T.C.
ERCİYES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

NÜMERİK OPTİMİZASYON PROBLEMLERİNDE YAPAY ARI KOLONİSİ (ARTIFICIAL BEE COLONY) ALGORİTMASININ PERFORMANS ANALİZİ

Tezi Hazırlayan
Bahriye AKAY

Tezi Yöneten
Prof. Dr. Derviş KARABOĞA

Bilgisayar Mühendisliği Anabilim Dalı
Doktora Tezi

Kasım 2009
KAYSERİ

T.C.
ERCİYES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

NÜMERİK OPTİMİZASYON PROBLEMLERİNDE YAPAY ARI KOLONİSİ (ARTIFICIAL BEE COLONY) ALGORİTMASININ PERFORMANS ANALİZİ

Tezi Hazırlayan
Bahriye AKAY

Tezi Yöneten
Prof. Dr. Derviş KARABOĞA

Bilgisayar Mühendisliği Anabilim Dalı
Doktora Tezi

Bu çalışma Erciyes Üniversitesi Bilimsel Araştırma Projeleri Birimi tarafından FBA-06-22 kodlu proje ile desteklenmiştir

Kasım 2009
KAYSERİ

Prof. Dr. Derviş KARABOĞA danışmanlığında **Bahriye AKAY** tarafından hazırlanan **"Nümerik Optimizasyon Problemlerinde Yapay Arı Kolonisi (Artificial Bee Colony) Algoritmasının Performans Analizi"** adlı bu çalışma, jürimiz tarafından Erciyes Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalında **Doktora** tezi olarak kabul edilmiştir.

04/11/2009

JÜRİ :

Başkan : Prof. Dr. Ahmet ARSLAN

Üye : Prof. Dr. Derviş KARABOĞA

Üye : Doç. Dr. Coşkun ÖZKAN

Üye : Yrd. Doç. Dr. Veysel ASLANTAŞ

Üye : Yrd. Doç. Dr. Mete Çelik

ONAY :

Bu tezin kabülü Enstitü Yönetim Kurulunun 10/11/2009 tarih ve 2009/39-02 sayılı kararı ile onaylanmıştır.

10 / 11 / 2009

Prof. Dr. Nusret AYYILDIZ

Enstitü Müdürü

TEŞEKKÜR

"Nümerik Optimizasyon Problemlerinde Yapay Arı Kolonisi (Artificial Bee Colony) Algoritmasının Performans Analizi" konulu tez çalışmamın yürütülmesinde, sonuçlandırılmasında ve sonuçlarının değerlendirilmesinde bilgi, tecrübe ve yardımlarını esirgemeyen değerli hocam sayın Prof. Dr. Derviş Karaboğa'ya ve tez izleme komitemde yer alan hocalarım Prof. Dr. Ahmet Arslan'a, Doç. Dr. Coşkun Özkan'a ve Yrd. Doç. Dr. Veysel Aslantaş'a, Yrd. Doç. Dr. Mete Çelik'e saygı ve teşekkürlerimi sunarım.

Hayatım boyunca verdikleri tüm desteklerden ötürü haklarını ödeyemeceğim aileme, anlayış ve desteğinden dolayı eşim Rüştü Akay'a, sağladığı motivasyondan ötürü oğlum Ali Batu'ya teşekkür ederim.

Erciyes Üniversitesi Bilimsel Araştırma Projeleri Birimi'ne FBA-06-22 kodlu proje desteğinden dolayı teşekkürlerimi sunarım.

NÜMERİK OPTİMİZASYON PROBLEMLERİNDE YAPAY ARI KOLONİSİ (ARTIFICIAL BEE COLONY) ALGORİTMASININ PERFORMANS ANALİZİ

Bahriye AKAY
Erciyes Üniversitesi, Fen Bilimleri Enstitüsü
Doktora Tezi, Kasım 2009
Tez Danışmanı : Prof. Dr. Derviş KARABOĞA

ÖZET

Optimizasyon alanında, belirli problem türlerinde çok iyi performans sergileyen tekniklerin geliştirilmesi yerine, genel olarak çoğu problem türünde iyi performans sergileyen algoritmaların geliştirilmesi oldukça önemlidir. Optimizasyon algoritmalarının bu karakteristiğe sahip olup olmadığının ortaya konması için performanslarının analiz edilmesi ve literatürdeki bilinen algoritmalarla kıyaslanarak davranışlarının incelenmesi gerekmektedir. Bu tez çalışmasında, son zamanlarda literatüre kazandırılmış, arıların yiyecek arama davranışını modelleyen, kısa sürede oldukça popüler olan yapay arı koloni (ABC) algoritmasının detaylı performans analizi yapılmıştır. ABC algoritması, tam sayı programlama problemlerinde kullanılacak şekilde geliştirilerek, literatürdeki algoritmalarla performansı kıyaslanmıştır. Sınırlamasız karma problemlerde algoritmanın performansını artırmak amacıyla algoritma yapısında değişiklikler yapılmıştır. Araştırma uzayının bazı eşitlik ve eşitsizliklerle sınırlandığı ve optimum çözümün kabul edilebilir bölge içinde olması gerekliliğini taşıyan sınırlamalı test problemlerini çözmek için ABC algoritmasının yeni versiyonu önerilmiştir. Optimizasyon literatüründe bilinen test fonksiyonlarının yanı sıra, ABC algoritması, bir gerçek dünya problemi olan endüstriyel süreçlerin denetiminde yaygın olarak kullanılan PID denetleyicilerin tasarımında kullanılmıştır. Bunlara ilave olarak genellikle doğrusal olmayan karakteristiğe sahip, sınırlamalı bazı makine mühendisliği tasarım problemlerinin çözümü için ABC algoritması kullanılmış ve oldukça başarılı sonuçlar elde edilmiştir.

Anahtar Kelimeler: Yapay Arı Kolonisi algoritması, sınırlamasız problemler, sınırlamalı problemler, tam sayı problemler, karma problemler, parametre ayarlaması, gerçek dünya problemleri

PERFORMANCE ANALYSIS OF ARTIFICIAL BEE COLONY ALGORITHM ON NUMERICAL OPTIMIZATION PROBLEMS

Bahriye AKAY
Erciyes University, Graduate School of Natural and Applied Sciences
Ph.D. Thesis, November 2009
Thesis Supervisor: Prof. Dr. Derviş KARABOĞA

ABSTRACT

In optimization field, it is important to develop models that show good performance on wide range of problem types rather than models that demonstrate excellent performance on a limited range of the problem types. In order to investigate that an optimization algorithm has this characteristic, it is required to conduct a detailed performance analysis and to make reliable comparisons against the other well-known techniques in the literature. In this study, a comprehensive performance analysis of the recently proposed Artificial Bee Colony algorithm which simulates the foraging behaviour of honey bees was conducted. Artificial Bee Colony algorithm was modified to be able to cope with integer programming problems and its performance on integer programming problems was compared to the state-of-the-art approaches. The performance of the ABC algorithm on unconstrained hybrid composite functions was improved by introducing some modifications in the algorithm. A new version of the ABC algorithm was proposed to solve constrained problems of which the optimal solution must be located in the feasible space bounded by the equality and/or inequality constraints. In addition to analyzing the ABC algorithm on benchmark problems considered in optimization field, the ABC algorithm was also applied to solve PID controller design and several nonlinear and constrained mechanical engineering design problems.

Keywords: Artificial Bee Colony algorithm, unconstrained problems, constrained problems, integer problems, hybrid problems, parameter tuning, real world problems

İÇİNDEKİLER

KABUL VE ONAY i

TEŞEKKÜR ii

ÖZET iii

ABSTRACT iv

TABLOLAR LİSTESİ xii

ŞEKİLLER LİSTESİ xix

1. BÖLÜM

GİRİŞ 1

1.1. Optimizasyon 3

1.1.1. Tasarım Değişkenleri 6

1.1.2. Amaç ve Uygunluk Fonksiyonu 7

1.1.3. Sınırlamalar 8

1.2. Optimizasyon Metotları 9

1.2.1. Klasik Teknikler 9

1.2.2. Modern Sezgisel Algoritmalar 12

1.2.3. Gelişime Dayalı Optimizasyon Algoritmaları 14

1.2.3.1. Gösterim (Representation) 16

1.2.3.2. Uygunluk Fonksiyonu (Fitness Function) 17

1.2.3.3. Popülasyon 17

1.2.3.4. Ebeveyn Seçme Mekanizması 18

1.2.3.5. Değişim Operatörleri 18

1.2.3.6. Hayatta Kalanı Seçme Mekanizması 19

1.2.3.7. Algoritmanın Başlatılması (Initialization) 20

1.2.3.8. Durdurma Kriteri 20

1.2.4. Sezgisel Metotların Performanslarının Değerlendirilmesi 21

1.2.4.1. Çözümlerin Kalitesi 25

1.2.4.2. Hesaplama Maliyeti 25

1.2.4.3. Gürbüzlük 26

2. BÖLÜM

SÜRÜ ZEKASI 27

2.1. Arıların Doğası ve Arılardaki Sürü Zekası 29

2.2. Arılardaki Sürü Zekasını Temel Alan Çalışmalar 34

2.2.1. Kraliçe Arı Benzetimleri 34

2.2.2. Dans ve Haberleşme Benzetimleri 35

2.2.3. Görev Paylaşımı Benzetimleri 37

2.2.4. Toplu Karar Alma ve Yuva Yeri Seçimi Benzetimleri 37

2.2.5. Çiftleşme, Üreme Benzetimleri 38

2.2.6. Yiyecek Arama Davranışı Benzetimleri 40

2.2.7. Floral, Feromen Bırakma Benzetimleri 46

2.2.8. Navigasyon Benzetimleri 46

2.3. Sonuç 47

3. BÖLÜM

YAPAY ARI KOLONİSİ (ARTIFICIAL BEE COLONY, ABC) ALGORİTMASI 53

3.1. Gerçek Arıların Yiyecek Arama Davranışları 53

3.2. Yapay Arı Kolonisi Algoritması 57

3.2.1. Başlangıç Yiyecek Kaynağı Bölgelerinin Üretilmesi 59

3.2.2. İşçi Arıların Yiyecek Kaynağı Bölgelerine Gönderilmesi 59

3.2.3. Gözcü Arıların Seleksiyonda Kullanacakları Olasılık Değerlerinin Hesaplanması (Dans Benzetimi) 61

3.2.4. Gözcü Arıların Yiyecek Kaynağı Bölgesi Seçmeleri 61

3.2.5. Kaynağı Bırakma Kriteri: Limit ve Kaşif Arı Üretimi 62

3.2.6. Seleksiyon Mekanizmaları 62

3.2.7. ABC Algoritmasının Adımları 64

3.2.8. ABC'nin Temel Özellikleri 65

4. BÖLÜM

SINIRLAMASIZ OPTİMİZASYON PROBLEMLERİ ÜZERİNDE YAPAY ARI KOLONİSİ ALGORİTMASININ PERFORMANS ANALİZİ 66

4.1. Basit Sınırlamasız Fonksiyonlar Üzerinde Çalışmalar 66

4.1.1. Kontrol Parametrelerinin Etkisi 67

4.1.1.1. Koloni Büyüklüğü ve Limitin Etkisi 69

4.1.1.2. Değişim Oranı ve Ölçekleme Faktörünün Etkisi 82

4.1.1.3. Başlangıç Popülasyonunun Oluşturulma Aralığının Etkisi 90

4.1.2. Diğer Algoritmalarla ABC Algoritmasının Kıyaslanması 105

4.1.2.1. Çalışma 1: ABC, GA, DE ve PSO Algoritmalarının Kıyaslanması . . 105

4.1.2.2. Çalışma 2: ABC ve ES-1 Algoritmalarının Kıyaslanması 109

4.1.2.3. Çalışma 3: ABC ve ES-2 Algoritmalarının Kıyaslanması 111

4.2. Tam Sayılı Test Fonksiyonları Üzerinde Çalışmalar 131

4.2.1. Çalışma 1: Tam sayı programlama problemleri için QPSO, PSO-In ve ABC 133

4.2.2. Çalışma 2: Tam sayı programlama problemleri için PSO Türevleri, BB ve ABC 134

4.3. Karma Test Fonksiyonları 140

4.3.1. ABC Algoritmasının Karma Fonksiyonlara Uygulanması 143

4.3.2. Literatürdeki Diğer Algoritmalarla ABC Algoritmasının Kıyaslanması 144

5. BÖLÜM

SINIRLAMALI OPTİMİZASYON PROBLEMLERİ ÜZERİNDE YAPAY ARI KOLONİSİ ALGORİTMASININ PERFORMANS ANALİZİ 161

5.1. Giriş 161

5.2. Literatürde Mevcut İlgili Çalışmalar 162

5.3. Sınırlamalı Problemlere ABC Algoritmasının Uyarlanması 167

5.4. Literatürdeki Diğer Algoritmalarla ABC'nin Performansının Kıyaslanması 172

5.5. Kontrol Parametrelerinin Etkisi 179

6. BÖLÜM

GERÇEK DÜNYA PROBLEMLERİNİN ÇÖZÜMÜNDE ABC ALGORİTMASININ KULLANILMASI 199

6.1. PID Kontrolör Tasarımı 199

6.1.1. Geçici-Durum Cevabı 201

6.1.2. PID Kontrolör 202

6.1.2.1. Mutlak Hatanın İntegrali (Integral of the Absolute Error, IAE) . . . 203

6.1.2.2. Karesel Hatanın İntegrali (Integral of the Square Error, ISE) 204

6.1.2.3. Zaman Ağırlıklı Mutlak Hatanın İntegrali (Integral of the Time-weighted Absolute Error, ITAE) 204

6.1.2.4. Çıkış Farkı ve Yükselme Zamanına Bağlı Mutlak Hata (PWAE) . . . 205

6.1.2.5. Çıkış Farkı ve Yükselme Zamanına Bağlı Zaman ile Ağırlıklandırılmış Mutlak Hata (PWTAE) 205

6.1.2.6. Çıkış Farkı, Yükselme Zamanı ve Maksimum Taşmaya Bağlı Zaman ile Ağırlıklandırılmış Mutlak Hata (PWOTAE) 205

