T.C.
BALIKESİR ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

GÖÇMEN KUŞLAR OPTİMİZASYON ALGORİTMASININ PARALEL BİLGİSAYARLARDA UYGULANMASI

YÜKSEK LİSANS TEZİ

ABDULLAH TÜLEK

BALIKESİR, HAZİRAN - 2019

T.C.
BALIKESİR ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

GÖÇMEN KUŞLAR OPTİMİZASYON ALGORİTMASININ PARALEL BİLGİSAYARLARDA UYGULANMASI

YÜKSEK LİSANS TEZİ

ABDULLAH TÜLEK

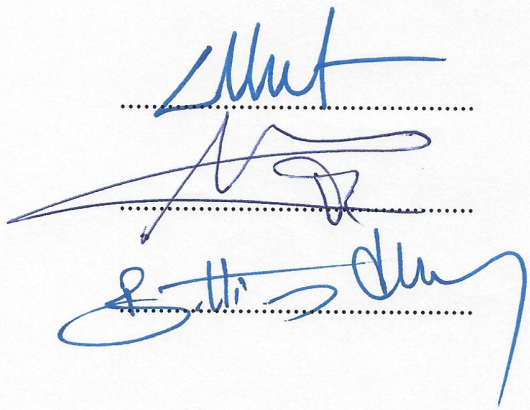
Jüri Üyeleri : Dr. Öğr. Üyesi Gültekin KUVAT (Tez Danışmanı)
Prof. Dr. Metin DEMİRTAŞ
Dr. Öğr. Üyesi Burhanettin DURMUŞ

BALIKESİR, HAZİRAN - 2019

KABUL VE ONAY SAYFASI

Abdullah TÜLEK tarafından hazırlanan "**GÖÇMEN KUŞLAR OPTİMİZASYON ALGORİTMASININ PARALEL BİLGİSAYARLARDA UYGULANMASI**" adlı tez çalışmasının savunma sınavı 14.06.2019 tarihinde yapılmış olup aşağıda verilen jüri tarafından oy birliği ile Balıkesir Üniversitesi Fen Bilimleri Enstitüsü Elektrik-Elektronik Mühendisliği Anabilim Dalı Yüksek Lisans Tezi olarak kabul edilmiştir.

Jüri Üyeleri	İmza
Danışman Dr. Öğr. Üyesi Gültekin KUVAT	.....
Üye Prof. Dr. Metin DEMİRTAŞ	.....
Üye Dr. Öğr. Üyesi Burhanettin DURMUŞ	.....



Jüri üyeleri tarafından kabul edilmiş olan bu tez Balıkesir Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulunca onanmıştır.

Fen Bilimleri Enstitüsü Müdürü

Prof. Dr. Necati ÖZDEMİR .....

ÖZET

GÖÇMEN KUŞLAR OPTİMİZASYON ALGORİTMASININ PARALEL BİLGİSAYARLARDA UYGULANMASI
YÜKSEK LİSANS TEZİ
ABDULLAH TÜLEK
BALIKESİR ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI
(TEZ DANIŞMANI: DR. ÖĞR. ÜYESİ GÜLTEKİN KUVAT)

BALIKESİR, HAZİRAN - 2019

Bu tez çalışmasında, metasezgisel optimizasyon algoritmalarından biri olan Göçmen Kuşlar Optimizasyon (GKO) algoritması paralel bilgisayarlarda uygulanarak Paralel Göçmen Kuşlar Optimizasyon (PGKO) algoritması geliştirilmiştir. Uygulama, TÜBİTAK ULAKBİM tarafından araştırmacıların kullanımına sunulan yüksek başarımlı hesaplama merkezi TRUBA altyapısı üzerinde, C++ dili ile Open MPI kütüphanesi kullanılarak geliştirilmiştir. GKO ve PGKO algoritmaları 50 boyutlu altı test fonksiyonunun çözümü için dört farklı alt popülasyon sayısı kullanılarak 30 defa bağımsız olarak çalıştırılmış ve elde edilen ortalama, en iyi ve en kötü sonuçlar verilmiştir. PGKO algoritması, beş farklı göç oranı ve aralığı için uygulanmış, sonuçlar GKO sonuçları ile karşılaştırılarak göç işleminin algoritmaya etkisi gösterilmiştir. Ayrıca, göç işleminde kullanılan farklı parametre değerlerinin PGKO algoritmasına etkisi incelenerek başarılı sonuçların üretildiği durumlar ortaya konmuştur. Bunun yanında, her iki algoritmadan elde edilen sonuçlara t-testi uygulanmış ve göç işleminin anlamlı bir fark oluşturduğu gösterilmiştir.

ANAHTAR KELİMELER: paralel göçmen kuşlar optimizasyon algoritması , göç oranı , göç aralığı , paralel optimizasyon

ABSTRACT

IMPLEMENTATION OF MIGRATING BIRDS OPTIMIZATION ALGORITHM ON PARALLEL COMPUTERS

MSC THESIS

ABDULLAH TÜLEK

BALIKESIR UNIVERSITY INSTITUTE OF SCIENCE

ELECTRICAL AND ELECTRONICS ENGINEERING

(SUPERVISOR: ASSIST. PROF. DR. GÜLTEKİN KUVAT)

BALIKESİR, JUNE 2019

In this thesis, Migrating Birds Optimization (MBO) algorithm which is one of the metaheuristic optimization algorithms is applied on parallel computers and Parallel Migrating Birds Optimization (PMBO) algorithm has been developed. The application is developed by using the Open MPI library with C++ language on the TRUBA infrastructure, which is provided by TÜBİTAK ULAKBİM. Both MBO and PMBO algorithms are run 30 times independently with using four different subpopulation numbers for the solution of six 50-dimensional test functions and the obtained average, best and worst results are given. PMBO algorithm is applied for five different migration rates and intervals, and the results are compared with MBO results and the effect of migration process on the algorithm is shown. In addition, the effect of different parameter values used in migration process on the PMBO algorithm is analyzed and the cases where generate successful results are presented. Furthermore, t-test is applied to the results acquired from both algorithms and it is proven that migration process makes the significant difference.

KEYWORDS: parallel migrating birds optimization algorithm , migration rate , migration interval , parallel optimization

İÇİNDEKİLER

Sayfa

ŞEKİL LİSTESİ

Sayfa

TABLO LİSTESİ

Sayfa

KISALTMALAR LİSTESİ

YAK : Yapay Arı Kolonisi
PSO : Parçacık Sürü Optimizasyonu
GA : Genetik Algoritma
KK : Karınca Kolonisi
DG : Diferansiyel Gelişim
KO : Kaos Optimizasyonu
BAÇO : Bal Arısı Çiftleşme Optimizasyonu
MPI : Message Passing Interface
TRUBA : Türkiye Ulusal Bilim e-Altyapısı
GKO : Göçmen Kuşlar Optimizasyonu
PGKO : Paralel Göçmen Kuşlar Optimizasyonu
KAP : Karesel Atama Problemi
GSP : Gezgin Satıcı Problemi
MSE : Mean Squared Error (Ortalama Karesel Hata)
CA : Certificate Authority
IP : Internet Protocol
SSH : Secure Shell
VPN : Virtual Private Network

ÖNSÖZ

Yüksek lisans eğitimime başladığım günden beri ders aşamasında ve tez çalışmamda beni cesaretlendiren, yönlendiren ve tez yazım aşamasında değerli bilgi birikimini ve zamanını benimle paylaşan çok değerli hocam ve danışmanım sayın Dr. Öğr. Üyesi Gültekin KUVAT'a,

İstatistiksel konularda bana destek olan kıymetli hocam Dr. Öğr. Üyesi Özlem KUVAT'a

Uygulama geliştirme aşamasında bana desteğini esirgemeyen çok kıymetli meslektaşım Kürşat TÜRKAY'a,

Çalışmalarım sırasında bana destek olan, beni cesaretlendiren sevgili eşim Özlem TÜLEK'e,

Saygı, sevgi ve teşekkürlerimi sunarım.

Bu araştırmada yer alan tüm nümerik hesaplamalar TÜBİTAK ULAKBİM, Yüksek Başarım ve Grid Hesaplama Merkezi'nde (TRUBA kaynaklarında) gerçekleştirilmiştir. Teşekkür ederiz.

Abdullah TÜLEK

1. GİRİŞ

Optimum, en iyi, en ideal anlamına gelmekte olup bir optimizasyon probleminde amaç en iyi sonucu elde etmektir. En iyi sonuç, matematiksel bir fonksiyonun minimum ya da maksimum noktası olabilir. Son zamanlarda meydana gelen teknolojik gelişmeler, çok bilinmeyenli tasarım ve üretim problemlerinde karmaşıklığın artması, problem boyutlarının büyümesi, hesaplama gücüne olan ihtiyacı hızla arttırmaktadır. Problemlerin çözümü için geliştirilmiş olan klasik yöntemler, önemli ölçüde kaynak ve zaman israfına neden olmakta ve matematiksel olarak modellenemeyen problemlerin çözümüne katkı sunamamaktadır. Bu türden çok boyutlu ve çözümü zor problemler, araştırmacıları yeni çözüm yöntemleri üretmeye yöneltmektedir. Günümüzde en yaygın olarak kullanılan yöntemlerden olan metasezgisel algoritmalar, genellikle doğadan esinlenen, rastlantısal ve en iyi çözümü arayan yöntemlerdir [1,2].

Genellikle doğal olaylardan, canlıların hareketinden veya sürü davranışı modellerinden yola çıkılarak geliştirilmiş olan metasezgisel algoritmalar, arama uzayı içinde en iyi çözümleri ararlar. Bir popülasyon üzerinde arama yapan metasezgisel algoritmalar, problemler zorlaştığında veya problem boyutu büyüdüğünde yetersiz kalabilmekte, yerel en iyilere takılarak başarısız sonuçlar üretebilmektedir. Birbirinden bağımsız olarak üretilen başlangıç popülasyonlarında arama yapabilme yeteneği, göç işlemi kullanılarak iyi çözümlerin paylaşılabilmesi ve bunun sonucu olarak algoritma performansında elde edilen iyileşmeler, çok sayıda alt popülasyon kullanılarak yapılan çalışmaların hızla artmasını sağlamaktadır. Örneğin yapay arı kolonisi (YAK) [3,4], parçacık sürü optimizasyonu (PSO) [5,6], genetik algoritmalar (GA) [7-9], karınca kolonisi (KK) [10], diferansiyel gelişim (DG) [11], kaos optimizasyonu (KO) [12] gibi farklı algoritmalar ile yapılan paralel optimizasyon çalışmaları mevcuttur.

Paralel metasezgisel algoritmalar ile yapılan çalışmalar, tek popülasyonda yapılan çözümlere göre genellikle daha iyi sonuçlar vermektedir [13]. Bunun nedeni

alt popülasyonlar arasında belli aralıklar ile belli oranda çözümün paylaşımını sağlayan göç işlemidir. Göç aralığı, göç oranı, gönderilecek çözümlerin seçimi, gelen çözümlerin alt popülasyona yerleştirilmesi ve haberleşme modeli göç parametrelerini oluşturur [14]. Bu parametreler üzerinde yapılan değişiklikler algoritma performansını etkiler.

Göç işleminin algoritma performansına katkısı iki şekilde açıklanmaktadır;

- Göç işlemi sayesinde alt popülasyonlar içindeki iyi çözümlerin, komşu alt popülasyonlar ile paylaşılması genel bir iyileşme sağlamaktadır,
- Alt popülasyon içerisine yeni katılan çözümler, metasezgisel algoritmaların sık görülen problemlerinden biri olan yerel optimumlara takılma riskini azaltmaktadır [15].

Bu çalışmada, Message Passing Interface (MPI) protokolü kullanılarak alt popülasyonlar arasında iyi çözümlerin göç etmesi sağlanmıştır. İyi çözümlerin, belirlenen oran ve aralıklarda komşu alt popülasyona gönderilmesi algoritma performansının iyileşmesini sağlamıştır. Göç parametreleri için seçilen farklı değerlerin, elde edilen sonuçları etkilediği gösterilmiştir.

MPI kullanılarak kümeli bilgisayar yapısı üzerinde alt popülasyonlar oluşturarak çözümün arandığı çeşitli çalışmalar mevcuttur [3,13,16]. Aynı şekilde, bu çalışmada yapılan hesaplamalar, TÜBİTAK ULAKBİM tarafından araştırmacıların hizmetine sunulan, yüksek başarımlı ve grid hesaplama merkezi TRUBA altyapısı üzerinde Open MPI kütüphanesi kullanılarak gerçekleştirilmiştir.

Bu tez çalışmasında, Göçmen Kuşlar Optimizasyon (GKO) algoritması paralel bilgisayarlarda uygulanarak Paralel Göçmen Kuşlar Optimizasyon (PGKO) algoritması geliştirilmiştir. Farklı alt popülasyon sayıları, göç oranı ve göç aralığı değerleri kullanılarak PGKO algoritması ile altı farklı test fonksiyonu çözülmüş ve en uygun göç parametrelerinin tespiti üzerinde çalışılmıştır. Her bir test fonksiyonu için, en başarılı sonuçların üretildiği göç parametrelerinden elde edilen PGKO ve GKO sonuçları kullanılarak t-testi uygulanmış, anlamlılık düzeyinde sonuçlar alınmış ve göç işleminin algoritma performansına katkısı gösterilmiştir.

Bu çalışmanın 2. bölümünde GKO ve PGKO algoritmaları açıklanmış, 3. bölümde uygulama platformu ve araçları ile test fonksiyonları tanıtılmış, 4. bölümde uygulama ve t-testi sonuçları sunulmuş ve 5. bölümde de sonuç ve öneriler verilmiştir.

2. PARALEL GÖÇMEN KUŞLAR OPTİMİZASYON ALGORİTMASI

Bu bölümde GKO ve PGKO algoritmaları açıklanmış, bu algoritmalara ait sözde kodlar, akış diyagramları ve ilgili açıklamalar verilmiştir. Bunun yanında PGKO algoritmasının yapısı, çalışma prensibi ve parametreleri sunulmuştur.

2.1 Göçmen Kuşlar Optimizasyon Algoritması

Metasezgisel algoritmalar genellikle doğadan esinlenilerek geliştirilmiş algoritmalardır. GKO algoritması da Λ şeklinde uçan göçmen kuşların davranışları modellenerek geliştirilmiştir [17]. Yapılan bir araştırmaya göre göçmen kuşların Λ şeklinde uçuşu onların maruz kaldığı hava sürtünmesini azaltarak daha uzun mesafelere, daha az enerji harcayarak uçmalarına olanak vermektedir. Bir kuşun kanat çırpması sonrası, kuşun sağ ve sol kanatları arkasında yukarı doğru hava akımı meydana gelmekte ve öndeki kuşun kanatları arkasında bulunan kuşlar bu hava akımından faydalanarak daha az enerji harcayarak daha uzun mesafelere uçabilmektedir. Bu araştırmaya göre 25 kuştan oluşan bir sürü, Λ şeklinde uçarak menzilini %70'e kadar artırabilmektedir [18]. Bu davranıştan yola çıkılarak geliştirilen GKO algoritmasında kullanılan parametreler aşağıda verilen Tablo 2.1'de [1,17] gösterilmiştir.

Tablo 2.1: GKO algoritmasında kullanılan parametreler.

Parametre	Açıklama
n	Popülasyondaki birey (kuş) sayısı
k	Her bir birey için üretilecek olan komşuluk sayısı
x	Bir sonraki birey (kuş) ile paylaşılacak olan çözüm sayısı
m	Lider kuş değişim aralığı
K	Maksimum iterasyon sayısı

GKO algoritması komşu arama tekniği kullanan bir algoritmadır. Algoritma şu şekilde çalışmaktadır:

Sürüdeki her bir kuş, optimizasyon problemindeki bir çözüme karşılık gelmektedir. Başlangıçta n adet kuş için rastgele pozisyon bilgisi üretilir. Bu işlemde denklem (2.1) [1,19] kullanılarak, problemin tanımlı olduğu aralıkta düzgün dağılımlı rastgele çözümler elde edilir.

$$x_{ij} = x_j^{min} + rand.\left(x_j^{max} - x_j^{min}\right) \tag{2.1}$$

Denklemde i, [1, n] aralığında bir tamsayı (n sürüdeki kuş sayısı), j, [1, d] aralığında bir tam sayı (d problemin boyutu), x_{ij}, i. kuşun j. boyutundaki çözümü, x_j^{min} ve x_j^{max}, problemin j boyutunda alabileceği en küçük ve en büyük değerler ve $rand$, [0, 1] aralığında rasgele bir sayıdır.

Sürüdeki kuşlardan biri lider kuş olarak kabul edilir ve diğer kuşların yarısı sağ, diğer yarısı da sol tarafa geldiği varsayılarak sanal Λ şekli elde edilir. Üretilen her bir çözüm için denklem (2.2) [1,19] kullanılarak, lider kuş için k adet komşuluk üretilir.

$$\hat{x}_{ij} = x_{ij} + \emptyset.\left(x_{ij} - x_{kj}\right) \tag{2.2}$$

Denklemde $\hat{x}_{ij}$, i. çözümün j. boyutundaki çözümünden üretilecek olan yeni komşuluk, x_{ij}, i. çözümün j. boyutundaki çözümü, $\emptyset$, [-1, 1] aralığında rasgele bir sayı ve x_{kj}, k. kuşun j. boyutundaki çözümüdür (k rasgele seçilmiş ve i'den farklı bir indeks numarası). Eğer üretilen komşuluk çözümü problemin j boyutunda alabileceği en küçük ve en büyük değerleri aşarsa, $\hat{x}_{ij}$ değeri denklem (2.3) [1,19] kullanılarak belirtilen limitlerde tutulur.

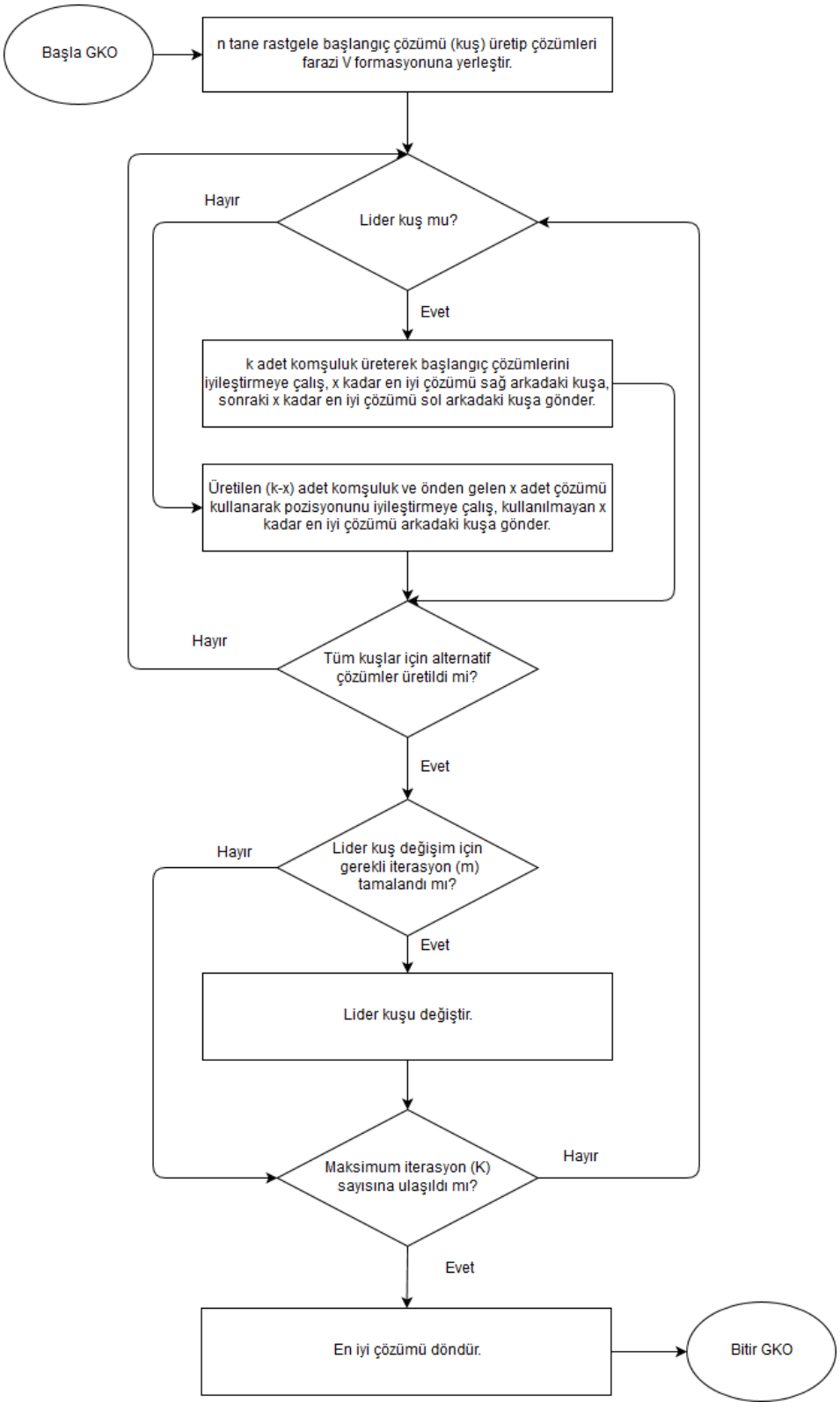
$$\hat{x}_{ij} = \begin{cases} x_j^{min}, & \hat{x}_{ij} \le x_j^{min} \\ \hat{x}_{ij}, & x_j^{min} < \hat{x}_{ij} < x_j^{max} \\ x_j^{max}, & \hat{x}_{ij} \ge x_j^{max} \end{cases} \tag{2.3}$$

