bazı durumlarda kullanılan dizi yapısında değişikliğe gidilerek

uygun olmayan çözümlere izin verilmez. Bu amaçla algoritmaya bilgiye dayalı adımlar eklenebilir.

3.1.3 Değerlendirme Fonksiyonu

Şekil 3.1, Adım-5'te görüleceği gibi, dizilerin bir sonraki yığına taşınması uygunluk fonksiyonu değerlerine bağlıdır. Uygunluk fonksiyonu değeri yüksek olan dizilerin yığından yığına taşınma şansı (olasılığı) yüksektir. Bu nedenle enbüyükleme problemlerinde uygunluk fonksiyonu olarak dizinin amaç fonksiyonu değeri kullanılabilir. Ancak enküçükleme problemlerinde uygunluk fonksiyonu değeri yerine amaç fonksiyonu değerinin kullanılması, uygunluk fonksiyonu düşük dizilerin sonraki nesile taşınmasına neden olmaktadır.

Bu durumu ortadan kaldırmak için enküçükleme problemlerinde kullanılabilecek çeşitli yaklaşımlar vardır [11]:

i. Amaç fonksiyonunun tersi (1/amaç fonksiyonu değeri) uygunluk fonksiyonu değeri olarak kullanılabilir.
ii. Her dizinin amaç fonksiyonu değeri, problemin çözüm değerleri dikkate alınarak belirlenen büyük bir sabit sayıdan çıkarılarak elde edilen sayı uygunluk fonksiyonu değeri olarak kullanılabilir.

3.1.4 Yeniden Üretim Operatörü

GA'da her nesilde bir sonraki nesile taşınacak diziler, Bölüm 3.1.3'te belirtilen uygunluk fonksiyonu değerine göre olasılıklı olarak seçilmektedir. Buna göre, uygunluk fonksiyonu değeri yüksek olan bir dizinin bir sonraki nesile geçirilme olasılığı da yüksektir. Yani Darwin'in "en iyi olan yaşar" prensibine göre en iyi uygunluk fonksiyonu değerine sahip dizi hayatta kalmaya devam etmektedir.

Daha önce belirtildiği gibi GA, olasılıklı seçim kuralları kullanmaktadır. Literatürde seçim mekanizmaları için farklı sınıflandırmalar yapılmıştır. Back [4], seçim mekanizmalarını; sıralı seçim mekanizmaları turnuva seçim mekanizması, orantılı seçim mekanizmaları, $(\mu + \lambda)$ ve (μ, λ) seçim mekanizmaları olarak sınıflandırmıştır.

Goldberg [11] ise, seçim mekanizmalarını orantılı yeniden üretim mekanizması, sıralı üretim mekanizması, turnuva üretim mekanizması ve denge durumu üretim mekanizması olarak sınıflandırmıştır. Adı geçen seçim mekanizmaları bu bölümün devamında açıklanmaktadır.

3.1.4.1 Orantılı yeniden üretim mekanizmaları

Orantılı yeniden üretim mekanizmaları, rulet çemberi, stokastik artan ve stokastik üniversal seçim mekanizmalarını içermektedir.

3.1.4.1.1 Rulet çemberi seçim mekanizması

Orantılı yeniden üretim mekanizmalarından literatürde en çok bilineni rulet çemberi yöntemidir. Bu yönteme göre bir çember yığında bulunan dizi sayısı kadar aralığa bölünür ve her bir aralığın genişliği, o aralığın ifade ettiği dizinin uygunluk fonksiyonu değeri ile orantılıdır.

Uygunluk fonksiyonu değeri yüksek dizilerin, bu çember üzerinde sahip oldukları aralığın genişliği daha fazla, uygunluk fonksiyonu değeri küçük olan dizilerin sahip oldukları aralık genişliği daha küçük olmaktadır.

Her iterasyonda 0 ile 1 arasında üretilen rassal sayı çember üzerinde hangi aralığa düşüyor ise, bu aralığa ait dizi bir sonraki nesile aktarılmak üzere seçilmektedir. Bu işlem yığın genişliğine ulaşılıncaya kadar devam eder.

3.1.4.1.2 Stokastik artan seçim mekanizması

Stokastik artan seçim mekanizması rulet çemberi seçim mekanizmasına benzerlik göstermektedir. Bu seçim mekanizmasına göre yeni nesile dizileri geçirirken her dizinin beklenen kopya sayısı hesaplanmakta ve ilgili dizinin bu değerin tamsayı kısmı kadar kopyası yeni nesile geçirilmektedir. Tüm diziler için aynı işlem tekrarlandıktan sonra yığın genişliğine ulaşılmadıysa, beklenen kopya sayısı değerinin ondalıklı bölümleri dikkate alınmaya başlar.

3.1.4.1.3 Stokastik üniversal seçim mekanizması

Rulet seçim mekanizmasına ek olarak çemberin dışında kalan bölge yığın genişliği kadar eşit aralıklara bölünmektedir. Bu mekanizmada çemberin dışında kalan aralık sayısı ilgili dizinin kopya sayısına eşit olmaktadır.

3.1.4.2 Sıralı seçim mekanizması

Sıralı seçim mekanizması, yığında bulunan dizilerin uygunluk fonksiyonu değerine göre en iyiden en kötüye doğru sıralanarak, kopya sayılarını da bu sırada azalan bir fonksiyon ile belirmektedir. Bu fonksiyon doğrusal veya doğrusal olmayan bir fonksiyon olabilir.

3.1.4.3 Turnuva seçim mekanizması

Turnuva seçim mekanizması, mevcut yığın içerisinden rassal olarak seçilen bir grup dizi içerisinden uygunluk fonksiyonu değeri yüksek olan dizinin yeni nesile kopyalanması işlemidir. Bu işlem nesil genişliğine ulaşılıncaya kadar devam etmektedir. Grup genişliği iki yada daha fazla sayıda diziden oluşabilmektedir. Literatürde grup genişliği genellikle iki olarak kullanılmaktadır [4].

3.1.4.4 Denge durumu seçim mekanizması

Denge durumu seçim mekanizması ise doğrusal olarak azalan sıralı seçim mekanizması kullanılarak, seçilen bir yada iki dizinin genetik operatörlerle değişime uğratılmasıyla oluşturulan yeni dizilerin, yığında bulunan ve uygunluk fonksiyonu değeri en düşük diziler ile değiştirilmesi işlemidir.

3.1.4.5 $(\mu + \lambda)$ Seçim mekanizması

$(\mu + \lambda)$ seçim mekanizmasında, μ yığın genişliğine eşit olmak üzere ebeveyn dizilerin sayısını, λ ebeveyn dizilerin çaprazlanması ile oluşan çocukların sayısını ifade etmektedir. Bu seçim mekanizmasında çaprazlama sonucunda ebeveynler silinmemekte, seçim ebeveynler ve çocuklar arasından yapılmaktadır. Seçimde, ebeveynler ve çocuklardan oluşan yığındaki diziler uygunluk fonksiyonu değerine göre iyiden kötüye doğru sıralanır ve ilk μ adet dizi bir sonraki nesile taşınmak üzere seçilir [4].

3.1.4.6 (μ, λ) Seçim mekanizması

Bu seçim mekanizmasında, $(\mu + \lambda)$ seçim mekanizmasından farklı olarak ebeveyn diziler dikkate alınmamakta, ebeveynlerin çaprazlanmasıyla oluşan çocuklar arasından seçim yapılmaktadır [4].

3.1.5 Genetik Operatörler

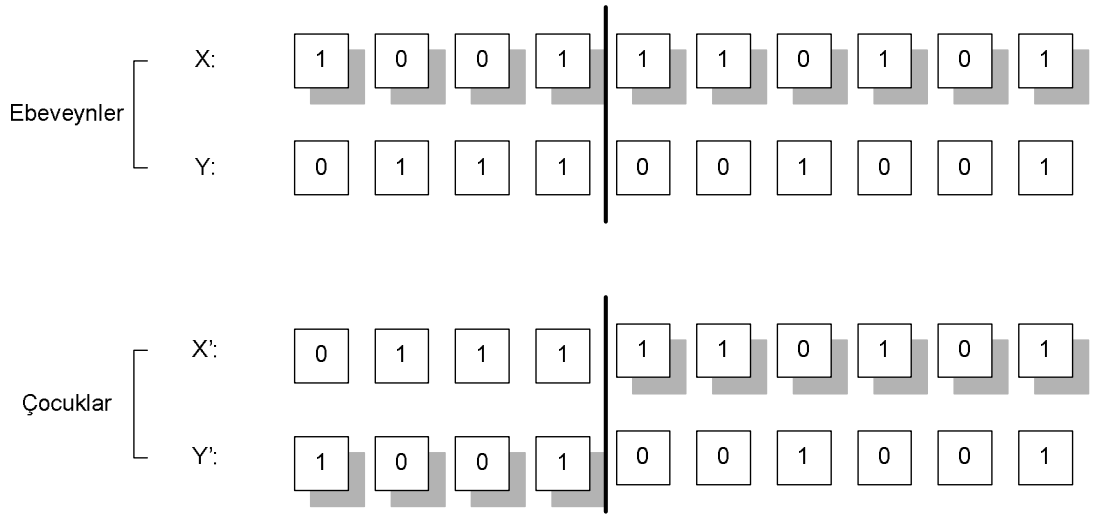
Genetik Algoritmada yeniden üretim işlemi gerçekleştirildikten sonra yığında bulunan diziler çaprazlama ve mutasyon operatörleri kullanılarak değişime uğratılırlar. Bu operatörler, belirli olasılık değerlerine göre yığında bulunan dizilere uygulanarak aramada mevcut yığından farklı noktaların elde edilmesini sağlar.

3.1.5.1 Çaprazlama operatörü

Çaprazlama operatörü, GA'nın farklı nesillerinde yer alan dizilerin aynı kalmaması için yığında yer alan diziler arasında bilgi değişimi yapar. Mevcut iyi dizilerin özellikleri kullanılarak arama uzayında farklı noktaların aranması için kullanılmaktadır. Çaprazlamanın gerçekleşip gerçekleşmeyeceği, rassal olarak 0 ile 1 arasında üretilen bir sayının p_c notasyonu ile gösterilen çaprazlama olasılığı değerinde küçük olup olmamasına göre belirlenir. Eğer üretilen rassal sayı p_c olasılık değerinden küçük ise çaprazlama gerçekleşir, aksi halde seçilen iki dizi arasında bilgi değişimi olmaz. Çaprazlama operatörü, rassal olarak seçilen iki dizi arasında bilgi değişiminin, bu iki dizinin hangi noktalarından itibaren yapılacağına göre farklılıklar göstermektedir. Genel olarak kullanılan çaprazlama operatörleri şunlardır;

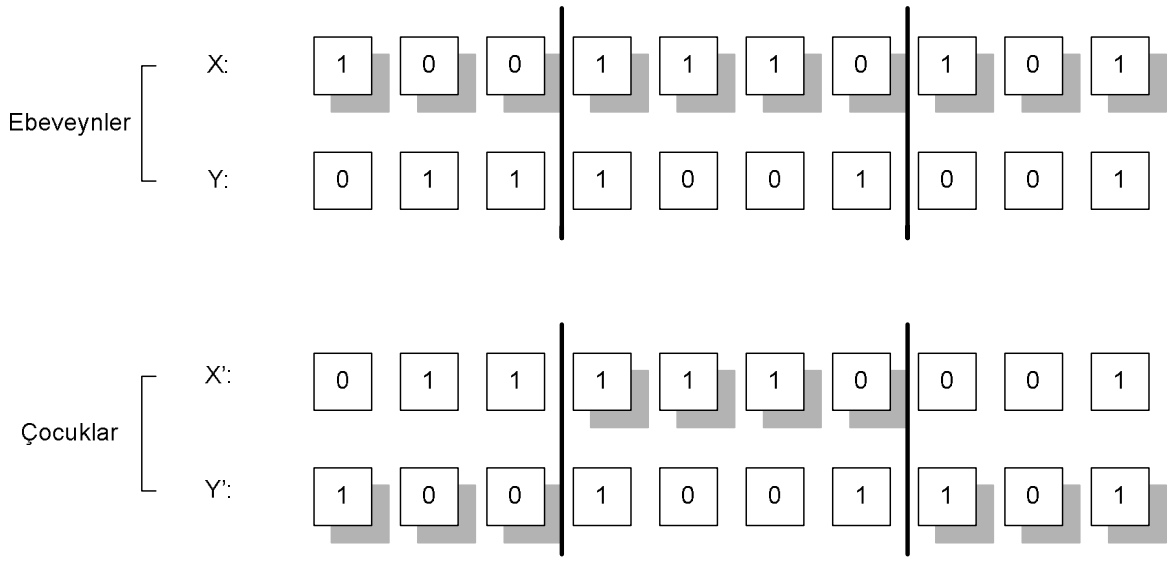
I. Tek noktadan çaprazlama
II. İki noktadan çaprazlama
III. Uniform çaprazlama

Tek noktadan çaprazlamada, adından da anlaşılacağı üzere iki dizi arasında bilgi değişimi, rassal olarak seçilen bir noktadan itibaren gerçekleşmektedir. Örneğin, X ve Y dizileri GA'nın herhangi bir neslinde yığın içerisinde yer alan ve on adet genden oluşan farklı iki dizi olsun. Çaprazlama noktası dört olarak seçilirse yeni çözümler X' ve Y' elde edilir. X, Y, X' ve Y' dizileri Şekil 3.2'de görülmektedir.



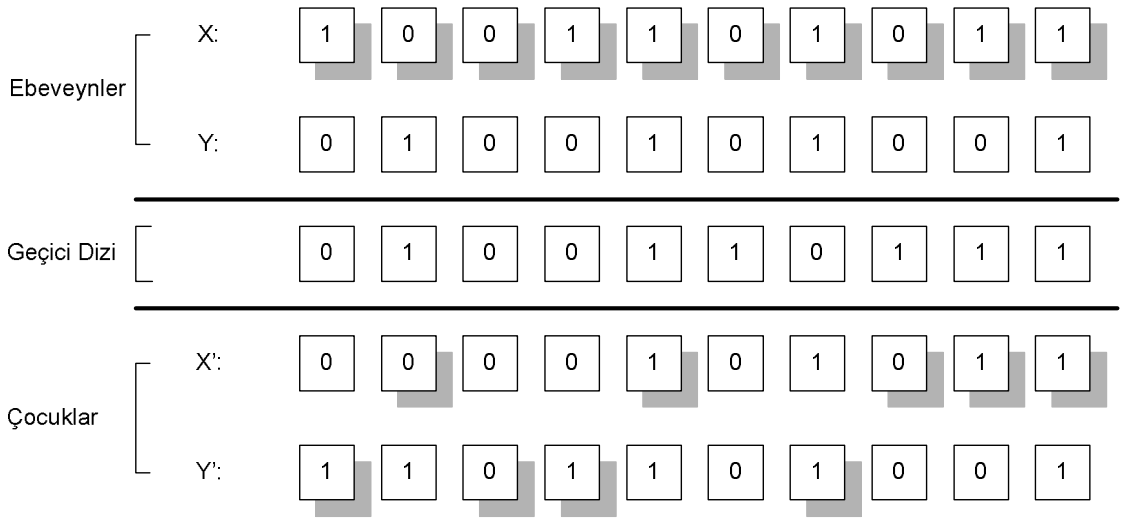
Şekil 3.2 Tek noktadan çaprazlama örneği

İki noktadan çaprazlama operatöründe, iki dizi arasındaki gen değişimi, dizilerin rassal olarak seçilen ve aynı olmayan iki noktası arasında kalan bölümleri arasında gerçekleşir. 1. çaprazlama noktası 3, ikinci çaprazlama noktası 7 olarak seçilirse ve yukarıdaki örnekte gösterilen X ve Y dizileri üzerinde iki noktadan çaprazlama denenirse Şekil 3.3'te görülen X' ve Y' dizileri elde edilir.