6.1.3. ABC Algoritması ile PID Kontrolör Parametrelerinin En İyilenmesi . 206

6.1.3.1. Sistem 1 [1] 207

6.1.3.2. Sistem 2 [2] 210

6.2. Sınırlamalı Mühendislik Tasarım Problemleri 213

6.2.1. Kaynak Yapılmış Kiriş (Welded Beam) Tasarım Problemi 213

6.2.2. Basınç Tankı (Pressure Vessel) Tasarım Problemi 214

6.2.3. Yay (Tension/Compression Spring) Tasarım Problemi 215

6.2.4. Hız İndirgeyici (Speed Reducer) Tasarım Problemi 216

6.2.5. Dişli Takımı (Gear Train) Tasarım Problemi 217

6.3. Çalışmalar ve Testler 218

7. BÖLÜM

SONUÇLAR 226

7.1. Gelecekte Yapılacak Çalışmalar 229

KAYNAKLAR 231

EK-1. BÖLÜM

Karşılaştırma Yapılan Algoritmalar 260

1.1. Gelişim Stratejileri 260

1.2. Gelişim Algoritmaları 262

1.2.1. Genetik Algoritma 262

1.2.2. SPC-PNX 263

1.2.3. Diferansiyel Gelişim Algoritması 264

1.2.4. Adaptif Ayrışmalı Sınırlamalı Gelişim Algoritması 266

1.2.5. Biçimdeş Haritalamalı Gelişim Algoritması 266

1.3. Parçacık Sürüsü Optimizasyon Algoritması 266

1.3.1. Kuantum Davranışlı PSO 268

1.3.2. Rekombinasyon ve Dinamik Bağlantılı PSO 269

1.3.3. Dinamik Çok Sürülü PSO 269

1.4. Dal ve Sınır Tekniği 270

EK-2. BÖLÜM

Fonksiyonlara Ait Veri Tabloları 271

EK-3. BÖLÜM

Karma Fonksiyonlar 278

3.1. F_1: Kaydırılmış Sphere Fonksiyonu 278

3.2. F_2: Kaydırılmış Schwefel's Problem 1.2 278

3.3. F_3: Kaydırılmış Döndürülmüş Yüksek Dereceli Eliptik Fonksiyon . . 278

3.4. F_4:Shifted Schwefel's Problem 1.2 Gürültülü 278

3.5. F_5: Genişletilmiş Schwefel's Problem 2.6 Global Optimum Sınırlar Üzerinde 279

3.6. F_6: Kaydırılmış Rosenbrock Fonksiyonu 279

3.7. F_7: Kaydırılmış DöndürülmüşGriewank Fonksiyonu Parametre Sınırlamasız 279

3.8. F_8: Kaydırılmış Döndürülmüş Ackley Fonksiyonu Global Minimum Sınırlar Üzerinde 279

3.9. F_9: Kaydırılmış Rastrigin Fonksiyonu 280

3.10. F_{10}: Kaydırılmış Döndürülmüş Rastrigin Fonksiyonu 280

3.11. F_{11}: Kaydırılmış Döndürülmüş Weierstrass Fonksiyonu 280

3.12. F_{12}: Schwefel Problem 2.13 280

3.13. F_{13}: Kaydırılmış Genişletilmiş Griewank + Rosenbrock Fonksiyonu . 281

3.14. F_{14}: Kaydırılmış Döndürülmüş Genişletilmiş Schaffer F6 Fonksiyonu 281

3.15. F_{15}: Karma Birleşim Fonksiyonu 281

3.16. F_{16}: Döndürülmüş Karma Birleşim Fonksiyonu 282

3.17. F_{17}: Gürültülü Döndürülmüş Karma Birleşim Fonksiyonu 282

3.18. F_{18}: Döndürülmüş Karma Birleşim Fonksiyonu 283

3.19. F_{19}: Global Optimumu Dar Alanda Olan Döndürülmüş Karma Birleşim Fonksiyonu 283

3.20. F_{20}: Global Optimumun Sınır Üzerinde Olduğu Döndürülmüş Karma Birleşim Fonksiyonu . . . 284

3.21. F_{21}: Döndürülmüş Karma Birleşim Fonksiyonu . . . 284

3.22. F_{22}: Yüksek Dereceli Döndürülmüş Karma Birleşim Fonksiyonu . . . 285

3.23. F_{23}: Süreksiz Döndürülmüş Karma Birleşim Fonksiyonu . . . 285

3.24. F_{24}: Döndürülmüş Karma Birleşim Fonksiyonu . . . 286

3.25. F_{25}: Araştırma Bölgesi Sınırlamasız Döndürülmüş Karma Birleşim Fonksiyonu . . . 286

EK-4. BÖLÜM

Sınırlamalı Test Problemleri . . . 288

4.1. g01: . . . 288

4.2. g02: . . . 288

4.3. g03: . . . 289

4.4. g04: . . . 289

4.5. g05: . . . 290

4.6. g06: . . . 291

4.7. g07: . . . 291

4.8. g08: . . . 292

4.9. g09: . . . 292

4.10. g10: . . . 293

4.11. g11: . . . 293

4.12. g12: . . . 294

4.13. g13: . . . 294

EK-5. BÖLÜM

ABC Algoritmasının Kodları . . . 295

ÖZGEÇMİŞ . . . 301

TABLOLAR LİSTESİ

Tablo 2.1. Algoritmaların ve uygulamalarının kategorisel olarak listesi 48

Tablo 4.1. Parametre analizinde kullanılan test fonksiyonları 73

Tablo 4.2. PSO, DE ve ABC algoritmalarının farklı problem boyutları için deneysel sonuçları, $SN = 100$, Ort:Ortalama, SD: Standar sapma. 74

Tablo 4.3. PSO, DE ve ABC algoritmalarının farklı popülasyon büyüklükleri için deneysel sonuçları, $D = 100$, Ort:Ortalama, SD: Standart sapma. 75

Tablo 4.4. ABC algoritmasının farklı limit değerleri için sonuçları, $D = 100$, $SN = 100$, Ort:Ortalama, SD: Standart sapma. 76

Tablo 4.5. ABC algoritmasının farklı limit değerleri için sonuçları, $D = 100$, $SN = 10$, Ort:Ortalama, SD: Standart sapma. 77

Tablo 4.6. Popülasyonun araştırma uzayının sol çeyrek bölümünde oluşturulması durumunda ve farklı limit değerleri kullanılarak ABC algoritması ile elde edilen sonuçlar, $D = 100$, $SN = 10$, Ort:Ortalama, SD: Standart sapma. 78

Tablo 4.7. Değişim oranı (MR) ve ölçekleme faktörünün (SF) analizinde kullanılan tek modlu ve çok modlu basit sınırlamasız test fonksiyonları 86

Tablo 4.8. Farklı MR ve SF değerleri ile koşulan ABC algoritmasının ve PSO'nun türevlerinin ilk dört fonksiyon için sonuçları, MM:Çok modlu, UM:Tek modlu 87

Tablo 4.9. Farklı MR ve SF değerleri ile koşulan ABC algoritmasının ve PSO'nun türevlerinin son dört fonksiyon için sonuçları, MM:Çok modlu, UM:Tek modlu 88

Tablo 4.10. Popülasyon başlangıç aralığının etkisi analizinde kullanılan DE, PSO ve ABC algoritmalarının kontrol parametrelerinin değerleri, SN: Koloni büyüklüğü, PS: Popülasyon büyüklüğü, SS: Sürü büyüklüğü, MCN: Maksimum Çevrim Sayısı, MGN: Maksimum jenerasyon sayısı, CR: Çaprazlama oranı, F: Ölçekleme Faktörü, ϕ_1: Bilişsel bileşen, ϕ_2: Sosyal bileşen, ω: Atalet(inertia) 92

Tablo 4.11. Çalışmalarda kullanılan kullanılan test fonksiyonları 93

Tablo 4.12. PSO algoritmasının farklı başlatma aralıkları ile elde edilen 30 koşmasının ortalaması. 99

Tablo 4.13. DE algoritmasının farklı başlatma aralıkları ile elde edilen 30 koşmasının ortalaması. 100

Tablo 4.14. ABC algoritmasının farklı başlatma aralıkları ile elde edilen 30 koşmasının ortalaması. 101

Tablo 4.15. PSO, DE ve ABC algoritmalarının ANOVA testi ile elde edilen anlamlılık sonuçları 102

Tablo 4.16. Çalışma 1'in sonuçları 115

Tablo 4.17. GA ve ABC algoritması için anlamlı farklılık testi t: student t-testine ait t-değeri, SED: Farkların standart hatası, p: t-değerine karşılık gelen p-değeri, R: p-değerinin rankı, I.R.:p-değerinin invers rankı, Sign: Farklılık. 122

Tablo 4.18. PSO ve ABC algoritmaları için anlamlı farklılık testi t: student t-testine ait t-değeri, SED: Farkların standart hatası, p: t-değerine karşılık gelen p-değeri, R: p-değerinin rankı, I.R.:p-değerinin invers rankı, Sign: Farklılık. 123

Tablo 4.19. DE ve ABC algoritmaları için anlamlı farklılık testi t: student t-testine ait t-değeri, SED: Farkların standart hatası, p: t-değerine karşılık gelen p-değeri, R: p-değerinin rankı, I.R.:p-değerinin invers rankı, Sign: Farklılık. 124

Tablo 4.20. Çalışma 2'de kullanılan test fonksiyonları 125

Tablo 4.21. CES [3], FES [4], ESLAT [3], CMA-ES [3] ve ABC Algoritmalarının sonuçları. D:Boyut, Ort:Ortalama, SD:Standart Sapma 128

Tablo 4.22. CES [3], FES [4], ESLAT [3], CMA-ES [3] ve ABC Algoritmalarının değerlendirme sayısına göre hesaplanan çözüm maliyetleri 129

Tablo 4.23. ESLAT [3], CMA-ES [3] and ABC Algoritmalarının başarı oranları(%). Koyu renkle yazılanlar en iyi sonucu göstermektedir. 130

Tablo 4.24. Çalışma 3'de kullanılan test fonksiyonları. D:Boyut, C:Karakteristik, U:Tek modlu, M:Çok modlu, S:Ayrıştırılabilir, N:Ayrıştırılamaz 130

Tablo 4.25. SOM-ES [5], NG-ES [5], CMA-ES [5] ve ABC Algoritmalarının istatistiksel sonuçları, Ort. Değ.: Ortalama Değerlendirme Sayısı . 131

Tablo 4.26. PSO-In [6], QPSO [6] ve ABC algoritmalarının $F_1 - F_5$, F_8, F_9 fonksiyonları için maksimum iterasyon sayıları (MNI), sürüdeki eleman sayıları (SS), başarı oranları (SR), ortalama iterasyon sayıları (MI). Koyu renkle yazılanlar en iyi sonucu göstermektedir. D:Problemin boyutu 136

Tablo 4.27. Çalışma 2'de kullanılan fonksiyonların boyutları ve bu fonksiyonlar için sürü boyutları 137

Tablo 4.28. PSO varyantları [7], BB Tekniği [7] ve ABC algoritmalarının F_1 fonksiyonu için başarı oranları (SR), değerlendirme sayılarının ortalama, median ve standard sapmaları (Std). Koyu renkle yazılan sonuçlar en iyi sonucu göstermektedir. 138

Tablo 4.29. PSO varyantları [7], BB Tekniği [7] ve ABC algoritmalarının F_2-F_7 fonksiyonları için başarı oranları (SR), değerlendirme sayılarının ortalama, median ve standard sapmaları (Std). Koyu renkle yazılan sonuçlar en iyi sonucu göstermektedir. 139

Tablo 4.30. 1-8 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 10, Koloni Büyüklüğü: 10, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 100000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması 146

Tablo 4.31. 9-17 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 10, Koloni Büyüklüğü: 10, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 100000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması 147

Tablo 4.32. 18-25 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 10, Koloni Büyüklüğü: 10, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 100000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması 148

Tablo 4.33. 1-8 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 30, Koloni Büyüklüğü: 30, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 300000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması 149

Tablo 4.34. 9-17 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 30, Koloni Büyüklüğü: 30, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 300000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması 150

Tablo 4.35. 18-25 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 30, Koloni Büyüklüğü: 30, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 300000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması 151

Tablo 4.36. 1-8 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 50, Koloni Büyüklüğü: 50, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 500000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması 152

Tablo 4.37. 9-17 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 50, Koloni Büyüklüğü: 50, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 500000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması 153

Tablo 4.38. 18-25 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 50, Koloni Büyüklüğü: 50, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 500000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması 154

Tablo 4.39. ABC algoritmasının karmaşıklığı 155

Tablo 4.40. 1-5 numaralı karma problemler için tüm algoritmaların sonuçları, D:10, Koloni Büyüklüğü: 20, Çevrim Sayısı: 5000, PSO-RDL: [8], DMS-PSO: [9], SPC-PNX [10], DE: [11], SADE [12], restartCMA-ES: [13] 156

Tablo 4.41. 6-10 numaralı karma problemler için tüm algoritmaların sonuçları, D:10, Koloni Büyüklüğü: 20, Çevrim Sayısı: 5000, PSO-RDL: [8], DMS-PSO: [9], SPC-PNX [10], DE: [11], SADE [12], restartCMA-ES: [13] 157

Tablo 4.42. 11-15 numaralı karma problemler için tüm algoritmaların sonuçları, D:10, Koloni Büyüklüğü: 20, Çevrim Sayısı: 5000, PSO-RDL: [8], DMS-PSO: [9], SPC-PNX [10], DE: [11], SADE [12], restartCMA-ES: [13] 158

Tablo 4.43. 16-20 numaralı karma problemler için tüm algoritmaların sonuçları, D:10, Koloni Büyüklüğü: 20, Çevrim Sayısı: 5000, PSO-RDL: [8], DMS-PSO: [9], SPC-PNX [10], DE: [11], SADE [12], restartCMA-ES: [13] 159

Tablo 4.44. 21-25 numaralı karma problemler için tüm algoritmaların sonuçları, D:10, Koloni Büyüklüğü: 20, Çevrim Sayısı: 5000, PSO-RDL: [8], DMS-PSO: [9], SPC-PNX [10], DE: [11], SADE [12], restartCMA-ES: [13] 160

Tablo 5.1. 13 test problemine ait ρ değerleri, D:problemin boyutu, LI: doğrusal eşitsizlik sayısı, NI: doğrusal olmayan eşitsizlik sayısı, LE: doğrusal eşitlik sayısı, NE: doğrusal olmayan eşitlik sayısı [14].173

Tablo 5.2. ABC algoritmasının 240000 değerlendirme ile 13 fonksiyon üzerinde elde edilen 30 koşmaya ait istatistiksel sonuçlar. 175

Tablo 5.3. HM, SR, ISR, OPA, ASCHEA, SMES, GA, DE, PSO ve ABC algoritmalarının 13 fonksiyon üzerinde 30 koşmalarının en iyi sonuçları, – Herhangi bir kabul edilebilir çözümün bulunamadığını, Na = Sonucun olmadığını göstermektedir. 182

Tablo 5.4. HM, SR, ISR, OPA, ASCHEA, SMES, GA, DE, PSO ve ABC algoritmalarının 13 fonksiyon üzerinde 30 koşmalarının en kötü sonuçları, – Herhangi bir kabul edilebilir çözümün bulunamadığını, Na = Sonucun olmadığını göstermektedir. 183

Tablo 5.5. HM, SR, ISR, OPA, ASCHEA, SMES, GA, DE, PSO ve ABC algoritmalarının 13 fonksiyon üzerinde 30 koşmalarının ortalama sonuçları – Herhangi bir kabul edilebilir çözümün bulunamadığını, Na = Sonucun olmadığını göstermektedir. Koyu renkle yazılan sonuçlar en iyi sonuçtur. 184

Tablo 5.6. ABC algoritması ile diğer algoritmaların kıyaslanması ile elde edilen başarı oranları. + algoritmanın daha iyi olduğunu - daha kötü olduğunu göstermektedir. 187

Tablo 5.7. ABC algoritmasının 350000 değerlendirme ile 13 fonksiyon üzerinde elde edilen 30 koşmaya ait istatistiksel sonuçlar. 188

Tablo 5.8. 350000 değerlendirme yapan ABC algoritması ile diğer algoritmaların kıyaslanması ile elde edilen başarı oranları. + algoritmanın daha iyi olduğunu - daha kötü olduğunu göstermektedir. 189

Tablo 5.9. 9 farklı MR değeri ile elde edilen hata değerleri için ANOVA Tablosu 190

Tablo 5.10. 5 farklı *limit* değeri ile elde edilen hata değerleri için ANOVA Tablosu 190

Tablo 5.11. 5 farklı SPP değeri ile elde edilen hata değerleri için ANOVA Tablosu 191

Tablo 5.12. 9 farklı MR değeri (0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9) ile elde edilen hata değerleri için ANOM Tablosu 191

Tablo 5.13. 5 farklı *limit* değeri (0.1xSNxD, 0.5xSNxD, SNxD, 2xSNxD, 4xSNxD) ile elde edilen hata değerleri için ANOM Tablosu 192

Tablo 5.14. 5 farklı SPP değeri (0.1xSNxD, 0.5xSNxD, SNxD, 2xSNxD, 4xSNxD) ile elde edilen hata değerleri için ANOM Tablosu 192

Tablo 6.1. Orantı, İntegral ve Türev karakteristikleri 203

Tablo 6.2. Sistem 1 için ZN, CHR ve ABC ile gerçekleştirilen tasarımların K_d, K_p, K_i, PO (yüzde maksimum aşma), t_r, t_s ve en iyilenme yapıldığı hata değerleri 208

Tablo 6.3. Sistem 2 için ZN, Kitamori, Tabu Araştırma ve ABC ile gerçekleştirilen tasarımların K_d, K_p, K_i, PO (yüzde maksimum aşma), t_r, t_s ve hata değerleri 211

Tablo 6.4. Sistem 1 ve Sistem 2 için ABC algoritmasının 30 koşmasına ait ortalama PWAE hata ve standard sapma değerleri 213

Tablo 6.5. Algoritmaların kontrol parametrelerinin değerleri. D: Problem boyutu, CS: Medeniyet sayısı, TS: Zaman aralığı, σ: Normal dağılım varyansı, SS: Sürü boyutu, MGN: Maksimum jenerasyon sayısı, ω: inertia, c_1: Bilişsel bileşen, c_2: Sosyal bileşen, S_r: Seleksiyon oranı, LR: Öğrenme oranı, MSS: Mutasyon adım büyüklüğü, χ: Konstriksiyon faktörü, NR: Komşuluk çapı, UF: Unification Faktörü, MtR: Mutation Oranı, CSabc: Koloni büyüklüğü, MCN: Maksimum çevrim sayısı, SPP: Kaşif arı üretme peryodu, MR: Değişim oranı 221

Tablo 6.6. Algoritmaların mühendislik problemleri için ürettikleri istatistiki sonuçlar. Koyu yazılı değerler en iyi sonucu göstermektedir. . . . 222

Tablo 6.7. Kaynak yapılmış kiriş problemi için en iyi sonuca ait parametre ve sınırlama değerleri 223