İlk üretilen çözümler ile sonradan üretilen komşulukların uygunluk değerleri hesaplanır. Uygunluk değeri en iyi olan çözüm lider kuşta bırakılır. Sonraki en iyi 2x

çözümden x adeti sağ arkadaki kuşa, x adeti de sol arkadaki kuşa aktarılır. Diğer kuşlar içinse öndeki kuştan gelen x adet çözüm, üretilen (k-x) adet çözüm ile birleştirilerek çözümlerin uygunluk değerleri hesaplanır ve en iyi çözüm ilgili kuşta bırakılarak sonraki en iyi x adet çözüm bir sonraki kuşa gönderilir. Üretilen komşu çözümlerin pozisyon bilgileri, problem uzayının ilgili boyutu için belirlenen sınırların dışına taşarsa, denklem (2.3) kullanılarak çözümler belirlenen sınırlara çekilir. Sürüdeki tüm kuşlar için alternatif çözüm üretme işlemi tamamlandığında algoritmanın bir iterasyonu tamamlanmış olur. Önceden belirlenmiş olan m parametresi kadar iterasyon tamamlandığında lider kuşun yorulduğu varsayılarak lider kuşun değişimi gerçekleştirilir. Lider kuş değişiminde, yorulduğu varsayılan lider kuş önce Λ şeklinde yerleştiği varsayılan sürünün sağ en arkasına gönderilerek sağ koldaki kuşlar birer sıra ilerleyerek yeni lider kuş belirlenir. Bir sonraki kuş değişim aralığı geldiğinde ise Λ şeklinde yerleştiği varsayılan sürünün sol kolunda aynı işlem gerçekleştirilir. Maksimum iterasyon sayısına (K) ulaşıldığında veya durma kriteri sağlandığında algoritma sonlandırılır ve mevcut çözümler içerisinde, en iyi uygunluk değerine sahip çözüm, problemin sonucu olarak verilir. GKO algoritması sözde kodu Şekil 2.1'de, akış diyagramı Şekil 2.2'de verilmiştir.

```
n tane rastgele başlangıç çözümü (kuş) üretip çözümleri farazi V formasyonuna yerleştir.
M=lider kuş değişim aralığı, K=maksimum iterasyon sayısı.
for (i=0;i<K;i++)
    for (j=0;j<m;j++)
        for each kuş in sürü
            if lider==true
                k adet komşuluk üreterek başlangıç çözümlerini iyileştirmeye çalış,
                x kadar en iyi çözümü sağ arkadaki kuşa, x kadar en iyi çözümü sol arkadaki kuşa gönder.
            else
                üretilen (k-x) kadar komşuluk ve önden gelen x kadar çözümü kullanarak pozisyonunu iyileştirmeye çalış,
                kullanılmayan x kadar en iyi çözümü arkadaki kuşa gönder.
            endif
        endfor
    endfor
    lider çözümü (kuşu) değiştir.
endfor
return sürüdeki en iyi çözüm.
```

Şekil 2.1: GKO algoritması sözde kodu.



Şekil 2.2: GKO algoritması akış diyagramı.

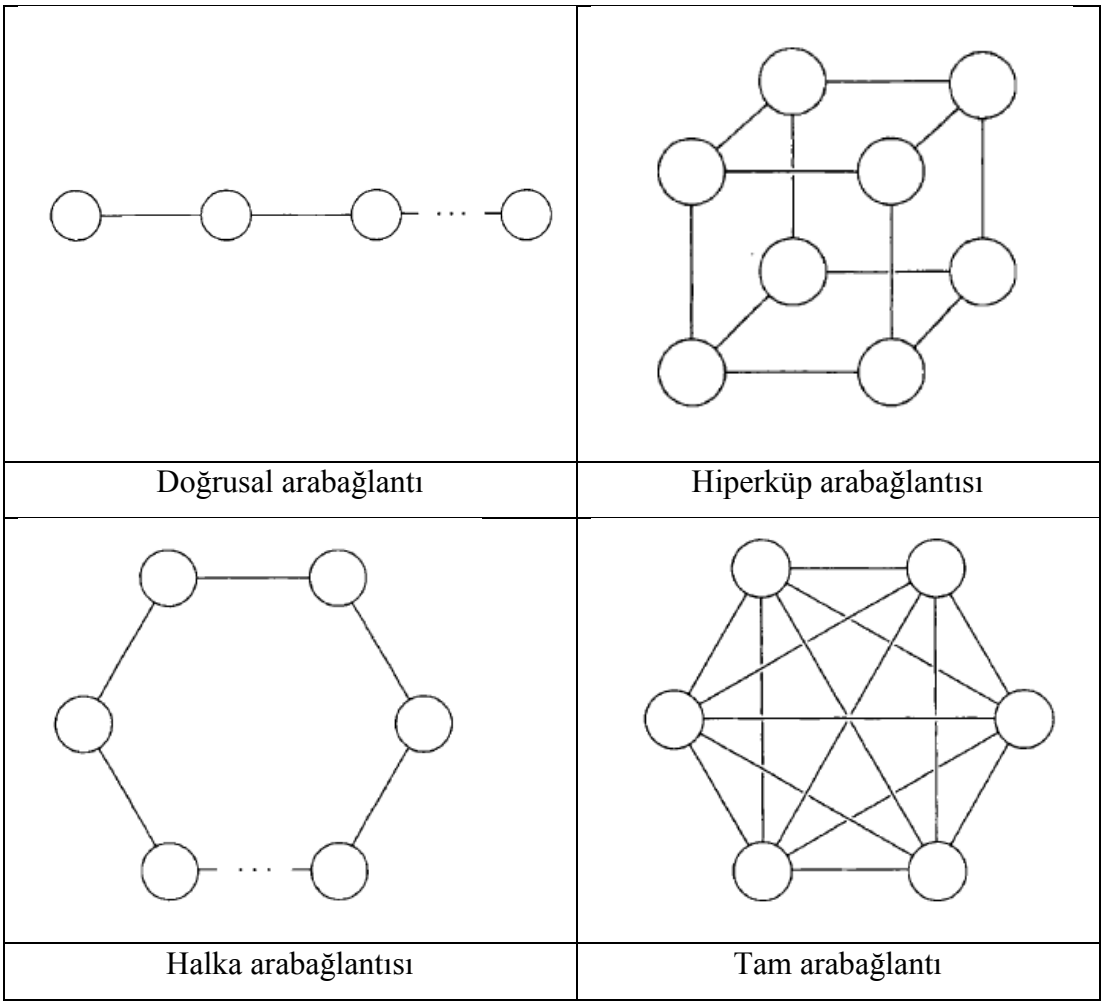
GKO algoritması, birçok araştırmacı tarafından farklı çalışmalarda kullanılmaktadır. Örneğin [20]'de yapılan çalışmaya göre rastlantısal veriye ihtiyaç duyan başlangıç çözümlerinin oluşturulmasında ve komşuluk üretme adımlarında çeşitli kaotik eşlem fonksiyonları kullanılmış ve algoritmanın yakınsama performansına olan etkileri gösterilmiştir. Yapılan diğer bir çalışmada [21] tek boyutlu kesme problemi GKO algoritması ile çözülmüş, komşuluk yapısı olarak araya ekleme metodu kullanılmış ve daha iyi sonuçlar verdiği ifade edilmiştir. [22]'de ise GKO'nun karesel atama problemlerinde (KAP) başarılı sonuçlar üretirken gezgin satıcı probleminde (GSP) aynı performansı gösteremediği ortaya konmuştur. Daha iyi sonuçlar elde etmek için yedi farklı komşuluk yöntemi GSP için uygulanarak, tercih edilen komşuluk yöntemine göre algoritma performansının etkilendiği gösterilmiştir. Bir diğer çalışma olan [23]'de ise GKO ile YAK algoritmaları paralel olarak çalıştırılıp, belirli iterasyon adımlarında en iyi çözümler karşılıklı olarak değiş tokuş edilip, en kötü çözümle yer değiştirilmiştir. Elde edilen sonuçlarla, bu yöntemin çözümün gelişmesine katkı sağladığı gösterilmiştir. Ayrıca, aynı çalışmada GKO algoritmasında komşuluk sayısını ifade eden k parametresi lineer olarak azaltılarak daha iyi sonuçlar üretildiği ifade edilmektedir. Bununla birlikte, GKO algoritmasının çizelgeleme problemlerinin çözümünde de yaygın olarak kullanıldığı görülmektedir [24-30].

Yapılan diğer bir çalışmada [31] GKO algoritmasında iki farklı tipte çaprazlama yöntemi kullanılarak komşu çözümler üretilmiş, elde edilen sonuçlar Taguchi deney tasarımı yöntemi ile analiz edilmiş ve ideal parametrelerin belirlenmesine çalışılmıştır. [32]'de GKO algoritması özel bir komşuluk üretme yöntemi ile çalıştırılarak çok amaçlı bir görev atama probleminin çözümü üzerinde çalışılmıştır. Elde edilen sonuçlara göre, GKO algoritmasının performansının iyileştiği belirtilmiştir. Özel komşuluk fonksiyonu ile çalıştırılan GKO algoritması, GA, bal arısı çiftleşme optimizasyonu (BAÇO) algoritması ve bilinen GKO algoritması ile karşılaştırılmış ve geliştirilen yöntemin en başarılı sonuçları ürettiği gösterilmiştir. [33]'de GKO algoritması, YAK algoritması, PSO algoritması, DG algoritması ve GA ile 10 farklı test fonksiyonu ve 5 farklı boyut için karşılaştırılmış ve sonuçlar analiz edilmiştir. Yapılan çalışmada 5 farklı sistem için parametre değerleri yukarıda verilen algoritmalar ile belirlenmiş ve üretilen ortalama karesel hata (MSE) değerleri karşılaştırılmıştır. Yapılan analizlere göre GKO algoritmasının başarılı sonuçlar

ürettiği gösterilmiştir. [34]'de ise YAK algoritması ile GKO algoritması birlikte çalıştırılarak, YAK algoritmasının arama fazındaki başarısı ile GKO algoritmasın iyileştirme fazındaki başarısı kullanılarak yeni bir metot önerilmiştir. Önerilen yeni yönteme göre önce YAK algoritması çalıştırılarak yerel optimumlardan kaçınılması sağlanmış ve sonrasında GKO algoritması çalıştırılarak eldeki çözümler kullanılarak daha iyi çözümler üretilmesi hedeflenmiştir. GKO algoritması, YAK algoritmasının farklı versiyonları, PSO algoritması, DG algoritması ve GA önerilen yeni yöntem ile 10 farklı test fonksiyonu ve 5 farklı boyut için karşılaştırılmış ve elde edilen sonuçlara göre önerilen metodun yüksek oranda başarılı sonuçlar ürettiği gösterilmiştir.

2.2 Paralel Göçmen Kuşlar Optimizasyon Algoritması

GKO algoritması üzerine birçok çalışma mevcuttur. Ancak daha iyi sonuçlar elde etme isteği GKO algoritmasının birden fazla alt popülasyon kullanılarak paralel bilgisayarlarda uygulanması fikrini ortaya çıkarmıştır. Paralel programlamada amaç her zaman problemin çözümünü hızlandırmak olmayabilir. Her işlemcide aynı problem çalıştırılıp, belirli aralıklarla işlemciler arası veri alışverişi yapılarak, aramanın hızlandırılması ve iyileştirilmesi de sağlanabilir. Bu çalışmada geliştirilen PGKO algoritması ada modeli kullanılarak oluşturulmuştur. GKO algoritması, alt popülasyon içerisinde belirlenen iterasyon adımı boyunca çalıştırıldıktan sonra seçilen çözümler belirlenen topolojiye göre bir alt popülasyondan diğerine taşınmaktadır. Bu işleme göç ismi verilir. Şekil 2.3 [35]'de yaygın olarak kullanılan topolojiler gösterilmektedir.



Şekil 2.3: Arabağlantı topolojileri.

Paralel metasezgisel optimizasyon algoritmalarında göç işleminin yapısı aşağıda verilen faktörlere [14] göre belirlenmektedir.

- Topoloji: Göç edecek çözümlerin gideceği alt popülasyonu belirler.
- Göç Oranı: Göç edecek çözümlerin sayısını belirler. Alt popülasyon boyutuna göre yüzde oran ile ifade edilir.
- Göç Aralığı, Göç Frekansı: İki göç işlemi arasındaki iterasyon adımı veya belirlenen kriterdir.
- Göç Politikası:
    - Göç edecek çözümlerin seçim yöntemidir.
        - En iyilerden seç,
        - Rasgele seç.
    - Göç sonrası yer değiştirme yöntemini belirler.
        - En kötüler ile yer değiştir,

        - Rasgele yer değiştir.
- Haberleşme Modeli: Senkron / asenkron.

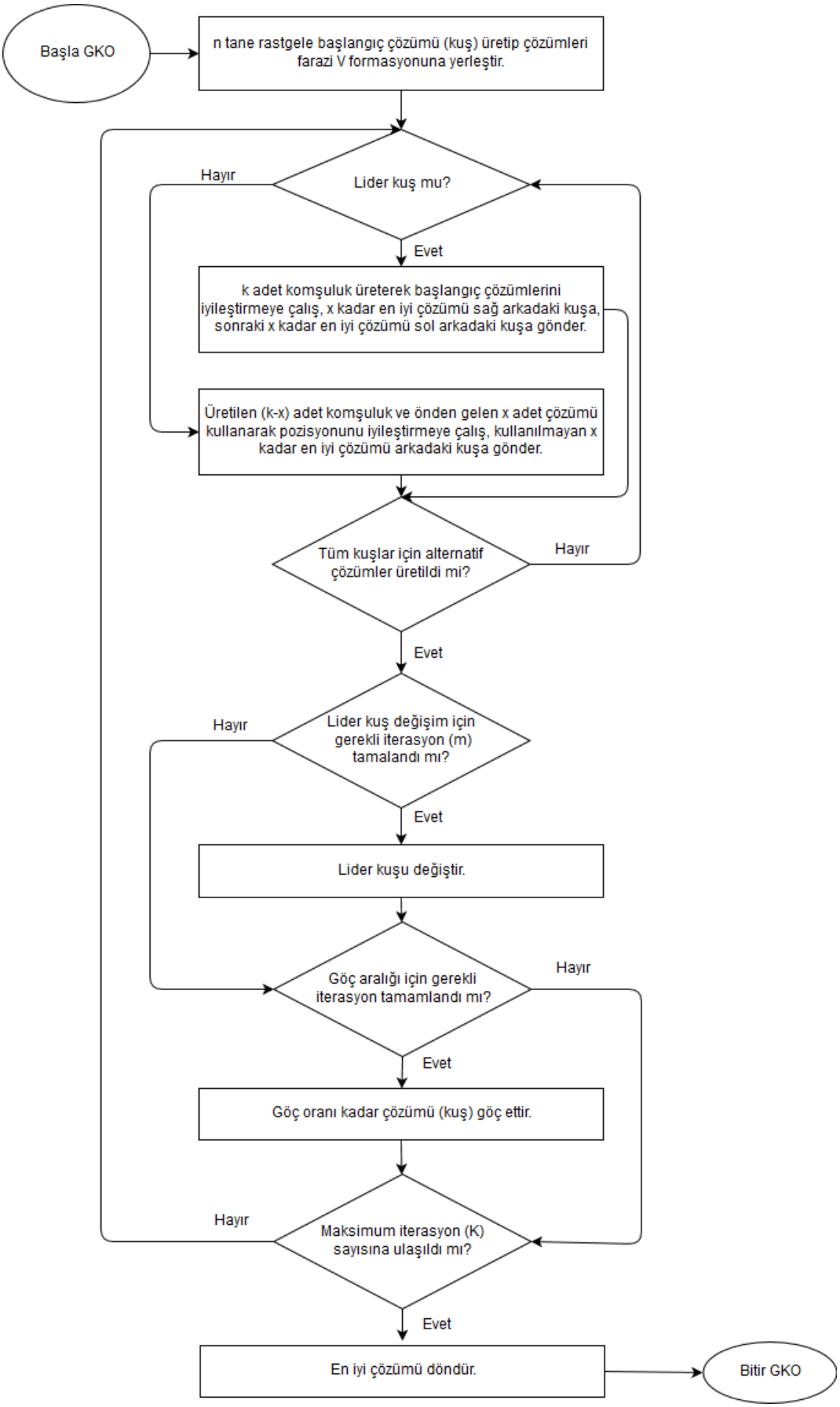
Bu çalışmada;

- Topoloji: Halka arabağlantı topolojisi kullanılmıştır.
- Göç Oranı: %2, %6, %10, %20, %40 olarak belirlenmiştir.
- Göç Aralığı, Göç Frekansı: 5, 10, 25, 50, 100 iterasyon olarak belirlenmiştir.
- Göç Politikası:
    - Göç edecek çözümlerin seçim yöntemi:
        - En iyilerden seç,
    - Göç sonrası yer değiştirme yöntemi:
        - En kötüler ile yer değiştir işlemi uygulanmıştır.
- Haberleşme Modeli: Senkron model kullanılmıştır.

Aşağıda verilen Şekil 2.4'de PGKO algoritmasının sözde kodu, Şekil 2.5'te de akış diyagramı gösterilmiştir.

```
n tane rastgele başlangıç çözümü (kuş) üretip çözümleri farazi V formasyonuna yerleştir.
M=lider kuş değişim aralığı, K=maksimum iterasyon sayısı, GA=göç aralığı, GO=Göç oranı.
for (i=0;i<K;i++)
    for (s=0;s<GA;s++)
        for (j=0;j<M;j++)
            for each kuş in sürü
                if lider==true
                    k adet komşuluk üreterek başlangıç çözümlerini iyileştirmeye çalış,
                    x kadar en iyi çözümü sağ arkadaki kuşa, x kadar en iyi çözümü sol arkadaki kuşa gönder.
                else
                    üretilen (k-x) kadar komşuluk ve önden gelen x kadar çözümü kullanarak pozisyonunu iyileştirmeye çalış,
                    kullanılmayan x kadar en iyi çözümü arkadaki kuşa gönder.
                endif
            endfor
        endfor
        lider çözümü (kuşu) değiştir.
    endfor
    GO kadar kuşu göç ettir.
endfor
return sürüdeki en iyi çözüm.
```

Şekil 2.4: PGKO algoritması sözde kodu.



Şekil 2.5: PGKO algoritmasının akış diyagramı.

3. UYGULAMA ORTAMI

Bu bölümde uygulama ortamı ve kullanılan test fonksiyonları açıklanmıştır.

3.1 Uygulama Platformu ve Araçları

Bu çalışmada yapılan tüm hesaplamalar TÜBİTAK ULAKBİM tarafından sağlanan TRUBA altyapısı kullanılarak gerçekleştirilmiştir.

3.1.1 TRUBA Altyapısı

TRUBA, TÜBİTAK ULAKBİM tarafından 2003 yılında kurulan ve ihtiyaçlar doğrultusunda gelişmekte olan yüksek başarımlı hesaplama altyapısıdır. 114 farklı üniversiteden 1900'den fazla araştırmacıya bilimsel çalışmalarına katkıda bulunmak üzere doğrudan hizmet veren, farklı kamu kurumları ve araştırma enstitüleriyle ortak projeler gerçekleştirilen ulusal e-altyapıdır. Bu altyapı, araştırmacılara 19800 adet işlemci çekirdeğini 4 Pbyte yüksek performanslı depolama alanı ile birlikte sunmaktadır [36].

Grid üzerinde çalışan uygulamalara sahip Türkiye'deki tüm üniversite, araştırma merkezi ve enstitülerden araştırmacılar, öğretim üyeleri, öğrenciler TRUBA kaynaklarını kullanmak üzere kullanıcı olabilirler [36].

TRUBA altyapısını kullanmak isteyen araştırmacıların öncelikle sisteme üye olmaları gerekmektedir. Üyelik işlemi üç adımdan oluşmaktadır. Araştırmacılar üyelik işlemlerini tamamladıktan sonra TRUBA kaynaklarını ücretsiz olarak kullanabilmektedir. Bu adımlar sırasıyla;

1- “https://portal.truba.gov.tr” sayfasında yer alan formun eksiksiz doldurulması gerekmektedir. Başvuru sırasında kurumdan alınan e-posta adresinin kullanımı zorunludur.