Şekil 3.3 İki noktadan çaprazlamaya örnek

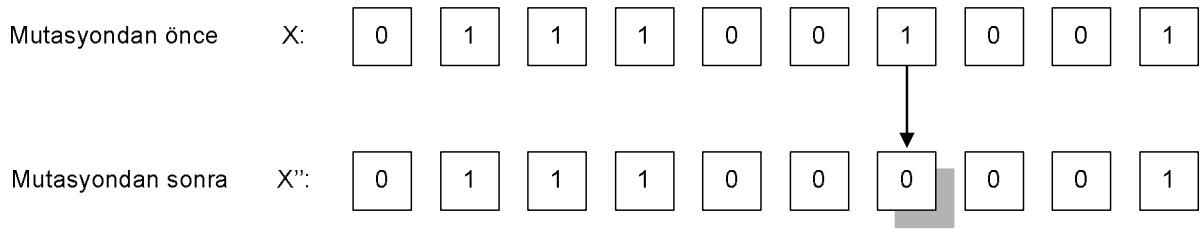
Uniform Çaprazlama Operatöründe ise belirlenen bir olasılık değerine göre oluşturulan ve asıl diziler ile aynı sayıda genden oluşan geçici bir dizi oluşturulur. Oluşturulan geçici dizide genler 0 yada 1 değeri alabilmektedir. 1 değerleri, oluşturulan 1. dizinin ilgili yerine 1. dizinin geninin geleceğini, 0 değeri 2. dizinin ilgili geninin geleceğini göstermektedir. Tersi durumda ise, yani oluşturulan 2. dizide ise geçici dizide yer alan 0 değerleri ilgili noktaya 1.dizinin ilgili geninin geleceğini, 1 değeri ise ilgili noktaya 2.dizinin ilgili geninin geleceğini göstermektedir. Geçici dizi 0 1 0 0 1 1 0 1 1 1 olmak üzere, aynı örnek üzerinde uniform çaprazlama uygulanırsa Şekil 3.4'te gösterilen diziler elde edilir.



Şekil 3.4 Uniform çaprazlamaya örnek

3.1.5.2 Mutasyon operatörü

Mutasyon operatörü, dizi içinden rassal olarak seçilen bir veya birkaç geni değişime uğratarak yığının yerel eniyiye takılmasını önler. Böylece yeni dizilerin aranması sağlanır. Mutasyonun gerçekleşip gerçekleşmeyeceği ise yine 0 ile 1 arasında üretilen rassal bir sayının p_m notasyonu ile gösterilen mutasyon olasılığından küçük olup olmamasına göre belirlenir. Üretilen rassal sayı p_m olasılık değerinden küçük ise, ilgili dizide mutasyon gerçekleşir, değilse hiçbir değişiklik olmayacaktır. Seçilen bir X dizisi, rassal olarak belirlenen yedinci geninde mutasyona uğratılırsa aşağıda Şekil 3.5'te gösterilen X'' dizisi elde edilir.



Şekil 3.5 Mutasyona örnek

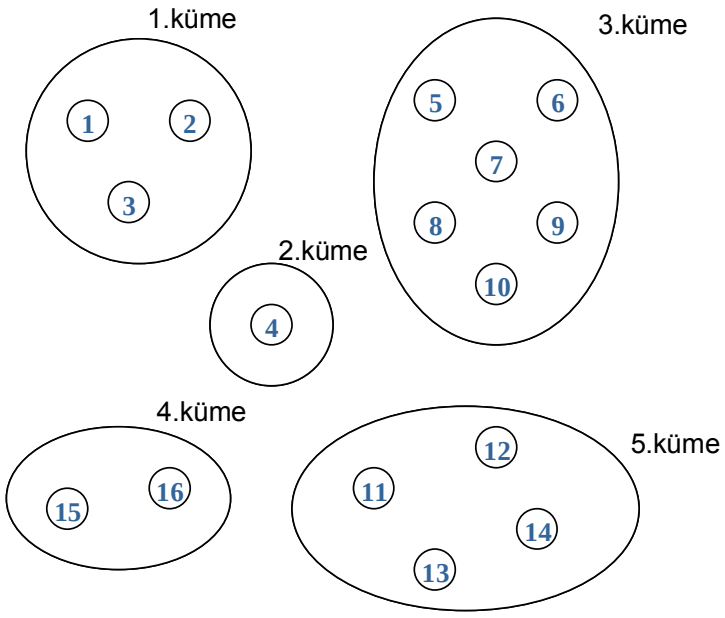
3.2 Geliştirilen Genetik Algoritma

GYP için probleme özgü özellikler içeren GA tabanlı bir algoritma geliştirilmiştir. Bu bölümde geliştirilen bu algoritmada kullanılan yapılar ve özellikler açıklanmaktadır.

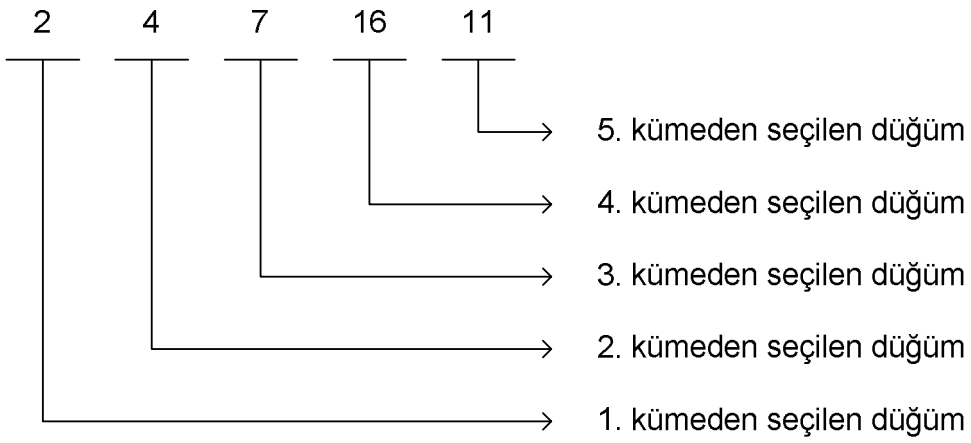
3.2.1 Dizi Gösterimi

Bölüm 2'de belirtildiği gibi, GYP için aday çözümler, her kümeden yalnız bir düğümün seçilerek, bu düğümler arasında EYP'nin çözülmesiyle elde edilebilmektedir. Bu durum için ikili düzende dizi tanımlama veya düğüm numaralarından oluşan dizi tanımlama yöntemleri seçilebilir. Ancak, ikili düzende dizi yapısı kullanımı, büyük boyutlu problemlerde dizi uzunluğunun uzamasına yol açmakta ve her dizi için sürekli olarak uygunluk kontrolünün yapılmasını ve gerekirse düzeltici algoritmalar kullanılmasını gerektirmektedir.

Geliştirilen GA'da dizilerin kodlanmasında gerçek değerli gösterim kullanılmıştır. Bu dizi gösterim şekli literatürde Golden ve diğerleri [12] tarafından da GYP için kullanılmıştır. Şekil 3.6'da gösterilen beş kümeli, toplam on-altı düğümlü GYP için dizi yapısı Şekil 3.7'de verilmiştir.



Şekil 3.6 GYP örneği



Şekil 3.7 GA'da kullanılan dizi örneği

Şekil Y'de gösterilen gerçek değerli dizi yapısının kullanılmasıyla;

i. Dizilerdeki değişken sayısı, GYP'deki küme sayısı ile sabit tutulmuş,
ii. Bilgisayar hafızasında daha az yer tutulması sağlanmış,
iii. Çözümler daha kolay anlaşılabilir hale gelmiştir.

3.2.2 Başlangıç Yığınının Oluşturulması

GA, bilindiği gibi, tek bir çözümle değil, mümkün çözümlerden oluşan bir yığın ile aramaya başlar. Bu yığın, kullanılacak bir başka sezgisel yöntemle oluşturulabileceği gibi, rassal olarak da oluşturulabilmektedir. Bu çalışmada, klasik bir GA'da olduğu gibi, başlangıç yığını rassal olarak üretilmiştir. Buna göre, kullanılan dizi yapısında başlangıç çözümler şöyle üretilmektedir;

1) Diziler ilgili problemin küme sayısı kadar değişkenden oluşmakta ve değişkenlerin dizide yer aldığı sıra numarasına ait kümeden, o kümeye ait herhangi bir düğümün numarasını, dizinin ilgili değişkeni olarak almaktadır.

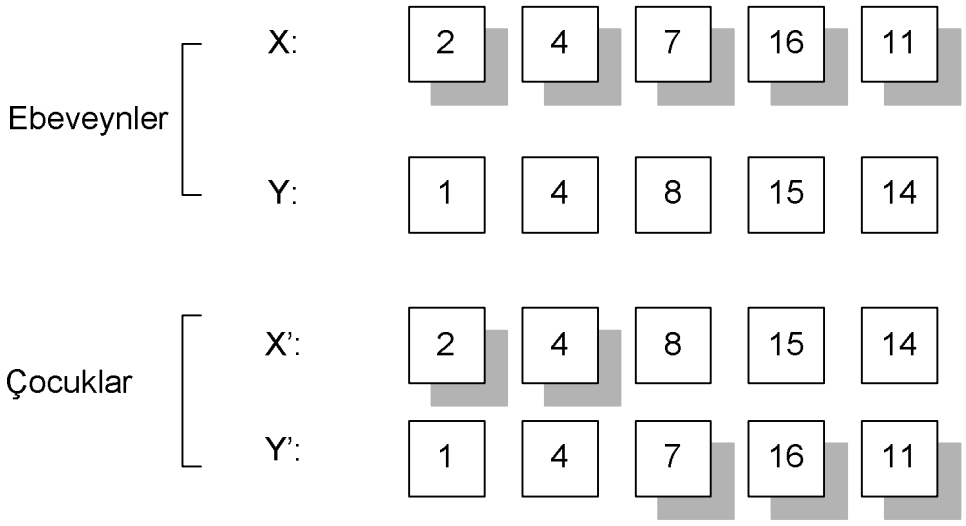
2) Bu şekilde, Cevap Yüzeyi metodu ile belirlenen 40 adet dizi rassal olarak oluşturulmaktadır.

3.2.3 Kullanılan Amaç ve Uygunluk Fonksiyonu

Bölüm 3.1.3'te belirtildiği gibi, kodlanan dizilerin uygunluk fonksiyonu değerlerinin hesaplanması gerekir. Bu çalışmada uygunluk fonksiyonu değerinin hesaplanmasında (1/amaç fonksiyonu) değerinin uygunluk fonksiyonu değeri olarak kullanması tercih edilmiştir. Bilindiği üzere GYP'de amaç fonksiyonu toplam uzaklığın ($\sum_{(i,j)\in A} c_{ij} x_{ij}$) enküçüklenmesi şeklindedir.

3.2.4 Çaprazlama Operatörü

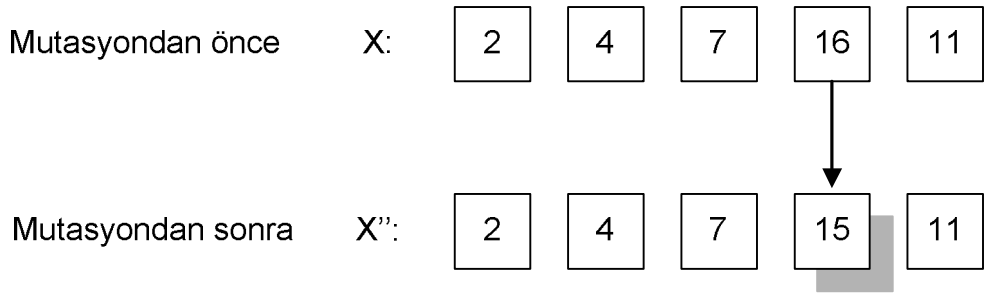
GYP için geliştirilen GA'da çaprazlanacak olan diziler turnuva seçim mekanizması kullanılarak seçilmektedir. Buna göre yığın içerisinden iki çift dizi seçilmekte, diziler uygunluk fonksiyonu değerlerine göre karşılaştırılarak, her çiftten uygunluk fonksiyonu değeri iyi olan diziler rassal olarak seçilen tek noktadan birbiriyle çaprazlanmaktadır. Örneğin Şekil 3.6'da görülen GYP için turnuva seçim mekanizmasıyla seçilen ebeveynler X ve Y dizileri, çaprazlama noktası 2 olarak seçilirse, çaprazlama Şekil 3.8'de gösterildiği gibi gerçekleşir.



Şekil 3.8 Kullanılan çaprazlama operatörü

3.2.5 Mutasyon Operatörü

Bu tezde kullanılan mutasyon operatörü Bölüm 3.1.5.2'de anlatılan şekliyle kullanılmıştır. Buna göre mutasyona uğratılacak dizi p_m olasılığına göre rassal olarak seçilen tek noktasında mutasyona uğratılır. Örneğin, Şekil 3.6'da görülen GYP için rassal olarak seçilen mutasyon noktası 4 olmak üzere X dizisinin mutasyona uğratılması Şekil 3.9'da görülmektedir.



Şekil 3.9 Kullanılan mutasyon operatörü

Yapılan denemelerde, GA'nın son nesillerinde yerel eniyiye takıldığı gözlenmiştir. Bunu azaltmak için bu tezde dinamik mutasyon olasılığı kullanılmıştır. Buna göre, Bölüm 4'te anlatıldığı gibi p_m değerinin başlangıç değeri 0.01 olarak seçilmiş, yapılan denemelere göre her nesilde mutasyon olasılığı $p_m = p_m + 0.001$ olarak güncellenmiş, bu sayede son nesillerde mutasyon olasılığının 0.2'ye kadar çıkarılarak yerel arama artırılmıştır.

3.2.6 Geliştirilen Genetik Algoritma

Bir sonraki nesile geçilirken kopyalanacak diziler ise $(\mu + \lambda)$ seçim mekanizması kullanılarak seçilmiştir. Buna göre, yığında yer alan μ (=n) adet dizi ile çaprazlama sonucu oluşan yeni diziler (λ), uygunluk fonksiyonu değerine göre iyiden kötüye doğru sıralanarak, tekrarlanan diziler silinir ve kalan diziler içinden ilk μ adet dizi yeni yığına kopyalanır. Eğer tekrarlanan dizilerin silinmesiyle μ'den daha az sayıda dizi kalırsa, rassal olarak oluşturulan diziler ile yığındaki dizi sayısı μ'ye

tamamlanır. $(\mu+\lambda)$ seçim mekanizmasında ebeveyn ve çocuklar içinden eniyi çözümler bir sonraki nesile taşındığından, elitist mekanizma kullanılmasına gerek kalmakaktadır. Rassal olarak oluşturulan bu diziler ile algoritmaya farklı noktaları da arama özelliği kazandırılmış olmaktadır. GYP'nin çözümü için geliştirilmiş olan GA'nın adımları Şekil 3.10'da verilmektedir.