Tablo 6.8. Basınç tankı problemi için en iyi sonuca ait parametre ve sınırlama değerleri 223

Tablo 6.9. Yay problemi için en iyi sonuca ait parametre ve sınırlama değerleri 223

Tablo 6.10. Hız indirgeyici problemi için en iyi sonuca ait parametre ve sınırlama değerleri 224

Tablo 6.11. Dişli takımı problemi için en iyi sonuca ait parametre ve sınırlama değerleri 224

Tablo EK-2.1 a and c parameters of Langerman Function 272

Tablo EK-2.2 A parameter of Fletcher-Powell Function 273

Tablo EK-2.3 B parameter of Fletcher-Powell Function 274

Tablo EK-2.4 α parameter of Fletcher-Powell Function 275

Tablo EK-2.5 a parameter of FoxHoles Function 276

Tablo EK-2.6 a and b parameters of Kowalik Function 277

Tablo EK-2.7 a and c parameters of Shekel Functions 277

Tablo EK-2.8 a, c and p parameters of 3-parameter Hartman Function 277

Tablo EK-2.9 a, c and p parameters of 6-parameter Hartman Function 277

ŞEKİLLER LİSTESİ

Şekil 1.1. Optimizasyon Metotlarının Sınıflandırılması [15] 11

Şekil 1.2. Klasik Algoritmaların ve Modern Sezgisel Algoritmaların Başarım Spektrumu 13

Şekil 2.1. Arılarda Görev Paylaşımı 32

Şekil 2.2. Arı kolonilerinin zeki davranışları üzerine yapılan yayın sayılarının yıllara göre dağılımı 47

Şekil 3.1. Arılarda Dans 56

Şekil 3.2. Yiyecek Arama Çevrimi 57

Şekil 3.3. ABC algoritmasının akış diyagramı 63

Şekil 4.1. Gelişim algoritmalarının parametre ayarlanması ile ilgili sınıflandırması 69

Şekil 4.2. Sphere, Rosenbrock ve Rastrigin fonksiyonları için başlangıç popülasyonun tüm uzayda ve sol çeyrek uzayda oluşturulması durumlarında limit parametresinin etkisinin analizi, $D = 100$, $SN = 10$. 79

Şekil 4.3. Griewank, Schwefel ve Ackley fonksiyonları için başlangıç popülasyonun tüm uzayda ve sol çeyrek uzayda oluşturulması durumlarında limit parametresinin etkisinin analizi, $D = 100$, $SN = 10$. 80

Şekil 4.4. Step, Penalized ve Dixon-Price fonksiyonları için başlangıç popülasyonun tüm uzayda ve sol çeyrek uzayda oluşturulması durumlarında limit parametresinin etkisinin analizi, $D = 100$, $SN = 10$. 81

Şekil 4.5. Farklı kontrol parametreleri ile bulunan en iyi sonuçların ortalaması 89

Şekil 4.6. ABC algoritmasının yakınsama grafikleri 90

Şekil 4.7. Algoritmaların ortalama mutlak hata değerleri 109

Şekil 4.8. ABC algoritmasının Test Problem 1'in farklı boyutları için logaritmik ölçekli yakınsama grafikleri 136

Şekil 4.9. ABC algoritmasının Test Problemleri 2-5,8,9'un farklı boyutları için logaritmik ölçekli yakınsama grafikleri 137

Şekil 4.10. 30 boyutlu durum için 25 koşmanın median değerine ait koşmanın logaritmik ölçekli yakınsama grafikleri 155

Şekil 5.1. g01-g06 problemlerinin ortalama sonuçlarına ait çubuk grafikler . 185

Şekil 5.2. g07-g13 problemlerinin ortalama sonuçlarına ait çubuk grafikler . 186

Şekil 5.3. 9 farklı MR değeri için g01-g06 problemlerinin ANOM grafikleri . 193

Şekil 5.4. 9 farklı MR değeri için g07-g13 problemlerinin ANOM grafikleri . 194

Şekil 5.5. 5 farklı *limit* değeri için g01-g06 problemlerinin ANOM grafikleri 195

Şekil 5.6. 5 farklı *limit* değeri için g07-g13 problemlerinin ANOM grafikleri 196

Şekil 5.7. 5 farklı SPP değeri için g01-g06 problemlerinin ANOM grafikleri 197

Şekil 5.8. 5 farklı SPP değeri için g07-g13 problemlerinin ANOM grafikleri 198

Şekil 6.1. Açık çevrimli sisteme ait blok şema 200

Şekil 6.2. Kapalı çevrimli sisteme ait blok şema 200

Şekil 6.3. Geçici-durum cevabı parametreleri 202

Şekil 6.4. Sistem 1 için ZN, CHR ve ABC ile gerçekleştirilen tasarımların birim basamak cevabı 209

Şekil 6.5. Sistem 1 için ABC algoritmasının farklı performans indislerine göre koşulmasıyla gerçekleştirilen tasarımların birim basamak cevabı 209

Şekil 6.6. Sistem 1 için PWAE performans indisi kullanan ABC algoritmasının 30 koşmasının en iyi değerlerinin ortalamalarına ait gelişim grafiği 210

Şekil 6.7. Sistem 2 için ZN, Kitamori, Tabu Araştırma ve ABC ile gerçekleştirilen tasarımların birim basamak cevabı 212

Şekil 6.8. Sistem 2 için ABC algoritmasının farklı performans indislerine göre koşulmasıyla gerçekleştirilen tasarımların birim basamak cevabı 212

Şekil 6.9. Sistem 2 için PWAE performans indisi kullanan ABC algoritmasının 30 koşmasının en iyi değerlerinin ortalamalarına ait gelişim grafiği 213

Şekil 6.10. Kaynak yapılmış kiriş tasarım problemi 214

Şekil 6.11. Basınç tankı tasarım problemi 215

Şekil 6.12. Yay tasarım problemi 216

Şekil 6.13. Hız indirgeyici tasarım problemi 216

Şekil 6.14. Dişli takımı tasarım problemi 217

Şekil 6.15. ABC algoritmasının 30 koşmasında en iyi sonuçların ortalamalarının gelişimi 225

1. BÖLÜM

GİRİŞ

Gerçek hayatta karşılaşılan problemlerin modellenerek bilgisayar ortamına aktarılması bilgisayar destekli çözümlerin bulunması gereksinimini ortaya çıkarmıştır. Bu problemlerin bir çoğu kombinasyonel veya sürekli tipteki optimizasyon problemleridir.

Optimizasyon; bilgisayar bilimi, yapay zeka, yöneylem araştırması ve ilgili araştırma alanlarında önemli bir konudur. Bu bilimlerin dışında optimizasyon, "daha iyiyi yapma" anlamına gelmektedir. Bu bilimlerde ise optimizasyon, genelde belli bir zaman sınırı olan optimizasyon problemine getirilen olası çözümlerden kabul edilebilir bölge içerisinde en iyi olanı (amaç fonksiyonunu minimum yada maksimum yapan) bulma sürecidir. Optimizasyon problemi farklı olası çözümleri olan ve çözüm kalitesinin açık bir ifadeyle yazılabildiği bir problemdir. Yani farklı aday çözümlerin anlamlı şekilde karşılaştırılıp değerlendirilebilmesi sözkonusu ise bir optimizasyon problemi var demektir.

Optimizasyon problemlerinin çözümünde optimizasyon teknikleri kullanılmaktadır. Problem boyutunun artması, problem yapısının lineer olmayan türde olması ve problem uzayının kabul edilebilir bölgesinin çok dar bir alan olması gibi sebeplerden dolayı kullanılan optimizasyon tekniğinin bu gibi hususlarla başa çıkabilecek ve olumsuz koşullara rağmen iyi sonuç verebilecek yetenekte olması beklenir. Klasik optimizasyon algoritmalarının bu zorluklarla başarılı bir şekilde başa çıkamamalarından dolayı literatüre bazı sezgisel teknikler önerilmiştir (Isıl İşlem (simulated annealing, SA) [16, 17], Tabu Araştırma (tabu search, TS) [18, 19], Genetik Algoritma (genetic algorithm, GA) [20]). Genetik algoritma ve ısıl işlem

gibi doğal olaylara dayalı algoritmaların geliştirilmesi, araştırmacıları doğada var olan başka olayları da incelemeye ve bunların modellenmesine ve benzetimini yapmaya teşvik etmiştir. Doğada sürüler halinde yaşayan varlıkların toplu olarak gösterdikleri zeki davranışlar (sürü zekası davranışı olarak adlandırılmaktadır) da bu doğal olaylardan birisidir. Özellikle sürülerin hayatta kalmaları açısından hayati önem taşıyan yiyecek bulma davranışlarının modellenmesiyle oldukça başarılı algoritmalar ortaya çıkmıştır. Bunlardan Karınca Koloni Optimizasyon (ant colony optimization, ACO) algoritması karıncaların yiyecek kaynağı ve yuva arasındaki yol uzunluğunu en aza indirgemeye yönelik davranışlarına dayalı bir algoritmadır [21]. Yapay Arı Kolonisi (Artificial Bee Colony, ABC) algoritması Karaboga tarafından arıların topluluk olarak yiyecek arama davranışlarını temel alarak geliştirilmiş bir optimizasyon algoritmasıdır [22]. Algoritmanın nümerik optimizasyon alanında, tam sayı programlamada, sınırlamalı optimizasyonda birçok probleme uygulanarak performans karşılaştırmalarının yapılması, birimlerinin analiz edilmesi ve gerekli iyileştirmelerin yapılması bu tezin amacıdır.

Tezin organizasyonu aşağıda belirtildiği gibidir:

Bu bölümde bir optimizasyon probleminin tanımı yapılarak bu türden problemlerin çözümünde kullanılan optimizasyon metotlarının klasik ve sezgiseller olarak gruplandırılması yapılacaktır. Sezgisel algoritmaların bileşenleri, başlatılmaları ve durdurulma kriterleri, başarımlarının ölçülmesi, bu türden metotların birbiri ile kıyaslanmasında kullanılacak ölçütler ve çözüm kalitesinin belirlenmesi gibi konular ele alınacaktır. Bu bölüm ile optimizasyon süreci ile ilgili genel bilgilerin verilmesi amaçlanmaktadır.

2. Bölümde sürü zekasının temel tanımları ve prensipleri verilerek bir sürünün zeki davranış gösteriyor olarak kabul edilebilmesi için gerekli şartlar anlatılacaktır. Arıların da zeki olarak nitelenecek davranışları anlatılarak, literatürde bu davranışların benzetimi ve uygulamaları üzerine yapılmış çalışmalar tanıtılarak, bu çalışmaların yıllara göre dağılımı sunulacaktır.

Arıların davranışlarının genel olarak incelenmesinin ardından 3. Bölümde özel olarak yiyecek arama davranışının tanıtılması amaçlanmıştır. Öncelikle gerçek arılardaki

yiyecek arama davranışının nasıl ortaya çıktığı, ne tür birim ve işleyişe sahip olduğundan bahsedilerek bu davranışın benzetimi ile geliştirilmiş Yapay Arı Kolonisi (ABC) algoritması detaylı olarak anlatılacaktır.

Tezin amacı, ABC algoritmasının başarımının ölçülmesi ve literatürdeki yerinin ve öneminin belirlenmesi amacıyla başka algoritmalarla kıyaslanması olduğu için 4. Bölümde ABC algoritmasının sınırlamasız problemler üzerindeki başarımı, 5. Bölümde ise sınırlamalı problemler üzerindeki başarımıyla ilgili yapılan çalışamalar sunulacaktır. 6. Bölümde ise sınırlamasız gerçek dünya problemleri olan PID kontrolör tasarımında ve sınırlamalı gerçek problemler olan bazı mühendislik problemlerinin çözümünde ABC algoritmasının kullanılması anlatılmıştır ve elde edilen sonuçlar sunulmuştur.

Son olarak 7. Bölümde tezden çıkarılan sonuçlar, tezin katkısı ve ileriye yönelik yapılması planlanan çalışmalar verilmiştir.

1.1. Optimizasyon

Optimizasyon, bir sistemin olası tasarımları arasından en iyisini bulmak demektir. Daha teknik bir tanım yapmak gerekirse S araştırma uzayı, F, $F \subseteq S$ şeklinde kabul edilebilir çözümler bölgesi ve f amaç fonksiyonu olsun. Optimizasyon, her $\vec{y} \in F$ için minimizasyon problemlerinde $f(\vec{x}) \leq f(\vec{y})$' yi, maksimizasyon problemlerinde $f(\vec{x}) \geq f(\vec{y})$' yi sağlayan $\vec{x} \in F$ çözümünün bulunması işlemidir. Bu durumda $\vec{x}$ küresel en iyi olmaktadır. Amaç fonksiyonu f, nümerik ($f : S \rightarrow R$) yada kombinasyonel ($f : SxS \rightarrow S$) olabilmektedir.

Bölgesel Optimum: Uzaklık ölçütü $dist : SxS \rightarrow R$ olmak üzere tüm $\vec{x} \in S$ için, $\vec{x}$'in komşuluğu $N(\vec{x})$ şu şekilde tanımlanır:

$$N(\vec{x}) = \{\vec{y} \in S | dist(\vec{x}, \vec{y}) \leq \varepsilon\} \tag{1.1}$$

Tüm $\vec{y} \in N(\vec{x})$ değerleri için $f(\vec{x}) \leq f(\vec{y})$ sağlanıyorsa $\vec{x} \in F$ çözümü, bölgesel minimumdur veya tüm $\vec{y} \in N(\vec{x})$ değerleri için $f(\vec{x}) \geq f(\vec{y})$ sağlanıyorsa bölgesel

maksimumdur denir. Dolayısıyla bölgesel optimum komşuluk tanımına oldukça bağlıdır. Dış bükey problemlerde bölgesel minimum global minimumdur. Yani belli bir komşulukta en iyi ancak global minimumdan daha kötü olan çözümlere bölgesel minimum denmektedir.

Bir problem ele alındığında bir minimumunun olup olmadığını bilinemez. Bazı durumlarda minimumun nasıl bulunacağı bilinmese bile bir minimumun varlığından emin olunabilir. Weierstrass teoremi bazı şartların sağlanması durumunda bunu garanti edebilir [23].

Weierstrass Teoremi: Boş olmayan, kapalı ve sınırlı S kabul edilebilir çözümler kümesi S'de $f(x)$ sürekli ise, S'de $f(x)$'in bir minimumu vardır.

Eğer S kümesi tüm sınır noktaları içeriyor ve her ardışıl nokta set içinde bir noktaya yakınsayan dizilime sahipse S kümesi kapalıdır ve c sonlu bir sayı olmak üzere $x \in S$ noktası için $x^T x < c$ şartı sağlanıyorsa set sınırlıdır denir. Bu şartların sağlanmaması durumunda da global minimum olabilir. Bir fonksiyonunun minimumu bulunurken gradyent vektörü ve hessian matrisleri önemlidir. Gradyent vektörü (birinci türev) fonksiyonun maksimum değişimin yönünü verirken simetrik hessian matrisi (ikinci türev) optimallik yeterlilik şartlarında önemli rol oynamaktadır.

Tasarım sürecinde iki temel adım söz konusudur:

- Probleme ait bir model kurulması
- Çözüm üretmek için bu modelin kullanılması

Belli bazı problemlerin modellerini kullanmadan da çözümlerini bulmak mümkündür. Doğrudan fiziksel sistem üzerinde çalışarak sistemden alınan geri beslemelerden faydalanmak suretiyle farklı çözümlerin denenmesi gerçeklenebilir. Bu durumda bir modele ihtiyaç yoktur. Ancak bu yaklaşımı uygulamak çoğu kez mümkün değildir. Örneğin bir köprü inşasının deneme yanılma yaklaşımı ile yapıldığı düşünüldüğünde imkansızlığı açıkca görülebilir. Problemi

modellemenin getirdiği diğer bir avantaj da bilgisayar simülasyonlarının gerçek dünya uygulamalarına göre çok daha hızlı gerçekleştirilebilmesidir. Bu nedenle uygulamalarda önce bir model geliştirilir sonrasında bu modele göre çözümlemeler yapılır. Bu iki adımda optimizasyon sürecinde kritik öneme sahiptir. Model yanlış kurulmuşsa elde edilen çözüm elimizdeki gerçek problemin çözümü olmayacağı için bir anlam taşımayacaktır. Yine, modele ait iyi bir çözüm üretilemiyorsa model doğru olsa da olmasa da optimizasyon mümkün olamayacaktır. Bir problemin formülize edilmesi modelleme aşamasının ilk önemli adımıdır. Formülasyon aşaması sistemi tanımlayan tasarım değişkenlerinin belirlenmesini, optimize edilecek amaç (objective) yada maliyet (cost) fonksiyonunun ve sistemin güvenilir ve doğru şekilde çalışmasını sağlayan sınırlamaların tanımlanmasını gerektirmektedir. Problemin türüne ve sistem performans gereksinimlerine bağlı olarak çeşitli tipte tasarım değişkenleri, sınırlamalar ve amaç fonksiyonları tanımlanabilir. Eğer probleme ait tüm fonksiyonlar doğrusal ise problem doğrusal programlama (LP) problemi adını alır. Amaç fonksiyonu quadratic ve sınırlamalar doğrusal ise problem quadratic programlama (QP) problemi, herhangi biri doğrusal olmayan fonksiyon olması halinde ise doğrusal olmayan programlama (NLP) problemi adını alır. NLP problemleri için geliştirilen nümerik algoritmaların çoğu iteratif süreçleriyle LP ve QP problemlerini de çözebilmektedir. NLP problemleri için geliştirilmiş modellerle ilgili olarak şu hususlar önemlidir [24]:

1. Model, sürekli tasarım değişkenlerine sahip tüm problemlere ve üzerinde küçük değişiklikler yapılarak uyarlanmasıyla çok amaçlı ve ayrık değişkenli problemlere de uygulanabilmelidir.