2- Başvurusu kabul edilen araştırmacılardan çalıştıkları kuruma ait statik IP adresi, geçerli bir kimlik belgesi ve çalıştığı kuruma ait kimlik belgesi istenmektedir. Çalışmalarını kurum dışı bilgisayarlardan sürdürmek isteyen araştırmacılar, sertifika başvurusu yapmak zorundadır. Tüm bu belgeleri eksiksiz olarak kuruma gönderen araştırmacılar için sisteme kullanıcı hesabı tanımlanmaktadır.

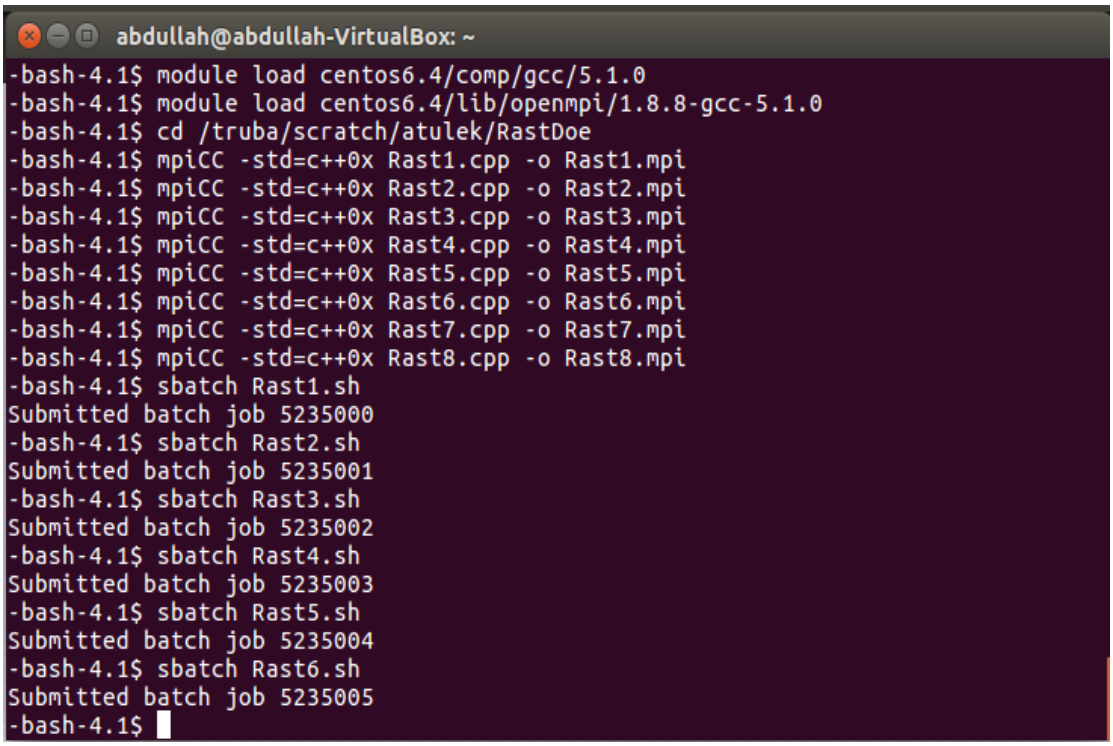
3- TRUBA altyapısında yetkilendirme işlemi için, tüm kullanıcıların TRUBA CA tarafından onaylanmış, sayısal sertifikaya sahip olması gerekmektedir (Statik IP dışında bağlananlar için). Başvuru sırasında sisteme kaydedilen kurumsal e-posta adresine gönderilen bağlantı ile sertifika başvurusu yapılması gerekmektedir.

3.1.2 Sisteme Bağlantı

Uzak sisteme bağlantı için başvuru sonucunda tanımlanan kullanıcı adı ve şifre kullanılmalıdır. Statik IP sahibi olanlar herhangi bir SSH (Secure Shell) istemcisi kullanarak "levrek1.ulakbim.gov.tr " adresinden sisteme bağlanabilmektedirler. Statik IP dışında bağlantı için başvurunun üçüncü adımında anlatılan sertifika ile OpenVPN programı kullanarak sisteme bağlanabilmektedir. OpenVPN programı için kullanılacak olan işletim sistemine göre (Windows, Linux, MAC OS) kurulum adımlarının açıklamaları ile birlikte “TRUBA.ovpn” ayar dosyası örneği ve "http://wiki.truba.gov.tr/index.php/OpenVPN_Baglantısı" adresindeki yönergeler incelenmelidir. OpenVPN bağlantısı yapıldıktan sonra kullanıcı adı ve şifre ile herhangi bir SSH (Secure Shell) istemcisi kullanarak "levrek1.ulakbim.gov.tr" adresine bağlanılabilmektedir.

Bu çalışma, uzak bağlantı yapılmadan önce Ubuntu Linux üzerinde sanallaştırma yöntemi ile Open MPI kütüphanesinin 1.10.2 sürümü ve GCC derleyicisinin 5.4.0 sürümü kullanılarak denenmiştir. Denemelerden başarılı sonuçlar alındıktan sonra gerçek küme bilgisayarlara (TRUBA) yine Ubuntu Linux işletim sistemi üzerinden OpenVPN programı ile bağlanılarak çalışılmıştır.

C++ dili ile geliştirilen uygulama dosyası uzak sistemde GCC derleyicisi 5.1.0 ve Open MPI 1.8.8 sürümleri (uygulama çalıştırıldığında sistem tarafından desteklenen en güncel sürümler) kullanılarak derlenmiştir. Geliştirilen uygulama dosyasının çalıştırılabilmesi için "/truba/scratch/kullanıcı_adı" dizininde bulundurulması gerekmektedir. Derleme işlemi için Open MPI komutlarından mpiCC komutu kullanılmış, uygulamada kullanılan çeşitli C++ kütüphaneleri C++11 sürümü tarafından desteklendiğinden "-std=c++0x" parametresi ile derleme işlemi gerçekleştirilmiştir. Derleme işleminden önce kullanılacak GCC ve Open MPI sürümleri sisteme "module load" komutu ile tanıtılmalıdır. Sistem tarafından desteklenen tüm uygulama ve kütüphanelerin listesine "module avail" komutu ile ulaşılabilmektedir. TRUBA kullanıcı ara yüzü aşağıdaki Şekil 3.1'de verilmiştir.



Şekil 3.1: TRUBA kullanıcı ara yüzü.

Derleme işlemi sonucu ortaya çıkan çalıştırılabilir dosya SLURM (kaynak yöneticisi) betiği ile kuyruğa dahil edilebilmektedir. Örnek bir betik dosyası Şekil 3.2'de verilmiştir.

Şekil 3.2: Örnek SLURM betiği.

3.2 Çalışmada Kullanılan Test Fonksiyonları

Bu bölümde GKO ve PGKO algoritmalarının performanslarını karşılaştırmak için, her biri minimizasyon problemi olan test fonksiyonları unimodal (tek modlu) ve multimodal (çok modlu) olmak üzere iki başlık altında verilmiştir.

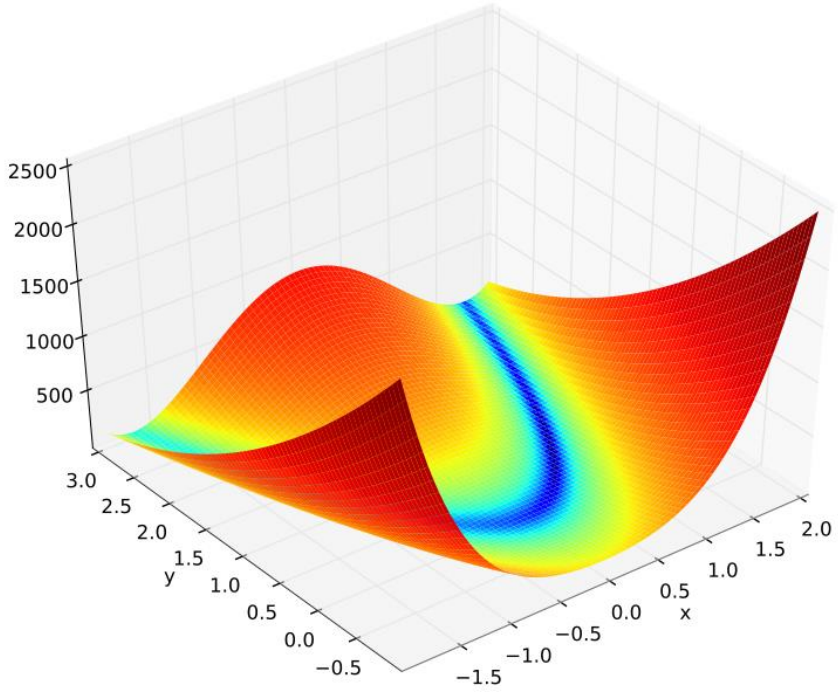
3.2.1 Unimodal (Tek Modlu) Fonksiyonlar

3.2.1.1 Rosenbrock Fonksiyonu

$$f(x) = \sum_{i=1}^{d-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2] \tag{3.1}$$

$x_i \in [-2.048, 2.048]$ aralığında, $i = 1, \ldots, d$

Global minimum $x^* = f(1, \ldots, 1)$ için $f(x^*) = 0$ [37].



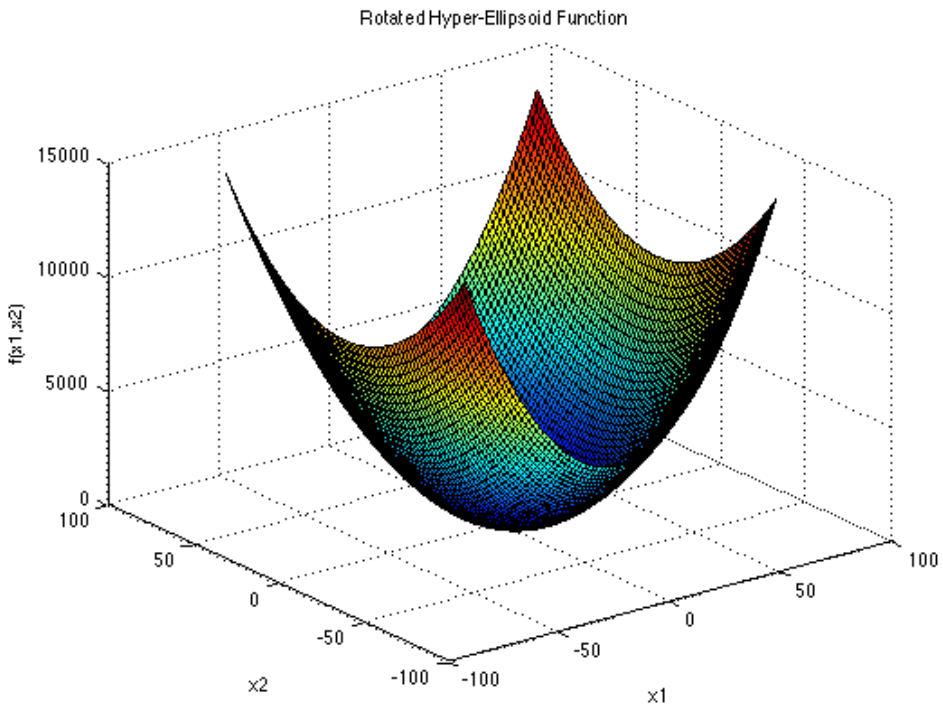
Şekil 3.3: Rosenbrock fonksiyonu 3-boyutlu görünümü.

3.2.1.2 Rotated Hyper-Ellipsoid Fonksiyonu

$$f(x) = \sum_{i=1}^{d} \sum_{j=1}^{i} x_j^2 \tag{3.2}$$

$x_i \in [-65.536, 65.536]$ aralığında, $i = 1, \ldots, d$

Global minimum $x^* = f(0, \ldots, 0)$ için $f(x^*) = 0$ [38].



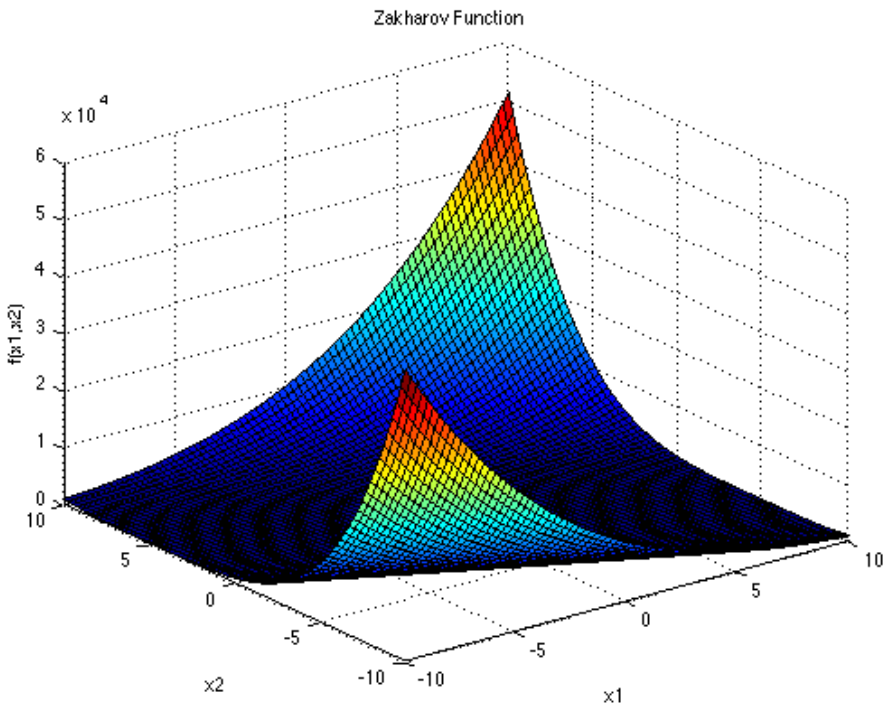
Şekil 3.4: Rotated Hyper-Ellipsoid fonksiyonu 3-boyutlu görünümü.

3.2.1.3 Zakharov Fonksiyonu

$$f(x) = \sum_{i=1}^{d} x_i^2 + \left(\sum_{i=1}^{d} 0.5ix_i\right)^2 + \left(\sum_{i=1}^{d} 0.5ix_i\right)^4 \tag{3.3}$$

$x_i \in [-5, 10]$ aralığında, $i = 1, \ldots, d$

Global min. $x^* = f(0, \ldots, 0)$ için $f(x^*) = 0$ [38].



Şekil 3.5: Zakharov fonksiyonu 3-boyutlu görünümü.

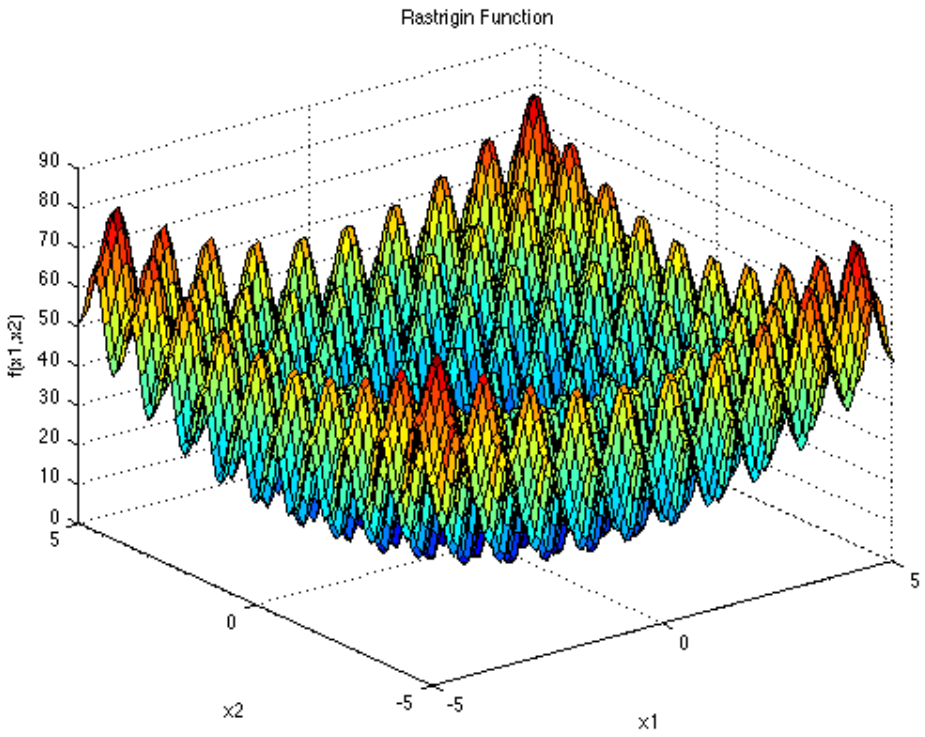
3.2.2 Multimodal (Çok Modlu) Fonksiyonlar

3.2.2.1 Rastrigin Fonksiyonu

$$f(x) = 10d + \sum_{i=1}^{d} [x_i^2 - 10\cos(2\pi x_i)] \tag{3.4}$$

$x_i \in [-5.12, 5.12]$ aralığında, $i = 1, \ldots, d$

Global minimum $x^* = f(0, \ldots, 0)$ için $f(x^*) = 0$ [39].



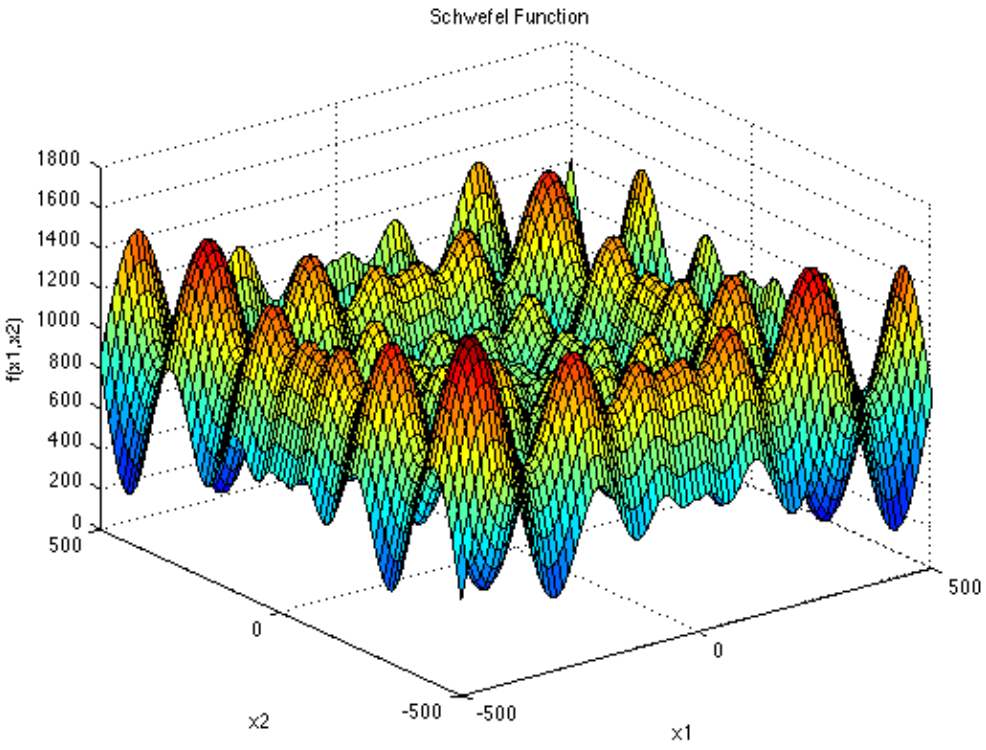
Şekil 3.6: Rastrigin fonksiyonu 3-boyutlu görünümü.

3.2.2.2 Schwefel Fonksiyonu

$$f(x) = 418.9829d - \sum_{i=1}^{d} x_i \sin\left(\sqrt{|x_i|}\right) \tag{3.5}$$

$x_i \in [-500,500]$ aralığında, $i = 1, \dots, d$

Global minimum $x^* = f(420.9687, \dots, 420.9687)$ için $f(x^*) = 0$ [40].