Adım-1: Rassal olarak başlangıç yığınını üret.

Adım-2: Üretilen dizilerin uygunluk fonksiyonu değerlerini hesapla.

Adım-3: *n*/2 kez turnuva seçim mekanizmasını kullanarak ebeveynleri belirle ve $p_{Ç}$ olasılığı ile çaprazla.

Adım-4: Mevcut yığına çaprazlama ile elde edilen yeni dizileri ekle.

Adım-5: Yığındaki dizileri p_m olasılığı ile mutasyona uğrat.

Adım-6: Yığında aynı olan diziler varsa, tekrarlanan dizileri sil.

Adım-7: Prim Algoritması ile yığındaki tüm dizilerin amaç fonksiyonu değerini hesapla, uygunluk fonksiyonu değerlerini bul.

Adım-8: Yığındaki dizileri uygunluk fonksiyonu değerlerine göre iyiden kötüye doğru sırala.

Adım-9: Yığındaki dizi sayısı yığın genişliğinden az ise, yığın genişliğine ulaşıncaya kadar rassal diziler üret. Eğer yığın genişliğinden daha fazla sayıda dizi varsa, iyiden kötüye doğru sıralanmış dizilerden ilk μ adet diziyi bir sonraki nesile taşı.

Adım-10: Nesil sayısını bir artır. p_m olasılığını güncelle. Belirlenen nesil sayısına ulaşılmış ise DUR, ulaşılmamış ise **Adım-3**'e dön.

Şekil 3.10 GYP için geliştirilen GA'nın adımları

Yapılan denemeler sonucunda algoritmanın 3.adımında çaprazlama için *n*/2 kez turnuva seçim mekanizması kullanılmıştır. Turnuva seçim mekanizması ile her seferinde iki dizi seçildiğinden, çaprazlama sonucunda *n* adet çocuk elde edilmektedir. Mevcut yığına çocukların eklenmesiyle toplam *2n* genişliğinde bir yığın elde edilmekte, $(\mu + \lambda)$ seçim kuralına göre bu dizilerin iyiden kötüye doğru sıralanması sonucu ilk μ (= *n*) adet dizi yeni nesile taşınmaktadır.

3.3 Genelleştirilmiş Yayılma Problemi Tavlama Benzetimi Algoritması

Tavlama Benzetiminin (TB) kesikli eniyileme problemlerinde kullanımı 1980'lerin başına kadar uzanmaktadır [24]. TB'nin temelini oluşturan mantık ilk olarak Metropolis tarafından 1953 yılında ortaya atılmıştır. Orijini teorik fizik ve istatistiksel Monte-Carlo metoduna dayanan TB, bir katının düzenli bir kristal yapıya ulaşması için yüksek sıcaklık derecelerine kadar ısıtılıp, düzenli olarak soğutulmasını taklit eden, benzetimini yapan sezgisel bir tekniktir.

İlk olarak Kirkpatrick, Gelatt ve Vecchi (1983) tarafından bir birleşi enküçükleme problemine uygulanmıştır. Metodun uygulamasında, mümkün çözümler ile sistemin durumları ve amaç fonksiyonu arasında bir analoji kurulur. Bu durumda en iyi çözüm sistemin minimum enerjiye sahip olduğu durumda elde edilir. Tavlamada adı geçen işlem ve durumların TB'deki karşılıkları Çizelge 3.1'de görülmektedir [24].

Çizelge 3.1: Tavlamada adı geçen işlem ve durumların TB'deki karşılıkları

Tavlamadaki işlem ve durumlar	**TB'deki karşılığı**
Sistemin durumu	Uygun çözümler
Enerji	Maliyet
Durum değişimi	Komşu çözüm arama
Sıcaklık	Kontrol parametresi
Donma durumu	Sezgisel çözüm

TB, komşu arama metoduna dayalı algoritmalardan biridir. Basit bir komşu arama yöntemi olan İniş Algoritması rassal olarak üretilen bir başlangıç çözüm ile aramaya başlar. Daha sonra uygun bir hareket mekanizması kullanarak bu çözümün komşusu üretilir ve her iki çözümün amaç fonksiyonu değeri karşılaştırılır. Komşu çözümün amaç fonksiyonu değeri daha iyi ise yeni mevcut çözüm olarak kabul edilir, aksi halde mevcut çözüm değiştirilmez. Mevcut çözümün hiçbir komşusu amaç fonksiyonu değerinde iyileşme sağlamayana kadar bu işlem tekrarlanır. İniş Algoritması, enküçükleme problemlerinde amaç fonksiyonu değerini artıracak hiçbir komşuluk hareketine izin vermemektedir. Bu durum, algoritmanın yerel eniyiye takılmasına ve daha iyi çözümlere ulaşamamasına neden olmaktadır. TB ise algoritmanın yerel eniyiye takılmasını önlemek üzere geliştirilmiş bir algoritmadır. Bu, rassal sayı üretimi ile sıcaklık adı verilen bir kontrol parametresi kullanılarak sağlanmaktadır.

TB, kabul fonksiyonuna göre giderek azalan bir olasılıkla mevcut çözümden daha kötü çözümlerin de kabul edilmesiyle aramaya devam etmektedir [31]. Böylece arama uzayında farklı bölgelere sıçramaya imkan sağlanarak yerel eniyi tuzağından kurtulma sağlanmaktadır. Kabul fonksiyonu, amaç fonksiyonunda δE kadar bir yükselmeye yol açan hareketin kabul edilme olasılığıdır. t, fiziksel tavlamadaki sıcaklığa karşılık gelen bir kontrol parametresi, k Boltzman sabiti olarak bilinen fiziksel bir sabit olmak üzere, kabul fonksiyonu $\exp(-\delta E / kt)$ eşitliği ile ifade edilir [2]. t değeri yüksek olduğunda hareketlerin çoğu kabul edilmekte, azaldıkça amaç fonksiyonunda artışa neden olan hareketlerin çoğu reddedilmektedir. Aramaya yüksek t değeriyle başlanması, algoritma performansını olumlu yönde etkilemektedir. TB'de, her iterasyonda Metropolis filtresi kullanılır. Metropolis filtresi şu şekilde çalışır [24];

I. Bir başlangıç çözümü üreterek katının durumunu rassal olarak değiştir ve enerji değişimini dikkate al.

II. Eğer yeni durum eskisinden daha iyiyse bunu artık eniyi çözüm olarak kabul et, daha kötü ise bu çözümü $p(\delta E) = \exp(-\delta E / kt)$ olasılığı ile kabul et.

Standart bir Tavlama Benzetimi algoritmasının genel akışı Şekil 3.11'de gösterilmiştir.

Adım 1: Bir başlangıç çözümü üret (*T*), başlangıç sıcaklık değerini (*t*) ve her sıcaklık derecesindeki tekrarlama sayısını (*r*) belirle.

Adım 2: Bir durdurma kriteri sağlanana kadar aşağıdaki adımları tekrarla:

I. Gelinen sıcaklık derecesinde aşağıdaki adımları *r* kez tekrarla:

i. Mevcut çözüme rassal bir değişiklik uygulayarak yeni çözümü (*T'*) elde et. İki çözüm arasındaki fark $\delta E = \delta E(T') - \delta E(T)$ değerini hesapla.

ii. O ile 1 arasında rassal bir x değeri üret.

iii. Eğer $\delta E < 0$ veya $x < \exp(-\delta E / t)$ ise $T=T'$ olarak değiştir.

II. *t* sıcaklık parametresini güncelle.

Adım 3: Mevcut çözüm *T*'yi en iyi çözüm olarak yazdır.

Şekil 3.11 Standart bir Tavlama Benzetimi algoritması

TB'de kullanılan parametrelerin değerlerinin belirlenmesiyle tavlama planı hazırlanır. Tavlama planı aşağıdaki parametreleri içermektedir [2]:

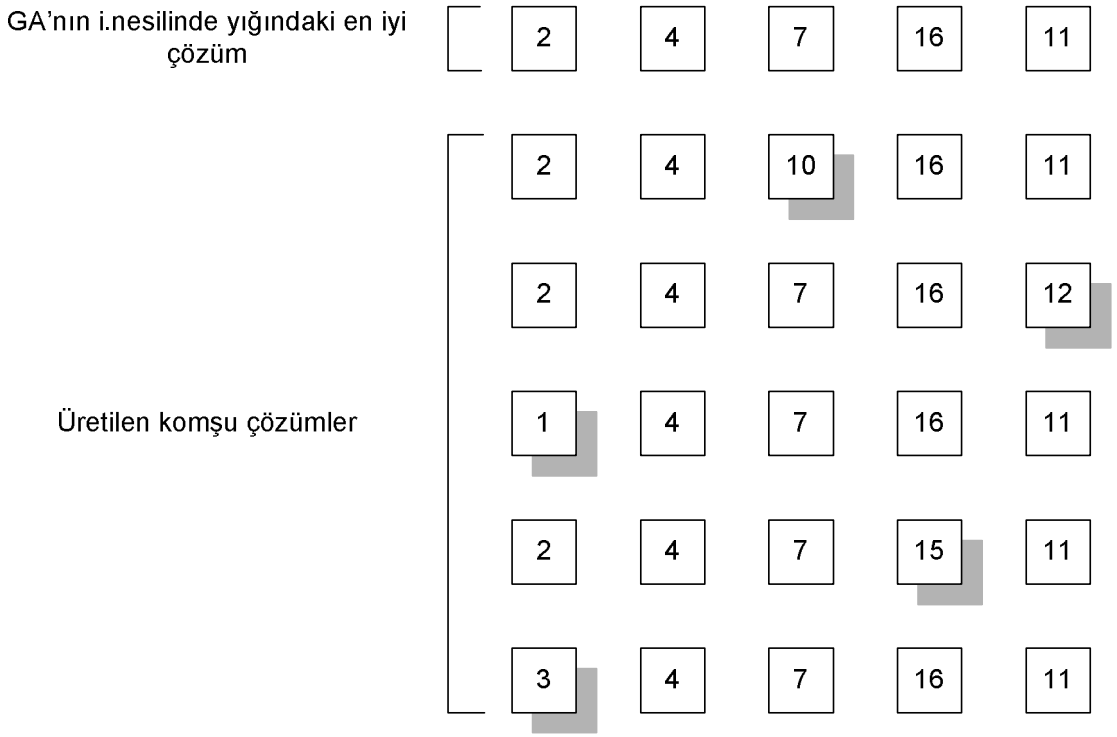
I. *t* sıcaklık parametresinin başlangıç değeri
II. Her sıcaklıkta gerçekleştirilecek *r* tekrar sayısı
III. Algoritmanın durdurulacağı *t* sıcaklık parametresinin son değeri veya sıcaklık azaltma sayısı.

TB'de sıcaklığın azaltılması, her iterasyonun sonucunda *t* sıcaklık parametresinin güncellenmesiyle gerçekleştirilir. *t(i)*, *i.* iterasyondaki sıcaklık parametresi değeri olmak üzere, *t(i+1) = s.t(i)* eşitliğine göre sıcaklık parametresi değeri güncellenir. Buradaki *s*, 1'den küçük bir değer alan, genellikle 0.80 ile 0.99 arasında değerler verilerek kullanılan soğutma katsayısıdır [2].

3.3.1 Geliştirilen Tavlama Benzetimi Algoritması

Daha önce de bahsedildiği üzere, GA ile TB'nin birlikte kullanılmasıyla GYP için bir melez yöntem geliştirilmesi amaçlanmıştır. Kullanılan TB algoritması, GA ile aynı dizi yapısını kullanmaktadır. Geliştirilen melez yöntemde, GA'nın her neslinde yığında yer alan en iyi çözüm TB'nin başlangıç çözümü olarak kabul edilmiş, belirlenen iterasyon sayısı boyunca TB algoritması kullanılmış, ele alınan çözümden daha iyi bir çözüm bulunması durumunda yeni nesile bu çözüm taşınmış, aksi takdirde mevcut çözüm bir değişikliğe uğratılmadan yeni nesile taşınmıştır.

TB'de mevcut çözümün komşuları, GA'daki en iyi dizinin rassal olarak seçilen tek bir geninde değişiklik yapılarak elde edilmiştir. Örneğin, Bölüm 3, Şekil 3.6'da görülen GYP için GA'nın herhangi bir neslindeki yığında yer alan en iyi çözüm (2-4-7-16-11), TB'de her sıcaklıkta tekrar sayısı (*r*) beş olsun. TB algoritmasında üretilen komşu çözümler Şekil 3.12'de görülmektedir.



Şekil 3.12 Tavlama Benzetimi algoritmasında komşu üretme

GYP için geliştirilen TB algoritmasının adımları Şekil 3.13'te sunulmuştur.

Adım-1: GA'nın *i*. neslindeki en iyi çözümü başlangıç çözümü olarak al, başlangıç sıcaklık değerini (*t*) ve her sıcaklık derecesindeki tekrarlama sayısını (*r*) belirle.

Adım-2: Durdurma kriteri sağlanana kadar aşağıdaki adımları tekrarla:

i. Gelinen sıcaklık derecesinde aşağıdaki adımları *r* kez tekrarla:
Mevcut çözümün bir genine rassal bir değişiklik uygulayarak yeni çözümü (*T'*) elde et. Prim Algoritması ile her iki çözümün amaç fonksiyonu değerlerini hesapla, İki amaç fonksiyonu değeri arasındaki fark olan $\delta E = \delta E(T') - \delta E(T)$ değerini hesapla.

ii. O ile 1 arasında rassal bir *x* değeri üret

iii. Eğer $\delta E < 0$ veya $x < \exp(-\delta E / kt)$ ise T=T' olarak değiştir.

iv. t sıcaklık parametresi değerini *t(i+1) = s.t(i)* olarak güncelle.

Adım-3: Mevcut çözüm *T'*yi en iyi çözüm olarak GA'nın (i+1).nesline ekle.

Şekil 3.13 GYP için geliştirilen TB algoritması

Algoritma ile ilgili parametre değerlerinin hassas bir şekilde belirlenmesi için Bölüm 4'te açıklanan Cevap Yüzeyi metodu kullanılmış, dikkate alınan düzeyler ise algoritma ile yapılan ön denemelere göre belirlenmiştir.

Bu tezde ele alınan GYP için TB'nin temel hali uygulanarak elde edilen algoritma literatürde yer alan test problemleri üzerinde uygulanmış ve sonuçları Bölüm 5'te değerlendirilmiştir.