2. Probleme ait fonksiyonların iki kez türevlenebilir olduğu kabul edilmektedir. Türevlenemeyen fonksiyonların olduğu problemler ise ekstra hesaplama maliyeti ile ele alınabilmelidir.

3. Amaç ve / veya sınırlama fonksiyonları tasarım değişkenlerine bağlı açık yada kapalı fonksiyonlardır.

4. Nümerik metotlarla optimizasyon yapılırken fonksiyonların türevlerine ihtiyaç duyulmaktadır.

Bazı gerçek dünya problemleri (gezgin satıcı problemi, SAT problemi, k-renklendirme ve ağ akış problemleri, çizelgeleme problemleri, sıkıştırma problemleri vb.) için öncelikle basitleştirilmiş bir model kurulup, klasik bir teknikle basitleştirilmiş modele tam çözüm bulunurken, bir diğer yöntemde basitleştirmeden tam modele yaklaşık bir çözüm bulunmasıdır. Pratik uygulamalarda ikinci yaklaşım ilkine göre daha iyi performans göstermektedir [15].

1.1.1. Tasarım Değişkenleri

Sistemi tanımlamak için seçilen parametrelere tasarım değişkenleri denir. Bu değişkenlere değerler atanmasıyla sistem tasarımı bilinir hale gelir. Tasarımcı bu değişkenlere istediği değeri atayabildiğinde serbesttirler. Değişkenler için atanan değerler problemin tüm sınırlamalarını sağlamıyorsa tasarım kabul edilebilir değildir (infeasible). Kabul edilebilir bir tasarım amaç fonksiyonu açısından en iyi olmayabilir ancak kullanılabilir bir tasarımdır. Problemin düzgün bir şekilde formülize edilmesinde sistem tasarım değişkenlerinin neler olduğunun yani sistemin hangi değişkenlere bağlı olarak tasarlanacağının belirlenmesi oldukça önemlidir. Tasarım değişkenleri doğru olarak belirlenemezse formülasyon elbette hatalı olacak yada imkansız hale gelecektir. Sistemi etkileyen değişkenlerin çok iyi analiz edilmesi gerekir. Başka bir önemli noktada değişkenlerin birbirinden olabildiğince bağımsız olması gerekliliğidir. Aksi takdirde problem fazladan karmaşıklığa sahip olacaktır. En az sayıda değişkenle problemin ve tasarımın doğru bir şekilde ifade edilmesi en uygundur [23]. Tasarım değişkeni verilen set içerisinden belli değerler alabiliyorsa bu değişkene ayrık (discrete) değişken adı verilir. Tam sayı değişkenler yalnız tam sayı değerler alabilirler. Ayrık ve tam sayı değerleri analiz eden özel metotlar bulunmaktadır (dinamik programlama, ayır ve bağla metodu vb.). Ancak hesaplama karmaşıklıkları fazladır. Sürekli değişkenlerle çalışan nümerik metotlar kullanılarak bu türden değişkeni olan tasarım problemlerine çözüm üretilerek bunların yuvarlanması veya kesilmesi ile tamsayı veya ayrık çözümler elde edilebilir.

1.1.2. Amaç ve Uygunluk Fonksiyonu

Bir sistem için bazılarının diğerlerine göre daha iyi olduğu çok sayıda kabul edilebilir tasarım olabilmektedir. En iyi tasarım bulunmaya çalışılırken deneme yanılma tabanlı optimizasyon algoritmasının aday çözümlerin hangisinin daha iyi olduğuna karar verebilmesi için çözümlerin kalitesini kıyaslaması gerekmektedir. Bunun için uygunluk (kalite) fonksiyonu kullanılır. Uygunluk fonksiyonu, çözüm uzayındaki aday çözümler için ya gerçel nümerik değerler üretir ya da iki aday çözümü alarak hangisinin daha iyi olduğu bilgisini verir. Nümerik uygunluk fonksiyonunda bir çözümün değerine, o çözümün uygunluk değeri denir. Genellikle $\vec{x}$ tasarım değişkenleri olmak üzere $f(\vec{x})$ olarak gösterilen amaç fonksiyonu çoğu zaman doğrudan uygunluk fonksiyonu olarak kullanılırken, uygunluk fonksiyonu amaç fonksiyonu ile aynı olmak zorunda değildir. Maksimizasyon problemi için $f(\vec{x})$, $-f(\vec{x})$ olarak alınabilir [23]. Optimum çözümün uygunluğunun en yüksek uygunluk değerine sahip olması gerekir. Uygunluk fonksiyonunun sınırları araştırma uzayıdır. Araştırma uzayını ve boyutunu problemin kendisi belirlemez. Seçilen model ve bu modelin gösterimi, veri yapısı belirler. Bundan dolayı seçilen model iyi uygunluk değerine sahip çözümlerin bulunabilme ihtimalini etkilemektedir. Araştırma uzayı dışındaki tüm çözümler modele ait sınırlamaları sağlamadığından geçerli çözümler olmayacaktır. Çözümün sınırlamaları sağlamaması durumunda kabul edilebilir olmadığı (infeasible) söylenir. Geçerli çözümler araştırma uzayının kabul edilebilir (feasible) bölgesinde ($F \subseteq S$) aranır [15]. Bazı durumlarda amaç fonksiyonu birden fazla olabilmektedir. Örneğin, aynı anda bir malzemenin ağırlığı minimize edilirken, belli bir noktadaki gerilim ve deplasman da minimize edilmeye çalışılabilir. Bu tür problemlere çok amaçlı optimizasyon problemleri denir. Bu tür problemlerin çözümü için genel ve güvenilir bir metot yoktur. Ancak bazı düzenlemelerle çözülebilir hale getirilmektedirler. Örneğin, tüm fonksiyonların ağırlıklandırılıp toplanmasıyla birleşik tek bir amaç fonksiyonu (composite cost function) oluşturulabilir ve çok amaçlı bir problem tek amaçlı bir problem gibi çözülebilir. Ağırlıklandırılırken kullanılan değerlerin doğru olarak belirlenmesi önemlidir. Çünkü herhangi bir fonksiyon diğerlerinin üzerinde baskın hale gelebilir. Diğer bir yol ise bir fonksiyonun amaç fonksiyonu olarak ele alınıp diğerlerinin

sınırlama gibi düşünülmesidir. Sınırlamaların limit değerlerinin değiştirilmesiyle farklı tasarımlar elde edilebilir. Tüm amaç fonksiyonları için bu olası eğriler üretilir ve tasarım sürecinde kullanılabilir.

1.1.3. Sınırlamalar

Bir sistemin tasarımı, tasarım değişkenlerine atanan değerler kümesidir. Bu tasarım sıra dışı (örneğin çap değişkeninin negatif olması gibi) veya yetersiz olsa da bu bir tasarımdır. Ancak bazı tasarımlar kullanılabilir, bazıları değildir. Tüm gereksinimlerin sağlandığı tasarıma kabul edilebilir tasarım denir. Herhangi birinin sağlanmadığı durumda ise kabul edilebilir olmayan tasarım denir.

Her bir sınırlama bir yada daha fazla tasarım değişkenine bağlı olmalıdır. Ancak bu şekilde anlamlı olur ve optimum tasarım üzerinde etkisi bulunur. Tasarım değişkenlerinin alt ve üst sınırları gibi bazı sınırlamaları belirlemek oldukça basit iken bazıları oldukça karmaşık olabilmektedir. Örneğin, belli bir noktadaki burulma tasarıma oldukça bağlıdır. Ancak bu burulma, basit yapılar dışında tasarım değişkenlerine bağlı açık (explicit) bir fonksiyon ile ifade edilemez. Bunlara kapalı (implicit) sınırlama denir [23].

Sınırlama fonksiyonlarının çoğu tasarım değişkenleri ile ilgili birinci dereceden terimler içerirler. Bunlar doğrusal (lineer) sınırlamalardır. Daha önce belirtildiği gibi doğrusal programlama problemleri sadece doğrusal sınırlamalara sahiptir. Ancak daha genel problemler doğrusal olmayan sınırlamalara da sahiptir. Bundan dolayı hem doğrusal hemde doğrusal olmayan sınırlamaları ele alacak metotlar geliştirilmelidir.

Tasarım problemleri eşitlik sınırlamaları ve/veya eşitsizlik sınırlamaları içerebilirler. Eşitsizlik sınırlamalarını sağlayan bir çok tasarım olabilirken, eşitlik sınırlamalarını sağlayan tasarım bir yüzey üzerinde bulunur. Dolayısıyla eşitsizlik sınırlamalarının kabul edilebilir bölgesi, aynı ifadenin eşitlik sınırlaması olarak belirtildiği durumdaki kabul edilebilir bölgeye göre oldukça büyüktür [23].

Bazı durumlarda birden fazla optimum tasarım olabilir. Bu durum herhangi bir

sınırlamanın amaç fonksiyonuna paralel olması halinde ortaya çıkar. Sınırlama optimum noktada aktif ise problemin çok sayıda çözümü olur.

1.2. Optimizasyon Metotları

Bölgesel ve global optimumların bulunmasına yönelik çalışan optimizasyon metotlarına tek boyutta minimizasyon yapan (Golden Section), çok boyutta arama yapan türeve dayalı olmayan (konjuge yönler), birinci türeve dayalı teknikler (Quassi Newton), ikinci türeve dayalı teknikler (Newton, Levenberg Mardquardt) ve global optimizasyon teknikleri olarak da adlandırılan modern sezgisel algoritmalar (ısıl işlem, tabu araştırma, genetik algoritma vb) örnek verilebilir.

Optimizasyon metotları temel olarak iki kategori altında toplanabilir (Şekil 1.1). Bu gruplar, kesin çözüm üreten klasik teknikler ile yaklaşık optimal çözüm üreten modern sezgisel tekniklerdir. Kesin teknikler de polinomal zaman içerisinde çok iyi bir çözümün bulunması ve sonlu çalışma zamanında problemin her sonlu durumu için optimalliğin korunması garanti edilmektedir [25].

Klasik teknikler çözümlerin üretiliş şekline göre analitik ve yapısal metotlar olmak üzere alt gruplara ayrılmaktadır. Analitik metotlar kategorisinde doğrusal programalama, türeve dayalı metotlar ve bölgesel araştırma metotları bulunmaktadır. Yapısal modeller ise dinamik programlama, ayır ve bağla, böl ve keşfet gibi metotları içermektedir.

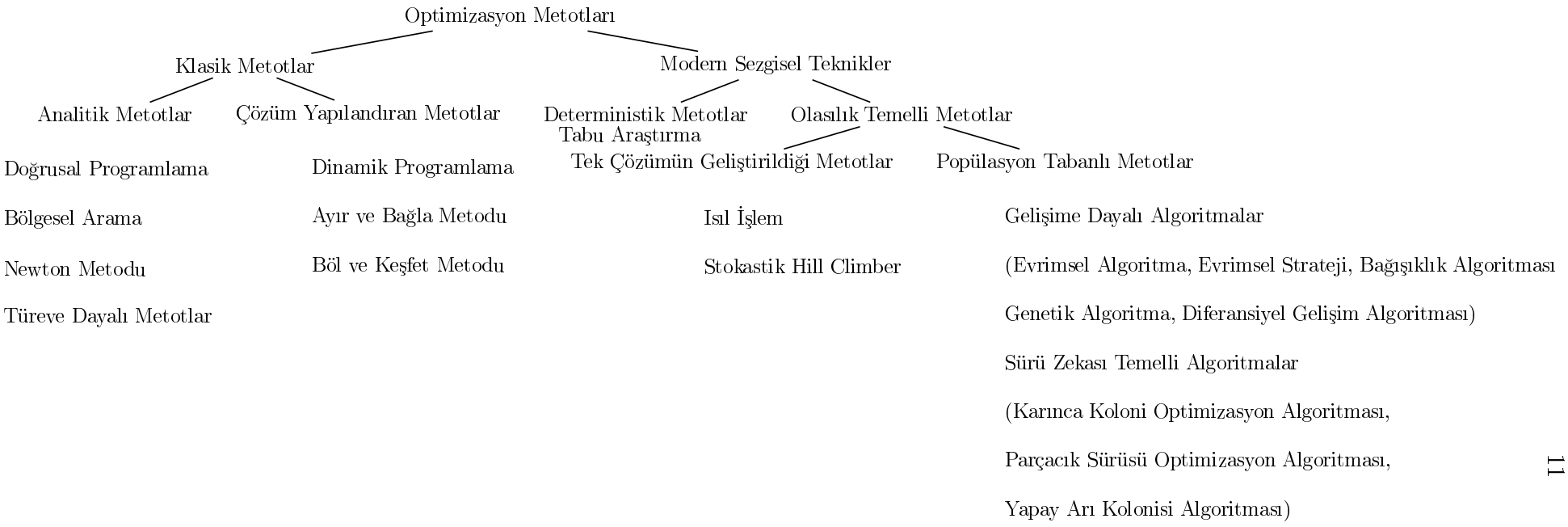
Modern sezgisel tekniklerde genel olarak deterministik ve olasılık temelli metotlar olmak üzere iki alt kategoriye bölünebilir. Deterministik metotlara tabu araştırma algoritması örnek verilebilir. Olasılık temelli metotlar ise tek bir çözümün alınıp iteratif olarak geliştirildiği metotlar veya çözüm kümesinin topluca geliştirildiği popülasyon tabanlı metotlar şeklinde sınıflandırılabilir. Popülasyon tabanlı metotlar örnek aldığı olaya göre özelleşmektedir [15].

1.2.1. Klasik Teknikler

Klasik teknikler, deterministiklik, hızlılık ve tam sonuç verme gibi avantajları söz

konusu olduğundan optimizasyonda bazı durumlarda oldukça faydalı olmaktadırlar. Klasik algoritmaların bir grubu ayrıntılı arama (exhaustive search) fikrine dayanmaktadır. Basit ayrıntılı arama, tüm çözümleri kontrol ederek en iyi çözümü bulmaya çalışır. Bazı durumlarda tüm çözümlerin taranmasına da gerek kalmayabilir. Örneğin, doğrusal programlamada dış bükey araştırma uzayının sınırları içerisinde kabul edilebilir çözümler üretilir. Dinamik programlama, böl ve keşfet (divide and conquer), ayır ve bağla (branch and bound), dallandır ve kes (branch-and-cut) ve Lagrangean Relaxation gibi araştırma metotları araştırma uzayını alt parçalara bölerek çalışmaktadırlar. Detaylı arama yöntemi kullanan algoritmalar her zaman global optimumu bulabilirler. Bölgesel arama ve türev temelli metotlar detaylı aramanın tersine çalışmaktadır. Bölgesel aramada bir çözüm iteratif olarak kendisinden türeyen yeni bir çözüm ile kıyaslanır, yeni çözüm daha iyi ise o anki çözümün yerine geçer. Kendinden türeyen yeni çözüm mevcut çözümün komşusu demektir. Newton metodu gibi türeve dayalı yada sabit nokta (fix point) metotları analitik bölgesel arama metotlarıdır. Türeve dayalı metotlarda türev ya da yaklaşık türev bilgisi hesaplanarak araştırma uzayında hangi yönde hareket edileceği belirlenir. Sabit nokta metotları, bir fonksiyonun ürettiği değerin sabit noktasının bulunması ile optimal değere iteratif olarak yakınsarlar.