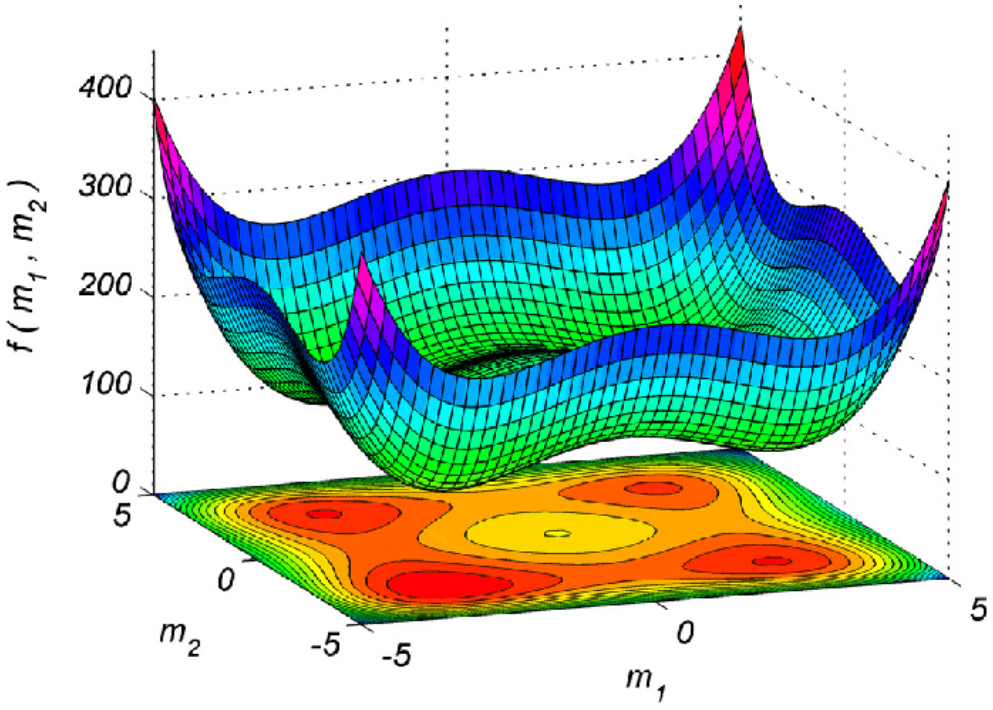
Şekil 3.7: Schwefel fonksiyonu 3-boyutlu görünümü.

3.2.2.3 Styblinski-Tang Fonksiyonu

$$f(x) = \frac{1}{2}\sum_{i=1}^{d}(x_i^4 - 16x_i^2 + 5x_i) \tag{3.6}$$

$x_i \in [-5,5]$ aralığında, $i = 1, \dots, d$

Global minimum $x^* = f(-2.903534, \dots, -2.903534)$ için $f(x^*) = -39.16599$ [41].



Şekil 3.8: Styblinski-Tang fonksiyonu 3-boyutlu görünümü.

4. UYGULAMA SONUÇLARI

Bu bölümde GKO ve PGKO algoritmaları, altı farklı test fonksiyonu için dört farklı alt popülasyon (8,16,24,32), beş farklı göç oranı (2,6,10,20,40) ve beş farklı göç aralığı (5,10,25,50,100) kullanılarak uygulanmıştır. 30 bağımsız denemeden elde edilen ortalama, en iyi ve en kötü değerleri tablo ve grafikler halinde verilmektedir.

Bunun yanında, her test fonksiyonu için 32 alt popülasyonda PGKO uygulanarak en başarılı ve en başarısız ortalama sonucun elde edildiği göç parametrelerindeki davranışı incelemek amacı ile 30 bağımsız denemenin her bir iterasyon adımındaki sonuçlarının ortalaması grafik halinde verilmiştir. Aynı sonuçlar grafik üzerinde GKO için sunularak karşılaştırma yapılabilmesi sağlanmıştır.

Ayrıca, 32 alt popülasyon için tüm göç parametrelerinde elde edilen ortalama sonuçları kullanılarak algoritma performansı farklı göç parametreleri için grafiksel olarak gösterilmiş ve sonuçlar analiz edilmiştir.

Her iki algoritmada da [17]'de önerilen algoritma parametreleri kullanılmıştır. Kullanılan parametreler Tablo 4.1'de gösterilmiştir.

Tablo 4.1: GKO ve PGKO algoritmasında kullanılan parametre değerleri.

Parametre	**Açıklama**	**Bu Çalışmada Kullanılan Değerler**
n	Popülasyondaki birey (kuş) sayısı	51
k	Her bir birey için üretilecek olan komşuluk sayısı	3
x	Bir sonraki birey ile paylaşılacak olan çözüm sayısı	1
m	Lider kuş değişim aralığı	10
K	Maksimum iterasyon sayısı	2500

4.1 Rosenbrock Fonksiyonu İçin Sonuçlar

Rosenbrock fonksiyonundan elde edilen sonuçlar aşağıda verilen Tablo 4.2'de gösterilmektedir. 2500 iterasyon sonunda elde edilen sonuçlar farklı alt popülasyon sayıları için analiz edilmiştir. PGKO, en başarılı ortalama sonucunu alt popülasyon sayısı 8 ve 32 olduğunda %10-5, 16 ve 24 olduğunda %20-5 durumları için üretmiştir. Tablodaki veriler incelendiğinde tüm alt popülasyonlarda en başarılı sonuçların, göçün sık olduğu durumlarda alındığı görülmektedir. Göç aralığı arttıkça algoritmanın aldığı sonuçların başarısı düşmektedir. En iyi sonuçların en başarılısı alt popülasyon sayısı 8 olduğunda %40-5, 16 olduğunda %40-10, 24 olduğunda %10-5 ve 32 olduğunda %6-5 değerleri için elde edilmiştir.

PGKO'nun ortalama sonuçlarının tümü incelendiğinde 100 durumdan 26 durumda GKO'dan daha kötü sonuç ürettiği görülmektedir. Bunlardan 11 tanesi alt popülasyon sayısının 16 olduğu durumlarda, 14 tanesi alt popülasyon sayısının 24 olduğu durumda ve 1 tanesi de alt popülasyon sayısının 32 olduğu durumdur. En iyi sonuçlara baktığımızda ise 70 durumda GKO'nun daha başarılı olduğu, diğer durumlarda PGKO'nun daha başarılı olduğu görülmektedir. En kötü sonuçlar incelendiğinde PGKO'nun 5 durum için GKO'dan daha başarısız sonuç ürettiği görülmektedir.

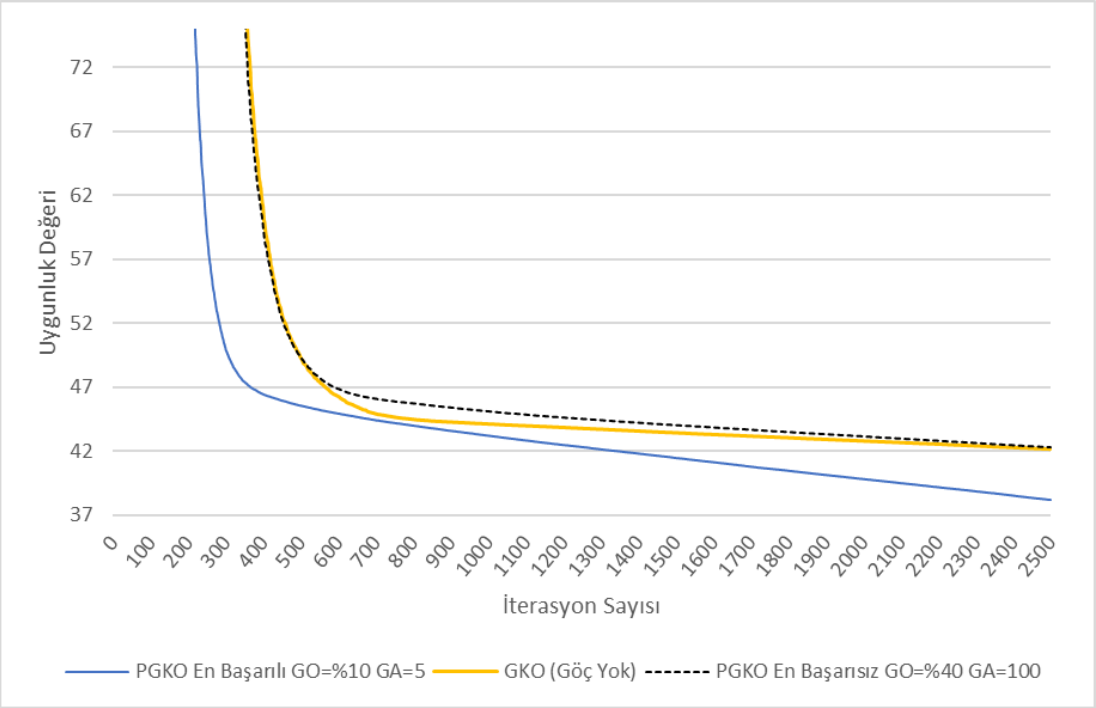
Tablo 4.2: Rosenbrock fonksiyonu için elde edilen sonuçlar.

Rosenbrock		Alt Popülasyon Sayısı											
		8			16			24			32		
Göç Oranı	Göç Aralığı	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü
Göç Yok		42.8003	39.9944	43.6839	41.7795	37.1348	43.5224	41.2192	36.1843	43.5378	42.2141	40.0936	43.4766
2	5	42.2588	39.5672	**94.6499	39.6733	38.7665	40.2659	39.3234	36.3136	40.0512	39.0897	36.7913	40.0107
	10	41.1747	40.4636	41.7137	40.6881	37.2553	41.453	40.8655	40.0973	41.2797	40.7338	40.1424	41.3918
	25	41.9561	41.3854	42.4258	41.8149	39.8422	42.3731	41.7857	37.8961	42.1396	41.6045	36.9264	42.0827
	50	42.4508	41.9047	42.7406	42.2943	40.3981	42.6241	42.101	38.4793	42.5271	42.0254	41.1277	42.5006
	100	42.3187	34.9392	42.97	42.4133	40.8004	42.867	42.0634	39.1925	42.7427	41.8991	38.2051	**42.8111
6	5	39.6524	32.484	44.5874	38.9701	38.3971	*39.8961	38.4481	34.8082	39.3717	38.2041	*35.1375	39.1488
	10	40.8588	38.622	41.4322	40.4712	39.8509	40.929	40.3118	38.0623	40.9157	40.3477	39.9989	40.5997
	25	41.7559	40.9672	42.2681	41.6735	41.2049	42.0452	41.601	41.2939	42.0461	41.4932	40.5403	41.7935
	50	42.2055	41.8493	42.7463	42.0662	40.8021	42.4324	41.7661	37.8805	42.3481	42.0251	**41.4201	42.231
	100	42.5158	42.0546	42.8275	42.1977	40.7196	42.6972	41.8093	39.7661	42.6779	42.1724	40.8098	42.508
10	5	*39.5098	37.8695	41.4098	38.9564	37.6967	43.3842	38.4342	*34.315	39.4961	*38.1868	35.3181	39.0951
	10	40.6535	39.5142	*41.3234	40.3027	39.3861	40.9126	40.1821	39.1598	40.6441	40.0364	38.2201	40.4822
	25	41.791	41.2859	42.2856	41.5546	40.7439	41.892	41.4273	40.9397	41.7723	41.4837	41.1623	41.8778
	50	42.1601	41.755	42.5234	42.0475	41.2442	42.4513	40.8995	39.62	42.4009	41.6011	38.6662	42.2299
	100	42.4176	41.1767	42.7913	42.1719	40.4356	42.6852	42.1031	34.8877	42.5702	41.8039	39.6212	42.5255
20	5	40.032	38.0905	45.6403	*38.7953	36.0749	40.4274	*38.4238	36.7053	*39.2318	38.2634	36.7329	*39.0775
	10	40.9245	38.9683	45.1013	40.2843	39.0987	40.9706	40.1138	38.8377	40.6016	39.899	38.7701	40.5663
	25	41.7408	41.2815	42.1898	41.4237	40.7194	41.9414	41.3863	40.751	41.6927	41.3686	40.7957	41.6842
	50	42.1333	41.5526	42.5914	41.9256	39.6513	42.4306	41.9511	**41.5343	42.2339	41.8375	40.2857	42.2636
	100	42.587	42.2115	43.1323	42.4074	41.1029	42.9327	42.2162	39.2286	42.6996	42.1519	39.4544	42.6474
40	5	39.5278	*31.9134	43.5778	39.1595	37.6661	**43.9231	38.6324	35.5834	39.4096	38.4346	36.9022	39.2844
	10	40.8208	40.0613	42.9327	40.2178	*36.0049	40.7672	40.2754	39.7024	40.6555	39.9858	36.8547	40.6033
	25	41.8291	41.6092	42.2596	41.4839	40.1517	42.0642	41.5473	41.1723	41.9406	41.4066	40.6107	41.8272
	50	42.3579	42.061	42.6125	42.1552	**41.8694	42.6154	41.8912	40.73	42.4705	41.9804	39.5831	42.5461
	100	**42.7639	**42.2437	43.1522	**42.4796	41.0327	43.0053	**42.5049	41.4306	**42.8819	**42.2825	40.7511	42.7835
PGKO > GKO		0	16	4	11	23	1	14	21	0	1	10	0

* PGKO'nun en başarılı çözümü.

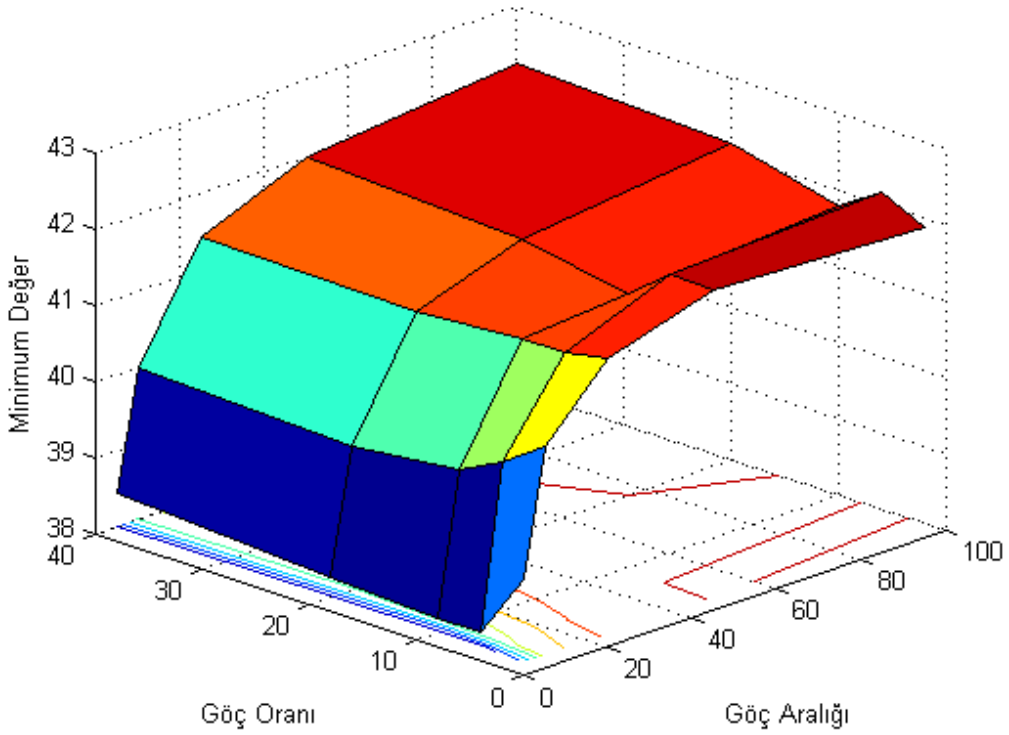
** PGKO'nun en başarısız çözümü.

Şekil 4.1'de, alt popülasyon sayısı 32 için ortalama sonuçlarından elde edilen GKO, en başarılı ve en başarısız PGKO grafikleri verilmektedir. En başarılı durum (%10-5) grafiğine bakıldığında algoritmanın arama fazında GKO'ya göre daha hızlı arama yaptığı, iyileştirme fazında ise detaylı bir arama yaparak iyi sonuçlar üretmeye devam ettiği görülmektedir. En başarısız durum (%40-100) grafiğinde PGKO'nun GKO'ya benzer bir davranış göstermesine rağmen, iyileştirme fazında arama yapmaya devam ettiği görülmektedir.



Şekil 4.1: Rosenbrock fonksiyonu global minimuma yakınsama grafiği.

Şekil 4.2'de, alt popülasyon sayısı 32 alındığında farklı göç parametreleri için elde edilen en başarılı ortalama sonuçları grafiksel olarak verilmektedir. Buna göre algoritmanın 32 alt popülasyon için göç oranı %10, göç aralığı 5 olduğunda en başarılı sonucu verdiği görülmektedir. Göç aralığı incelendiğinde, aralık arttıkça başarının düştüğü görülmektedir. Rosenbrock fonksiyonu incelendiğinde, problemin yapısı gereği farklı boyutlardaki değerlerin ilişkisinin sonucu etkilediği görülmektedir. Bu nedenle göç ile alt popülasyona katılan yeni bireyler genel bir iyileşme sağlamaktadır. Göç aralığı düşük olduğunda, yani sık göç uygulandığında algoritma performansı artmaktadır. Göç oranı düşük olduğunda daha başarılı sonuçlar elde edilirken yüksek olduğunda ise başarının azaldığı görülmektedir.



Şekil 4.2: Rosenbrock fonksiyonu göç parametreleri grafiği.

4.2 Rotated Hyper-Ellipsoid Fonksiyonu İçin Sonuçlar

Rotated Hyper-Ellipsoid fonksiyonundan elde edilen sonuçlar aşağıda verilen Tablo 4.3'de gösterilmektedir. Elde edilen sonuçlar farklı alt popülasyon sayıları için analiz edilmiştir. PGKO, ortalama, en iyi ve en kötü sonuçların en başarılılarını tüm alt popülasyonlar için göç oranının %40, göç aralığının 5 olduğu durumlar için üretmiştir. Tablodaki veriler incelendiğinde tüm alt popülasyonlarda en başarılı sonuçların, göçün sık olduğu ve göç eden bireylerin fazla olduğu durumlarda elde edildiği görülmektedir. Göç aralığı arttıkça algoritmanın aldığı sonuçların başarısı düşmektedir. PGKO'nun ortalama sonuçlarının tümü incelendiğinde, 100 durumun tamamında GKO'dan daha iyi sonuçlar ürettiği görülmektedir. En iyi sonuçlara baktığımızda ise sadece 2 durumda GKO'nun daha başarılı olduğu, diğer 98 durumda PGKO'nun daha başarılı olduğu görülmektedir. En kötü sonuçlar incelendiğinde ise yine PGKO'nun tüm durumlar için GKO'dan daha başarılı sonuç ürettiği görülmektedir.

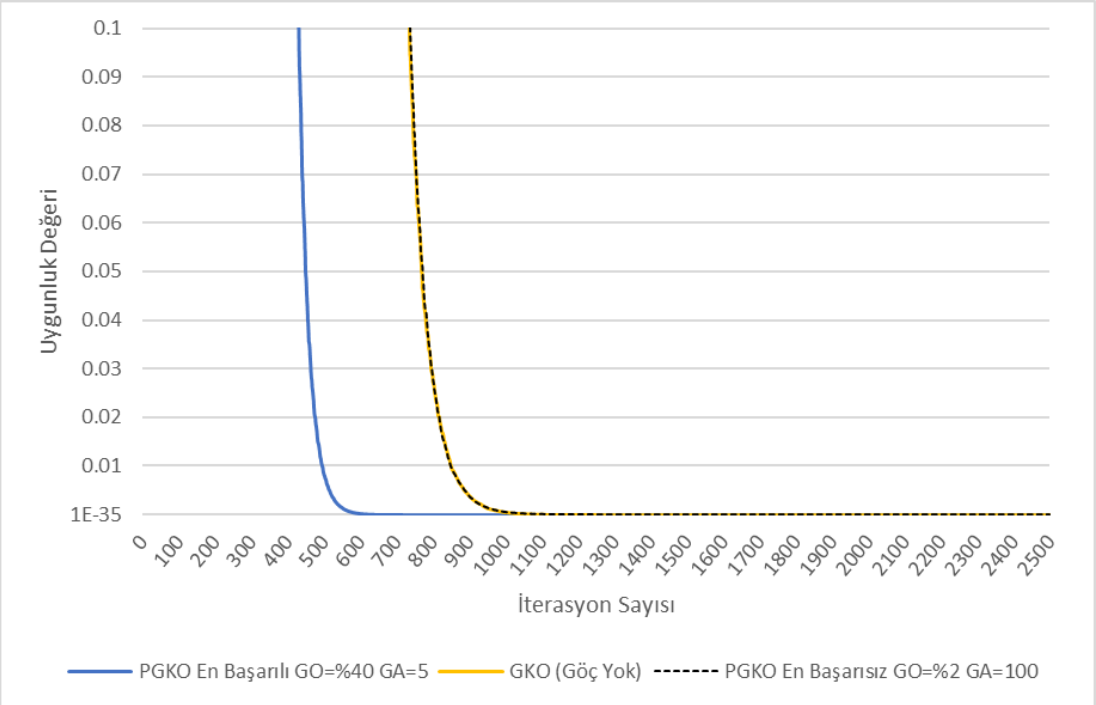
Tablo 4.3: Rotated Hyper-Ellipsoid fonksiyonu için elde edilen sonuçlar.