3.4 Genelleştirilmiş Yayılma Problemi İçin Bir Melez Yaklaşım

Birleşi eniyileme problemlerinde kullanılan sezgisel yöntemler bazen istenilen kalitede çözümler üretememektedir. Bu durumda algoritma yerel eniyiye takılarak, daha iyi çözümler üretememiş ve en iyi çözümden yüksek oranlarda sapan çözümler

elde etmiş olabilir. Bu durumla başa çıkmak için algoritmanın performansını iyileştirici çalışmalar yapılmasına rağmen bazen bu işlemler tek başına yeterli olamamaktadır.

Bu problemleri ortadan kaldırmak için son yıllarda çok kullanılan diğer bir yaklaşım sezgisel yöntemlerden birkaçını birlikte kullanmaktır. Literatürde, evrimsel algoritmalarla birlikte bir yerel arama yönteminin kullanılmasının, daha iyi kalitede çözümler üretilmesine olanak sağladığı belirtilmiştir [6].

Literatürde birçok birleşi probleminin çözümü için çok sayıda melez yöntem önerilmiştir. NP-zor bir problem olan İletişim hatlarının ana omurga ağının tasarımı probleminde Yerel arama ile GA birlikte kullanılmış ve bu yöntemin daha önce kullanılan yöntemlerden daha iyi çözümler ürettiğini belirtmişlerdir [20]. Buradan hareketle, bu tezde ele alınan GYP için GA ve TB'nin birlikte kullanılması fikriyle yola çıkılmış ve bir melez algoritma elde edilmiştir.

3.4.1 Geliştirilen Melez Algoritma

Bu bölümde, GYP için geliştirilen melez yöntem ele alınmaktadır. Bu yöntem daha önceki bölümlerde detaylı olarak incelenen GA ile TB algoritmalarının birleştirilmesiyle oluşturulmuştur. Buna göre, GA belirtilen nesil sayısı kadar çalıştırılmakta, bir nesilden diğerine geçilirken yığında bulunan en iyi çözüm TB algoritmasının başlangıç çözümü olarak ele alınmakta, belirtilen iterasyon sayısı kadar TB çalıştırılmakta, mevcut çözümden daha iyi bir çözüm elde edilirse yeni yığına bu çözüm taşınmakta, TB ile iyileşme sağlanamazsa mevcut çözüm yeni yığına taşınarak GA ile aramaya devam edilmektedir. Bu adımlar Bölüm 3.3'te anlatılan TB algoritması içinde yer almaktadır.

Bu melez yapıda, Bölüm 3.2'de anlatılan GA'nın sadece mutasyon olasılığını güncelleme işlemi kullanılmamıştır. Geliştirilen melez yöntemin genel akışı Şekil 3.14'te görüldüğü gibidir.

Başla

Adım 1: *i=1* olarak nesil sayacı değerini ata.

Adım 2: *i.* nesil için GA adımlarını gerçekleştir.

Adım 3: Yığında yer alan eniyi dizi için TB algoritmasını çalıştır.

Adım 4: *i=i+1* olarak güncelle. Nesil sayısına ulaşılmışsa dur, değilse Adım 2'ye dön

Bitir

Şekil 3.14 Geliştirilen melez yöntemin adımları

3.5 Genelleştirilmiş Yayılma Problemi İçin Kuş Sürüsü Algoritması

Kuş Sürüsü Algoritması (KSA) nümerik ve nitel problemlerin çözümünde kullanılmak için geliştirilmiş yeni bir algoritmadır [18]. İlk olarak 1995 yılında, sosyal-psikolog olan James Kennedy ve elektrik mühendisi Russel Eberhart tarafından balık ve kuş sürülerinin toplu hareketlerinden ilham alınarak geliştirilmiştir. Yiyecek bulmaya çalışan bir kuş veya balık sürüsü yiyeceğin gerçek yerini bilmemelerine rağmen, sürüde yer alan diğer bireylerin hareketlerinden yiyeceğe ne kadar uzakta olduklarını belirlemeye çalışırlar. Bunun için ise yiyeceğe en yakın konumda bulunan kuşu izlemeye yönelirler. KSA, bu fikirden yola çıkarak, her iterasyonda sürüde yer alan tüm kuşların, en iyi kuşun pozisyonuna göre kendi pozisyonlarını güncellemesiyle çalışmaktadır.

Kuş Sürüsü Algoritması, rassal olarak üretilen ve sürü olarak adlandırılan başlangıç yığını ile aramayı başlatır. Sürüdeki her kuş üç bileşenden oluşmaktadır. Bunlar hız, konum ve uygunluk değerleridir. Arama uzayında, tüm kuşların elde ettiği deneyimler, sosyal olarak paylaşılarak sürünün tüm üyeleri tarafından kullanılmaktadır.

3.5.1 Kuş Sürüsü Algoritması İle İlgili Temel Kavramlar

Doğadaki balık ve kuş sürülerinin hareketlerini taklit eden KSA, temel olarak aşağıdaki üç ana adımdan oluşmaktadır [18].

i. Değerlendirme işlemi
ii. Karşılaştırma
iii. Taklit etme

Bu adımlar aşağıda detaylı olarak açıklanmaktadır.

i) Değerlendirme İşlemi

Sürüde yer alan kuşların, ele alınan problemin amacını dikkate alarak birbirleriyle kıyaslanabilmesi, içlerinden iyi olanın belirlenmesi için dikkate alınacak kriter değerlendirme aşaması olarak adlandırılmaktadır. Algoritmada öğrenmenin sağlanabilmesi için, sürüdeki kuşların değerlendirilmesi gerekmektedir. GYP bir enküçükleme problemi olduğundan, sürüdeki en iyi kuş, amaç fonksiyonu değeri en küçük olan kuştur. Bundan dolayı amaç fonksiyonu değeri, kuşların değerlendirilmesinde doğrudan kullanılabilmektedir.

ii) Karşılaştırma

Değerlendirme işleminden sonra sürüdeki kuşlar, amaç fonksiyonu değerlerini komşularıyla kıyaslayarak, sonraki adımlarında edindikleri bilgileri kullanabilmeleri gerekmektedir. Buradaki kıyaslama işlemi karşılaştırma olarak isimlendirilmektedir. KSA'daki tüm kuşların birbirleriyle edindikleri bilgi ve tecrübeleri paylaşmaları için karşılaştırma işlemi önem taşımaktadır.

iii) Taklit Etme

Doğada olduğu gibi, öğrenmenin etkili yollarından biri taklit etmektir. KSA'da sürüdeki kuşlar edindikleri bilgi ve tecrübeleri birbirleriyle paylaştıktan sonra, kendi yollarına devam ederken, bu bilgi ve tecrübeleri kullanmaktadırlar.

KSA'da kullanılan parametreler ve açıklamaları aşağıdaki gibidir;

x_{ijs}^{k} : k.çevrimde i.kuşun j.kümesinin s.elemanının konum değeri

x_{mak} : Herhangi bir kuşun konum değerinin alabileceği en yüksek değer

$x_{\min}$: Herhangi bir kuşun konum değerinin alabileceği en küçük değer

v_{ijs}^{k} : k.çevrimde i.kuşun j.kümesinin s.elemanının hız değeri

v_{mak} : Herhangi bir kuşun hız değerinin alabileceği en yüksek değer

$v_{\min}$: Herhangi bir kuşun hız değerinin alabileceği en küçük değer

pb_{ij}^{k} : k.çevrime kadar i.kuşun eniyi amaç fonksiyonu değerini aldığı durumda j.kümesinin ilk konum değeri

gb_{j}^{k} : k.çevrime kadar sürüdeki tüm kuşların içinden eniyi amaç fonksiyonu değerini alan kuşun j.kümesinin ilk konum değeri

w : Eylemsizlik faktörü (0.4-1.4)

c_1 : Öğrenme faktörü (1.5-2)

c_2 : Öğrenme faktörü (2-2.5)

k : Çevrim sayacı

k_{mak} : Algoritmanın çalıştırılacağı çevrim sayısı

r_1, r_2 : *U(0,1)* olan rassal sayılar

Standart bir KSA kısaca Şekil 3.15'te gösterildiği gibi çalışır [28].

Adım-1: Rassal olarak sürü genişliği kadar kuş üret.

Adım-2: Üretilen her kuşun başlangıç konumunu belirle.

$$x_{ijs}^{0} = x_{\min} + (x_{mak} - x_{\min}) * U(0,1)$$

Adım-3: Üretilen her kuşun başlangıç hızını hesapla.

$$v_{ijs}^{0} = v_{\min} + (v_{mak} - v_{\min}) * U(0,1)$$

Adım-4: Kuşları konum değerlerine göre değerlendir ve karşılaştır.

Adım-5: k.çevrimde sürüdeki kuşlardan amaç fonksiyonu değeri enküçük olan kuşun konum değerlerini pb_{ijs}^{k} olarak sakla. O ana kadarki en iyi kuşun konum değerlerini gb_{j}^{k} olarak sakla.

Adım-6: Çevrim sayısını bir artır. Eğer belirlenen çevrim sayısına ulaşılmış ise DUR, değilse **Adım-7**'e geç.

Adım-7: Sürüdeki her kuşun hız değerlerini güncelle.

$$v_{ijs}^{k} = wv_{ijs}^{k-1} + c_1 r_1 (pb_{ij}^{k-1} - x_{ijs}^{k-1}) + c_2 r_2 (gb_{j}^{k-1} - x_{ijs}^{k-1})$$

Adım-8: Sürüdeki her kuşun konum değerlerini güncelle. **Adım-4**'e dön.

$$x_{ijs}^{k} = x_{ijs}^{k-1} + v_{ijs}^{k}$$

Şekil 3.15 Standart bir KSA'nın adımları

KSA yakın bir zamanda geliştirilmiş olduğundan, literatürde yapılmış çalışma sayısı oldukça sınırlıdır. Güç ve voltaj kontrolü problemi [30], optimal güç akışı tasarımı problemi [1], iş atama problemi [26] gibi alanlarda kullanılmıştır. KSA'yı ilk kullanan araştırmacılar Kennedy ve Eberhart [19], lineer olmayan sürekli fonksiyonlarda başarıyla uygulamıştır.

2004 yılında yapılan bir çalışmada KSA, tek makineli toplam ağırlıklandırılmış gecikme probleminde uygulanmıştır. Bu çalışmanın dikkat çekici yönü, sürekli optimizasyon problemlerinde uygulanan KSA'nın kesikli birleşi eniyileme problemlerinde de uygulanabileceğinin gösterilmiş olmasıdır [28].

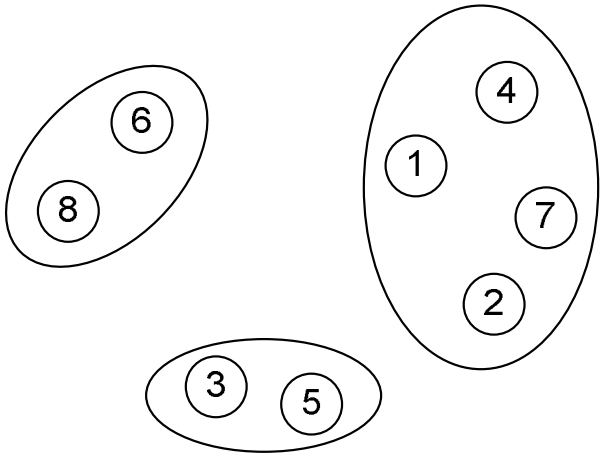
Bir diğer çalışmada ise, literatürde parti büyüklüğü problemi olarak bilinen toplam stok tutma ve sipariş verme maliyetini enküçükleme problemi için KSA geliştirilmiştir [29]. Geliştirilen algoritma 20 test problemi üzerinde denenmiş ve aynı

problem için geliştirilen GA ile sonuçları karşılaştırılmıştır. Geliştirilen KSA'nın, 19 problem için eniyi sonucu bulduğu, GA'nın ise 14 problem için eniyi çözümü verdiği belirtilmiştir.

3.5.2 Geliştirilen Kuş Sürüsü Algoritması

Literatürde, GYP için KSA daha önce uygulanmamıştır. Bu tezde, KSA GYP'ye uyarlanmış ve test problemleri üzerinde denenmiştir.

KSA'da kuş olarak adlandırılan uygun çözümlerin gösterimi GA'daki dizilere denk gelmektedir. GA'da olduğu gibi, KSA'daki kuşlar ikili ve tamsayılı düzende oluşturulabilmektedir. Ancak, KSA'da farklı olarak her kuş ayrıca rassal olarak üretilen hız ve konum değerlerine sahiptir. Örneğin, Şekil 3.16'da yer alan üç kümeli ve toplam sekiz düğümlü bir GYP görülmektedir.



Şekil 3.16 Üç kümeli, sekiz düğümlü GYP örneği

Üç kümeden oluşan bu GYP için oluşturulacak kuşlarda her bir kümenin elemanları bulunmalıdır. Bunun için kuşlarda *j=3* olacak şekilde bir kümeleme yapılmıştır. Her bir kümenin elemanları ise ilgili *j* değerinin bulunduğu sütunda yer almaktadır. Çizelge 3.2'de örnek kuş gösterimi yer almaktadır. x_{ijs}^{k} konum değerleri 0 ile 4 arasında, v_{ijs}^{k} hız değerleri -4 ile 4 arasında rassal olarak atanmış olsun.

Çizelge 3.2: Örnek kuş gösterimi

j	j=1		j=2				j=3	
s	6	8	1	2	4	7	3	5
x_{ijs}^{k}	1,03	0,77	3,31	2,38	0,08	1,33	1,32	2,36
v_{ijs}^{k}	-0,88	2,27	3,83	0,29	-3,34	2,06	2,66	1,64
sıralama	8	6	4	7	2	1	3	5

Çizelge 3.2'de görüleceği üzere bu gösterimin, konum değerlerine göre sıralamadan sonra ifade ettiği uygun çözüm (8-4-3) düğümlerinden oluşmaktadır. Burada 1.kümenin elemanları olan 6 ve 8 numaralı düğümlerden konum değeri düşük olan 8 numaralı düğüm, ikinci kümeden 4 numaralı düğüm ve üçüncü kümeden 3 numaralı düğüm seçilmiştir. Seçilen bu düğümler arasında Prim algoritması kullanılarak enküçük yayılan ağaç elde edildiğinde bulunan çözüm, üç kümeli GYP için bir uygun çözüm olarak ortaya çıkmaktadır.

GYP'nin çözümünde ilk defa bu çalışmada önerilen KSA'nın adımları Şekil 3.16'da gösterilmektedir.

Adım-1: Rassal olarak sürü genişliği kadar kuş üret.

Adım-2: Üretilen her kuşun başlangıç konumunu belirle.

$$x_{ijs}^{0} = x_{\min} + (x_{mak} - x_{\min}) * U(0,1)$$

Adım-3: Üretilen her kuşun başlangıç hızını hesapla.

$$v_{ijs}^{0} = v_{\min} + (v_{mak} - v_{\min}) * U(0,1)$$

Adım-4: Kuşların içinde yer alan her küme için, konum değerlerini küçükten büyüğe doğru sırala.

Adım-5: Oluşan yeni sıralamada her kümenin ilk düğümünü seç. Seçilen düğümler arasında Prim algoritması kullanarak enküçük yayılan ağacı elde et, amaç fonksiyonu değerini hesapla.