Klasik algoritmalarda en iyi değerinin bulunmaya çalışıldığı amaç fonksiyonunun tam olarak, doğru bir şekilde matematiksel yolla ifade edilmesi gerekmektedir. Gerçek dünya problemlerinin çoğunda klasik algoritmaların çözeceği tarzda bir matematiksel model bulunmamaktadır. Bununla birlikte klasik algoritmaların uygulanabildiği problem türü oldukça azdır. Genellikle probleme özgüdürler. Yani, bir algoritma bir problemde çok iyi bir performans sergilerken, başka bir problemde aynı performansı sergileyememektedirler.



Şekil 1.1. Optimizasyon Metotlarının Sınıflandırılması [15]

Klasik tekniklerin çözdüğü problemlerin çoğunun boyutu küçüktür ve problem boyutunun artmasıyla birlikte tekniğin hesaplama maliyeti çok fazla artmaktadır. Ayrıca kesin tekniklerin şu dezavantajları da bulunmaktadır: (i) Hafıza tüketimi problemi programın erken sonlanmasına neden olmaktadır. (ii) Bir problem için çok iyi sonuç veren kesin tekniğin formülasyondaki bir değişime yeniden uyarlanması oldukça zordur. (iii) Bir problem için çok iyi sonuç veren kesin teknik başka bir probleme uyarlanması oldukça zordur. Ancak avantajları da şunlardır: (i) Algoritma başarılı olarak sonuca ulaştığında bulunan çözüm optimumdur. (ii) Alt ve üst sınırlardaki optimum çözüme götüren bilgiler elde edilebilir. (iii) Optimal çözümün olmadığı bölgeleri elemine edebilmektedirler [25].

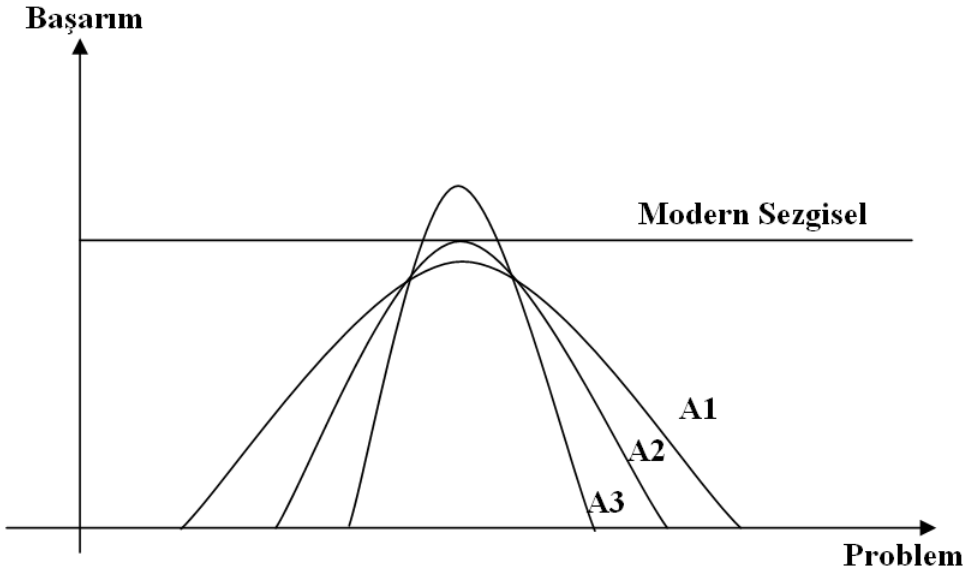
Çok sayıda optimum değerin olduğu varsayılırsa, bölgesel arama stratejileri bölgesel optimumu bulabilirler. Ancak, bölgesel optimuma takılmasını önlenmek için komşuluk boyutu artırılabilir. Fakat bu durumda komşuluk boyutunun artması da algoritmanın yakınsama hızını düşürecektir ve zor problemlerde kabul edilebilir olmayan ve üstel olarak artan bir çalışma süresine neden olacaktır [15].

1.2.2. Modern Sezgisel Algoritmalar

Kesin çözüm üreten metotların yukarıda belirtilen dezavantajlarından dolayı optimum çözümü bulması oldukça zaman almakta veya gerçek dünya problemlerine uygulanamamaktadır. Araştırma uzayı geniş ise hesaplama karmaşıklığını azaltmak için uzayın boyut indirgemesinin yapılması gerekir ki bu da gerçek dünya problemlerinde mümkün değildir. Şayet problemi çözmek için arama metodu kullanılmak istenirse bu durumda da bölgesel optimumlara takılma problemi ortaya çıkmaktadır.

Klasik algoritmalar probleme özgü iken modern sezgisel algoritmalar genel amaçlıdır. Modern sezgiseller değişik problemlere uygulanabilirken, bir problem için özel tasarlanmış bir metot kadar o problem üzerinde iyi sonuç vermeyebilir. Örneğin Lin-Kernighan metodu gezgin satıcı problemi (TSP) için modern sezgisel algoritmalardan daha iyi sonuç verebilir. Ancak gelişime dayalı algoritmalar da daha geniş spektrumdaki problemleri çözmek amacıyla kullanılabilir. Şekil 1.2'de

A1, A2 ve A3 gibi farklı klasik algoritmaların ve modern sezgisellerin farklı problem türlerinde başarımlarının değişimi gösterilmektedir. A1, A2 ve A3 belli problem türlerinde çok iyi başarım sağlarken, modern sezgisel algoritma ise daha geniş problem türü üzerinde iyi performans göstermektedir.



Şekil 1.2. Klasik Algoritmaların ve Modern Sezgisel Algoritmaların Başarım Spektrumu

İteratif araştırma yapan modern sezgisel algoritmalardan ikisi ısıl işlem ve tabu araştırma algoritmalarıdır. Isıl işlem algoritması bölgesel arama metodunun olasılıksal varyantıdır, ancak ısıl işlemin lokal optimumlardan kurtulabilme özelliği vardır [16,17]. Bölgesel arama metodundan esinlenerek ortaya çıkmış bir diğer metot tabu araştırma algoritmasıdır [18,19]. Tabu araştırma algoritması bölgesel araştırma metodundan, daha önce denenen çözümlerin tabu listesini tutması ile farklılaşır. Tabu listesindeki çözümler dışında araştırma yapıldığından tabu listesi araştırma uzayının yeni bölgelerinin araştırılmasını sağlar. Isıl işlemin tersine tabu araştırma algoritması deterministiktir.

Modern sezgisel algoritmaların önemli bir grubu da popülasyon tabanlı algoritmalardır. Bu gruptaki algoritmalar, probleme ait aday çözümlerden oluşan bir popülasyonu çeşitli operatörlerle geliştirerek yaklaşık çözümler üretirler. Popülasyon tabanlı algoritmaları gelişime dayalı algoritmalar ve sürü zekası temelli algoritmalar

olarak iki alt gruba ayırmak mümkündür.

1.2.3. Gelişime Dayalı Optimizasyon Algoritmaları

Modern sezgisel algoritmaların en temeli gelişime dayalı algoritmalardır. Birçok optimizasyon algoritması benzer şekilde operatörler kullanırken, operatörlerin nasıl işlev yaptığı ve hangi durumda devreye girdikleri modelledikleri olaya göre değişmektedir. Gelişime dayalı algoritmaların birçok çeşidi bulunmaktadır. Bunların hepsinin altında yatan temel fikir aynıdır. Bireylerin oluşturduğu popülasyon çevresel etmenlerle en iyi durumun keşfine yönelik doğal seleksiyona maruz kalmakta ve bu da popülasyonun kalitesinin artmasına neden olmaktadır. Maksimize edilecek kalite fonksiyonu için rasgele aday çözümlerden oluşan bir popülasyon üretilir ve bu çözümlerin uygunluk değerleri kalite fonksiyonunda değerlendirilir. Bu uygunluk değerlerine göre bazı bireyler diğer kuşağa aktarılmak üzere seçilirler. Seçilen bireylere yeniden oluşum (rekombinasyon) ve/veya mutasyon operatörleri uygulanır. Yeniden oluşum operatöründe iki yada daha fazla ebeveyn çözüm seçilerek bunlardan yeni bireyler oluşturulur. Mutasyon operatörü tek bir bireye uygulanır ve işlem sonrası yine tek birey oluşur. Bu süreç belli bir adım süresince veya yeterince iyi çözüm bulununcaya kadar devam ettirilir. Bu süreçte iki önemli etken vardır [26]:

- Değişim operatörleri: Popülasyonun farklılığını sağlar.
- Seleksiyon: Popülasyonun kalitesini artırır.

Bu süreç sonunda optimizasyonun gerçekleştiği, en azından çözümlerin optimuma yakınsadığı görülebilir. Gelişim aslında bir uyum süreci olarak da düşünülebilir. Bu durumda uygunluk amaç fonksiyonu cinsinden değil, çevresel gereksinimlerin bir ifadesi olarak karşımıza çıkmaktadır. Gelişim süreci popülasyonun çevresel etmenlere daha iyi uyum sağlamasınına sebep olmaktadır.

Gelişim sürecinin birçok bileşeni stokastiktir. Seleksiyon esnasında yüksek uygunluk değerine sahip bireylerin seçilme olasılığı daha fazla olmasına rağmen düşük kaliteli bireylerinde popülasyonda kalma ihtimalide bulunmaktadır. Yeniden oluşumda

hangi bireylerin hangi parçalarının değişime uğrayacağı, mutasyonda da hangi parçaların mutasyona uğrayacağı rasgele olarak belirlenmektedir. Gelişime dayalı algoritmaların temel adımları şu şekildedir:

1: Başlangıç
2: Değerlendir
3: **repeat**
4: Ebeveynleri seç
5: Ebeveynlerden yeni bireyler oluştur
6: Oluşan yeni bireylere mutasyon uygula
7: Yeni bireyleri değerlendir
8: Bir sonraki kuşağa aktarılacak bireyleri seç
9: **until** Durma kriteri sağlanan kadar

Genel olarak bakıldığında bu algoritmanın üret ve test et mantığına dayandığını görürüz. Uygunluk fonksiyonu çözüm kalitesinin sezgisel kestirimini ifade ederken arama süreci değişim ve seleksiyon operatörleri ile yönlendirilir. Gelişime dayalı algoritmaların üret ve test et grubu içinde yer almasını sağlayan bazı özellikleri:

- Popülasyon tabanlıdırlar. Tüm aday çözümler değerlendirilir.
- Birden fazla aday çözümün bilgisini tek bir çözümde birleştirmek amacıyla yeniden oluşum operatörünü kullanırlar.
- Gelişime dayalı algoritmalar stokastiktir

Gelişime dayalı algoritmaların teknik detaylar açısından farklılık arz eden türevleri bulunmakadır. Örneğin Genetik Algoritmada (GA) çözümlerin gösterimi dizi şeklindedir, Gelişim Stratejilerinde (ES) gerçel değerli vektörler şeklinde, klasik Evrimsel Programlamada (EP) sonlu durum makineleri şeklinde, Genetik Programlama (GP) da ağaç yapısı şeklinde bir gösterim söz konusudur. Bu farklılıklar algoritmaların felsefesinden kaynaklanmaktadır. Eğer probleme bir gösterim daha iyi uyuyorsa o gösterim seçilebilir. Yeniden oluşum ve mutasyon operatörü gösterim şekline uygun olarak uyarlanmalıdır. GP'da yeniden oluşum operatörünün ağaç yapısına uygun olması gerekir. Değişim operatörlerinin

tersine, seleksiyon operatörü sadece uygunluk fonksiyonunu dikkate alır; dolayısıyla gösterimden bağımsızdır [26].

Gelişime dayalı algoritmaların en önemli bileşenleri şunlardır:

- Gösterim (bireylerin tanımı)
- Değerlendirme fonksiyonu (uygunluk fonksiyonu)
- Popülasyon
- Ebeveyn seçme mekanizması
- Değişim operatörleri: yeniden oluşum ve mutasyon
- Hayatta kalacak bireyin belirlenmesi (seçme) mekanizması

Gelişime dayalı bir algoritmanın tanımlanabilmesi bu bileşenlerin her birinin belirlenmesini gerektirmektedir. Ayrıca çalışma sırasında algoritmanın başlatılması için bir prosedür ve sonlandırma kriteri tanımının yapılması da gerekmektedir.

1.2.3.1. Gösterim (Representation)

Gelişime dayalı algoritmalarda ilk adım gerçek dünya problemini bu algoritmanın anlayacağı forma dönüştürülmesidir. Orijinal probleme ait çözümlere fenotip, bu çözümlerin gelişime dayalı algoritmada kodlanmış haline de genotip adı verilmektedir. Gösterim, bazen fenotip uzayı ile genotip uzayı arasında köprü anlamında kullanılırken bazen aday çözümlerin kodlanması anlamına gelmektedir (örneğin binary kodlama gibi). Örneğin tam sayı değerler alan optimizasyon probleminde tamsayılar fenotip iken bunların ikili olarak kodlanması ve algoritmaya sunulması hali genotiptir. Fenotip uzayı ile genotip uzayı birbirinden çok farklı olabilir. Algoritma araştırmayı durdurduktan sonra en iyi genotipin çözülmesiyle (decode) fenotip uzayında çözüm elde edilir [26].

Standart gelişime dayalı algoritmalarda gösterim ikili formdaki bit dizileri şeklindedir. Başka tipte dizilerde kullanılabilir. Bu dizilerin sabit uzunluklu olmasıyla temel yeniden oluşum operatörü kolaylıkla uygulanabilir. Değişken

uzulunluklu dizilerde de kullanılabilir ancak çok daha karmaşık hale gelir. Genetik programlamada ağaç gösterimi kullanılmaktadır.

1.2.3.2. Uygunluk Fonksiyonu (Fitness Function)

Uygunluk fonksiyonu belli bir gösterime sahip bireylerin (genotiplerin) kalitesini ölçer. Seleksiyon için gerekli bilgiyi oluşturur. Uygunluk fonksiyonu değerlendirme biriminin bir bölümünü teşkil eder. Gerçek problem, doğrudan algoritma tarafından çözülecek optimizasyon problemi niteliğinde ise değerlendirme birimi ile problem aynı olmakta ve bu durumda amaç fonksiyonu adını almaktadır.

1.2.3.3. Popülasyon

Popülasyonun rolü, belli formdaki olası çözümleri tutmaktır. Başka bir deyişle genotiplerin kümesidir. Belli bir gösterimdeki popülasyonu tanımlamak popülasyonda kaç birey olacağını yani popülasyon büyüklüğünü belirlemek kadar kolaydır. Bazı daha karmaşık gelişime dayalı algoritmalarda, uzaklık ölçütü yada komşuluk ilişkisi şeklinde ek bir yapı daha vardır. Böyle algoritmalarda ek yapının da tam olarak tanımlanması gerekir. Değişim operatörleri bir yada daha fazla ebeveyn üzerinde işlem yapıyor iken seleksiyon operatörleri (ebeveyn seçimi ve hayatta kalacak bireyin seçimi) genellikle popülasyon düzeyinde gerçekleşir. Genel olarak tüm popülasyonu dikkate alırlar ve seçimler eldeki bireylere göre yapılır. Örneğin bir sonraki kuşağa aktarılmak üzere en iyi birey seçilir yada en kötü birey başka bir birey ile değiştirilir.

Seleksiyon stratejisine ve seleksiyon havuzuna göre farklı popülasyon modelleri tanımlamanabilir: global, bölgesel veya belli bir alanı kapsayan (regional) şeklinde.

Araştırmada tek bir nokta yerine popülasyonun kullanılması, gelişime dayalı algoritmaların bölgesel minimumlara takılmasını engeller.

Popülasyondaki farklılık (diversity) farklı çözümlerin sayısının bir ölçütüdür. Farklılığın ifadesi için tek bir ölçüt yoktur. Farklı uygunluk değerlerinin sayısı, farklı fenotiplerin sayısı, farklı genotiplerin sayısı da farklılık ölçütü olarak kullanılabilir.

Bunlardan farklı olarak entropi gibi istatistiksel ölçütlerde kullanılabilir. Tek bir uygunluk değerinin oluşu, tek bir fenotip olmasını ve tek bir fenotip olması tek bir genotip olmasını gerektirmez. Ancak tersi doğru değildir. Yani, tek bir genotip bir fenotip ve bir uygunluk değerine sahiptir [26].

1.2.3.4. Ebeveyn Seçme Mekanizması

Ebeveyn seçimi yada eş seçimi, kalitelerine göre bireylerin ayırt edilmesi ve daha iyi bireylerin bir sonraki kuşakta ebeveyn olabilmesi için yapılır. Yeni bir birey üretilmesi amacıyla değişime uğramak üzere seçilmiş olan birey ebeveyn ismini alır. Hayatta kalacak bireyin seçilmesi ve ebeveyn seçimi ortalama kalitenin artmasından sorumludur. Evrimsel hesaplamada ebeveyn seçimi olasılık tabanlıdır. Dolayısıyla yüksek kaliteli adayların seçilme olasılıkları düşük kaliteli olanlara nazaran yüksektir. Ancak düşük kaliteli olanlarında seçilme ihtimali vardır. Aksi takdirde seçme işlemi çok aç gözlü (greedy) olur ve bölgesel optimumlara takılır [26].