Rotated Hyper-Ellipsoid		Alt Popülasyon Sayısı											
		8			16			24			32		
Göç Oranı	Göç Aralığı	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü
Göç Yok		1.61E-16	3.158E-17	5.42E-16	1.01E-16	2.05E-17	2.51E-16	9.53E-17	3.706E-17	2.403E-16	7.30E-17	2.089E-17	1.369E-16
2	5	2.47E-22	6.811E-23	6.39E-22	1.61E-22	5.308E-23	3.55E-22	1.57E-22	5.03E-23	4.05E-22	1.55E-22	6.25E-23	2.81E-22
	10	2.31E-19	7.28E-20	5.02E-19	2.14E-19	1.04E-19	4.99E-19	1.78E-19	6.04E-20	3.79E-19	1.62E-19	9.34E-20	2.46E-19
	25	1.31E-17	4.91E-18	2.42E-17	1.03E-17	3.82E-18	2.16E-17	8.20E-18	3.63E-18	1.28E-17	9.13E-18	5.25E-18	1.79E-17
	50	4.75E-17	***1.50E-17**	8.78E-17	3.50E-17	1.28E-17	5.35E-17	2.83E-17	9.88E-18	5.26E-17	2.48E-17	9.53E-18	4.76E-17
	100	***8.66E-17**	1.41E-17	***1.72E-16**	***7.08E-17**	***2.17E-17**	***1.39E-16**	***4.84E-17**	1.24E-17	***9.43E-17**	***5.04E-17**	***2.81E-17**	***7.94E-17**
6	5	2.33E-26	3.77E-27	5.66E-26	1.55E-26	2.93E-27	3.48E-26	1.25E-26	4.76E-27	3.36E-26	9.43E-27	2.80E-27	1.67E-26
	10	2.33E-21	8.55E-22	5.15E-21	1.84E-21	9.48E-22	3.79E-21	1.72E-21	5.76E-22	4.53E-21	1.41E-21	6.64E-22	2.98E-21
	25	2.44E-18	7.78E-19	4.71E-18	1.66E-18	5.77E-19	3.35E-18	1.50E-18	8.23E-19	2.60E-18	1.37E-18	5.26E-19	2.27E-18
	50	2.24E-17	9.03E-18	3.95E-17	1.49E-17	6.81E-18	2.66E-17	1.55E-17	7.15E-18	3.20E-17	1.22E-17	4.86E-18	2.26E-17
	100	5.96E-17	1.47E-17	1.54E-16	4.53E-17	1.64E-17	9.45E-17	3.75E-17	***1.50E-17**	6.57E-17	3.38E-17	1.05E-17	5.07E-17
10	5	1.67E-28	3.13E-29	4.20E-28	8.38E-29	2.20E-29	2.64E-28	6.72E-29	2.03E-29	1.66E-28	6.08E-29	1.81E-29	1.70E-28
	10	1.78E-22	5.71E-23	3.65E-22	1.09E-22	4.74E-23	1.88E-22	1.00E-22	4.52E-23	2.29E-22	9.44E-23	4.89E-23	1.56E-22
	25	8.67E-19	3.45E-19	2.52E-18	8.11E-19	2.45E-19	1.90E-18	6.12E-19	2.82E-19	1.10E-18	5.50E-19	1.95E-19	8.13E-19
	50	1.25E-17	4.57E-18	2.32E-17	1.03E-17	2.79E-18	1.85E-17	9.61E-18	3.83E-18	1.70E-17	8.98E-18	3.55E-18	1.74E-17
	100	4.98E-17	1.26E-17	1.04E-16	3.76E-17	1.45E-17	7.89E-17	2.39E-17	1.18E-17	4.57E-17	2.66E-17	1.11E-17	5.07E-17
20	5	2.30E-31	4.37E-32	6.97E-31	8.96E-32	3.50E-32	2.00E-31	7.46E-32	2.14E-32	1.50E-31	6.74E-32	2.33E-32	1.56E-31
	10	4.62E-24	1.52E-24	1.08E-23	2.95E-24	6.22E-25	5.44E-24	2.62E-24	5.33E-25	5.44E-24	2.34E-24	1.14E-24	4.41E-24
	25	2.01E-19	7.32E-20	4.23E-19	1.35E-19	5.01E-20	2.78E-19	1.48E-19	1.04E-19	2.16E-19	1.16E-19	6.07E-20	2.14E-19
	50	6.29E-18	2.49E-18	1.37E-17	5.19E-18	2.26E-18	1.15E-17	4.63E-18	2.22E-18	1.04E-17	4.10E-18	2.48E-18	7.88E-18
	100	3.20E-17	6.88E-18	6.82E-17	2.48E-17	1.29E-17	4.75E-17	1.85E-17	5.24E-18	3.41E-17	1.89E-17	4.94E-18	3.68E-17
40	5	***3.58E-33**	***4.67E-34**	***9.36E-33**	***1.35E-33**	***1.51E-34**	***3.46E-33**	***1.05E-33**	***1.86E-34**	***1.88E-33**	***9.41E-34**	***4.04E-34**	***2.23E-33**
	10	4.27E-25	7.26E-26	1.57E-24	2.47E-25	1.24E-25	4.98E-25	1.66E-25	8.77E-26	3.38E-25	1.65E-25	6.83E-26	2.98E-25
	25	5.49E-20	1.78E-20	1.04E-19	3.71E-20	1.59E-20	6.48E-20	3.33E-20	1.36E-20	6.38E-20	3.32E-20	1.36E-20	6.23E-20
	50	3.11E-18	1.12E-18	6.34E-18	2.34E-18	6.32E-19	4.95E-18	2.16E-18	9.16E-19	3.56E-18	2.06E-18	1.16E-18	3.34E-18
	100	2.10E-17	5.90E-18	4.90E-17	1.75E-17	8.80E-18	3.35E-17	1.55E-17	6.59E-18	2.93E-17	1.22E-17	4.25E-18	2.37E-17
PGKO > GKO		0	0	0	0	1	0	0	0	0	0	1	0

* PGKO'nun en başarılı çözümü.

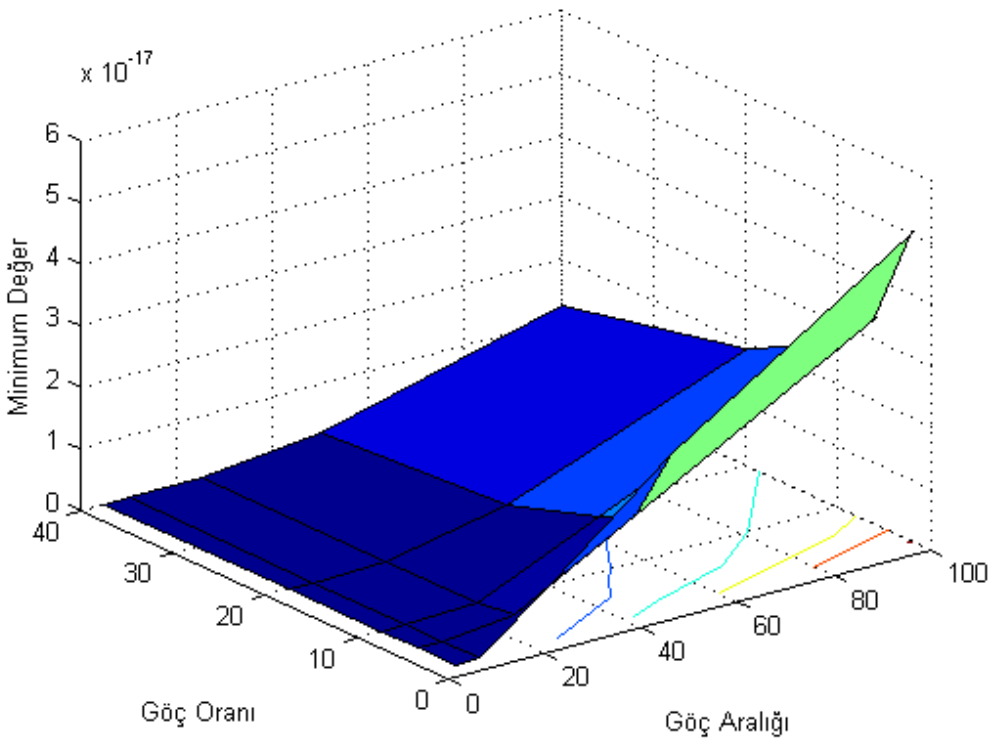
** PGKO'nun en başarısız çözümü.

Şekil 4.3'de, alt popülasyon sayısı 32 için en başarılı (%40-5) ve en başarısız (%2-100) ortalama sonuçlarının üretildiği PGKO yakınsama grafiği GKO ile birlikte verilmektedir. En başarılı PGKO durum grafiğinde algoritmanın arama fazında GKO'ya göre daha hızlı arama yaptığı ve iyi sonuçlara çok erken adımlarda ulaştığı, iyileştirme fazında ise çok küçük değerlere inildiğinden çok iyi görülmemekle birlikte, tablolardaki sonuçlar incelendiğinde detaylı bir arama yaparak iyi sonuçlar üretmeye devam ettiği görülmektedir. En başarısız PGKO grafiğinde ise GKO'ya benzer bir davranış gösterdiği, bunun yanında iterasyon sonunda ürettiği sonucun daha başarılı olduğu görülmektedir.



Şekil 4.3: Rotated Hyper-Ellipsoid fonksiyonu 32 alt popülasyon için global minimuma yakınsama grafiği.

Şekil 4.4'de, alt popülasyon sayısı 32 alındığında, uygulanan tüm göç parametreleri için elde edilen en başarılı ortalama sonuçlar verilmektedir. Buna göre algoritmanın göç oranı %40, göç aralığı 5 olduğunda en başarılı sonucu verdiği görülmektedir. Göç aralığının etkisi incelendiğinde, aralık arttıkça başarının düştüğü açıkça ortaya konmaktadır. Problemin yerel optimum noktası bulunmadığı için her göç işlemi alt popülasyonu optimum noktaya doğru yaklaştırarak performansı arttırıcı bir etki yapmaktadır. Göç oranı çok düşük olduğunda göç eden çözüm sayısı hedef alt popülasyona yeterince katkı sağlayamadığı için başarılı sonuçlar elde edilememiştir.



Şekil 4.4: Rotated Hyper-Ellipsoid fonksiyonu göç parametreleri grafiği.

4.3 Zakharov Fonksiyonu İçin Sonuçlar

Zakharov fonksiyonundan elde edilen sonuçlar Tablo 4.4'de verilmektedir. PGKO, en başarılı ortalama sonucunu 8 alt popülasyon için %10-10, diğer alt popülasyonlar için %40-5 durumları için üretmiştir. 16, 24 ve 32 alt popülasyon için en başarılı sonuçların, göçün sık olduğu ve göç eden bireylerin fazla olduğu durumlarda ürettiği görülmektedir. Göç aralığı arttıkça Rosenbrock ve Rotated Hyper-Ellipsoid fonksiyonlarında olduğu gibi algoritmanın ürettiği sonuçların başarısı düşmektedir. En iyi sonuçların en başarılısına baktığımızda tüm alt popülasyonlar için göç oranının %40, göç aralığının 5 olduğu durumlarda elde edildiği görülmektedir. PGKO'nun ortalama, en iyi ve en kötü sonuçlarının tümü incelendiğinde, 100 durumun tamamında GKO'dan daha başarılı sonuçlar ürettiği görülmektedir. Bu sonuç, göç işleminin PGKO algoritmasına katkısını açıkça ortaya koymaktadır.

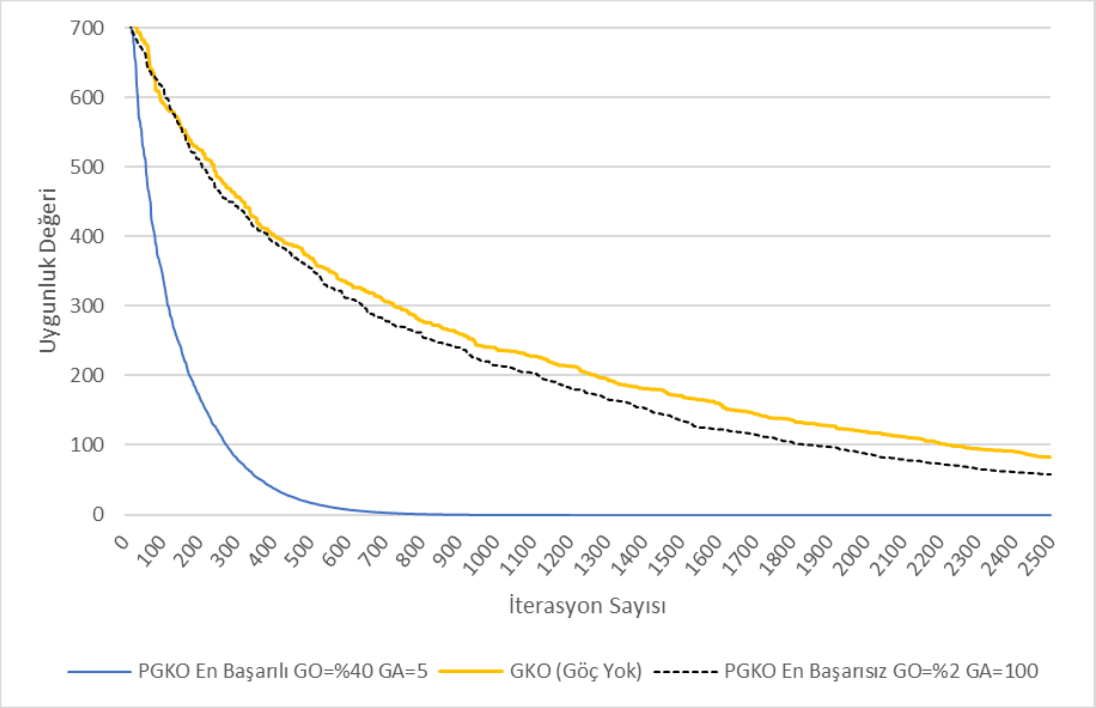
Tablo 4.4: Zakharov fonksiyonu için elde edilen sonuçlar.

Zakharov		Alt Popülasyon Sayısı											
		8			16			24			32		
Göç Oranı	Göç Aralığı	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü
Göç Yok		93.8771	70.0435	118.251	81.7517	67.2699	106.729	84.8536	72.2978	104.519	82.2964	65.7492	99.988
2	5	0.0101907	0.0013659	0.0360117	0.0063907	0.0025817	0.0173523	0.0048884	0.0020354	0.0103296	0.0050115	0.0014208	0.0103281
	10	0.450773	0.200806	1.02817	0.548355	0.226736	1.16203	0.363198	0.180111	0.798371	0.310389	0.138084	0.476949
	25	10.886	4.05724	20.476	8.25528	3.30277	18.8105	6.80191	3.16566	10.2186	7.03163	3.00492	11.8111
	50	35.2726	23.3807	49.5387	32.2779	18.4934	49.5676	31.5737	16.3226	44.4873	26.3844	16.7091	35.5674
	100	**72.5908	**42.4558	**100.798	**62.4603	**47.0551	**80.0816	**60.7321	**47.0531	**74.564	**58.0707	**43.9338	**68.447
6	5	2.03503	3.06E-05	2.62E+01	3.56E-05	1.35E-05	1.20E-04	2.55E-05	8.66E-06	5.20E-05	2.10E-05	9.68E-06	4.02E-05
	10	0.109779	0.0051539	2.72366	0.0135535	0.005918	0.0217226	0.0103905	0.004482	0.0215198	0.0104449	0.0048682	0.0178124
	25	2.24844	1.11446	3.56327	1.81175	0.636774	4.28654	1.67647	0.60925	3.32408	1.62604	0.951742	2.79852
	50	15.3115	5.83687	24.0823	12.9457	8.11872	22.0009	12.5136	7.01437	18.9888	11.298	6.686	14.7565
	100	48.5627	25.9552	63.0658	41.8873	29.6555	54.5256	37.5366	27.1351	49.7059	34.1294	24.412	43.4492
10	5	6.81495	3.61E-06	5.14E+01	3.51E-06	8.62E-07	6.12E-06	3.16E-06	1.24E-06	6.66E-06	2.35E-06	1.08E-06	4.00E-06
	10	*5.04E-03	0.0011858	*1.07E-02	0.0033381	0.0009477	0.009402	0.0023159	0.0006846	0.0043268	0.0024875	0.0013139	0.0065343
	25	1.21488	0.470678	1.98985	0.778279	0.402676	1.90388	0.865573	0.465125	1.6013	0.691632	0.404817	1.30459
	50	12.2899	5.98938	20.2851	9.14843	4.07839	14.5341	9.7071	6.48204	15.8171	8.19076	5.67124	11.6591
	100	38.1434	23.2486	60.2824	33.7014	20.0848	42.9621	30.9719	18.7766	43.0648	27.8168	17.7004	38.6728
20	5	4.55203	4.65E-07	2.60E+01	5.13E-07	2.03E-07	1.16E-06	2.85E-07	1.19E-07	4.70E-07	2.66E-07	1.05E-07	5.65E-07
	10	0.874037	0.0005544	26.1816	0.0007445	0.0002941	0.0013101	0.0005612	0.000172	0.0011046	0.0004571	0.0001641	0.0014737
	25	0.532848	0.194659	1.12118	0.385998	0.187969	0.670941	0.360321	0.171376	0.547603	0.272285	0.119121	0.446326
	50	7.84123	4.25199	13.2085	6.1407	3.22415	10.3858	5.21741	3.0881	8.77367	4.83538	2.81888	8.03985
	100	31.5177	23.1587	42.7883	25.5149	13.7807	35.8071	24.1574	16.0185	30.4268	22.1686	16.3611	28.7465
40	5	4.33176	*1.73E-07	2.65E+01	*1.72E-07	*3.59E-08	*4.55E-07	*1.59E-07	*6.96E-08	*4.42E-07	*9.44E-08	*2.09E-08	*2.63E-07
	10	0.869815	0.0002384	26.0717	0.000347	0.0001514	0.0006778	0.0002757	7.38E-05	5.27E-04	0.000262	0.0001402	0.0004507
	25	0.355839	0.156072	0.850028	0.241879	0.133309	0.497343	0.213502	0.0917454	0.338282	0.194893	0.0756049	0.389359
	50	5.19498	3.04103	8.98752	4.52212	2.20824	7.66356	4.00865	2.14857	6.89501	3.70168	1.38457	5.49313
	100	25.355	12.3991	35.3752	23.121	13.543	34.8333	20.8633	14.1536	28.3777	19.7934	10.9749	29.0433
PGKO > GKO		0	0	0	0	0	0	0	0	0	0	0	0

* PGKO'nun en başarılı çözümü.

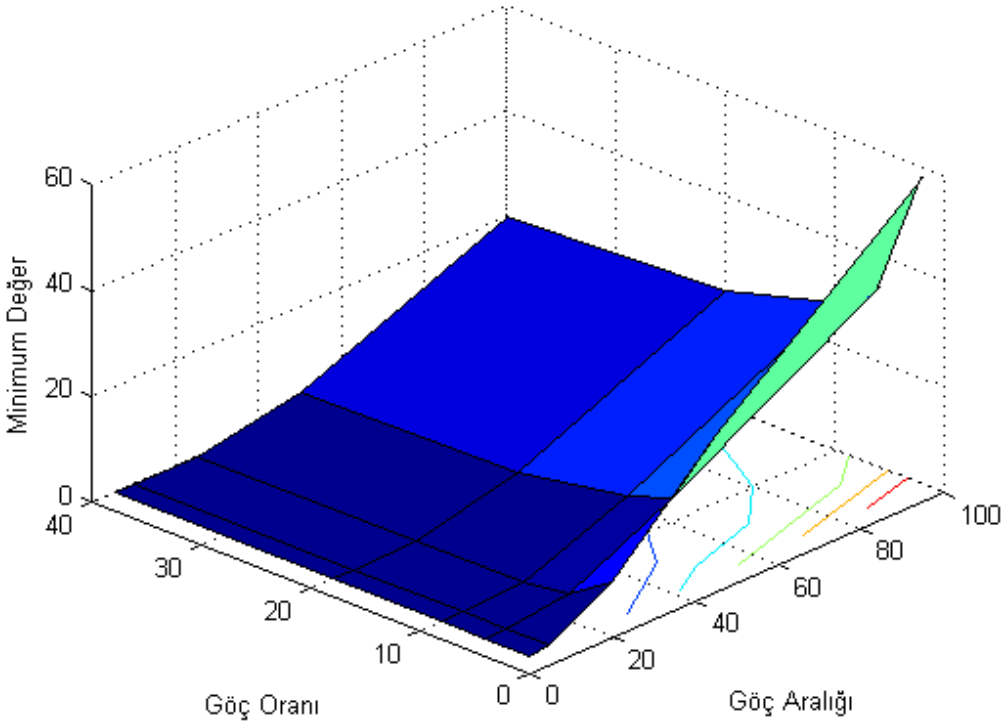
** PGKO'nun en başarısız çözümü.

Ortalama sonuçlarından elde edilen GKO, en başarılı (%40-5) ve en başarısız (%2-100) PGKO grafikleri Şekil 4.5'de verilmektedir. Buna göre GKO'nun çok yavaş bir arama yaptığı ve iterasyon adımlarının son bölümünde aramaya devam etmesine rağmen yeterince iyi sonuçlar üretemediği görülmektedir. En başarılı PGKO durum grafiğinde ise algoritmanın arama fazında oldukça hızlı arama yaptığı, iyileştirme fazında ise detaylı bir arama yaparak iyi sonuçlar üretmeye devam ettiği görülmektedir. En başarısız durum da PGKO'nun GKO'ya benzer bir davranış göstermesine rağmen, iyileştirme fazında daha iyi sonuçlar ürettiği görülmektedir.