Adım-6: k.çevrimde sürüdeki kuşlardan amaç fonksiyonu değeri enküçük olan kuşun konum değerlerini pb_{ijs}^{k} olarak sakla. O ana kadarki en iyi kuşun konum değerlerini gb_{j}^{k} olarak sakla.

Adım-7: Çevrim sayısını bir artır. Eğer belirlenen çevrim sayısına ulaşılmış ise DUR, değilse **Adım-8**'e geç.

Adım-8: Sürüdeki her kuşun hız değerlerini güncelle.

$$v_{ijs}^{k} = wv_{ijs}^{k-1} + c_1 r_1 (pb_{ij}^{k-1} - x_{ijs}^{k-1}) + c_2 r_2 (gb_{j}^{k-1} - x_{ijs}^{k-1})$$

Adım-9: Sürüdeki her kuşun konum değerlerini güncelle. **Adım-4**'e dön.

$$x_{ijs}^{k} = x_{ijs}^{k-1} + v_{ijs}^{k}$$

Eğer $x_{ijs}^{k} < x_{\min}$ ise $x_{ijs}^{k} = x_{\min}$ olarak, eğer $x_{ijs}^{k} > x_{mak}$ ise $x_{ijs}^{k} = x_{mak}$ olarak güncelle.

Şekil 3.17 GYP için geliştirilen KSA

GYP için geliştirilen KSA'nın parametre değerlerinin belirlenmesi ayrı bir eniyileme problemidir. Tüm sezgisel algoritmalarda olduğu gibi KSA'da da parametre kümesinin belirlenmesi önemli bir problemdir [16]. Bölüm 4'te bu tezde geliştirilen tüm algoritmaların uygun parametre setinin nasıl belirlendiği detaylı bir şekilde anlatılmaktadır.

4. GELİŞTİRİLEN ALGORİTMALAR İÇİN UYGUN PARAMETRE DEĞERLERİNİN BELİRLENMESİ

GYP'nin çözümü için geliştirilen algoritmaların eniyi parametre kümelerinin belirlenmesi amacıyla, her algoritma için ayrı ayrı deney tasarımı yapılmıştır. Geliştirilen algoritmaların parametreleri Çizelge 4.1'de verilmektedir.

Çizelge 4.1: Algoritmaların kullandığı parametreler

GA			MY						KSA		
$p_ç$	p_m	Yığın Genişliği	$p_ç$	p_m	Yığın Genişliği	t	r	Soğutma Katsayısı	w	c_1	c_2

Her algoritma için 2^k faktör tasarımı dikkate alınarak deneyler yapılmıştır. Buna göre, her parametre için iki düzey, seçilen bir GYP'nin amaç fonksiyonu değerlerine göre karşılaştırılmış, amaç fonksiyonu değerini daha küçük yapan parametre değeri seçilerek, algoritmalar bu parametre değerleriyle çalıştırılarak sonuçlar elde edilmiştir.

4.1 Geliştirilen Genetik Algoritma İçin Eniyi Parametre Kümesinin Belirlenmesi

Çaprazlama ve mutasyon olasılığı ile yığın genişliği GA'ların performansını etkileyen en önemli faktörlerdir. Literatürde yer alan çalışmalarda çaprazlama olasılığı genellikle (0.6-0.8) arasında, mutasyon olasılığı (0.01-0.05) arasında, yığın genişliği (20-40) arasında değerler verilerek kullanılmıştır. Ancak bu değerler ele alınan probleme göre değişiklik göstermektedir. Yani bu parametreler literatürde önerilen değerleriyle her probleme uygulanamamaktadırlar. Bu çalışmada kullanılan GA'nın bu üç parametre için uygun değerlerinin bulunması amacıyla yapılan 2^3

deney tasarımıyla, problem üzerinde algoritma performansının iyileştirilmesi sağlanmıştır. Stat-Ease deney tasarımı programı ile yapılan deney tasarımında TSPLIB'den seçilen 10hk48 problemi kullanılmıştır. Bu problem için deney tasarımı, parametre değerlerine en büyük ve en küçük değerleri verilerek yapılmıştır. Her parametre düzeyinde algoritma beşer kez çalıştırılmış ve ortalama değerler Çizelge 4.2'de görüldüğü gibi programa girilmiştir.

Çizelge 4.2: GA için deney tasarımı ve sonuçları

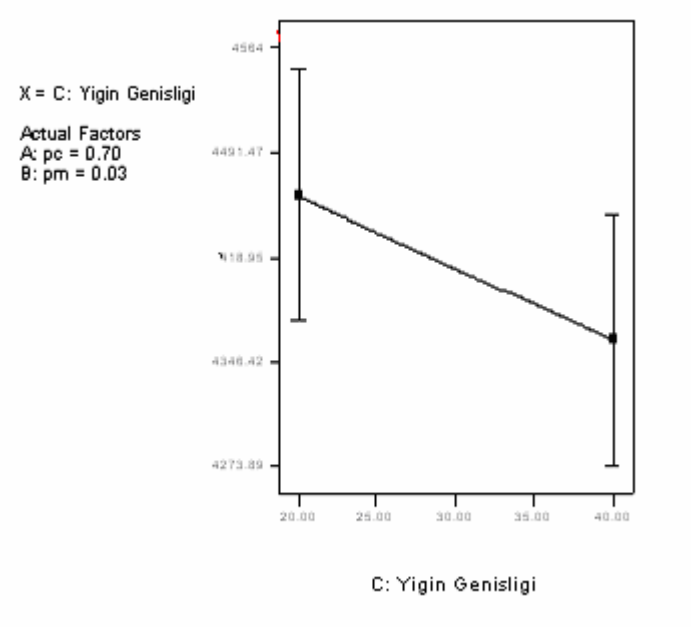
	Çaprazlama olasılığı	Mutasyon olasılığı	Yığın genişliği	
Deney	A	B	C	Cevap
1	0,6	0,01	20	4405
2	0,8	0,01	40	4376
3	0,6	0,05	40	4396
4	0,8	0,05	20	4441
5	0,8	0,01	20	4564
6	0,6	0,01	40	4326
7	0,6	0,05	20	4436
8	0,8	0,05	40	4347

Elde edilen ANOVA çizelgesi Çizelge 4.3'te verilmektedir.

Çizelge 4.3: GA için elde edilen ANOVA Çizelgesi

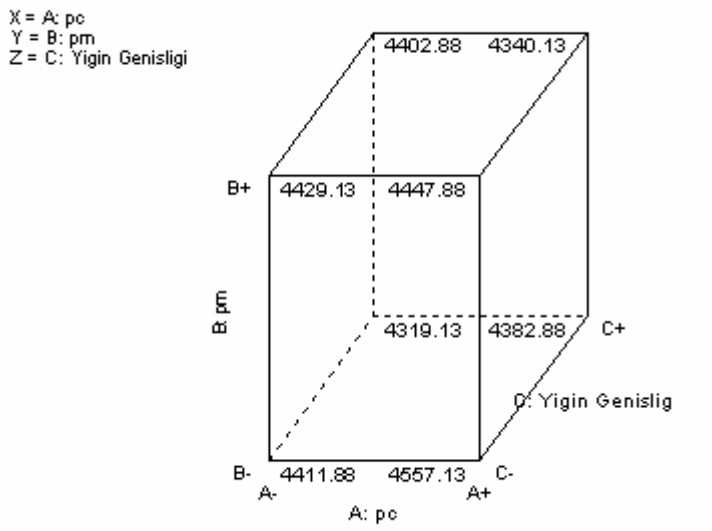
Kaynak	**Kareler toplamı**	**Serbestlik derecesi**	**Ortalama karesi**	**F değeri**	**P değeri**
Model	37361,75	6	6226,958333	16,46798898	0.1864
A	3403,125	1	3403,125	9	0.2048
B	325,125	1	325,125	0,859834711	0.5240
C	20100,125	1	20100,125	53,15735537	0.0868
AB	8001,125	1	8001,125	21,16	0.1363
AC	3321,125	1	3321,125	8,783140496	0.2072
BC	2211,125	1	2211,125	5,847603306	0.2496
Fark	378,125	1	378,125		
Kor Toplam	37739,875	7			

Çizelge 4.3'ten görüleceği üzere, $\alpha = 0.10$ güvenilirlik düzeyinde sadece yığın genişliği seviyeleri arasında algoritmanın performansında istatistiksel olarak anlamlı bir fark vardır.

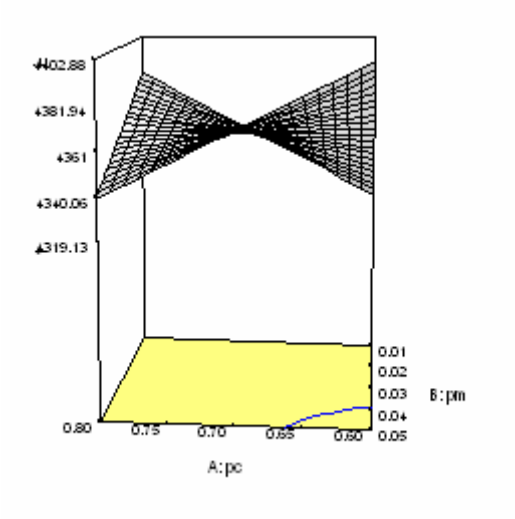


Şekil 4.1 Yığın genişliğinin algoritma performansı üzerine etkisi

Şekil 4.1'den de görüleceği üzere, yığın genişliği arttığında amaç fonksiyonu değeri anlamlı şekilde düşmektedir. Bu sebeple yığın genişliği 40 olarak seçilmiştir. Çaprazlama olasılığı ve mutasyon olasılığı seviyeleri arasında istatistiksel olarak anlamlı bir fark yoktur. Bu sebeple, bu faktörlerin herhangi bir seviyesi seçilebilir. Bu tezde, amaç fonksiyonunun eniyi değerini aldığı noktadaki değerleri seçilerek kullanılmıştır. Yapılan ön denemelere göre nesil sayısı 200 olarak belirlenmiştir. Analiz sonucu elde edilen verilerle Şekil 4.2'de gösterilen küp ve Şekil 4.3'te gösterilen cevap yüzeyi (response surface) elde edilmiştir.



Şekil 4.2 Genetik Algoritma için küp grafiği



Şekil 4.3 GA için elde edilen cevap yüzeyi

Buna göre, amaç fonksiyonun enküçük değerini aldığı noktada, $p_c = 0.6$, $p_m = 0.01$, yığın genişliğinin 40 değerlerini aldığı görülmektedir. Bu parametre setinin değerlerine göre, amaç fonksiyonunun değişimi Şekil 4.2'de görülmektedir. Şekil 4.3'te görüldüğü üzere, çaprazlama olasılığının mutasyon olasılığı ile birlikte azalması durumunda amaç fonksiyonu değerinin iyileştiği, benzer şekilde çaprazlama olasılığının mutasyon olasılığı ile birlikte artması durumunda amaç fonksiyonu değerinin iyileştiği, ancak ilk durumda daha iyi sonuçlar elde edildiği görülmektedir. *Y* benzetim çıktısı, A çaprazlama olasılığı faktörünü, B mutasyon olasılığı faktörünü ve C yığın genişliği faktörünü ifade etmektedir.

$$Y = 3741.81 + (1291.87)A + (825625)B + (6.75)C - (15812)AB - (20.37)AC + (83.12)BC \qquad (4.1)$$

Buna göre elde edilen modelin regresyon denklemi (4.1)'de görülmektedir.

4.2 Geliştirilen Melez Yöntem İçin Eniyi Parametre Kümesinin Belirlenmesi

Geliştirilen melez algoritma, GA ve TB'yi birlikte kullandığından, yapılan deney tasarımında her iki algoritmanın kullandığı parametreler dikkate alınmıştır. GA için yapılan deney tasarımında elde edilen parametre değerleri aynen kullanılmış, TB için ise deney tasarımı parametre değerleri ayrıca yeniden bulunmuştur.

Bu çalışmada kullanılan başlangıç sıcaklığı değeri (500-1000) düzeylerinde, her sıcaklık derecesindeki tekrarlama sayısı (3-5) düzeylerinde ve soğutma katsayısı (0.98-0.95) düzeylerinde incelenmiştir. Toplam üç parametre kullanıldığından, 2^3 faktör tasarımı gereği deneyler yapılmıştır. Her parametre düzeyinde algoritma beşer kez çalıştırılmış ve ortalama değerler Çizelge 4.4'te görüldüğü gibi elde edilmiştir.

Çizelge 4.4: TB için deney tasarımı ve sonuçları

	Başlangıç sıcaklığı	Her sıcaklıkta tekrar sayısı	Soğutma katsayısı	
Deney	A	B	C	Cevap
1	1000	3	0,98	4312
2	500	3	0,98	4442
3	500	5	0,98	4386
4	1000	5	0,98	4278
5	1000	5	0,95	4239
6	500	5	0,95	4346
7	500	3	0,95	4470
8	1000	3	0,95	4317

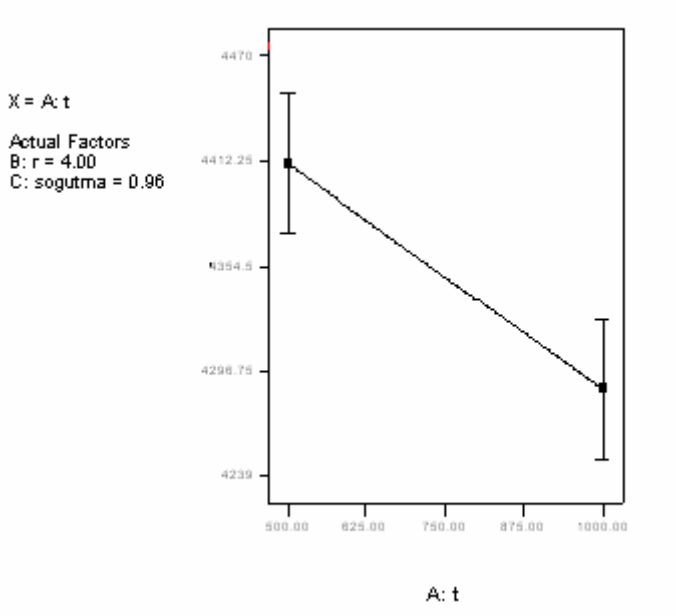
Elde edilen ANOVA çizelgesi Çizelge 4.5'te verilmektedir.