1.2.3.5. Değişim Operatörleri

Değişim operatörleri aracılığıyla mevcut bireylerden yeni bireyler üretilir. Değişim operatörleri üret ve test et sürecinde üret aşamasında görev alırlar. Evrimsel hesaplamada değişim operatörleri argüman sayılarına göre iki gruba ayrılırlar.

Mutasyon: Doğal gelişimdeki bir organizmanın DNA'sındaki mutasyon olayının rolünün benzetimidir. Gelişim algoritması bir yada daha fazla popülasyon üyesi üzerinde peryodik olarak rasgele değişimler yada mutasyonlar gerçekleştirir. Buda popülasyon üyelerinden daha iyi yada daha kötü olabilen yeni aday çözümlerin oluşmasını sağlar.

Mutasyon operatörü bir genotipe uygulanır ve bu genotipin değişiminden yeni bir birey meydana getirir. Mutasyon operatörü her zaman stokastiktir. Ürettiği çocuk rasgele serilerden oluşur. Farklı evrimsel algoritmalarda mutasyonun rolü de farklıdır. Örneğin, genetik programlamada her zaman kullanılmaz; genetik algoritmada gen havuzunu taze genlerle doldurmak için kullanılır; evrimsel programlamada tek değişim operatörü mutasyondur. Başka bir bakışla değişim

operatörleri araştırma uzayında atılacak adımları belirlermektedir. Yeni bir bireyin üretilmesi araştırma uzayında başka bir noktaya hareket edilmesi anlamındadır. Gelişime dayalı algoritmanın global optimumu bulabileceğini belirten teoremler olası bir çözümü ifade eden her bir genotipin değişim operatörleri ile elde edilebileceği özelliğine dayanır. Bu şartın sağlanmasının en kolay yolu mutasyon operatörünün her noktaya gidebilmesine izin vermektir.

Değişim operatörleri gösterime bağlıdır. Farklı gösterimlerde farklı değişim operatörleri tanımlanmalıdır. Genotipler bit dizileri şeklinde ise 0-1 yada 1-0 dönüşümü şeklinde bir mutasyon operatörü kullanılabilir ancak ağaç şeklinde bir gösterim kullanılıyorsa bu gösterime uygun başka bir mutasyon operatörü seçilmelidir [26].

Yeniden Oluşum (Rekombinasyon): Canlı gelişimindeki yeni oluşumların benzetimidir. Mevcut çözümlerin elemanlarının kombinasyonu ile ebeveynin bazı özelliklerine sahip yeni çözüm üretilmeye çalışılır.

İkili sistemde değişim operatörü, yeniden oluşum yada çaprazlama (crossover) olarak adlandırılır. Adından da anlaşıldığı üzere iki ebeveyn genotipteki bilgileri birleştirerek bir yada iki yeni genotipi oluşturan operatördür. Mutasyona benzer olarak yeniden oluşum da bir stokastik operatördür. Genotipe ait hangi parçaların birleştirileceği yine rasgele olarak belirlenmektedir. Farklı evrimsel hesaplama modellerinde yeniden oluşum operatörü farklı şekilde ele alınabilir. Yeniden oluşum operatörü genetik programlamada tek değişim operatörüdür; genetik algoritmada asıl araştırma operatörüdür; evrimsel programlama da hiç kullanılmaz. Çok sayıda ebeveyn kullanan yeniden oluşum operatörlerinin matematiksel olarak geliştirilmesi mümkündür anca biyolojik bir olayla ilişkisi yoktur. Bu nedenle birçok çalışmada gelişim üzerinde olumlu etkisinin olduğu gösterilmesine rağmen genel olarak çok kullanılmaz [26].

1.2.3.6. Hayatta Kalanı Seçme Mekanizması

Hayatta kalanı seçme yada çevresel seleksiyon kalite değerlerine göre bireylerin ayrılmasıdır. Ebeveyn seçme işlemine benzemesine rağmen gelişim çevriminde

farklı bir yerde kullanılmaktadır. Seçilen ebeveynlerden elde edilen yeni çözümlerin oluşturulmasından sonra devreye girer. Standart evrimsel hesaplamada popülasyon büyüklüğü sabit kaldığından üretilen çocuklarla birlikte hangi bireylerin bir sonraki kuşağa aktarılacağının belirlenmesi gerekir. Bunun belirlenmesi genelde uygunluk değerine göre yapılır. Yaş mantığının da uygulanmasına rağmen uygunluğu fazla olanın seçilmesi daha yaygındır. Ebeveyn seçme işleminin stokastik olmasının tersine, hayatta kalanın seçilmesi deterministiktir. Uygunluk değerlerine göre ebeveynler ve çocuklar sıralandıktan sonra en iyilerin, yada çocuklardan en iyilerin seçilmesi şeklindedir [26].

1.2.3.7. Algoritmanın Başlatılması (Initialization)

Başlangıç işlemleri çoğu gelişime dayalı algoritma uygulamalarında basit tutulmuştur. Öncelikle rasgele seçilen bireylerden bir popülasyon oluşturulur. Popülasyonun büyüklüğü problemin yapısına göre değişebilir. Bu adımda probleme özgü sezgisel işlemler kullanılarak başlangıç popülasyonunun yüksek uygunlukta olması veya araştırmanın optimal çözüm potansyelinin olduğu bölgelerde başlatılması sağlanabilir.

1.2.3.8. Durdurma Kriteri

Uygun bir durdurma kriteri için iki durumdan bahsedilebilir. Problem bilinen bir optimal uygunluk seviyesine sahipse, yani amaç fonksiyonunun bilinen bir optimum değeri söz konusu ise ϵ ($\epsilon > 0$) hassasiyetinde bu seviyeye erişildiği zaman algoritma durdurulabilir. Ancak evrimsel algoritmalar stokastik olduğundan ve bir optimuma erişme garantileri olmadığından yada problemin optimumunun ne olduğu bilinemediği durumlarda bu şart hiç sağlanmayabilir ve dolasıyla algoritma hiçbir zaman durmayabilir. Bu durumda algoritmanın kesinlikle durmasını sağlayan başka bir şart daha eklenebilir. Bu amaçla sıkça kullanılan şartlar şunlardır:

1. İzin verilen maksimum CPU zamanının dolmuş olması,

2. Toplam uygunluk hesaplamalarının belirlenen sayıya ulaşmış olması,

3. Belirli bir zaman periyodu içinde uygunluk değeri gelişiminin belirlenen bir eşik değeri altında kalması,

4. Popülasyon farklılığının (diversity) belli bir eşik değeri altında kalması.

1.2.4. Sezgisel Metotların Performanslarının Değerlendirilmesi

Yapay sinir ağları, gelişim algoritmaları, karınca koloni algoritması, ısıl işlem, tabu araştırma, parçacık sürüsü optimizasyon algoritması gibi birçok sezgisel yaklaşım literatüre sunulmuş ve farklı yeni sezgiseller de sunulmaya devam etmektedir. Sezgiseller yapay zeka alanındaki en başarılı alanlardan biridir. Literatürde sunulan bir çok çalışma bazı ölçüt yada metriklere dayanarak yüksek kalitede sonuçlar ürettiğini iddia etmektedir [27].

Ortaya konulan bir yaklaşımın belirli bir problem seti üzerindeki verimliliği teorik analizlerle ve deneysel çalışmalarla gösterilebilir. Literatüre yeni bir metot sunulduğu zaman, tarafsız olarak katkılarının ve yeniliklerinin bilimsel bir biçimde ortaya konması gerekir. Sezgisel metot kullanılarak yapılan bir çalışma değerlendirilirken genel optimizasyon algoritmaları ile ilgili kullanılan temel birçok kriterin yanı sıra algoritmaya has ek değerlendirme kriterleri de bulunabilir [28].

Aynı verilerden farklı metrikler kullanılarak farklı sonuç çıkarımları yapılabilir. Kesin metotlardan farklı olarak, zaman verimliliği başarı için temel ölçüttür. Çözümlerin elde edilme hızı ve bulunan çözümüm optimumdan uzaklığı arasında bir ödünleşme (trade-off) vardır. Sezgiselleri analiz ederken iki yaklaşım söz konusudur: teorik analiz (en kötü durum analizi, ortalama durum analizi vb.) ve deneysel analiz. Gerçel problemlerde sonuç elde edilememesi ve uygulamasının kısıtlı olması teorik analizin dezavantajlarıdır. Bundan dolayı sezgisel algoritmalar belli uygulamalara yönelik empririk olarak kıyaslanır [29].

Gerçek deneylerden soyutlama olduğu için bir deney yada çalışma algoritmik yöntemlerle dolaylı olarak gerçekleştirilebilir. Deney (experiment) bilinen bir gerçeği göstermek, hipotezin geçerliliğini kontrol etmek, yeni bir algoritmanın performansını görmek gibi bir amacı gerçekleştirmek amacıyla kontrollü şartlar

altında bazı testlerin gerçekleştirilmesidir. Faktör, deneylerde sonucu yada çıkışı etkileyen kontrol edilebilir değişkendir. Bir algoritmayla ilgili hesaplama testlerinde, bilgisayar simülasyonları vasıtasıyla test problemlerine çözüm sağlanır. Deney yapan kişinin test problemlerin seçiminde, algoritmanın gerçekleştiriminde, hesaplama ortamlarının belirlenmesinde, performans ölçütlerinin seçiminde, algoritmaya has seçeneklerin belirlenmesinde ve sonuçların raporlanmasında bilgi sahibi olması gerekir. Her bir faktörün seçimi sonuçlar ve deney başarımı üzerindeki etkisi büyüktür. Elde edilen sonuçların anlamlı olabilmesi için araştırmacı tarafından bunların raporlanması yapılmalı ve faktörlerin olası ihtimalleri göz önüne alınarak deneyler gerçekleştirilmelidir [28].

Bir deney sürecine ilişkin adımlar şu şekildedir:

1. Deneyin amacının ortaya konması,
2. İncelenecek faktörlerin ve performans metriklerinin belirlenmesi,
3. Deneyin tasarlanması ve gerçekleştirilmesi,
4. Elde edilen verinin analiz edilmesi ve sonuç çıkarılması
5. Deney sonuçlarının raporlanması.

Araştırmacı, deneyin amacına yönelik olarak ne tür sonuçlar elde etmek istediğini, hangi hipotezin test edileceğini ve bu bağlamda hangi testlerin yapılacağını, etkisi incelenecek faktörleri, kullanılacak metrikleri ve verileri belirler ve çalışmanın amacını, üzerinde çalışılan sezgisel algoritma ile ilişkilendirir. Algoritmanın kabul edilebilirliği ile ilgili bir standart olmamasına rağmen aşağıda tanımlanan özelliklerden en azından birine katkı sağlaması gerekir [28]:

- Hızlılık: Diğer yaklaşımlara göre daha hızlı olarak yüksek kaliteli çözümler üretmelidir.
- Doğruluk: Diğer yaklaşımların ürettiğinden daha kaliteli çözümler üretmelidir.

- Gürbüzlük: Problem karakteristiklerinden, veri kalitesinden ve parametre değerlerindeki değişimlere karşı diğer algoritmaların olduğundan daha az duyarlı olmalıdır.
- Basitlik: Geliştiriminin kolay olması gerekir.
- Etkililik: Yeni veya önemli bir problemi diğer yaklaşımlardan daha hızlı ve doğru bir şekilde çözmelidir.
- Genellenebilirlik: Çok sayıda problem türünde uygulanabilmelidir.
- Yenilik: Kendi alanında yenilik getirmelidir.

Bunlara ek olarak genel algoritma tasarımının ya da problem yapısının algoritma performansı üzerindeki etkisi ve algoritmanın davranışı açıklanmalıdır. Ayrıca, sınırlamaların çözüm kalitesi üzerindeki etkisi gibi teorik detaylar da verilmelidir.

Algoritmanın diğer algoritmalar ile karşılaştırılmasının yanı sıra performansını karakterize edecek açıklayıcı testler yapılmalıdır. Amaç, algoritmanın faktörlerden nasıl etkilendiğini bulmak ve çalışma mantığını görmektir. Bazı özel faktörlerin algoritma üzerindeki etkisini detaylı olarak görmek amacıyla varyans analizi gibi teknikler kullanılabilir.

Hesaplama testlerinde algoritmaların performansını etkileyen üç ana faktör bulunmaktadır: problem, algoritma ve test ortamı. Her bir kategori test sonuçları üzerinde çeşitli etkilere sahip olduğundan aşağıda belirtilen hususlara dikkat edilmesi gerekir:

- Hangi faktörlerin inceleneceği (problem boyutu, sezgisel metot, işlemci sayısı),
- Sabit durumlar (problem türleri, durma kriteri, hesaplama ortamı),
- Sonuç üzerinde herhangi bir etkisinin olmayacağı düşünülerek ihmal edilecek durumlar (problem maliyetinin dağılımı gibi).

Deney faktörlerinin belirlenmesi, karşılaştırma ve ortaya koyma testlerinin her ikisinde de önemlidir [28].

- **Problem Faktörleri:** Boyut, yapı ve parametrik dağılım gibi çeşitli problem karakteristikleri deney sonuçlarını etkileyebilir. Kodların gürbüzlüğü için de önemlidir. Bir minimum noktada problem boyutunun etkisi (değişken ve eşitlik sayısı gibi) de dahil edilmelidir. Bazı faktörler kolayca kontrol edilebilir ve istatistiksel tekniklerle doğrulukları analiz edilebilir.

- **Algoritma Faktörleri:** Sezgisel metodun seçimi, test edilecek bilgisayar kodlarının seçimi ve kullanılacak içsel kontrol değerleri gibi faktörlerlerdir. Algoritmaya özel parametreler açık şekilde tanımlanmalı ve probleme bağlı ise bu bağlılığın kuralları ortaya konmalıdır. Bu bağlılık istatistiksel analizle açıklanabilir. Durma kriteri de algoritma faktörleri arasındadır.

- **Test Ortamı Faktörleri:** Yarıştırılan algoritmaların aynı programcı tarafından kodlanıp, aynı test problemleri üzerinde, aynı bilgisayar konfigürasyonu kullanılarak koşulmalıdır. Zaman ve çözüm kalitesi bakımından her problemden alınan sonuçlar karşılaştırılabilir. Bu şekilde donanım (model, hafıza, CPU hızı ve sayısı vs), yazılım (işletim sistemi, dil, derleyici vs), sistem ve programcı (tecrübesi ve ayar kabiliyeti) faktörleri kontrol edilebilir.

Aslında belli bir problem üzerinde algoritma test edilirken şu sorulara cevap verilmesi gerekir:

1. Bulunan en iyi çözüm ne kadar kaliteli?
2. En iyi çözümün bulunması ne kadar sürmekte?
3. Metot ne kadar gürbüz?
4. En iyi çözüm ile diğer bulunan çözümler arasındaki fark nedir?
5. Kabul edilebilirlik ve çözüm kalitesi arasındaki ödünleşme (trade-off) nasıldır?

Bu sorulardan da anlaşılacağı üzere performans ölçütlerini üç başlık altında toplamak mümkün: çözüm kalitesi, hesaplama maliyeti ve gürbüzlük. Her bir kategorideki ölçütler çok iyi incelenmelidir.

1.2.4.1. Çözümlerin Kalitesi

Bir problem üzerinde algoritmayı test ederken, optimal çözüme yakınsama hızı ve oranı önemli konulardır. Sezgisel algoritmalar için sezgisel olarak bulunan çözümün optimuma yakınlığı en temel konudur. Sezgisel çözümlerin, optimum çözüm var ise optimal çözümle karşılaştırılması gerekir ve genellikle optimumdan yüzde olarak ne kadar saptığı verilir. Optimum çözüm yoksa, alt sınıra olan uzaklık bir ölçüt olarak kullanılmaktadır.

1.2.4.2. Hesaplama Maliyeti

Sezgisel algoritmalar iyi çözümler üretebilmeli ve aynı zamanda da hesaplama hızı iyi olmalıdır. Yani hız önemli bir faktördür.