Şekil 4.5: Zakharov fonksiyonu global minimuma yakınsama grafiği.

Şekil 4.6'da alt popülasyon sayısı 32 için göç parametrelerindeki değişimin PGKO algoritmasına etkisi gösterilmiştir. Buna göre algoritmanın göç oranı %40, göç aralığı 5 olduğunda en başarılı sonucu verdiği ortaya konmaktadır. Göç aralığı arttıkça PGKO'nun başarısı düşmektedir. Bunun nedeni fonksiyonda lokal optimum nokta bulunmadığından alt popülasyona yeni gelen her iyi çözümün genel iyileşmeye katkı sağlaması ve algoritma performansını arttırmasıdır.



Şekil 4.6: Zakharov fonksiyonu göç parametreleri grafiği.

4.4 Rastrigin Fonksiyonu İçin Sonuçlar

Rastrigin fonksiyonundan elde edilen sonuçlar aşağıda verilen Tablo 4.5'de gösterilmektedir. PGKO, en başarılı ortalama sonucunu alt popülasyon sayısı 8 olduğunda %40-100, 16 ve 24 olduğunda %10-50, 32 olduğunda %10-25 değerleri için üretmiştir. Alt popülasyon sayısı 8 olduğunda düşük göç aralıklarında başarısız sonuçlar üretildiği görülmektedir. En iyi sonuçların en başarılısına baktığımızda ise alt popülasyon sayısı 8 olduğunda %40-50, 16 ve 24 olduğunda %10-25, 32 olduğunda ise %20-5 değerleri için elde edildiği görülmektedir. PGKO'nun ortalama sonuçlarının tümü incelendiğinde 100 durumdan sadece 1 durumda GKO'dan daha kötü sonuç ürettiği görülmektedir. En iyi sonuçlara baktığımızda ise 16 durumda GKO'nun daha başarılı olduğu görülmektedir. Bunların 11 tanesi alt popülasyon sayısı 8 olduğunda oluşmuştur. En kötü sonuçlar incelendiğinde PGKO'nun sadece 1 durum için GKO'dan daha başarısız sonuç üretmiştir.

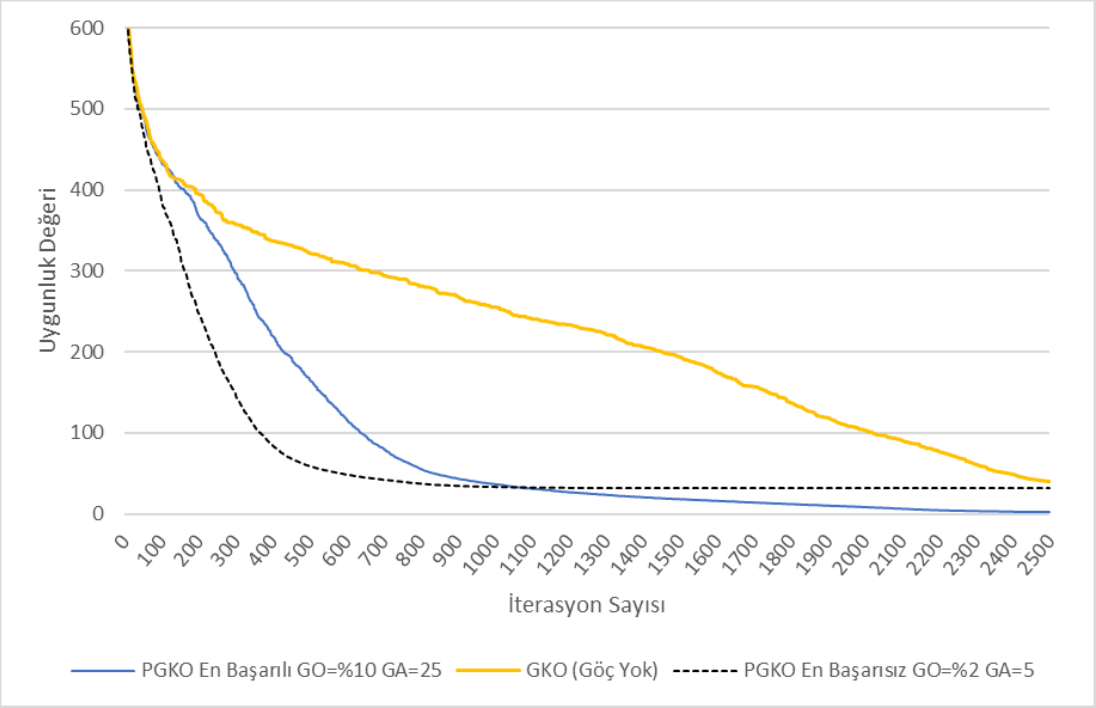
Tablo 4.5: Rastrigin fonksiyonu için elde edilen sonuçlar.

Rastrigin		Alt Popülasyon Sayısı											
		8			16			24			32		
Göç Oranı	Göç Aralığı	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü
Göç Yok		80.8992	23.051	189.203	56.907	22.0072	190.465	46.2378	18.5705	122.959	39.7725	15.9728	127.09
2	5	59.001	41.7883	82.5815	54.3247	**35.8185	91.5361	**43.5128	**17.9093	76.6117	**32.5683	**16.9143	61.6874
	10	50.5107	33.8286	88.5512	42.5511	26.8639	67.6571	26.5322	2.98488	46.763	14.5596	0.995088	45.7681
	25	34.6577	20.8941	55.7177	21.6238	3.98038	44.7731	7.99271	3.01948	30.8437	6.38475	1.00783	14.9259
	50	27.5603	14.926	41.7883	19.5828	7.00287	32.8336	15.8809	8.15446	23.9013	13.2975	8.77152	18.9287
	100	71.2448	21.1077	**250.478	28.5329	16.3077	55.219	25.3193	13.0168	44.7353	25.5664	14.9451	**68.447
6	5	64.4401	42.7832	97.5058	55.3197	19.8992	86.5613	35.7522	2.98488	67.6571	22.6187	4.9748	43.7782
	10	53.3629	34.8236	80.5916	43.8445	18.9042	77.6067	22.0218	6.96471	37.8084	10.3807	0.994959	36.8135
	25	42.8164	19.8992	69.647	22.3086	4.00197	38.8034	7.80231	0.996986	28.8538	2.94864	0.0039281	*5.97492
	50	29.8488	16.9143	43.7782	13.4374	6.96583	26.8639	7.24093	2.31418	12.9445	8.40255	4.00381	15.7013
	100	29.9385	13.9379	66.877	21.1806	15.9291	31.2627	18.4131	9.94979	27.8643	18.7882	13.149	27.9006
10	5	66.2311	**50.7429	108.45	**56.9116	25.8689	87.5563	40.8596	5.96975	65.6672	26.9302	2.98488	44.7731
	10	57.8071	35.8185	81.5865	39.0355	7.95967	69.647	22.4861	2.98488	41.7883	14.6922	7.805E-09	37.8084
	25	42.1199	24.874	52.7328	19.5679	*1.98999	54.7227	6.67282	*0.0124656	21.8891	*2.65718	0.0004413	9.94959
	50	32.704	7.042	51.7378	*10.9171	4.09152	25.8689	*5.94003	2.09469	*11.0112	6.16692	3.10815	11.5746
	100	27.146	13.934	101.42	19.1923	12.524	27.0215	16.9496	10.9451	23.0326	15.7165	10.3689	22.1609
20	5	67.8893	45.7681	97.5058	55.0544	20.8941	**92.5311	42.3521	14.9244	78.6016	25.4709	*0	53.7277
	10	57.0443	33.8286	96.5109	40.2958	13.9294	65.6672	23.9785	6.96471	50.7429	13.2661	1.653E-11	32.8336
	25	47.4263	21.8891	78.6016	25.6699	4.9748	47.758	9.64867	0.995332	24.874	3.52663	0.0016266	19.8992
	50	34.0608	9.95223	55.7177	12.7026	3.62376	25.8689	7.89783	2.31294	13.9319	7.46382	1.01466	13.3077
	100	23.5982	12.9345	49.2153	15.351	7.08351	*24.8765	16.8059	8.04243	23.2785	14.9346	6.20837	21.3037
40	5	**73.3117	38.8034	95.5159	56.0825	32.8336	77.6066	41.7551	11.9395	**79.5966	23.1494	3.97984	40.7933
	10	56.7548	34.8235	78.6017	37.9742	10.9445	65.6672	27.1624	2.98488	46.763	14.0621	3.18E-13	33.8286
	25	42.6174	20.8941	66.6622	24.874	4.97685	50.7429	14.5931	2.98494	32.8336	8.36664	0.0001092	24.874
	50	29.9999	*6.5234	44.7731	16.9628	6.85684	43.7782	11.2774	5.09783	20.9167	9.03287	3.06012	17.9111
	100	*23.4617	13.9773	*35.8187	18.8111	9.06622	27.2796	16.5466	9.94965	24.9143	16.0269	9.01948	20.9458
PGKO > GKO		0	11	1	1	4	0	0	0	0	0	1	0

* PGKO'nun en başarılı çözümü.

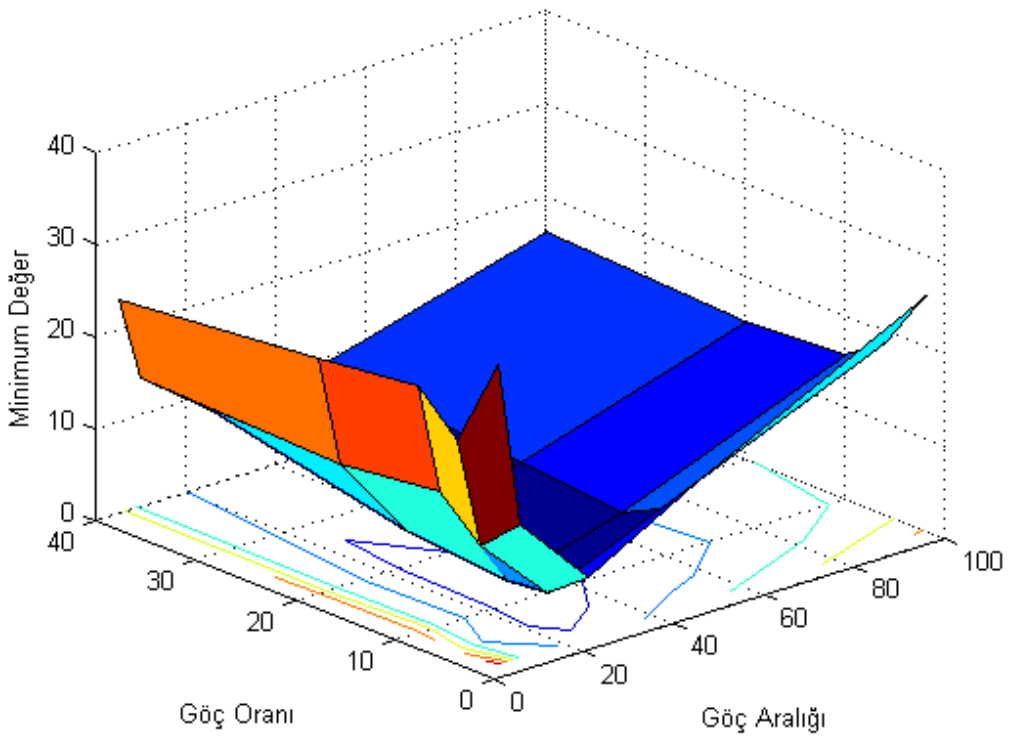
** PGKO'nun en başarısız çözümü.

Şekil 4.7'de, 32 alt popülasyonda Rastrigin fonksiyonunun GKO ve PGKO kullanılarak çözümünden elde edilen ortalama sonuç grafiği verilmektedir. Buna göre GKO'nun çok yavaş bir arama yaptığı ve iterasyon adımlarının son bölümünde daha iyi sonuçlar ürettiği görülmektedir. En başarılı PGKO (%10-25) grafiğinde ise algoritmanın arama fazında yavaş arama yaptığı, iyileştirme fazında ise detaylı bir arama yaparak iyi sonuçlar ürettiği görülmektedir. En başarısız PGKO (%2-5) grafiğinde, arama fazında çok sık göç yapıldığı için PGKO'nun hızlı bir arama gerçekleştirdiği, ancak iyileştirme fazında sonuçlarını geliştiremediği görülmektedir.



Şekil 4.7: Rastrigin fonksiyonu global minimuma yakınsama grafiği.

Şekil 4.8'de, 32 alt popülasyon kullanıldığında, üzerinde çalışılan tüm göç parametreleri için PGKO algoritmasından elde edilen en başarılı ortalama sonuçları verilmektedir. Buna göre, göç oranı %10, göç aralığı 25 olduğunda en başarılı sonuç elde edilmiştir. Verilen grafiğe göre göç aralığı çok azaldığında algoritma başarısının düştüğü görülmektedir. Bunun nedeni sık yapılan göç işleminden dolayı alt popülasyonların birbirine benzeşmesi, göç bireylerinin yeni popülasyonlara katkısını azaltmasıdır. Çok yüksek göç aralıklarında ise göç adımı daha az gerçekleştiği için göçün algoritma performansına olan katkısı azalmakta ve başarılı sonuçlar üretilememektedir.



Şekil 4.8: Rastrigin fonksiyonu göç parametreleri grafiği.

4.5 Schwefel Fonksiyonu İçin Sonuçlar

Farklı alt popülasyon sayısı, göç oranı ve göç aralıkları için Schewefel fonksiyonunun GKO ve PGKO algoritmaları ile çözümünden elde edilen sonuçlar Tablo 4.6'da verilmektedir. PGKO, en başarılı ortalama sonucunu alt popülasyon sayısı 8 olduğunda %10-100, 16 olduğunda %10-50, 24 olduğunda %20-50, 32 olduğunda %10-25 değerleri için üretmiştir. Alt popülasyon sayısı 8 olduğunda göç aralığının düşük olduğu değerlerde başarısız sonuçlar üretildiği görülmektedir. Bunun nedeni sık yapılan göç işleminden ve az sayıda alt popülasyondan dolayı alt popülasyonlardaki bireylerin hızlıca birbirine benzemeleridir. Alt popülasyonlar birbirine benzeşince arama uzayının farklı bölgeleri temsil edilememekte ve kaliteli bir arama yapılamamaktadır. En iyi sonuçların en başarılısına baktığımızda ise alt popülasyon sayısı 8 olduğunda %6-50, 16 ve 32 olduğunda %20-25 ve 24 olduğunda %6-25 değerleri için elde edildiği görülmektedir.

PGKO'nun ortalama sonuçlarının tümü incelendiğinde 100 durumdan sadece 11 durumda GKO'dan daha kötü sonuç ürettiği görülmektedir. Bunlardan 10 tanesi alt popülasyon sayısının 8 olduğu durumlarda, 1 tanesi de alt popülasyon sayısını 16 olduğu durumdadır. En iyi sonuçlara baktığımızda ise sadece 8 alt popülasyon için 4 durumda GKO, diğer durumlarda PGKO'nun daha başarılı olmaktadır. En kötü sonuçlar incelendiğinde PGKO'nun 31 durum için GKO'dan daha başarısız sonuç ürettiği görülmektedir. Schewefel fonksiyonundan elde edilen tüm sonuçlara göre, alt popülasyon sayısı arttıkça PGKO'nun başarımı artmaktadır.

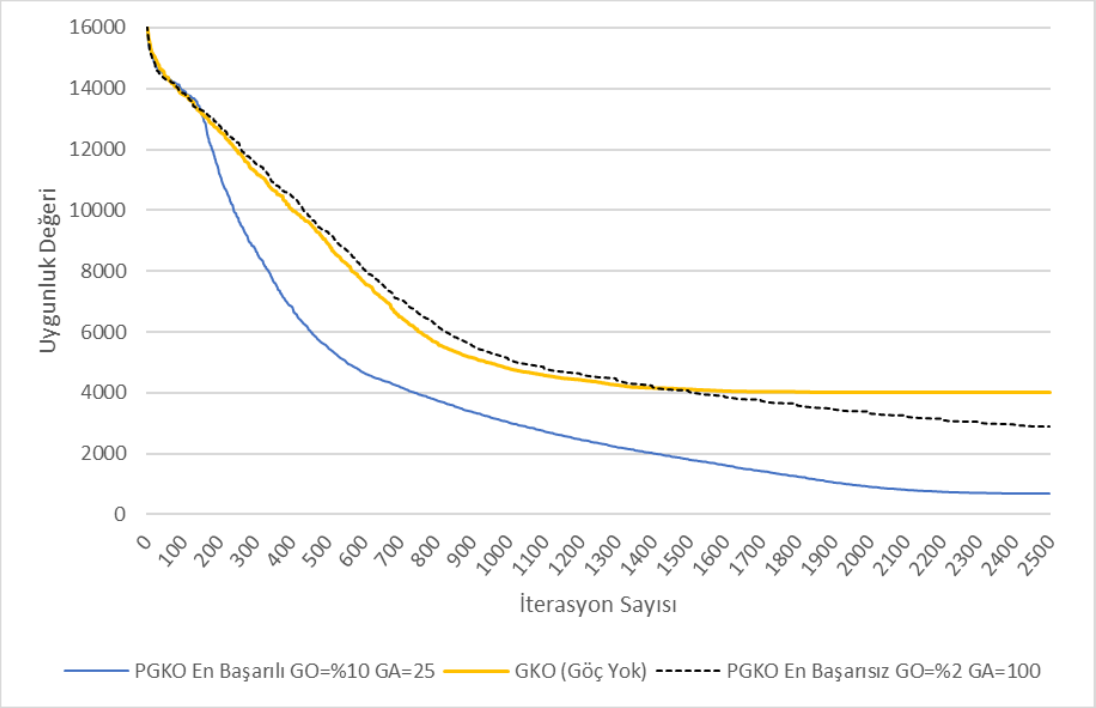
Tablo 4.6: Schwefel fonksiyonu için elde edilen sonuçlar.

Schwefel		Alt Popülasyon Sayısı											
		8			16			24			32		
Göç Oranı	Göç Aralığı	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü
Göç Yok		4544.07	3565.29	5354	4217.86	3565.29	4761.81	4032.07	3329.93	4757.26	4021.45	3326.89	4521.9
2	5	4911.48	**3923.64	5788.28	**4787.52	**3208.46	**5935.56	**3609.12	**2494.79	**5592.39	2824.34	830.586	**4163.55
	10	4573.93	3448.37	5472.44	3916.65	1188.94	5232.53	2756.77	830.586	4640.33	1895.16	592.192	3917.57
	25	4338.73	2851.62	6061.59	2601.59	1304.34	4165.06	1740.98	834.623	4280.47	1120.29	481.136	1779.61
	50	4242.83	2853.15	5829.27	2597.95	1544.28	3802.16	2122.45	1575.5	3088.5	2085.17	1541.85	2848.59
	100	3435.27	2255.18	5190.03	3224.61	2613.64	4512.79	2939.22	2018.01	3445.33	**2895.03	**2134.93	3679.55
6	5	4814.91	1996.75	6072.21	4114.05	1898.05	5475.47	2815.13	1185.9	5306.94	2247.59	592.192	3445.33
	10	4532.98	2734.7	5686.54	3790.83	2254.88	5829.27	1986.37	1068.98	3679.17	1803.85	593.709	3088.5
	25	4477.81	2139.48	6426.01	2533.56	1185.9	4752.7	1221.41	*236.878	2021.04	826.529	236.913	1661.17
	50	3827.47	*1781.15	5707.8	1815.06	1184.49	2848.59	1185.84	596.781	1667.22	1318.22	592.735	2134.98
	100	3550.61	2268.84	4989.58	2524.61	1779.48	3370.48	2459.69	1542.74	3332.49	2306.75	1422.2	2966.17
10	5	4690.55	3569.84	6257.5	3929.66	2371.8	5429.93	2942.58	1187.42	4997.17	2017.75	710.631	3805.2
	10	4798.86	3683.73	6172.43	3642.98	2018	5329.71	2195.61	593.709	3568.32	1586.21	592.192	2608.68
	25	4490.42	2968.54	5473.95	2319.31	1185.9	3922.12	1359.87	593.845	2371.8	*679.87	355.316	1782.64
	50	3768.6	2021.04	5354	*1441.89	713.536	*2490.24	1081.47	473.776	1659.83	849.619	241.91	*1421.41
	100	*3401.34	1898.74	*4755.74	2332.58	1541.31	3198.58	2148.44	1424.41	2724.32	2083.94	1421.29	2729.03
20	5	4869.98	3644.25	5943.18	3958.77	2368.77	5473.95	3432.53	2014.97	4500.64	2312.73	710.631	4040.56
	10	4676.58	3408.89	**6787.4	3449.84	1662.69	5476.99	2227.14	829.069	3805.2	1743.72	947.507	3088.5
	25	4448.26	3544.03	5592.39	2560.89	*473.754	4515.83	1419.52	475.277	3682.21	916.437	*118.578	2014.97
	50	3934.32	1784.16	4872.66	1799.43	599.548	3326.89	*1059.79	594.887	*1541.3	891.839	475.439	1787.09
	100	3477.22	1783.77	5114.09	2044.34	1421.31	2846.01	1941.82	1304.69	2610.24	1814.07	1066.19	2260.44
40	5	**4986.55	3220.63	6380.47	3592.77	2353.58	5709.31	2861.7	1304.34	4620.6	2449.95	592.192	3662.47
	10	4559.95	2971.58	6189.14	3578.14	2136.44	5232.53	2737.39	1188.94	4159	1735.73	592.192	3308.67
	25	4802.05	3328.41	6307.57	2920.97	1302.82	4632.75	1905.94	830.586	2851.62	1070.61	236.884	2371.8
	50	4448.15	2971.58	6545.97	2278.58	1189.52	3442.3	1692.78	948.453	2727.12	1430.57	711.887	1909.86
	100	3519.45	2251.87	4994.13	2274.72	1185.97	3085.59	2336.1	1778.16	2965.92	2173.94	1304.4	2725.79
PGKO > GKO		10	4	18	1	0	10	0	0	3	0	0	0

* PGKO'nun en başarılı çözümü.