Çizelge 4.5: Geliştirilen TB için elde edilen ANOVA çizelgesi

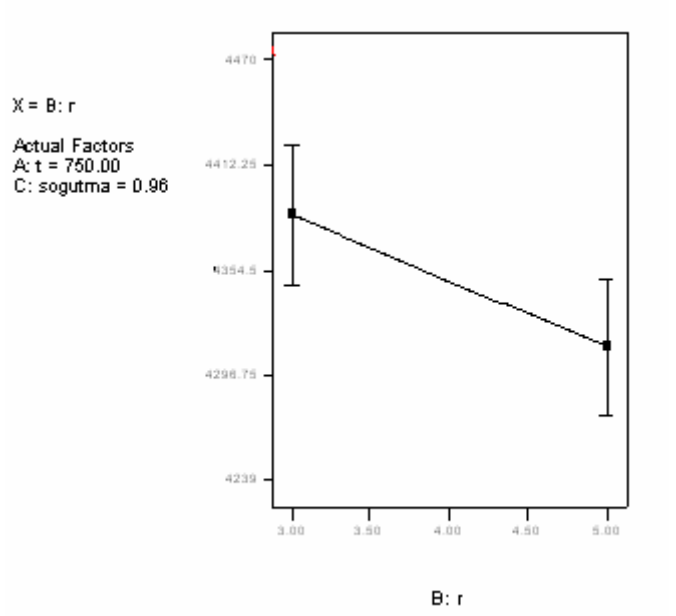
Kaynak	**Kareler toplamı**	**Serbestlik derecesi**	**Ortalama karesi**	**F değeri**	**P değeri**
Model	44129,50	6	7354,92	102,15	0.0756
A	31000,50	1	31000,50	430,56	0.0307
B	10658,00	1	10658,00	148,03	0.0522
C	264,50	1	264,50	3,67	0.3061
AB	578,00	1	578,00	8,03	0.2160
AC	60,50	1	60,50	0,84	0.5277
BC	1568,00	1	1568,00	21,78	0.1344
Fark	72,00	1	72,00		
Kor Toplam	44201,50	7			

Çizelge 4.5'te görüleceği üzere, $\alpha = 0.10$ güvenilirlik düzeyinde başlangıç sıcaklık değeri (*t*) ve her sıcaklıkta tekrar sayısı (*r*) yığın genişliği seviyeleri arasında algoritmanın performansında istatistiksel olarak anlamlı bir fark vardır. Soğutma katsayısının düzeyleri arasında istatistiksel olarak anlamlı bir fark yoktur.

Şekil 4.4 ve Şekil 4.5'te sırasıyla başlangıç sıcaklığının ve her sıcaklıktaki tekrar sayısının algoritma performansı üzerindeki etkisi görülmektedir.

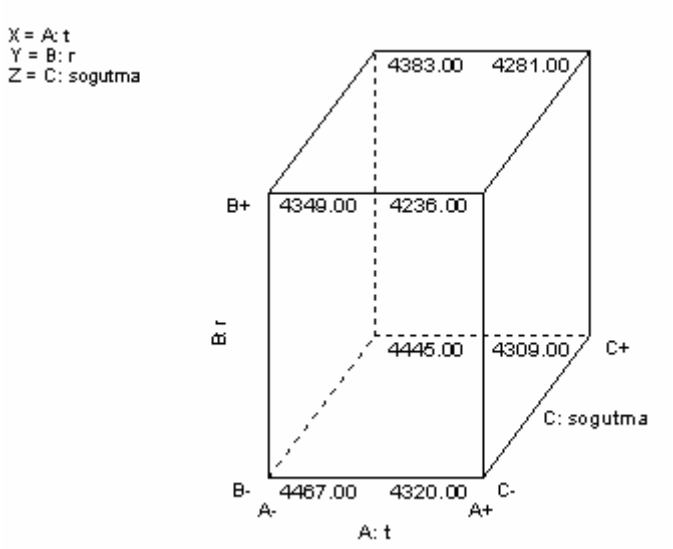


Şekil 4.4 Başlangıç sıcaklığının algoritma performansı üzerine etkisi

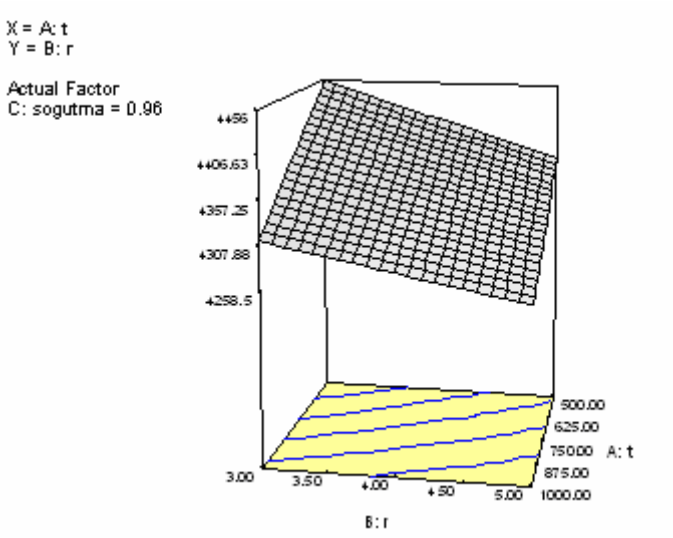


Şekil 4.5 Her sıcaklık algoritma performansı üzerine etkisi

TB için elde edilen küp ve cevap yüzeyi Şekil 4.6 ve Şekil 4.7'de yer almaktadır.



Şekil 4.6 Tavlama Benzetimi için küp grafik



Şekil 4.7 Tavlama Benzetimi için etki yüzeyi

Buna göre, amaç fonksiyonun enküçük değerini aldığı noktada, başlangıç sıcaklığı değerinin (*t*) 1000, her sıcaklık derecesinde yapılacak tekrar sayısının (*r*) 5 ve soğutma katsayısı 0.95 değerlerini aldığı görülmektedir. Bu parametre setinin değerlerine göre, amaç fonksiyonunun değişimi Şekil 4.7'de görülmektedir.

Şekil 4.7'de görüldüğü üzere, başlangıç sıcaklığı değerinin artması amaç fonksiyonu değerini düşürmektedir. Her sıcaklık derecesinde yapılacak gözlem sayısının artması, yine amaç fonksiyonu değerini düşürmektedir. Bundan dolayı yapılan denemelerde, başlangıç sıcaklığı değeri (*t*) 1000, her sıcaklık değerinde yapılacak gözlem sayısı (*r*) 5 ve soğutma katsayısı 0.95 olarak alınmıştır. Geliştirilen

algoritmaların çözüm uzayında inceledikleri nokta sayılarının yakın tutulması için GA 80 nesil, GA'nın her nesilinde kullanılan TB'inde sıcaklık 12 kez azaltılarak kullanılmıştır.

4.3 Geliştirilen Kuş Sürüsü Algoritması İçin Eniyi Parametre Kümesinin Belirlenmesi

Daha önce Çizelge 3'te belirtildiği gibi, geliştirilen KSA'da, w, c_1, c_2 parametre değerleri dikkate alınarak deney tasarımı yapılmıştır. Literatürde w, c_1, c_2 parametreleri için farklı değerler tanımlanmıştır. Bu çalışmada w değeri (0.4-1.4) düzeylerinde, c_1 öğrenme faktörü (1.5-2.5) düzeylerinde, c_2 öğrenme faktörü değeri ise (2-2.5) düzeylerinde dikkate alınmıştır.

Kennedy'nin [18] belirttiği gibi kuşların alabileceği, en küçük hız değeri $v_{\min}$ = -4, en büyük hız değeri v_{mak} = 4 olarak alınmıştır. Kuşların konum değerleri ise, en küçük konum değeri olarak $x_{\min}$ = 0, en büyük konum değeri olarak x_{mak} = 4 düzeyleri dikkate alınmıştır.

Parametrelerin eniyi kümesi Cevap Yüzeyi metodu kullanılarak elde edilmiştir. 2^k faktöryel tasarımı uyarınca yapılan denemeler sonucu elde edilen veri Stat-Ease programı kullanılarak Cevap Yüzeyi elde edilmiştir. TSPLIB'den seçilen 10hk48 problemi örnek problem olarak dikkate alınmıştır. Bu problem için, KSA parametre düzeylerinin oluşturduğu tüm kombinasyonlar için Çizelge 4.6'da görüldüğü gibi denemiştir.

Çizelge 4.6: KSA için deney tasarımı ve sonuçları

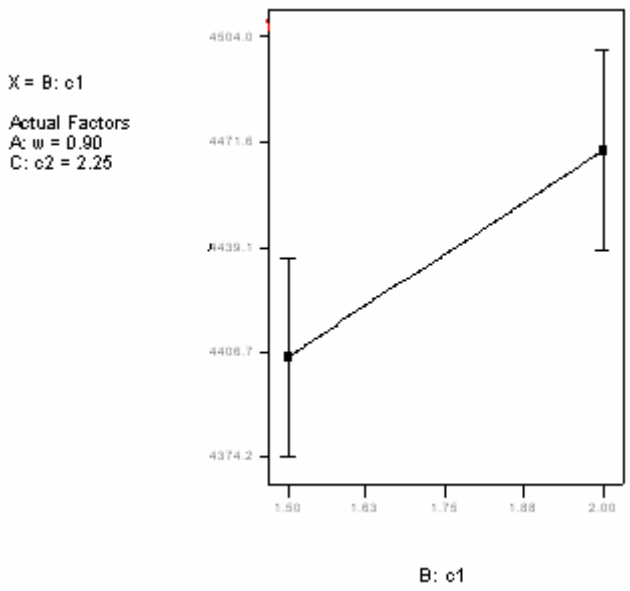
Deney	w değeri A	c1 değeri B	c2 değeri C	Cevap
1	0,4	1,5	2,5	4432
2	1,4	1,5	2,5	4385
3	0,4	2	2,5	4461
4	1,4	2	2,5	4486
5	0,4	2	2	4504
6	1,4	2	2	4425
7	1,4	1,5	2	4391
8	0,4	1,5	2	4412

Elde edilen ANOVA Çizelgesi Çizelge 4.7'de verilmektedir.

Çizelge 4.7: KSA için elde edilen ANOVA Çizelgesi

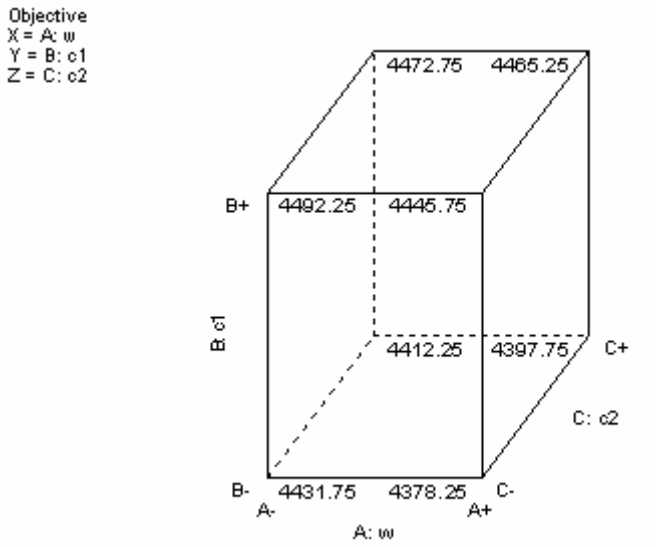
Kaynak	Kareler toplamı	Serbestlik derecesi	Ortalama karesi	F değeri	P değeri
Model	10837,50	4	2709,38	3,62	0.1592
A	1860,50	1	1860,50	2,49	0.2128
B	8192,00	1	8192,00	10,96	0.0454
AB	24,50	1	24,50	0,03	0.8679
AC	760,50	1	760,50	1,02	0.3874
Fark	2242,50	3	747,50		
Kor Toplam	13080,00	7			

Buna göre c_1 parametresinin düzeyleri arasında algoritmanın performansında istatistiksel olarak anlamlı bir fark vardır. c_1 faktörünün amaç fonksiyonuna etkisi Şekil 4.8'de görülmektedir.



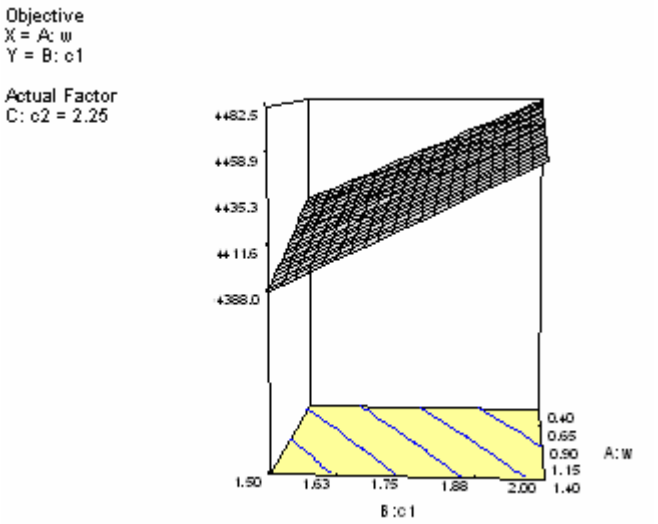
Şekil 4.8 c_1 faktörünün algoritma performansı üzerine etkisi

Çizelge 4.7'den de görüleceği üzere, $\alpha = 0.10$ güvenilirlik düzeyinde w ve c_2 parametrelerinin düzeyleri arasında istatistiksel olarak anlamlı bir fark yoktur. Bu sebeple, bu faktörlerin herhangi bir düzeyi seçilebilir. Bu çalışmada, yukarıdaki sonuçlardan hareketle, daha iyi sonuçların elde edildiği w ve c_2 düzeyleri seçilmiştir. Her parametre düzeyinde beşer kez algoritma 100 iterasyon için çalıştırılmış ve bu değerler programa girilerek Şekil 4.9'da gösterilen küp elde edilmiştir.



Şekil 4.9 Kuş Sürüsü Algoritması için küp grafiği

Buna göre, amaç fonksiyonun enküçük değerini aldığı noktada, w, c_1, c_2 parametrelerinin sırasıyla 1.4, 1.5 ve 2.5 değerlerini aldığı görülmektedir. Bu parametre kümesinin değerlerine göre, amaç fonksiyonunun değişimi Şekil 4.10'da görülmektedir.