- En iyi çözümün bulunma zamanı: Bu süreye tüm ön süreç boyunca gerçekleşen işlemlerde katılır (uzaklık matrislerinin hesaplanması gibi).
- Toplam koşma süresi: Bazı karmaşık algoritmaların durma kriteri olmadığından en iyi çözümün bulunduğu ve bunu üreten koşmanın toplam zamanı alınabilir.
- Her bir aşamaya ait süre: Başlangıç, geliştirme ve sonlandırma aşamaları ayrı olan karma yapılı algoritmalarda, her bir faz için gereken zaman ve her bir faz sonundaki çözümlerin kalitesi verilmelidir.

Farklı sistemlerde zamanın ölçülmesinde farklı yollar kullanılabilir: kullanıcı saati, sistem saati ve gerçek saat. Paralel gerçekleştirimler ek zaman komplikasyonları getirmektedir.

Koşma zamanları bir sistemden diğerine tam olarak dönüştürülemeyebilir. Bundan dolayı farklı bir ölçüt kullanmak daha uygun olabilir. Veri güncelleme sayıları, arama ağacındaki düğümlerin sayısı gibi ayrık ölçütler çalışma zamanı ile daha iyi ilişkilendirilebilir.

1.2.4.3. Gürbüzlük

Eğer bir sezgisel metot sadece belli bir problem türü üzerinde çok iyi sonuçlar üretiyorsa bu metot gürbüz değildir. Genel olarak gürbüzlük, çok farklı karakteristik özellikli test problemleri üzerinde iyi performans gösterebilmeye bağlıdır. Sezgisel metotlar ve kontrol parametreleri test problemlerine göre değişmemelidir. Yada problemin karakteristiğine göre otomatik olarak belirlenmelidir. Parametre değişimlerine karşı algoritmanın performansının değişimi de gürbüz olup olmadığının bir göstergesidir. Sonuçların olumsuz olanlarının da dikkate alınması gerekmektedir. Eğer algoritma birçok problem türü üzerinde iyi performans gösterip, özel bir problem türü üzerinde başarısız oluyorsa bunun saklanmaması gerekir ve algoritma kabul edilebilir bir çözüm bulmayı garanti edemiyorsa, bunun hangi şartlar altında gerçekleştiğinin ortaya konulması gerekir.

2. BÖLÜM

SÜRÜ ZEKASI

Sürü terimi kollektif davranış sergileyen balık, kuş kümeleri, karınca, termit ve arı kolonileri gibi hayvan ve böcek toplulukları için kullanılmaktadır. Bir sürüdeki bireyler herhangi bir danışma işlemi olmaksızın hareket ederler ve komşuluk bazında algılama yetilerinden dolayı bu davranışları stokastik yapıdadır. Küresel düzeyde herhangi bir ilişkileri olmaksızın bölgesel kurallar vasıtasıyla kendi kendine organize olabilen bireyler arasındaki etkileşim sürü zekası adı verilen kollektif zekanın ortaya çıkmasına sebep olmaktadır. Sürünün kollektif zekası sayesinde çevre ve kaynaklar verimli bir şekilde kullanılır. Kendi kendine organize olabilme sürü sisteminin temel özelliğidir. Bu özellik sayesinde daha düşük seviyeli (mikroskobik) bölgesel etkileşimler ile küresel seviyede (makroskobik) cevap oluşmaktadır. Bonabeau kendi kendine organize olabilmeyi dört özellik ile karakterize etmektedir [30]:

1. *Pozitif geri besleme*: Elverişli ve uygun durumların oluşmasını destekleyen davranışsal kurallar sayesinde oluşur. Karıncaların kimyasal madde yaymaları yada arıların dans etmeleri vasıtasıyla bireylerin bir işe yönlendirilmeleri pozitif geri beslemeye örnek olarak verilebilir. Yani yapılan bir işin daha çok yapılması için gerekli koşullar oluşturulur.

2. *Negatif geri besleme*: Sürünün kollektifliğini kararlı kılabilmek için pozitif geri beslemeyi dengeler nitelikte çalışan kurallar ile oluşur. Belli bir iş konusunda, yiyecek tüketiminde, yada yarışma durumlarında doyuma gitmeyi önlemek için negatif geri besleme mekanizmasına ihtiyaç duyulur.

3. *Dalgalanmalar*: Rasgele keşifler veya hatalar gibi bireylerin mevcut işlerini değiştirmeleri ile yeni çözümlerin keşfini sağladığından yaratıcılık ve yenilik

açısından önem taşımaktadır.

4. *Çoklu etkileşimler*: Sürüdeki bazı bireylerin diğer bireylerden gelen bilgiyi almalarını ve böylece sürü içince bilgi dağılımını sağlar.

Uzmanlaşmış bireylerin işlerini paralel olarak gerçekleştirmelerini sağlayan *iş bölümü* (division of labor) özelliği, kendi kendine organize olabilme gibi sürüye ait önemli bir özelliktir. Dışardan ve içerden gelen değişimlere karşı verilen tepkilerin esnekliği iş bölümünün bir başka boyutudur [31–34].

Millonas bir sürünün zeki olarak tanımlanabilmesi için sürünün sağlaması gereken beş temel özelliğin sağlanması gerektiğini ifade etmiştir [35]:

a) Sürü, temel uzay ve zaman hesaplamalarını yapabilmelidir (yakınlık prensibi).

b) Yiyeceklerin kalitesi veya yerin güvenliği gibi çevresel etkenleri değerlendirebilmeli ve tepki verebilmelidir (kalite prensibi).

c) Tüm kaynaklarını dar boğazlarda kullanmamalı ve kaynakları birden fazla noktaya dağıtabilmelidir (dağılım cevabı prensibi).

d) Çevrede oluşan her bir değişimde veya dalgalanmada çalışma modunu değiştirmemeli ve kararlılığını koruyabilmelidir (kararlılık prensibi).

e) Enerji tüketimine değecek maliyette ise çalışma modunu değiştirebilmelidir (uyarlanabilirlik prensibi).

Hayvan davranışlarını inceleyen bilimciler (ethologist) bir sürünün davranışını yukarda verilen özelliklerle mikro ve makro düzeyde modellemişlerdir [36]. Son zamanlarda araştırmacılar bu modellerden ilham almışlar ve zor olan gerçek hayat problemlerinin (trafik ve network yönlendirme, oyunlar, endüstriyel problemler, robotik, ekonomi problemleri) çözümü için sürü zekasına dayalı yeni problem çözme teknikleri geliştirmişlerdir. 1990'larda geliştirilen özellikle iki yaklaşım (1991 yılında Dorigo tarafından tanımlanan karınca koloni temelli algoritma [21] ve 1995 yılında Kennedy ve Eberhart tarafından kuş ve balık sürülerini temel alan parçacık sürüsü optimizasyon [37] algoritması) oldukça dikkat çekmiştir. Her iki yaklaşım da birçok

araştırmacı tarafından çalışılmış ve farklı alanlarda birçok probleme uygulanmıştır. Literatüre uygulamaları ile ilgili binlerce yayın sunulmuştur. Bu çalışmaları gözden geçiren araştırmalar (survey) da yine literatürde bulunmaktadır [38–41].

Bonabeau tarafından tanımlanan kendi kendine organize olabilme [30] ile iş bölümü ve Millonas tarafından tanımlanan sürü zekası için gerekli olan prensipler [35] arı kolonilerinde de görülmesine rağmen, arılarda varolan sürü zekasına dayalı problem çözme tekniklerinin geliştirilmesi 2000'li yılların başından itibaren başlamıştır. Takip eden bölümlerde 2000'li yıllardan bu yana arıların sürü zekasını temel alarak yapılan çalışmalar verilecektir.

Bir sonraki bölümde arı koloni sistemi özetlenecek, yukarıdaki bilgiler ışığında zeki sayılabilecek davranışları sunulacaktır.

2.1. Arıların Doğası ve Arılardaki Sürü Zekası

Doğadaki en ilginç sürülerden biri işlerini dinamik olarak dağıtabilen ve çevresel değişimlere karşı topluluk zekalarıyla uyarlanabilir cevaplar verebilen bal arısı sürüsüdür. Bal arıları fotoğrafik hafızaya, uzay çağı sensörlerine, navigasyon sistemine, sezgisel kavrama yeteneğine, yeni yuva yeri seçerken grup olarak karar verme özelliğine sahiptirler. Ayrıca yiyeceklerin saklanması, balın getirilmesi ve dağıtılması, iletişim ve yiyecek arama gibi işleri de en uygun şekilde yerine getirebilmektedirler. Bu karakteristikler araştırmacıları arıların davranışlarını modellemeleri yönünde teşvik edici olmuştur. Bu davranışları temel alan algoritmalara ve uygulamalarına geçmeden önce koloni davranışları tartışılcaktır.

Arılar koloniler halinde yaşayan sosyal böceklerdir. Bir kolonide üç çeşit arı bulunmaktadır: kraliçe arı (queen), erkek arı (drone) ve dişi olan işçi arılar (worker).

Kraliçe Arı (Queen Bee): Kraliçe arı bir kaç yıl yaşayabilmektedir. Kolonideki tüm arıların annesi olan kraliçe arı yumurtlama özelliğne sahip tek arıdır. Kraliçe arı genellikle yaşamı boyunca bir kez çiftleşir ve bu çiftleşmede depoladığı spermlerle

iki yıl boyunca yumurtlayabilir. Spermler tükenince döllenmemiş yumurtalar üretir ve kraliçenin kızlarından biri yani kolonideki dişi arılardan biri yumurtlamanın devamı için kraliçe olarak seçilir. Bir yumurta sırasıyla larva, pupa ve yetişkin arı evrelerinden geçer. Koloninin yiyecek miktarı azaldığı zaman kraliçe arı yumurtlamaya başlar. Koloni yeterince kalabalıklaştığı zamanda ise kraliçe arı yumurtlamaya ara verir. Sağlıklı bir kraliçe günde 2000 yumurta ve ortam şartlarına bağlı olarak yılda 175 000-200 000 arasında yumurta üretir.

Erkek Arılar (Drones): Erkek arılar koloninin babalarıdır. Erkek arılar döllenmemiş yumurtalardan oluşurken kraliçe ve işçi arılar döllenmiş yumurtalardan oluşurlar. 6 aydan daha fazla yaşamazlar. Yaz aylarında kolonide bir kaç yüz tane erkek arı bulunur. Erkek arıların temel görevi kraliçe arının döllenmesidir. Erkek arılar kraliçe arı ile çiftleşmeleri sonucu ölürler.

İşçi Arılar (Workers): İşçi arılar yiyecek toplanmasından, toplanan yiyeceklerin saklanmasından, ölü arıların ve molozların ortadan kaldırılmasından, kovanın havalandırılmasından ve güvenliğinden sorumludur. İşçi arılar, kraliçenin içerisine yumurtladığı bal mumu hücrelerini yaparlar ve tükürük bezlerindeki özel bir salgı ile kraliçe arıyı, erkek arıları ve larvaları beslerler. İşçi arının yapacağı görev yaşına ve koloninin gereksinimlerine bağlıdır. Yaşamının ikinci yarısında yiyecek arayıcı (forager) olarak çalışır. İlk başlarda kovanın yerini ve çevre topolojisini öğrenebilmek için kısa uçuşlar yapar. Yaz aylarında 6 hafta, kış aylarında ise 4-9 ay arasında yaşayabilmektedirler.

Çiftleşme uçuşu (Mating-Flight): Kraliçenin yuvadan uzaklara yaptığı uçuşlarda çiftleşme gerçekleşir. Çiftleşme uçuşu kraliçenin gerçekleştirdiği bir dans sonrasında başlar. Uçuş sırasında erkek arılar kraliçe arıyı takip ederler ve çiftleşme gerçekleşir. Bir erkek arının kraliçe arı ile çiftleşme olasılığı kraliçenin hızına ve kraliçe ile erkek arının uygunluğuna bağlıdır. Erkek arının spermleri kraliçenin spermetika adı verilen bir organında depolanır ve yeni yavruların genetik havuzunu

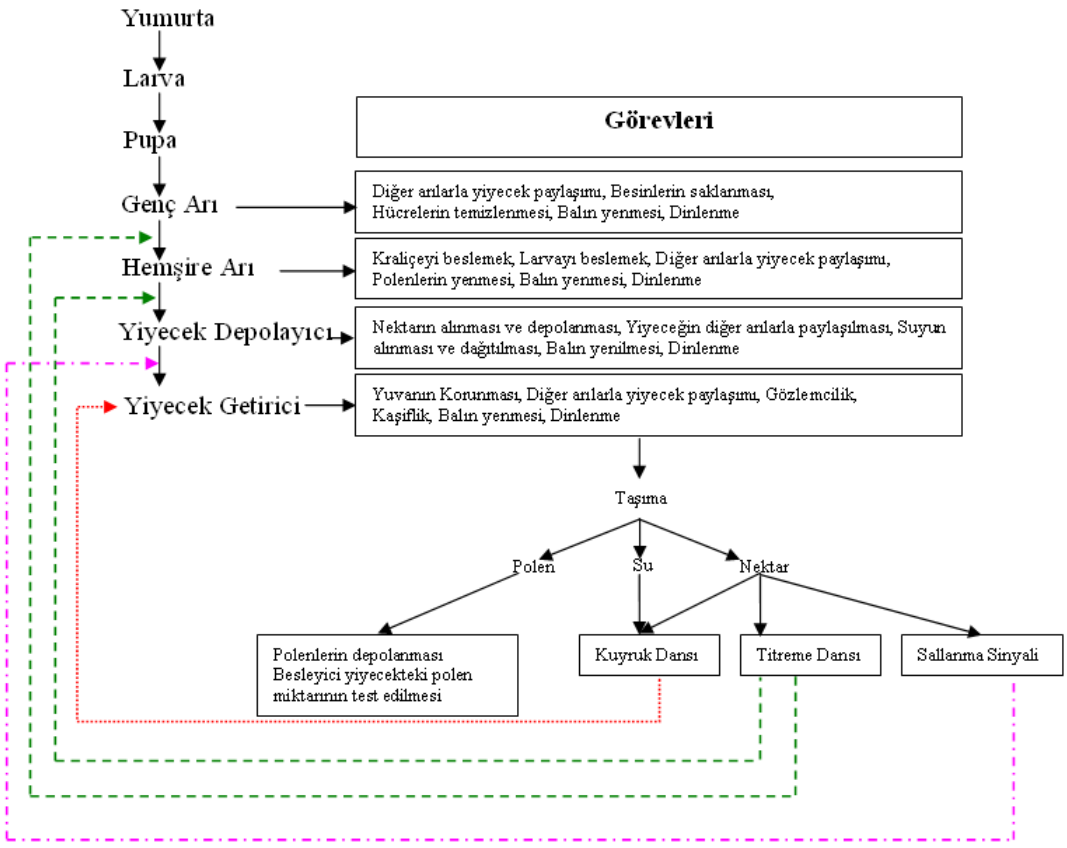
oluşturur.

Görev Paylaşımı (Task Selection): Bal arılarının kolonisinde mevcut iş gücünün dağıtılması gereksinimi olduğu için çok sayıdaki farklı iş için uygun sayıda bireyin tahsis edilmesi gerekmektedir [42]. Arılar kovandaki işlerin gerçekleştiriminde uzmanlaşmışlardır. Bu uzmanlaşmada ne tür faktörlerin (yaş, hormonlar, genetiklerinden kaynaklanan bireysel yatkınlıklar gibi) rol oynadığı konusunda çelişkiler vardır [43].

Şekil 2.1'de farklı davranışsal rollere ait aktiviteler ve bu rollere atanmış işçi sayısını düzenleyecek şekilde işçilerin rollerini değiştirecek geri beslemelerin şematik gösterimi verilmektedir [43]. Bir arı hayatına kraliçe arı tarafından bırakılan bir yumurta olarak başlar. Yeterince beslendikten sonra larvaya ve pupaya dönüşür ve en sonunda da genç arı olarak yumurtadan çıkar. Dolayısıyla algıladığı ilk işi beslenmek olacaktır. Yaşının ilerlemesi ile beslenme alanını terk edip, yeni görevler keşfedecektir. Yetişkin arılar yuva çıkışında dururken, genç arılar ve hemşire arılar beslenme alanında çalışırlar. Bu uzmanlaşmada yaşın dışında vücut salgılarının da rol oynadığına dair modeller bulunmaktadır. Yumurtadan yetişkin birey olmaya kadar geçen dönemde her arının besleme, depolama, bal ve polenlerin elde edilmesi ve dağıtılması, iletişim ve yiyecek arama gibi görevleri bulunur. Belli bir anda bir arının hangi işi gerçekleştireceği o anki davranışsal rolüne, çevreden topladığı algılara ve bu algılara gösterdiği tepki eşiğine bağlıdır. Davranışsal roller, arılar arasındaki fizyolojik farklılıkları modellemektedir. Bu nedenle bazı organların belli işleri yapan arılarda gelişmiş olması gerekir. Arıların yaşları ilerledikçe algıladıkları sinyallere (titreme dansı, sallanma sinyali gibi), içgüdülerine ve iş bulmak için gereksinim duydukları süreye göre görevleri değişebilmektedir. Bir sonraki görevin ne olacağının belirlenmesi harici bir uyarıcıya ve aktiviteye özel eşik değerine bağlıdır.