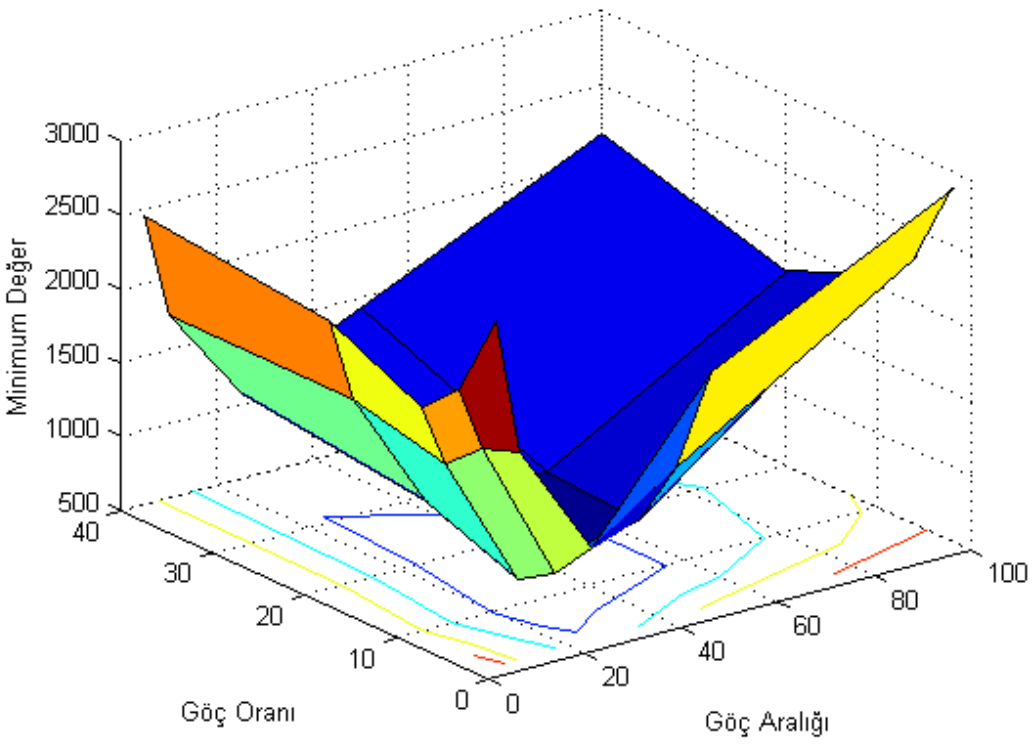
** PGKO'nun en başarısız çözümü.

Alt popülasyon sayısı 32 için ortalama sonuçlarından elde edilen GKO, en başarılı (%10-25) ve en başarısız (%2-100) PGKO grafikleri Şekil 4.9'da verilmektedir. Buna göre GKO çok yavaş bir arama yapmakta ve iyileştirme fazında aramaya devam edememektedir. En başarılı durum grafiğinde, algoritmanın arama fazında oldukça hızlı arama yaptığı, iyileştirme fazında ise detaylı bir arama yaparak iyi sonuçlar üretmeye devam ettiği görülmektedir. En başarısız durum grafiğinde de PGKO, GKO'ya benzer bir davranış göstermesine rağmen, iyileştirme fazında göç etkisiyle daha iyi sonuçlar üretmektedir.



Şekil 4.9: Schwefel fonksiyonu global minimuma yakınsama grafiği.

Şekil 4.10'da, alt popülasyon sayısı 32 için farklı göç parametreleri kullanılarak PGKO algoritmasının ürettiği en başarılı ortalama sonuçları grafiksel olarak verilmektedir. Buna göre algoritma, göç oranı %10, göç aralığı 25 olduğunda en başarılı sonucu vermektedir. Göç aralığı en iyi değer olan 25 değerinden 0'a yaklaştığında algoritma başarısının hızla kötüleştiği görülmektedir. Benzer şekilde 100'e doğru yaklaştığında da PGKO'nun başarısı azalmaktadır. Sık yapılan göç işleminde alt popülasyonların birbirine benzeşmesi, göç bireylerinin yeni popülasyonlara katkısını azaltmıştır. Çok yüksek göç aralıklarında ise göç adımı daha az gerçekleştiği için göçün algoritma performansına olan katkısı azalmakta ve başarılı sonuçlar üretilememektedir.



Şekil 4.10: Schwefel fonksiyonu göç parametreleri grafiği.

4.6 Styblinski-Tang Fonksiyonu İçin Sonuçlar

Styblinski-Tang fonksiyonundan elde edilen sonuçlar aşağıda verilen Tablo 4.7'de gösterilmektedir. Tablo incelendiğinde hem GKO hem de PGKO'nun en iyi sonuçlarının optimum noktayı genellikle bulduğu görülmektedir. Ortalama sonuçlara göre PGKO, alt popülasyon sayısı 8 ve 16 olduğunda 10 durum için, 24 olduğunda 5 durum için daha kötü sonuçlar üretmektedir. Alt popülasyon sayısı 32 olduğunda ise tüm durumlarda PGKO daha başarılı sonuçlar üretmektedir. En kötü sonuçlara bakıldığında ise alt popülasyon sayısı arttıkça PGKO'nun ürettiği başarısız sonuçların GKO'ya göre azaldığı görülmektedir.

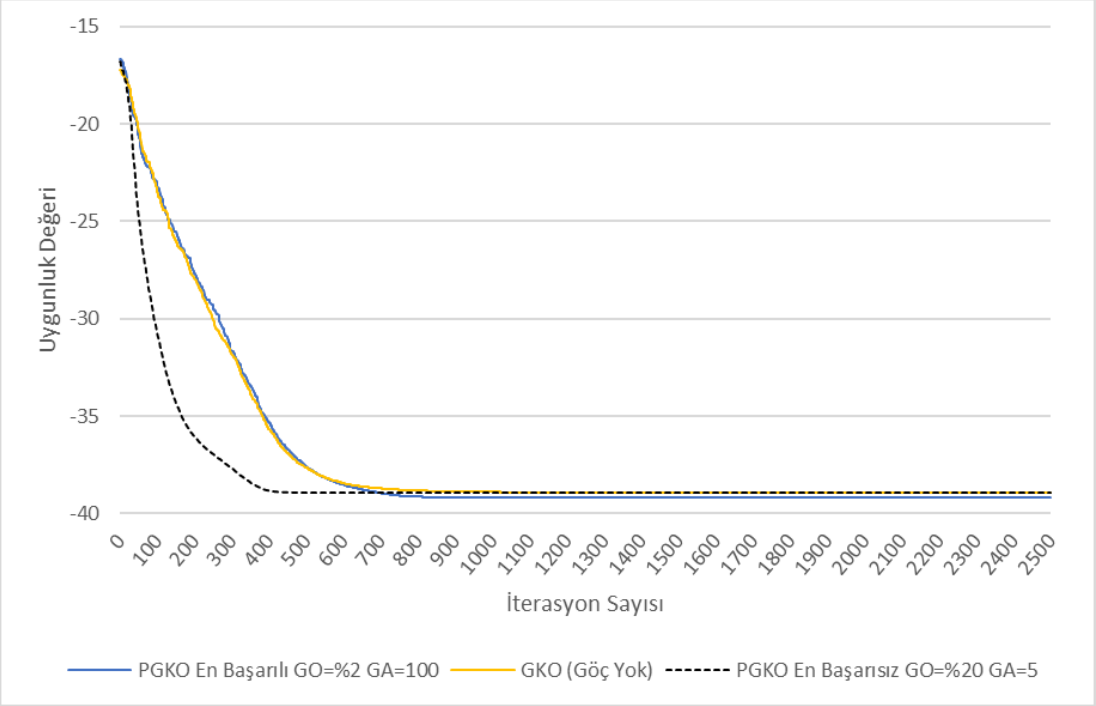
Tablo 4.7: Styblinski-Tang fonksiyonu için elde edilen sonuçlar.

Styblinski-Tang		Alt Popülasyon Sayısı											
		8			16			24			32		
Göç Oranı	Göç Aralığı	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü	Ortalama	En İyi	En Kötü
Göç Yok		-38.5913	-39.1662	-38.0352	-38.8175	-39.1662	-38.6007	-38.9965	-39.1662	-38.6007	-38.9117	-39.1662	-38.6007
2	5	-37.0551	-38.6007	-35.2079	-37.6394	*-39.1662	-36.0561	-38.6667	*-39.1662	-37.7525	-38.9494	*-39.1662	-38.318
	10	-37.8656	*-39.1662	-35.2079	-38.7515	*-39.1662	-36.9043	-39.1096	*-39.1662	-38.8834	*-39.1662	*-39.1662	*-39.1662
	25	-39.0248	*-39.1662	-38.318	-39.1473	*-39.1662	-38.8834	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662
	50	*-39.1662	*-39.1662	-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662
	100	*-39.1662	*-39.1662	-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662
6	5	-36.697	-38.6007	-34.6424	-37.8373	*-39.1662	**-35.7734	-38.5818	*-39.1662	**-36.9043	-38.9211	*-39.1662	**-37.7525
	10	-37.6017	*-39.1662	**-34.3597	-38.5065	*-39.1662	-37.187	-39.119	*-39.1662	-38.8834	*-39.1662	*-39.1662	*-39.1662
	25	-38.8457	*-39.1662	-38.0352	-39.1567	*-39.1662	-38.8834	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662
	50	-39.1473	*-39.1662	-38.8834	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662
	100	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662
10	5	-36.5462	-38.6007	-35.2079	**-37.6111	*-39.1662	-36.0561	-38.6478	*-39.1662	-37.187	-38.9494	*-39.1662	**-37.7525
	10	-37.5923	*-39.1662	-35.7734	-38.6384	*-39.1662	**-35.7734	-39.119	*-39.1662	-38.6007	-39.1473	*-39.1662	-38.8834
	25	-38.6761	*-39.1662	-36.9043	-39.1567	*-39.1662	-38.8834	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662
	50	-39.1379	*-39.1662	-38.6007	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662
	100	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662
20	5	-36.4708	-38.0352	**-34.3597	-37.7054	**-38.8834	-36.0561	**-38.4499	*-39.1662	-37.187	**-38.9117	*-39.1662	**-37.7525
	10	-37.0457	*-39.1662	-35.4906	-38.6949	*-39.1662	-37.4698	-39.0531	*-39.1662	-38.318	-39.1567	*-39.1662	-38.8834
	25	-38.7892	*-39.1662	-37.7525	-39.119	*-39.1662	-38.8834	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662
	50	-39.1379	*-39.1662	-38.8834	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662
	100	-39.1473	*-39.1662	-38.8834	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662
40	5	**-36.3859	**-37.7525	-34.6424	-37.6206	*-39.1662	**-35.7734	-38.4782	*-39.1662	**-36.9043	-38.9683	*-39.1662	-38.318
	10	-37.4792	*-39.1662	-35.4906	-38.629	*-39.1662	-37.4698	-39.0154	*-39.1662	-38.318	-39.1002	*-39.1662	-38.6007
	25	-38.6384	*-39.1662	-37.187	-39.1473	*-39.1662	-38.8834	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662
	50	-39.0248	*-39.1662	-38.6007	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662
	100	-39.1473	*-39.1662	-38.8834	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662	*-39.1662
PGKO > GKO		10	5	13	10	1	10	5	0	7	0	0	5

* PGKO'nun en başarılı çözümü.

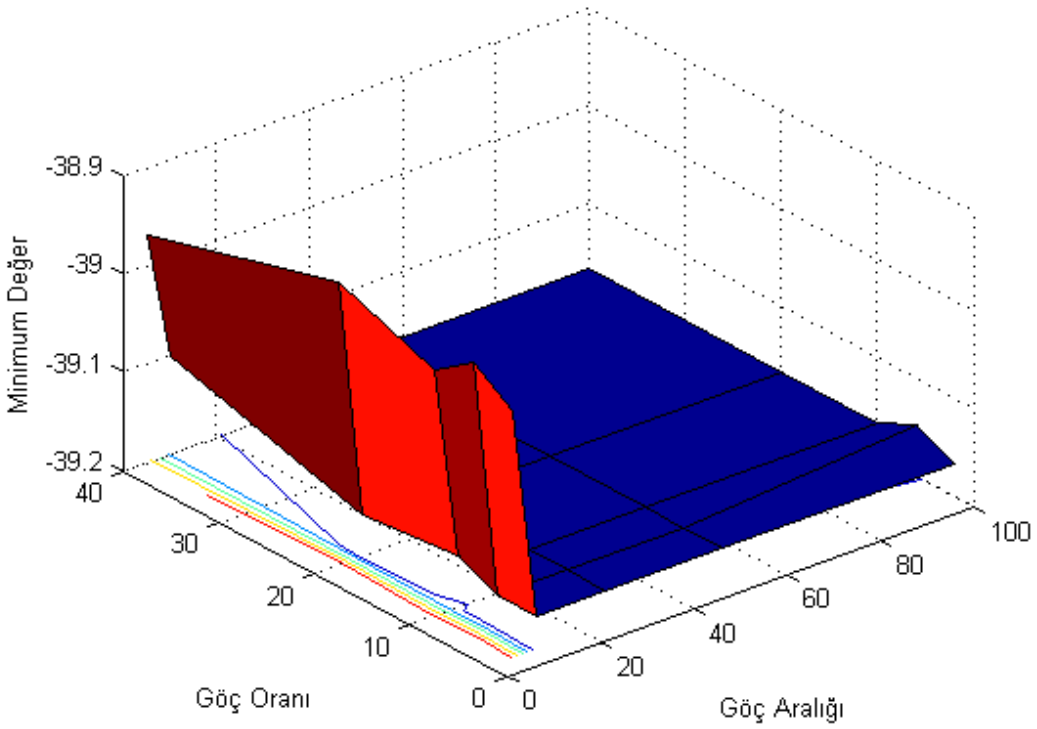
** PGKO'nun en başarısız çözümü.

Alt popülasyon sayısı 32 için ortalama sonuçlarından elde edilen GKO, en başarılı ve en başarısız PGKO grafikleri Şekil 4.11'de verilmiştir. Styblinski-Tang fonksiyonunda PGKO'nun en başarılı sonuçları %2 göç oranı ve 100 göç aralığı, en başarısız sonuçlar ise %20 göç oranı ve 5 göç aralığı değerleri için elde edilmiştir. En başarılı durum için birden fazla seçenek içinden optimum değere en erken ulaşan durum alınmıştır. PGKO en başarılı durumda GKO'ya benzer olarak arama fazında yavaş bir arama gerçekleşmesine rağmen optimum sonuç elde edilmiştir. En başarısız durumda ise arama fazında çok hızlı bir arama yapılmış, ancak yaklaşık 700. iterasyondan itibaren yerel optimuma takılarak optimum sonucu bulamamıştır. Arama fazında çok hızlı bir yakınsama gerçekleşmesinin nedeni düşük göç aralığı değeri (5) kullanılmasıdır.



Şekil 4.11: Styblinski-Tang fonksiyonu global minimuma yakınsama grafiği.

Aşağıda verilen Şekil 4.12'ye göre yüksek göç aralıklarında PGKO'nun optimum sonucu bulduğu, göç aralığı azaldığında ise daha kötü sonuçlar ürettiği görülmektedir. Göç oranı azaldığında optimum sonucun elde edildiği durumlar artarken göç oranı arttığında daha az durum için optimum sonuca ulaşılmıştır.



Şekil 4.12: Styblinski-Tang fonksiyonu göç parametreleri grafiği.

4.7 T-testi ve Elde Edilen Sonuçlar

Bağımsız iki grup sonucun anakütle ortalamaları arasında fark olup olmadığı ya da hangi grubun anakütle ortalamasının daha büyük olduğunun araştırılmasının istendiği durumlarda hipotez testleri gerçekleştirilmektedir. Örnek olarak satın alma davranışları konusunda kadın ve erkeklerin davranış farkları, örgütsel bağlılık açısından genç ve yaşlı çalışanların tutumlarının farkları inceleme konusu olabilir [42].

Çalışmanın bu bölümünde her bir alt popülasyon sayısı için ortalama ve en iyi sonuçların en başarılılarının elde edildiği göç parametreleri için PGKO ve GKO gruplarının durumlarına göre anakütle ortalamaları arasında fark olup olmadığının belirlenebilmesi için t-testi yapılmıştır.

T-testine geçmeden önce varyansların eşit olup olmadığının belirlenmesi gerekir. Bunun için öncelikle F-testi yapılmış, %5 önem düzeyine göre varyansların eşitliği kontrol edilmiştir. Eşit varyansa sahip olan gruplar için iki örnekli eşit varyans,

eşit varyansa sahip olmayan gruplar için iki örnekli eşit olmayan varyans hesabı kullanılarak t-testleri gerçekleştirilmiştir [43,44]. T-testi sonucunda elde edilen p değeri 0,05'ten küçük olduğunda %5 anlamlılık düzeyi için PGKO ile yapılan optimizasyonun GKO sonuçlarına göre anlamlı düzeyde başarılı sonuçlar üretebildiği gösterilmiştir. Aşağıda verilen Tablo 4.8 incelendiğinde GKO ile PGKO grupları arasında yapılan t-testi sonuçlarında sıfıra çok yakın değerler elde edildiği görülmektedir. GKO ve PGKO algoritmalarından elde edilen sonuçlar anlamlıdır. T-testi sonuçları göç işlemi sayesinde elde edilen sonuçların GKO sonuçlarını önemli derecede iyileştirdiğini, göç işlemin algoritma performansına önemli derecede katkı sağladığını göstermektedir.

Tablo 4.8: T-testi sonuçları.

T-testi Sonuçları		Alt Popülasyon Sayısı							
		8		16		24		32	
		Ortalama	En İyi	Ortalama	En İyi	Ortalama	En İyi	Ortalama	En İyi
Rastrigin	Göç Oranı (%) - Göç Aralığı	40-100	40-50	10-50	10-25	10-50	10-25	10-25	20-5
	t-testi (p)	4.85825E-07	3.48331E-06	1.98971E-06	5.0511E-05	1.21211E-08	1.67075E-08	2.19775E-08	0.007707741
Rosenbrock	Göç Oranı (%) - Göç Aralığı	10-5	40-5	20-5	40-10	20-5	10-5	10-5	6-5
	t-testi (p)	3.00548E-22	2.12217E-12	4.80867E-10	8.66316E-05	3.87773E-07	5.51972E-07	6.91624E-27	2.76263E-25
Zakharov	Göç Oranı (%) - Göç Aralığı	10-10	40-5	40-5		40-5		40-5	
	t-testi (p)	2.58191E-27	1.20487E-37	5.66365E-27		2.98526E-30		3.60087E-33	
Schwefel	Göç Oranı (%) - Göç Aralığı	10-100	6-50	10-50	20-25	20-50	6-25	10-25	20-25
	t-testi (p)	4.52282E-11	2.53768E-05	2.54409E-33	1.36872E-11	1.51109E-36	3.22888E-33	6.49824E-43	1.57795E-38
Rotated Hyper-Ellipsoid	Göç Oranı (%) - Göç Aralığı	40-5		40-5		40-5		40-5	
	t-testi (p)	5.94925E-09		1.18953E-11		4.52776E-12		7.63684E-15	

5. SONUÇ VE ÖNERİLER

Paralel metasezgisel algoritmaların başarılı sonuçlar üretebilmesini sağlayan en önemli işlem adımı olan göç, başarılı çözümlerin belirlenen aralık ve oranda tercih edilen topolojiye göre alt popülasyonlar arasında paylaşılması işlemidir. Göç, algoritma performansını etkileyen farklı parametreler içermektedir. Göç oranı ve göç aralığı bunların içerisinde önemli parametrelerdir. Doğru parametrelerin belirlenmesi konusunda yapılan çalışmalar paralel metasezgisel algoritmaların daha başarılı sonuçlar üretebilmesi için önemlidir.