Şekil 4.10 Kuş Sürüsü Algoritması için cevap yüzeyi

Şekil 4.10'da görüldüğü üzere, c_1 değerinin azalması, c_2 değerinin artırılması, w değerinin artırılması amaç fonksiyonu değerinin azalmasını sağlamaktadır. KSA'da kuş sayısı GA'da kullanılan yığın genişliğine eşit alınmış ve KSA 40 kuş için 200 iterasyon çalıştırılmıştır. Y benzetim çıktısı, A gösterimi w faktörü, B gösterimi c_1 faktörü ve C gösterimi c_2 faktörü olmak üzere, modelin regresyon denklemi (4.2)'de görüldüğüm gibi elde edilmiştir.

$$Y = 4437 - (15.25).A + (32).B + (1.75).A.B + (9.75).A.C \quad (4.2)$$

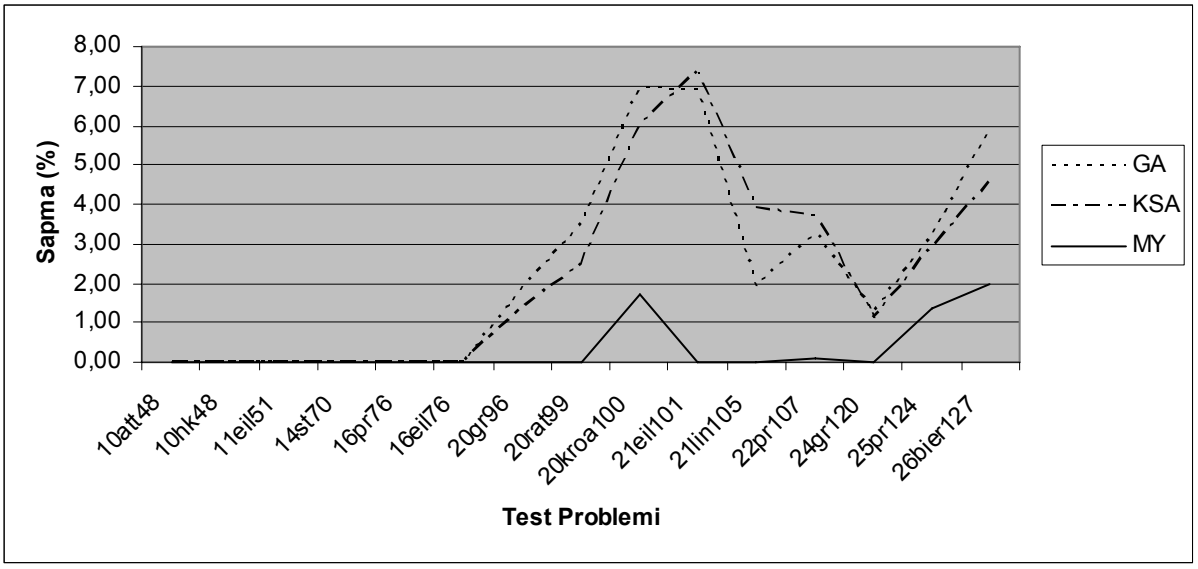
Bu bölümde elde edilen eniyi parametre değerleri kullanılarak algoritmalar TSPLIB'ten seçilen test problemleri üzerinde denenmiş ve elde edilen sonuçlar karşılaştırılmıştır. Detaylı karşılaştırmalar Bölüm 5'te tartışılmaktadır.

5. UYGULAMA VE TARTIŞMA

Bu tezde Genelleştirilmiş Yayılma Problemi için geliştirilen Genetik Algoritma, Melez Algoritma ve Kuş Sürüsü Algoritması, Bölüm 4'te incelenen parametre değerleriyle, Pentium (4) 1.5 Ghz işlemcili bilgisayarda VBA derleyicisi kullanılarak çalıştırılmıştır.

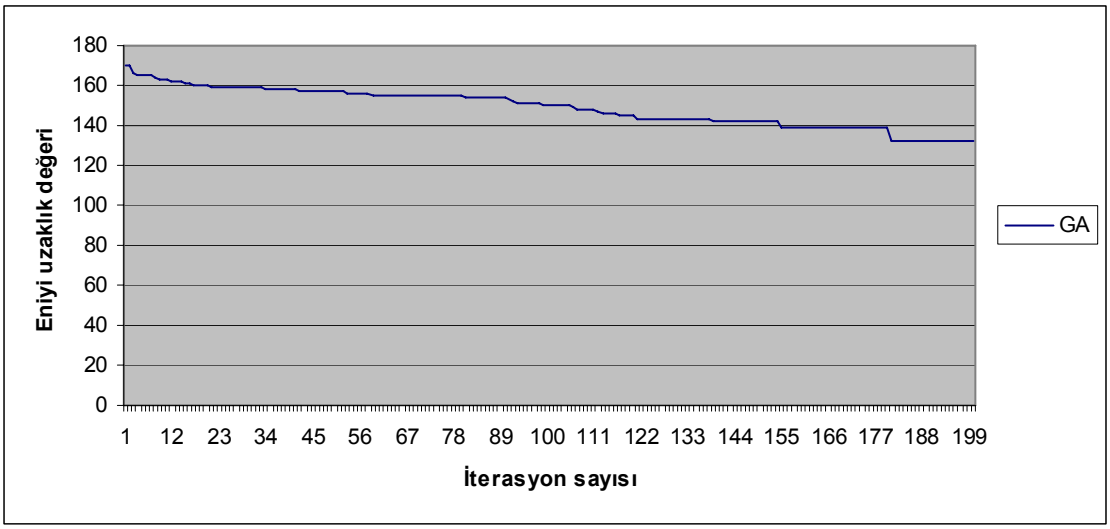
Eşit koşullarda bir karşılaştırma yapılabilmesi amacıyla her üç algoritma ile eşit sayıda arama yapılmasına çalışılmıştır. Geliştirilen algoritmaların yakınsama noktasına gelene dek arama uzayında inceledikleri nokta sayıları yaklaşık 8000 civarında tutulmuştur.

TSPLIB'de yer alan ve eniyi çözümü bilinen 15 test probleminin her biri için GA, KSA ve melez yöntemin performansları (bilinen eniyi sonuçlardan sapma değerleri) Şekil 5.1'de görülmektedir. Geliştirilen algoritmalarla elde edilen çözüm değerleri ve çözüm zamanları EK-1'de verilmektedir.

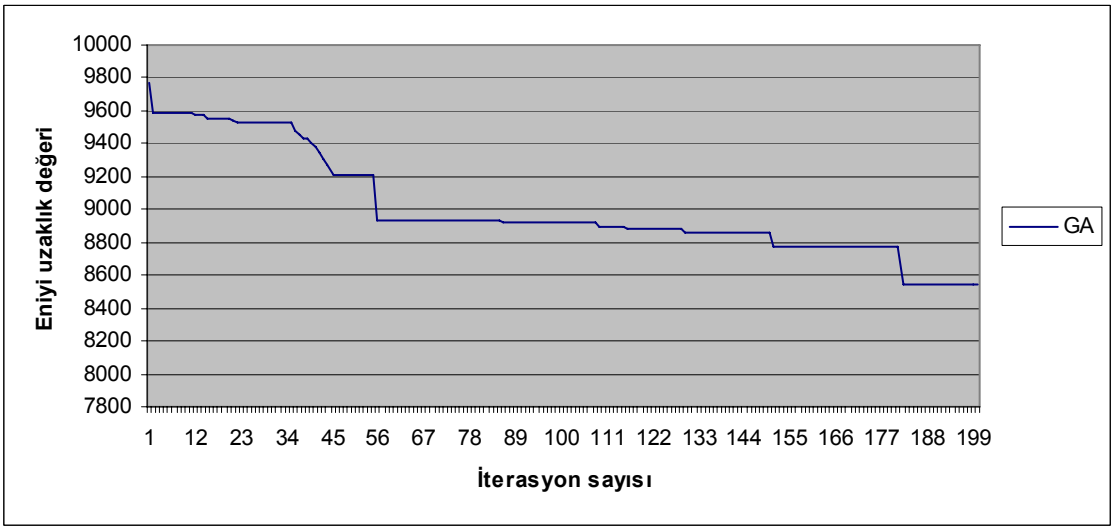


Şekil 5.1 15 Test Problemi İçin Algoritma Sonuçları

Buna göre, GA ortalama %2.31, KSA ortalama %2.23, melez yöntem ortalama %0.34 sapma ile eniyi çözüme yaklaşmaktadır. Algoritmalar küçük boyutlu problemler için eniyi sonucu elde edilirken problem boyutu büyüdükçe hata oranının arttığı görülmektedir. Geliştirilen melez yöntem, bu çalışmada geliştirilen GA'ya göre yaklaşık olarak %1.97 oranında iyileşme sağlamıştır. KSA ise GA'ya göre yaklaşık olarak %0.08 oranında üstünlük sağlamıştır. Çözüm zamanları açısından algoritma performansları, GA için ortalama 640.6, KSA için ortalama 606.8 ve melez yöntem için 694.07 saniyedir. Test problemlerinden bazıları için GA'nın yakınsama grafikleri Şekil 5.2 ve Şekil 5.3'te görülmektedir.

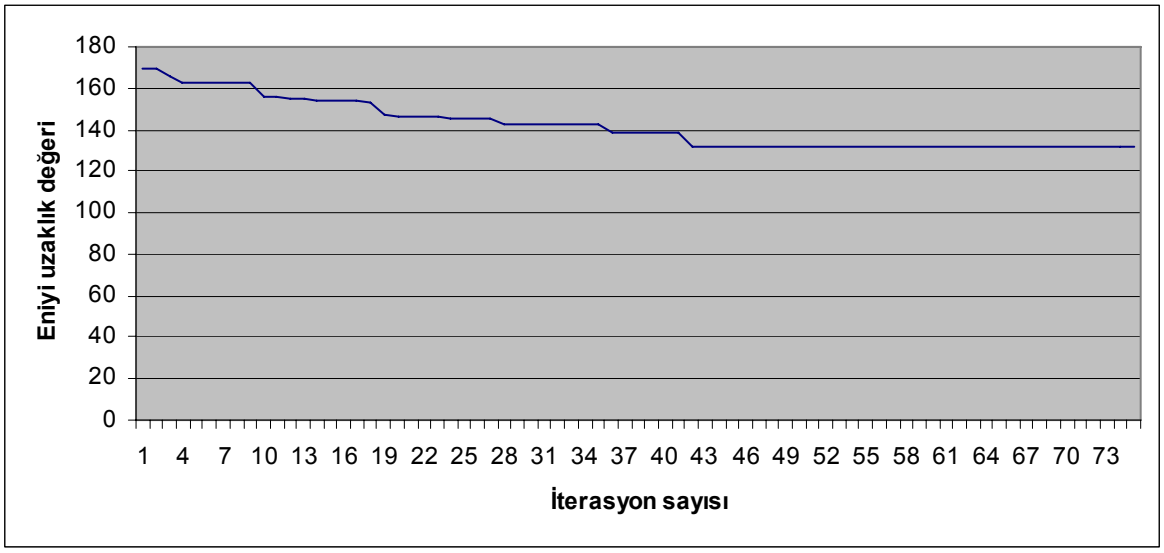


Şekil 5.2 GA'nın 11eil51 Problemi İçin Yakınsama Grafiği

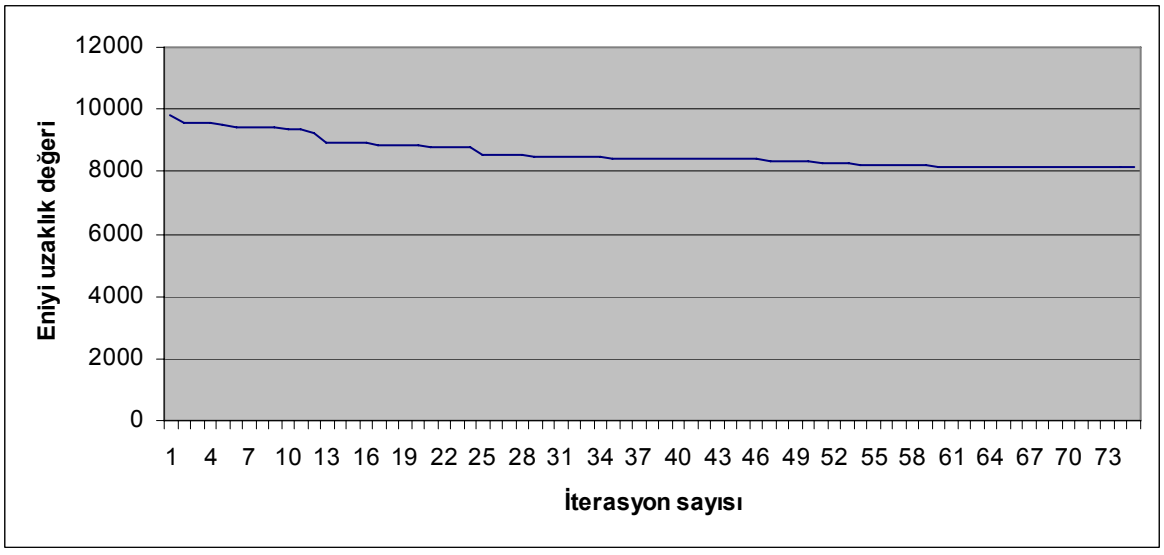


Şekil 5.3 GA'nın 20kroa100 Problemi İçin Yakınsama Grafiği

Geliştirilen melez yöntemin aynı test problemleri için yakınsama grafikleri Şekil 5.4 ve Şekil 5.5'da görülmektedir.

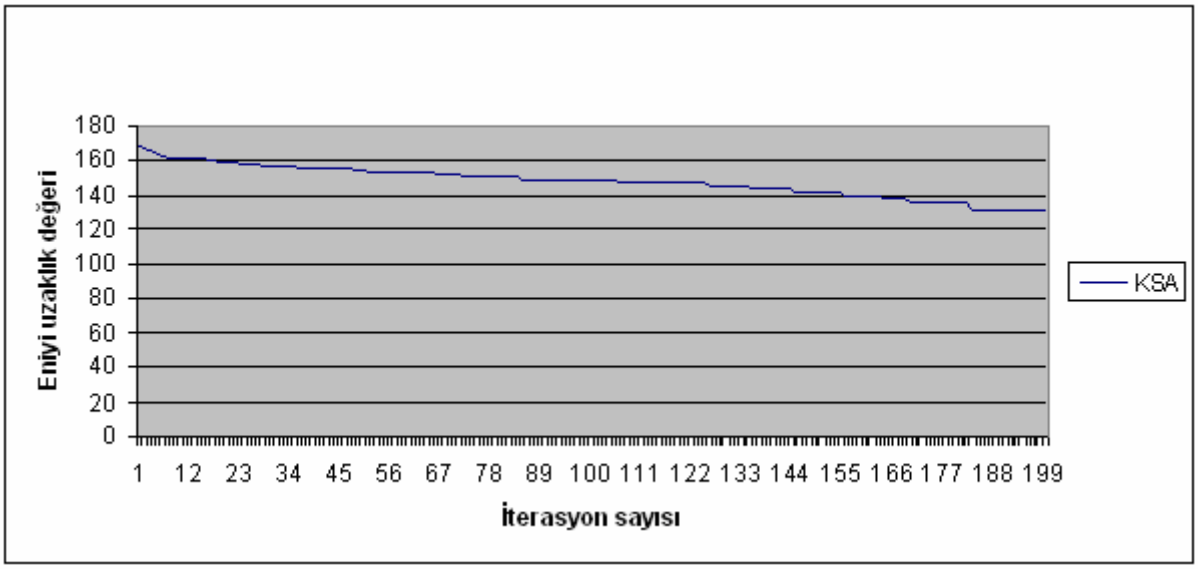


Şekil 5.4 Melez Yöntemin 11eil51 problemi için yakınsama grafiği

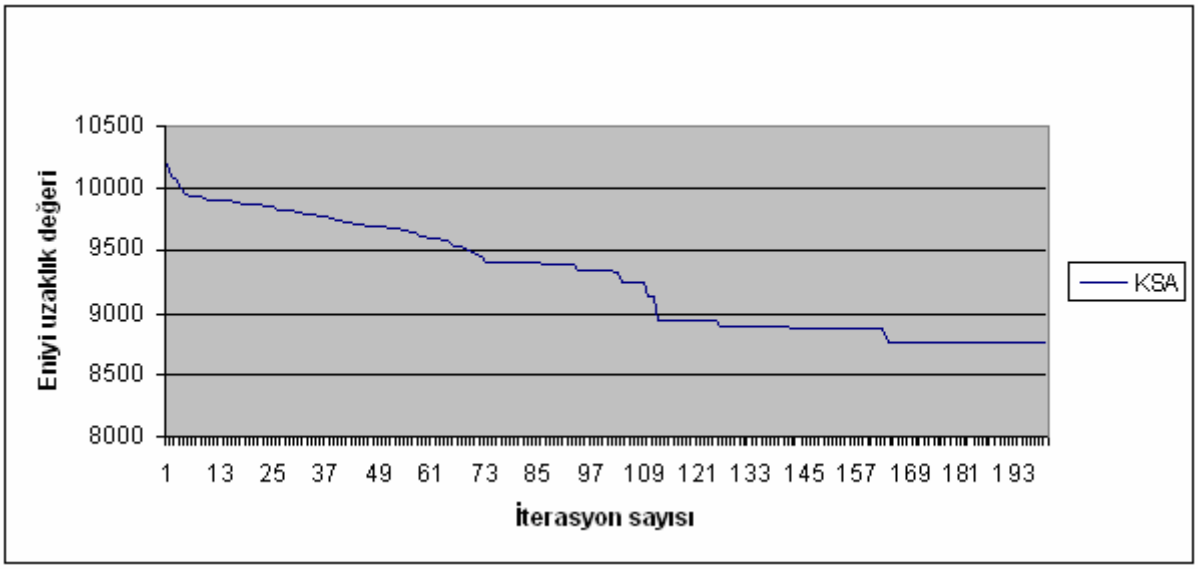


Şekil 5.5 Melez Yöntemin 20kroa100 problemi için yakınsama grafiği

Aynı problemler için KSA'nın yakınsama grafikleri Şekil 5.6 ve Şekil 5.7'de verilmektedir.



Şekil 5.6 KSA'nın 11eil51 problemi için yakınsama grafiği



Şekil 5.7 KSA'nın 20kroa100 problemi için yakınsama grafiği

Bu çalışmada diğer iki algoritmaya üstünlük sağlayan melez algoritma, literatürde yer alan ve eniyi çözümü bilinmeyen 40d198, 41gr202, 45ts225 ve 46pr226 problemleri üzerinde denenmiştir.

Bu test problemleri için literatürde geliştirilmiş sezgisel yöntemlerin çözüm değerleri ile bu çalışmada geliştirilen melez yöntemin bulduğu çözüm değerleri EK-2'de verilmiştir. Literatürdeki yöntemler farklı bilgisayarlarda farklı programlama dilleriyle kodlandığından, çözüm zamanı karşılaştırması yapılmamıştır.

Bu değerlere göre geliştirilen melez yöntem, 41gr202 probleminde literatürde daha önce bulunan çözüm değerlerinden daha iyi bir çözüm bulduğu görülmektedir. Daha önce bilinen eniyi çözüm değerinden yaklaşık %0.008 oranında daha iyi bir çözüm elde edilmiştir.

40d198 probleminde geliştirilen melez yöntem Ghosh'un [10] geliştirdiği algoritmalardan, Hu, Leither ve Raidl'in [15] geliştirdiği SA, VNDS, TS2 algoritmalarından ve Feremans'ın [9] geliştirdiği UB algoritmasından daha iyi çözüm değeri bulunmuştur. Ancak daha önce bilinen eniyi çözüm değerinden yaklaşık %0.0012 oranında daha kötü bir çözüm elde edilmiştir.

45ts225 ve 46pr 226 problemlerinde ise Ghosh'un [10] ve Feremans'ın [9] bulduğu çözümlerden daha iyi, ancak bilinen eniyi çözüm değerinden yaklaşık %0.0004 oranında daha kötü bir çözüm elde edilmiştir.

6. SONUÇ

Bu çalışmada, GYP için üç yeni sezgisel algoritma geliştirilmiştir. Bunlar, Genetik Algoritma, Genetik Algoritma ve Tavlama Benzetiminden oluşan bir melez yöntem ile GYP için ilk kez kullanılan Kuş Sürüsü Algoritmalarıdır.

GYP, problem boyutuna bağlı olarak arama uzayındaki nokta sayısı üstel artış gösterdiğinden NP-zor bir problemdir. Bu nedenle büyük boyutlu GYP'lerde klasik optimizasyon yöntemleri yetersiz kalmaktadır. Literatürde bu problem için önerilmiş sezgisel yöntemler bulunmaktadır.

Biyolojik sistemlerin doğal evrim mekanizmasını taklit ederek benzetimini yapan stokastik bir arama yöntem olan GA temel özellikleri yanı sıra probleme özgü bilgilerle geliştirilen GA'nın TSPLIB'den seçilen test problemlerinde ortalama %2.31 hata ile eniyi çözüm değerlerine yaklaşılabilmiştir. Golden, Raghavan ve Stanojevic'in [12] belirttiğine göre, Feremans'ın [9] çalışmasında yer alan GA ortalama %6.53 oranındaki sapma ile eniyi çözüm değerine yaklaşmaktadır [12]. Geliştirilen GA'nın performansının daha da iyileştirilmesi için GA ile bir yerel arama yönteminin birlikte kullanılmasıyla düşünülmüş ve yerel arama yöntemlerinden Tavlama Benzetimi seçilerek bir melez yöntem geliştirilmiştir.

GA ile TB, iki farklı şekilde birlikte kullanılarak melez yapı oluşturabilmektedir. Bunlardan ilki, GA'da mevcut yığından bir sonraki yığına kopyalanacak dizilere TB uygulanması, diğeri ise GA aramasını tamamladıktan sonra TB kullanılmasıdır. Bu çalışmada, ikinci yol seçilmiştir. Aynı test problemleri üzerinde denenen melez yöntem, ortalama %0.34 sapma ile eniyi çözümlere yaklaşmıştır.

GYP üzerinde ilk kez bu çalışmada kullanılan KSA, test problemlerinde ortalama %2.23 sapma ile bu çalışmada geliştirilen GA'dan yaklaşık %0.08 oranında daha iyi sonuçlar bulmuştur. Ancak bu yöntem, geliştirilen melez yöntem ile karşılaştırıldığında ortalama %1.89 oranında daha kötü çözümler ürettiği görülmektedir. Bu çalışmada geliştirilen üç algoritmadan en iyi çözüm değerlerini bulan melez yöntem eniyi çözümü bilinmeyen problemlerde de test edilmiştir.

İleride Yapılabilecek Araştırmalar

Bu çalışmada GYP üzerinde ilk kez denenen KSA'nın performansı GA'dan daha iyi olmasına rağmen, geliştirilen melez yöntemden daha iyi sonuçlar bulamamıştır. KSA'da, GA'ya benzer olarak mevcut çözümlerin bir kümesi ile aramaya devam ettiğinden, geliştirilen melez yöntemde olduğu gibi, bir yerel arama yöntemiyle birlikte kullanılarak performansı daha da artırılabilir. KSA'nın öncesinde bir yerel arama yöntemi kullanılarak, birbirinden farklı olmak koşuluyla başlangıç çözümler elde edilebilir ve bu çözümler KSA'nın başlangıç sürüsü olarak kullanılabilir. KSA'nın hemen ardından bir yerel arama yöntemi de kullanılabilir. Bunlar ileride yapılacak çalışmalarda araştırılabilecek konulardır.

KAYNAKLAR LİSTESİ

[1] Abido M.A., Optimal power flow using particle swarm optimization, Computers and Operations Research, vol.24, pp.563-571, 2002.

[2] Alabaş Ç., Tabu Arama ve Tavlama Benzetimi Algoritmalarıyla Bilgisayar Şebekelerinin Topolojik Optimizasyonu, Yüksek Lisans Tezi, Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara, 1999.

[3] Altıparmak F., Genetik Algoritma ile haberleşme şebekelerinin topolojik optimizasyonu, Doktora Tezi, Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara, 1996.

[4] Back, Thomas, Evolutionary Algorithms in theory and practice, Oxford University Press, 1996.

[5] Ball, M.O., Magnanti T.L., Monma C.L., Nemhauser G.L., Handbooks in Operations Research and Management Science, Elsevier, 1995.

[6] Bouhmala N., Combining local serach and genetic algorithms with the multilevel paradigm for the travelling salesman problem, first international workshop on hybrid metaheuristics, Velancia-Spain, pp.51-58, 2004.

[7] Dror M., Haouari M., Chauoachi J., Generalized spanning trees, European Journal of Operations Research 120, pp.583-592, 2000.

[8] Feremans C., Labbe M., Laporte G., A comperative analysis of several formulations for the generalized minimum spanning tree problem, Networks 39 pp.29-34, 2002.

[9] Feremans C., Generalized Spanning Trees and Extensions, Ph.D. thesis, Institute de Statistique et de Recherche Operationnelle Universite Libre de Bruxelles, Bruxelles, Belgium, 2001.

[10] Ghosh D., Solving Medium to Large Sized Euclidean Generalized Minimum Spanning Tree Problems, Working paper series, IIM Ahmedabad, India, 2003. E

[11] Goldberg D.E., Genetic Algorithms, Addison Wesley Longman, 1989.

[12] Golden B., Raghavan S., Stanojevic D., Heuristic Search for the Generalized Minimum Spanning Tree Problem, Informs Journal on Computing Vol.17, No.3, pp.290-304, 2005.

[13] Grimaldi R.P., Discrete and Combinatorial Mathematics, 4^{th} edition, Addison-Wesley Longman, 1999.

[14] Harris J.M., Hirst J.L., Mossinghoff M.J., Combinatorics And Graph Theory, Springer-Verlag, 2000.

[15] Hu B., Leitner M., Raidl R.G., Combining Variable Neighborhood Search with Integer Programming for the Generalized Minimum Spanning Tree Problem, Institute of Computer Graphics and Algorithms, Vienna University of Technology, Vienna, Austria, 2006.

[16] Jiang M., Luo Y.P., Yang S.Y., Stochastic convergency analysis and parameter selection of the standard particle swarm optimization algorithm, Information Processing Letters 102, pp.8-16, 2007.

[17] Kara İ., Ünlü Y., Çakıcı E., Bektaş T., Genelleştirilmiş Yayılma Problemi İçin Yeni Bir Tamsayılı Karar Modeli, YA'EM 2004, 24. Ulusal YA'EM Kongresi, Çukurova Üniversitesi, Adana-Türkiye, s.49-51, 2004.

[18] Kennedy J., Eberhart R.C., Swarm Intelligence, Morgan Kaufmann, 2001.

[19] Kennedy J., Eberhart R., Particle Swarm Optimization, Proceedings of the IEEE International Conference on Neural Networks, Perth-Australia, pp.1942-1945, 1995.

[20] Konak A., Smith A.E., A Hybrid Genetic Algorithm Approach for Backbone Design of Communication Networks, IEEE 0-7803-5536-9, 1999.

[21] Myung, Y-S., Lee, C-H., Tcha, D-W., On the generalized minimum spanning tree problem, Networks 26, pp.231-241, 1995.

[22] Pop P.C., Kern W., Still G.J., The generalized minimum spanning tree problem, Technical report, University of Twente, Twente, The Netherlands, 2000.

[23] Raghavan S., On modelling the generalized minimum spanning tree, Technical Report, The Robert H. Smith School of Business, University of Maryland, College Park, MD, 2002.

[24] Reeves C.R., Modern Heuristic Techniques For Combinatorial Problems, McGraw-Hill, 1995.

[25] Rosen R. H., Discrete Mathematics and Its Applications, pp. 593, 3^{rd} edition, Mc Graw Hill, 1995.

[26] Salman A., Ahmad I., Al-Madani S., Particle swarm optimization for task assignment problem, Microprocessors and Microsystems, vol.26, pp.363-371, 2003.

[27] Taha, H.A., Operations Research, 6^{th} edition, Prentice Hall, 1997.

[28] Tasgetiren M.F., Sevkli M., Liang Yun-Chia, Gencyilmaz G., Particle Swarm Optimization Algorithm for Single Machine Total Weighted Tardiness Problem, IEEE 0-7803- 8515-2/04, 2004.

[29] Tasgetiren M.F., Liang Y.C., A Binary Particle Swarm Optimization Algorithm for Lot Sizing Problem, Journal of Economics and Social Research 5(2), pp.1-20.

[30] Yoshida H., Kawata K., Fukuyama Y., Yakanishi Y., A particle swarm optimization for reactive power and voltage control considering voltage security assessment, IEEE Transactions on Power Systems, vol.15, pp.1232-1239, 2000.

[31] Zaim A.H, Reliable Network Topology Design, Bogazici University, 1996.

Çizelge E1 : Algoritma Performanslarının Karşılaştırma Çizelgesi

Problem	Eniyi Çözüm Değeri	GA			MY			KSA		
		Yaklasık Eniyi	Sapma (%)	CPU (sn)	Yaklasık Eniyi	Sapma (%)	CPU (sn)	Yaklasık Eniyi	Sapma (%)	CPU (sn)
10att48	10923	10923	0,00	311	10923	0,00	362	10923	0,00	279
10hk48	4119	4119	0,00	306	4119	0,00	341	4119	0,00	266
11eil51	132	132	0,00	318	132	0,00	365	132	0,00	294
14st70	233	233	0,00	434	233	0,00	489	233	0,00	398
16pr76	46514	46514	0,00	591	46514	0,00	621	46514	0,00	546
16eil76	186	186	0,00	566	186	0,00	603	186	0,00	553
20gr96	221	225	1,81	768	221	0,00	863	224	1,36	674
20rat99	402	416	3,48	761	402	0,00	809	412	2,49	713
20kroa100	7982	8536	6,94	756	8119	1,72	835	8463	6,03	740
21eil101	204	218	6,86	779	204	0,00	826	219	7,35	766
21lin105	6728	6859	1,95	774	6728	0,00	819	6992	3,92	754
22pr107	20398	21057	3,23	785	20415	0,08	840	21156	3,72	759
24gr120	2255	2284	1,29	805	2255	0,00	863	2280	1,11	776
25pr124	30174	31140	3,20	821	30590	1,38	874	31052	2,91	788
26bier127	58150	61562	5,87	834	59310	1,99	901	60801	4,56	796

EK-1 Geliştirilen Algoritmaların Performanslarının Karşılaştırması

Çizelge E2 : Geliştirilen Melez Yöntemin Performansının Literatürdeki Algoritmaların Performansları İle Karşılaştırması

	Golden, Raghavan, Stanojevic (2005)		Ghosh (2003)						Hu,Leither,Raidl (2006)					Feremans (2001)		
Problem	LS	GA	TS1	TS2	VND	RVNS	VNS	VNDS	TS2	VNDS	SA	GA	VNS	UB	Bu çalışmada geliştirilen MY	Çözüm Zamanı (CPU)
40d198	7044	7044	7070	7063	7151	7151	7313	7185	7062	7169	7468	7044	7044	7232	7053	1442
41gr202	244	243	242	242	250	250	250	250	242	249	258	243	242	250	240	1604
45ts225	62400	62315	63444	62369	62656	62656	62656	63444	62366	63139	67195	62315	62268	62506	62315	1898
46pr226	55515	55515	55518	55518	55518	55518	56424	55636	55515	55515	56286	55515	55515	55971	55860	1963

EK-2 Eniyi Çözümü Bilinmeyen Test Problemleri İçin Geliştirilen Melez Yöntemin Literatürde Yer Alan Algoritmalarla Karşılaştırması