Bu tez çalışmasında metasezgisel bir optimizasyon algoritması olan GKO algoritması, çok sayıda alt popülasyon kullanılarak paralel bilgisayarlarda uygulanmış ve PGKO algoritması geliştirilmiştir. Yapılan çalışmada alt popülasyon sayısı 8, 16, 24 ve 32, göç oranı %2, %6, %10, %20 ve %40, göç aralığı 5, 10, 25, 50 ve 100 değerleri kullanılarak tek modlu Rosenbrock, Rotated Hyper-Ellipsoid ve Zakharov, çok modlu Rastrigin, Schwefel ve Styblinski-Tang fonksiyonları olmak üzere altı test fonksiyonu 50 boyut için PGKO algoritması ile çözülmüştür. PGKO algoritmasında halka topoloji kullanılmıştır. Bunun yanında göç işleminin kullanılmadığı GKO algoritması aynı alt popülasyon sayıları için uygulanmıştır. Her farklı durum için 30 bağımsız denemeden elde edilen ortalama, en iyi ve en kötü sonuçlar verilmiştir.

Tek modlu fonksiyonlardan Rosenbrock fonksiyonu 8, 16, 24 ve 32 alt popülasyon sayılarında ortalama sonuçların en başarılılarını sırasıyla %10-5, %20-5, %20-5 ve %10-5 değerlerinde üretmektedir. En iyi sonuçlarda ise en başarılı değerleri %40-5, %40-10, %10-5 ve %6-5 değerleri için elde etmektedir. Bütün sonuçlar incelendiğinde Rosenbrock fonksiyonunun çözümünde düşük göç aralıkları için daha başarılı sonuçlar üretildiği görülmektedir. Algoritmanın genel davranışını gösteren ortalama sonuçlarına göre %10-%20 aralığında seçilecek göç oranı ve düşük göç aralıklarının tercih edilmesi, algoritmanın davranışında genel bir iyileştirme sağlamaktadır. Tüm çalışma incelendiğinde 100 farklı ortalama sonucunun içinde 26

farklı durumda GKO, 74 farklı durumda PGKO daha başarılı sonuçlar üretmiştir. En iyi sonuçlara göre ise 70 durumda GKO, 30 durumda ise PGKO daha başarılıdır. Bu sonuçlara göre, göç parametrelerinin Rosenbrock fonksiyonu üzerindeki etkisi oldukça fazladır. Doğru göç parametreleri seçildiğinde PGKO çok daha başarılı sonuçlar üretebilmektedir.

Rotated Hyper-Ellipsoid fonksiyonundan elde edilen sonuçlara göre PGKO algoritması, tüm alt popülasyon sayıları için en başarılı sonuçları %40-5 parametreleri için üretmektedir. Ortalama sonuçları için 100 farklı durumun tamamında PGKO, GKO'dan daha başarılı sonuçlar üretmiştir. En iyi sonuçlara göre ise PGKO 98, GKO 2 farklı durumda daha başarılı sonuçlar üretmiştir. Bu sonuçlara göre Rotated Hyper-Ellipsoid fonksiyonunun çözümünde göç işleminin PGKO algoritmasının performansına katkısı açıkça görülmektedir.

Zakharov fonksiyonu sonuçlarında ise bütün test durumları için PGKO, GKO'dan daha başarılı sonuçlar üretmiştir. Alt popülasyon sayısı 8 için ortalama sonuçların en başarılısı %10-10 değerleri için elde edilmiştir. Bunun dışındaki bütün durumlarda en başarılı sonucun %40-5 parametre değerleri için üretilmiştir.

Tek modlu fonksiyonlarda yerel optimumların bulunmaması nedeniyle göç bireyleri, hedef alt popülasyondaki diğer çözümleri arama uzayının daha iyi bir bölgesine yaklaştırmaktadır. Bu nedenle sık yapılan göç işlemi algoritma performansını arttırmaktadır. Bunun bir sonucu olarak PGKO'nun düşük göç aralıkları için daha başarılı sonuçlar ürettiği görülmektedir. Rotated Hyper-Ellipsoid ve Zakharov fonksiyonları birbirine çok benzeyen bir davranış göstermektedir. Rosenbrock fonksiyonu için ise doğru göç parametrelerinin seçimi oldukça zordur.

Çok modlu Rastrigin fonksiyonunda ortalama sonuçların en başarılıları alt popülasyon sırasına göre %40-100, %10-50, %10-50 ve %10-25 değerleri için üretilmiştir. En iyi sonuçların en başarılıları incelendiğinde ise %40-50, %10-25, %10-25 ve %20-5 değerleri için elde edildiği görülmektedir. PGKO, ortalama sonuçları için 100 farklı durumda 99, en iyi sonuçlar için 84 durumda GKO'dan daha başarılı sonuçlar üretmiştir.

Schwefel fonksiyonu sonuçlarına bakıldığında ise alt popülasyon sırasına göre %10-100, %10-50, %20-50 ve %10-25 parametre değerleri için ortalama sonuçların en başarılıları üretilmiştir. En iyi sonuçların en başarılıları ise sırasıyla %6-50, %20-25, %6-25 ve %20-25 değerleri için elde edilmiştir. Ortalama sonuçlarına göre PGKO 89, GKO 11 durumda daha başarılı sonuçlar üretmiştir. En iyi sonuçlar incelendiğinde ise PGKO 96, GKO 4 durumda daha başarılıdır.

Styblinski-Tang fonksiyonu için yapılan çalışmalar incelendiğinde GKO algoritması en iyi sonuçların tamamında fonksiyonun optimum değerine ulaşmıştır. PGKO algoritması ise sırasıyla 20, 24, 25 ve 25 durum için optimum değeri bulmuştur. Ortalama sonuçları incelendiğinde ise PGKO alt popülasyon sayısı 8 için 4, 16 için 10, 24 için 15 ve 32 için 17 durumda optimum değeri üretmiştir. Ortalama sonuçlarında optimum değeri üretmek 30 bağımsız denemenin tamamında optimum sonucun elde edildiği anlamına gelmektedir. GKO algoritması, ortalama sonuçlarında hiçbir alt popülasyon sayısı için optimum sonuca ulaşamamıştır.

Çok modlu fonksiyonların sonuçları birlikte incelendiğinde göç aralığı çok azaldığında algoritma performansının hızla kötüleştiği görülmektedir. Bunun nedeni, çözümlerin hızla yer değiştirmesi sonucu benzer çözümlerin alt popülasyonlarda artmasıdır. Alt popülasyonlar arama uzayının aynı bölgelerine yoğunlaşmakta, farklılık azaldığı için yeni çözümler üretilememekte ve yerel optimumlara takılma riski artmaktadır. Özellikle alt popülasyon sayısı ve göç aralığı düşük olduğunda bu etki daha fazla görülmekte, PGKO algoritmasının başarısı daha da kötüleşmektedir.

Aşağıda verilen Tablo 5.1'de, beş farklı test fonksiyonu için dört farklı alt popülasyonda elde edilen en başarılı sonuçlara ait göç parametreleri verilmiştir. Styblinski-Tang fonksiyonu, birçok farklı parametrede optimum sonuca ulaştığından tabloya dahil edilmemiştir.

Tablo 5.1: En başarılı sonuçların elde edildiği göç parametreleri.

	Alt Popülasyon Sayısı							
	8	16	24	32	8	16	24	32
	Ortalama				En iyi			
Rosenbrock	%10-5	%20-5	%20-5	%10-5	%40-5	%40-10	%10-5	%6-5
Rotated H-E	%40-5	%40-5	%40-5	%40-5	%40-5	%40-5	%40-5	%40-5
Zakharov	%10-10	%40-5	%40-5	%40-5	%40-5	%40-5	%40-5	%40-5
Rastrigin	%40-100	%10-50	%10-50	%10-25	%40-50	%10-25	%10-25	%20-5
Schwefel	%10-100	%10-50	%20-50	%10-25	%6-50	%20-25	%6-25	%20-25

Yapılan çalışmanın devamında algoritmanın iterasyon boyunca davranışı incelenmiştir. Bu amaçla alt popülasyon sayısı 32 için ortalama sonuçlarına göre PGKO'nun en başarılı olduğu, en başarısız olduğu parametrelerde üretilen sonuçlar ve GKO algoritmasının sonuçları 2500 iterasyon için grafiksel olarak verilmiştir. Rosenbrock fonksiyonunun grafiğine göre en başarılı (%10-5) PGKO diğer durumlara göre açık şekilde daha iyi bir arama yapmaktadır. En başarısız (%40-100) PGKO, GKO'ya göre daha yavaş bir arama gerçekleştirirken iterasyon sonunda yaklaşık olarak aynı sonuca ulaşmıştır. Rotated Hyper-Ellipsoid fonksiyonu incelendiğinde en başarılı (%40-5) PGKO'nun, en başarısız (%2-100) PGKO ve GKO'ya göre çok erken iterasyon adımlarında başarılı sonuçlar ürettiği görülmektedir. Zakharov fonksiyonu grafiğine göre ise en başarılı (%40-5) PGKO diğer durumlara göre çok daha iyi bir arama gerçekleştirmiştir. En başarısız (%2-100) PGKO'da GKO'ya göre daha iyi bir arama gerçekleştirmiş ve daha başarılı bir sonuç üretmiştir. Ancak her iki durumda da en başarılı PGKO'ya göre çok yetersiz bir sonuç üretilmiştir. Rastrigin fonksiyonunda PGKO, GKO'dan çok daha başarılı bir arama gerçekleştirmektedir. En başarılı (%10-25) PGKO, en başarısız (%2-5) PGKO'ya göre daha yavaş bir arama yapmasına rağmen ilerleyen iterasyon adımlarında daha başarılı sonuçlar üretmiştir. Bu sonuca göre göç aralığı düşük olduğunda arama fazında hızlı bir arama gerçekleştiği, ancak iyileştirme fazında başarılı sonuçlar üretilemediği görülmektedir. Schwefel fonksiyonu çözümünde en başarılı (%10-25) PGKO diğer durumlardan çok daha başarılı bir arama gerçekleştirerek iterasyon sonunda çok daha iyi bir sonuç üretmiştir. Styblinski-Tang fonksiyonu ise hem GKO hem de PGKO'nun optimum sonucu bulabildiği bir fonksiyondur. Elde edilen sonuçlara göre göç aralığı düşük olduğu için

en başarısız (%20-5) PGKO'nun arama fazında hızlı bir arama gerçekleştirdiği görülmektedir. En başarılı (%2-100) PGKO ise yaklaşık 700. iterasyonda diğer yöntemleri yakalayarak aramasını devam ettirmiş ve iterasyon sonunda daha başarılı bir sonuç üretmiştir.

Göç işleminin algoritma performansına yaptığı katkıyı incelemek amacıyla her alt popülasyon sayısı için en başarılı ortalama ve en iyi sonuçların alındığı durumda PGKO ile GKO algoritmalarının sonuçlarının farklılığı t-testi ile test edilmiştir. T-testi öncesinden eşit veya farklı varyans durumlarını incelemek amacıyla F-testi yapılmıştır. Elde edilen sonuçlara göre tüm durumlarda çok küçük değerler elde edildiği görülmüştür. Bu sonuçlar göç işleminin algoritma performansına yaptığı katkıyı göstermektedir.

Bu tez çalışmasında PGKO algoritmasında yapılan göç işleminin algoritma performansına olan katkısı, çok sayıda farklı durum ve problem için gösterilmiştir. Elde edilen sonuçlardan görüldüğü gibi göç parametrelerinin doğru seçimi, algoritma performansını önemli derecede etkilemektedir. Bu nedenle göç parametrelerinin belirlenmesi konusunda yapılacak çalışmalar, daha zor problemlerin çözümü için önemli bir altyapı oluşturacaktır.

6. KAYNAKLAR

[1] Makas H., "Güncel en iyileme algoritmalarının paralel ve birlikte uygulamaları ve performans analizleri", Doktora Tezi, *Sakarya Üniversitesi Fen Bilimleri Enstitüsü*, Bilgisayar ve Bilişim Mühendisliği Anabilim Dalı, Sakarya, (2015).

[2] Akay R., "Paralel Programlama Tekniklerinin Gelişime Dayalı Algoritmalar Üzerindeki Etkinlik Analizi", Doktora Tezi, *Erciyes Üniversitesi Fen Bilimleri Enstitüsü*, Bilgisayar Mühendisliği Anabilim Dalı, Kayseri, (2014).

[3] Baştürk A. and Akay R., "Performance Analysis of the Coarse-Grained Parallel Model of the Artificial Bee Colony Algorithm", *Information Sciences.,* 253, 34-55, (2013).

[4] Asadzadeh L., "A parallel artificial bee colony algorithm for the job shop scheduling problem with a dynamic migration strategy", *Computers & Industrial Engineering.,* 102, 359-367, (2016).

[5] Gülcü Ş. and Kodaz H., "A novel parallel multi-swarm algorithm based on comprehensive learning particle swarm optimization", *Engineering Applications of Artificial Intelligence.,* 45, 33-45, (2015).

[6] Tu K.-Y. and Liang Z.-C., "Parallel computation models of particle swarm optimization implemented by multiple threads", *Expert Systems with Applications.,* 38(5), 5858-5866, (2011).

[7] Alba E., Nebro A. J. and Troya J. M., "Heterogeneous Computing and Parallel Genetic Algorithms", *Journal of Parallel and Distributed Computing.,* 62(9) 1362-1385, (2002).

[8] Doorly D. J. and Peiró J., "Supervised Parallel Genetic Algorithms in Aerodynamic Optimisation", *Artificial Neural Nets and Genetic Algorithms*, Vienna, (1998).

[9] Mühlenbein H., Schomisch M. ve Born J., "The parallel genetic algorithm as function optimizer", *Parallel Computing.,* 17(6-7) 619-632, (1991).

[10] Randall M. and Lewis A., "A Parallel Implementation of Ant Colony Optimization", *Journal of Parallel and Distributed Computing.,* 62(9), 1421-1432, (2002).

[11] Wang H., Rahnamayan S. and Wu Z., "Parallel differential evolution with self-adapting control parameters and generalized opposition-based learning for solving high-dimensional optimization problems", *Journal of Parallel and Distributed Computing.,* 73, 62-73, (2013).

[12] Xiaofang Y., Ting Z., Yongzhong X. and Xiangshan D., "Parallel chaos optimization algorithm with migration and merging operation", *Applied Soft Computing.*, 35, 591-604, (2015).

[13] Kuvat G. ve Tülek A., "Göçmen Kuşlar Optimizasyon Algoritmasının Paralel Bilgisayarlarda Uygulanması", *5. Ulusal Yüksek Başarımlı Hesaplama Konferansı*, İstanbul, (2017).

[14] Alba E. and Troya J. M., "Improving flexibility and efficiency by adding parallelism to genetic algorithms", *Statistics and Computing.*, 12(2), 91-114, (2002).

[15] Rebaudengo M. and Reorda M. S., "An experimental analysis of the effects of migration in parallel genetic algorithms", *1993 Euromicro Workshop on Parallel and Distributed Processing*, Gran Canaria, (1993).

[16] Omkar S. N., Venkatesh A. and Mudigere M., "MPI-based parallel synchronous vector evaluated particle swarm optimization for multi-objective design optimization of composite structures", *Engineering Applications of Artificial Intelligence.*, 25(8), 1611-1627, (2012).

[17] Duman E., Uysal M. and Alkaya A. F., "Migrating Birds Optimization: A new metaheuristic approach and its performance on quadratic assignment problem", *Information Sciences.*, 217, 65-77, (2012).

[18] Lissaman P. B. and Shollenberger C. A., "Formation Flight of Birds", *Science.*, 168, 1003-1005, (1970).

[19] Karaboğa D., *Yapay Zeka Optimizasyon Algoritmaları*, Kayseri: Nobel Akademik Yayıncılık, 207-208, (2014).

[20] Öz D., "Göç Eden Kuşlar Algoritmasinda Kaos Fonksiyonlarinin Kullanilmasi," *Gazi Üniversitesi Fen Bilimleri Dergisi.*, 225-233, (2016).

[21] Tongur V. ve Ülker E., "Tek Boyutlu Kesme Probleminin Göçmen Kuşlar Optimizasyon Algoritması ile Çözümü", *Uluslararası Bilgisayar Bilimleri ve Mühendisliği Konferansı*, Tekirdağ, (2016).

[22] Tongur V. and Ülker E., "The Analysis of Migrating Birds Optimization Algorithm with Neighborhood Operator on Traveling Salesman Problem", *Intelligent and Evolutionary Systems.*, 5, 227-237, (2015).

[23] Makas H. and Yumuşak N., "New cooperative and modified variants of the migrating birds optimization algorithm", *Electronics, Computer and Computation (ICECCO), 2013 International Conference*, Ankara, (2013).

[24] Gao L. and Pan Q.-K., "A shuffled multi-swarm micro-migrating birds optimizer for a multi-resource-constrained flexible job shop scheduling problem", *Information Sciences.,* 372, 655-676, (2016).

[25] Zhang B., Pan Q.-K., Gao L., Zhang X.-L., Sang H.-Y. and Li J.-Q., "An effective modified migrating birds optimization for hybrid flowshop scheduling problem with lot streaming", *Applied Soft Computing.*, 52, 14-27, (2017).

[26] Gao K., Suganthan P. N. and Chua T. J., "An enhanced migrating birds optimization algorithm for no-wait flow shop scheduling problem", *2013 IEEE Symposium on Computational Intelligence in Scheduling (CISched),* Singapore, (2013).

[27] Pan Q.-K. and Dong Y., "An improved migrating birds optimisation for a hybrid flowshop scheduling with total flowtime minimisation", *Information Sciences.*, 277, 643-655, (2014).

[28] Ramanathan L. and Ulaganathan K., "Nature-inspired Metaheuristic Optimization Technique-Migrating bird's optimization in Industrial Scheduling Problem", *SSRG International Journal of Industrial Engineering (IJIE).,* 1(3) 1-6, (2014).

[29] Benkalai I., Rebaine D., Gagné C. and Baptiste P., "The migrating birds optimization metaheuristic for the permutation flow shop with sequence dependent setup times", *IFAC-PapersOnLine.*, 49(12), 408-413, (2016).

[30] Tongur V. and Ülker E., "Migrating Birds Optimization for Flow Shop Sequencing Problem", *Journal of Computer and Communications.,* 02(04), 142-147, (2014).

[31] Niroomand S., Hadi-Vencheh A., Şahin R. and Vizvári B., "Modified migrating birds optimization algorithm for closed loop layout with exact distances in flexible manufacturing systems", *Expert Systems with Applications.,* 42(19), 6586-6597, (2015).

[32] Öz D., "An improvement on the Migrating Birds Optimization with a problem-specific neighboring function for the multi-objective task allocation problem", *Expert Systems with Applications.,* 67, 304-311, (2017).

[33] Makas H. and Yumuşak N., "System identi cation by using migrating birds optimization algorithm: acomparative performance analysis", *Turkish Journal of Electrical Engineering & Computer Sciences.,* 24, 1879-1900, (2016).

[34] Makas H. and Yumuşak N., "Balancing exploration and exploitation by using sequential execution cooperation between artificial bee colony and migrating birds optimization algorithms", *Turkish Journal of Electrical Engineering and Computer Sciences.*, 24(6), 4935-4956, (2016).

[35] Pacheco P. S., *Parallel Programing with MPI*, San Francisco: Morgan Kaufmann Publishers, Inc., (1997).

[36] "TRUBA2018-Kitapcik.pdf [online]", (20.12.2018), http://www.truba.gov.tr/wp-content/uploads/2018/10/TRUBA2018-Kitapcik.pdf., (2018).

[37] Rosenbrock H. H., "An automatic method for finding the greatest or least value of a function", *The Computer Journa.l*, 3(3), 175-184, (1960).

[38] Rahnamayan S., Tizhoosh H. R. and Salama M. M., "A novel population initialization method for accelerating evolutionary algorithms",» *Computers & Mathematics with Applications.*, 53(10), 1605-1615, (2007).

[39] Dixon L. C. and Szego G. P., "The global optimization problem: an introduction", *Towards global optimization.*, 2, 1-15, (1978).

[40] Surjanovic S. and Bingham D., "Optimization Test Problems [online]", (01.02.2017), https://www.sfu.ca/~ssurjano/schwef.html., (2016).

[41] Zhang Y., Park C., Kim N. H.and Haftka R., "Function Prediction at One Inaccessible Point Using Converging Lines", *Journal of Mechanical Design.*, 139(5), 1-11, (2017).

[42] Akar C., *İş Analitiği, Excel Uygulamalı Yönetsel Karar Verme ve Veri Analizi*, Bursa: Dora Basım, 194, (2018).

[43] Bayram N., *Veri Analizi Excel ve SPSS Uygulamaları ile Birlikte*, Ankara: Siyasal Kitabevi, 150, (2012).

[44] Tekin V. N., *SPSS Uygulamalı İstatistik Teknikleri*, Ankara: Seçkin Yayınları, 67, (2009).