Görev dağılımı dinamik olarak da değişebilmektedir. Örneğin yiyecek tükendiğinde, hemşire arılar da yiyecek arama sürecine katılabilmektedirler.

Yiyecek Arama (Foraging): Yiyecek arama kovandaki en önemli aktivitedir. Bal arılarının yiyecek arama davranışlarını ne tür dış etkenlerin (koku, waggle dansdan gelen konum bilgisi, başka arılarında kaynağa yönlenmiş olmaları gibi) ve



Şekil 2.1. Arılarda Görev Paylaşımı

iç etkenlerin (hafızada tutulan kaynak yada koku gibi) etkilediğini inceleyen bir çok çalışma yapılmıştır [44–46]. Yiyecek arayan arılar kovanın dışında çalışmaktadırlar. Kovan dışından bal, polen ve su ile dönmektedirler. Yiyecek arama süreci bir yiyecek arayıcının (forager) kovandan ayrılmasıyla başlar. Kendine bir çiçek bulduktan sonra, arı bu çiçeğin nektarını bal deposunda saklar. Çiçeğin nektar yoğunluğu ve çiçeğin kovandan uzaklığı gibi durumlara bağlı olarak deposunu 30-120 dakika boyunca doldurur ve bu deposundaki nektarlar üzerine bir enzim salgılamasıyla bal yapma süreci başlar. Kovana döndükten sonra deposundaki nektarı boş petek hücrelerine boşaltırlar ve fermante olmaması ve bakterilenmemesi için ekstra maddeler eklerler. Bal ve enzimlerle dolan hücreler bal mumu ile kapatılırlar.

Buldukları kaynakla ilgili diğer arılara bazı hareketler yaparak bilgi aktarmaktadırlar. Bu hareketlere dans adı verilmektedir. Her bilgi aktarımında özellikle yiyecek kaynakları hakkında haberleşmek için özel bir dans çeşidi bulunmaktadır. Depolayıcı arı yiyecek getiricilerden toplanan nektarları alır ve

petekler içerisinde saklar. Kaşif arılar (scout) yeni kaynaklar arayışı içerisinde olan arılardır. Temel olarak kaşif ve yiyecek getiren arılar arasında farklılık yoktur. Yiyecek getirici arılar, belli bir nektar kaynağına bireysel derecede bağlıdırlar. İşçi arılardan bazıları kaynaklarını bırakarak kaşif arı haline gelir ve rasgele yiyecek aramaya başlarlar. Kaşif arıların sayısı tüm koloninin % 5-10'u arasında değişmektedir. Rasgele araştırma yapan bu kaşif arılar, yeni bir kaynak keşfettikleri zaman tekrar nektar toplamaya başlamakta ve yiyecek getirici arı halini almaktadırlar.

Dans: Yiyecek getirici arıların hepsi kaynaklarını kendileri bulmamaktadır. Diğer arıların çoğu, yiyecek getirici arıların bilgi aktarımı ile kaynak bulmaktadırlar. Kaynağın bulunabilmesi için kaynağa ait uzaklık ve yön bilgisinin verilmesi gerekir. Arılar sağır olduklarından, nektar kaynağından en iyi şekilde faydalanabilmek için bu kaynağa gidip gelen yiyecek getirici arılar gerçekleştirdikleri dansla bunu arkadaşlarına bildirirler. Bu arı dans ederken diğer arılarda ona antenleri ile dokunurlar ve bulduğu kaynağın tadı ve kokusu ile ilgili bilgi alırlar. Daha fazla arı yönlendirebilmek için kovandaki çeşitli alanlarda bu dansı gerçekleştirir ve kaynağına geri döner.

Yuva Yerinin Seçilmesi (Nest Site Selection): Kovan yapılacak yer belirlenirken, arılar yerin petekleri tutabilecek kadar ama en az darlıkta kavise sahip olabilme şartını, hava koşullarını, inşa süresi gibi hususları dikkate alırlar. En önemli husus ise tüm kovanın herhangi bir fikir ayrılığı olmadan tek bir yere karar vermesidir. Bunu başarabilmek için paralel çalışan çok sayıda kaşif arı, potansiyel yuva yerlerini araştırırlar ve buldukları yerlerle ilgili bilgileri dans aracılığı ile diğer kaşif arılarla paylaşırlar. Kuyruk dansı aracılığıyla diğer kaşif arıları etkileyerek bazı koalisyonlar kurulur. Dans aracılığıyla iyi olduğu iddia edilen kaynak yerinde incelendikten sonra gerçekten iyi ise kaşif arı o koalisyona katılır. Bu inceleme ile daha kötü olan yerlerin seçilmesi engellenmiş olur [47].

Navigasyon: Yiyecek arayıcı arılar yuva ile yiyecek kaynakları arasındaki uzaysal konum bilgisini haritaya benzer bir yapı ile kodlarlar. Bu yapı iki vektör arasındaki hesaplamalara veya noktalara dayanmaktadır. Hangisinin gerçekten

doğru olduğu bilinmemekle birlikte iki yaklaşım vardır. Birincisi arılar uçuşları sırasında uyarıcıdan faydalandıkları, ikincisi ise danslarındaki koordinat bilgisini harita vari bir uzaysal hafızaya kodladıkları yönündedir [48].

2.2. Arılardaki Sürü Zekasını Temel Alan Çalışmalar

Literatürde arıların önceki bölümde tanımlanan davranışlarını temel alan çeşitli optimizasyon yaklaşımları ve bunların problem çözümünde kullanımı mevcuttur. Bu bölümde temel alınan davranışa göre yaklaşımların gruplaması yapılarak tanıtılacaklardır.

2.2.1. Kraliçe Arı Benzetimleri

Jung, Genetik Algoritma (GA)'nın yeniden üretim (reproduction) mekanizmasına kraliçe arının davranışını katarak Kraliçe Arı Gelişim Modelini (KAGM) ortaya koymuştur [49]. Bu metot ile GA'nın araştırma performansı ve bulunun çözümlerden faydalanma süreçleri iyileştirilmiştir. Quin ve ark., KAGM algoritmasını doğrusal olmayan, sınırlamalı, karmaşık bir optimizasyon problemi olan ekononik güç sevkiyatı (economic power dispatch) problemine uygulamışlardır [50]. Azeem ve Saad, KAGM'yi ağırlıklandırılmış çaprazlama operetörü kullanarak değiştirmişler ve iki karmaşık doğrusal olmayan sistem için bulanık mantık bilgi tabanı kontrolörünün giriş ve çıkış ölçekleme faktörlerini ayarlamak amacıyla kullanmışlardır [51]. Azeem bu çalışmayı 4 farklı tip sisteme uygulamıştır [52].

Karci, bal arılarının çiftleşmelerinden esinlenerek bir çaprazlama operatörü geliştirmiştir [53]. Bu operatör çaprazlamanın ebeveyn bireyi olarak sırasıyla en iyi ve en kötü uygunluk değerine sahip çözümü kraliçe arı olarak seçmektedir.

Xu ve ark., bazı kombinasyonel ve termodinamik sınırlamaları sağlayan DNA dizilerinin tasarımı için Arı Sürüsü Genetik Algoritmasını (Bee Swarm Genetic Algorithm) geliştirmişlerdir [54]. Bu algoritmada GA'nın popülasyon bilgisinden faydalanma (exploitation) kabiliyetini artırmak için populasyonun en iyi bireyi kraliçe arı olarak seçilmekte ve popülasyondaki farklılığı artırmak için de rasgele popülasyon oluşturulmaktadır.

Lu ve Zhou, Çoklu Arı Popülasyonu Gelişim Algoritması (Multi-bee Population Evolutionary Algorithm,MBPEA) üzerine inşa edilmiş bir GA sunmuşlardır [55]. Bu algoritmada, popülasyonlardan birisi MBPEA tarafından üretilirken diğerleri rastgele seçilmektedir. Her bir popülasyonun iyisi olan kraliçe arı seçilen bir çözüm (drone) ile çaprazlama işlemine tabi tutulmaktadır.

Xiong ve ark., GA içerisinde kraliçe arı çaprazlama operatörünü kullanarak etiket-sınırlamalı minimum kapsayan ağaç (label-constrained minimum spanning tree) problemi için GA'yı daha verimli hale getirmişlerdir [56].

2.2.2. Dans ve Haberleşme Benzetimleri

Sato ve Hagiwara, arıların davranışlarına dayanan, Bee System olarak adlandırdıkları GA geliştirmişlerdir [57]. Modelde bir arı yiyecek bulduğunda yiyecek ile ilgili bilgiyi dans ederek diğer ortak çalıştığı arılara iletmektedir. Arılar GA'nın kromozomlarına karşılık gelmekte ve her bir kromozom tek başına iyi bir çözüm bulmaya çalışmaktadır. Bir kromozom diğerlerine göre daha iyi olduğunda, diğer kromozomlar çoklu populasyon oluşturarak bu kromozom komşuluğunda iyi bir çözüm arayışına girmektedirler. Yaptıkları çalışmalarda Sato ve Hagiwara, Bee System'in klasik GA'ya göre daha iyi başarım sergilediğini göstermişlerdir [57].

Walker, bal arılarının bilgiyi paylaşım ve işleme davranışlarının benzetimini yaparak bilgisayar içindeki, bir yerel ağdaki (LAN) ve tüm interneti kapsayan geniş alan ağlarındaki (WAN) bilgi akışına uyarlamıştır [58].

Gordon ve ark., aralarında haberleşebilen özdeş otonom robotların gridi üzerinde örüntü oluşturulması problemini çözmek amacıyla bir yaklaşım önermişlerdir [59]. Ayrık Arı Dansı (Discrete Bee Dance, DBD) adı verilen bu algoritma grid üzerinde koordine edilen arı dansları serisinden oluşmaktadır. Dans aracılığı ile robot ajanlar bilgi paylaşmakta ve karar vermede ortak bir çalışma sergilemektedirler.

Wedde ve ark., kovandaki haberleşmeden esinlenerek BeeHive adında bir algoritma geliştirmişler ve ağlarda yönlendirme problemini çözmek için kullanmışlardır [60]. Algoritmada, ajanlar yiyecek arama bölgeleri adı verilen ağdaki alanlara doğru

gitmekte ve yönlendirme tablolarının güncellenmesi için ağın durumu ile ilgili bilgi paylaşımını gerçekleştirmektedirler [61, 62]. Wedde ve Farooq, BeeHive algoritmasının performansını literatürdeki diğer algoritmalarla karşılaştırmışlardır [63]. Wedde ve ark., BeeHive algoritmasının güvenlik tehditlerini ortadan kaldırmak için kendilerinin güvenlik modeli ile genişleterek algoritmaya BeeHiveGuard ismini vermişlerdir [64]. Wedde ve ark., yapay bağışıklık sistemi ile BeeHive'i biraraya getirerek BeeHiveAIS adını vermişler ve BeeHiveAIS ve BeeHiveGuard'ın performanslarını karşılaştıran bir deneysel gerçekleştirim çatısı tasarlamışlardır [64]. Sonuçlar, BeeHiveAIS'ın işlem ve haberleşme maliyetinin BeeHiveGuard'a göre çok az olmasına rağmen BeeHiveAIS'ın BeeHiveGuard ile aynı seviyede güvenlik sağladığını göstermiştir [65]. Wang ve ark., BeeHive algoritmasına dayanan "her zaman en iyi bağlantılı olan desteklenir" yönlendirme şeması önermişlerdir [66].

Wedde ve ark., BeeHive algoritmasını kullanarak araba ve kamyon yönlendirmelerinin algoritmalarla gerçeklendiği çoklu katmanlarda merkezden yönetilmeyen çok ajanlı bir yaklaşım geliştirmişler ve BeeJamA ismini vermişlerdir [67]. Klasik yaklaşımlara göre üstün performans sergilediğini raporlamışlardır [68].

Navrat, örün aramasındaki süreçleri modellemek amacıyla dans alanı, toplantı alanı ve görev dağıtım bölmesinden oluşan arı kovanını temel alan bir yaklaşım (Bee Hive Metafor, BHM) önermiştir [69, 70]. Navrat ve ark., kullanıcının önceden tanımlı sayfalarında online arama için BHM'yi kullanmışlardır [71]. Yazarlar, BHM'nin arama yaparken en iyi rotayı takip ettiğini kötü olanları elediğini belirtmelerine rağmen kıyaslamalı bir çalışma henüz yapılmamıştır.

Olague ve Puente, görme hesaplamalarında kullanılabilecek gelişmiş aralıklı yeniden yapılandırma (improved sparse reconstruction) elde edebilmek amacıyla 3-B noktaların arılarda olduğu gibi birbiri ile haberleştiği Bal Arısı Arama (Honey Bee Search, HBS) algoritması geliştirmişlerdir. Yapılan çalışamalardan önerilen haberleşme sisteminin aykırı değerleri azalttığı sonucuna varılmıştır [72].

2.2.3. Görev Paylaşımı Benzetimleri

Nakrani ve Towey, isteklerin yerine getirilmesi için sunucuların dinamik olarak tahsis edilmesini sağlayan ve merkezi olmayan bir bal arısı (Honey Bee, HB) algoritması önermişlerdir [73]. Modelde, sunucu tahsisi ve bal arılarındaki arayıcı tahsisi olaylarının benzetiminden yola çıkmışlardır. Benzetilmiş istek dizileri ve ticari veriler üzerinde algoritmanın performansı bilge (omniscient) optimallik algoritması ile kıyaslanmıştır. Bal arısı algoritması fazla değişkenli istek yüklerinde statik ve aç gözlü algoritmalardan daha iyi sonuç verirken az değişkenli isteklerde aç gözlü algoritma daha iyi sonuç vermiştir [73]. Nakrani ve Towey başka bir çalışmalarında bal arısı dans protokolünü internet evsahibi (hosting) merkezlerindeki otonom sunucu orkestrasyonuna uygulamışlardır [74]. Yazarlar kendi kendine organize olabilen bal arısı modelini veri akışı ve geribesleme türünde tanımlamışlar ve iki problem arasındaki benzerliği analiz ederek hosting merkezleri için biyolojik taklitçi (biomimetic) algoritmayı türetmişlerdir. Deney sonuçları, algoritmanın oldukça fazla değişen dış çevreye karşı oldukça adaptif ve başarılı bir performans sergilediğini göstermiştir [75].

Gupta ve Koul, arı kovanının işçi arılar ile kraliçe arı tarafından yönetimine dayalı Swan adında bir mimari geliştirmişler ve ağ yazılımlarındaki yetersizlikleri giderebilmek amacıyla IP ağların yönetiminde kullanmışlardır [76].

Bal arısı ile bireylerin arasındaki takım çalışmaları arasındaki benzerlik Sadik ve ark.'nı iş verimliliğini artıracak bir takım mimarisi geliştirmeye sevketmiştir [77]. Sadik ve ark., bu mimariye daha sonraları Bal Arısı Takımı (Honey Bee Teamwork, HBT) mimarisi adını vermişlerdir [78].

2.2.4. Toplu Karar Alma ve Yuva Yeri Seçimi Benzetimleri

Yonezawa ve Kikuchi, bal arılarının toplu karar alma sürecinin temel prensiplerini tanımlamışlardır [79]. Geliştirdikleri ekolojik algoritmanın yapılandırmasını inceceleyerek simulasyonlar yapmışlardır. Passino, arıların yuva yeri seçimi süresinin

matematiksel modelini kurmuş ve dinamik karar verme süreçlerinin olası etkilerini vurgulamıştır [80]. Gutierrez ve Huhns, yuva yerinin belirlenmesinde çoğunluğun yeterli sayıya ulaşması anlayışını temel alarak yazılımların hatalarına yönelik çalışma yapmışlardır [81].

2.2.5. Çiftleşme, Üreme Benzetimleri

Abbass, arılardaki çiftleşme sürecini modelleyerek Marriage in Honey-Bees (MBO) adını verdiği bir optimizasyon algoritması geliştirmiştir [82]. Modelde önce bir kraliçe arı ailesi olmaksızın tek başına yaşamına başlarken daha sonra bu aileleri olan bir yada daha fazla kraliçeden oluşan bir koloniye dönüşmektedir. Abbass bu algoritmasını elli değişkenli ve 215 sınırlamaya sahip elli tane sağlanabilirlik (satisfiability) problemi önermesine uygulamıştır [82]. Abbass, başka bir çalışmasında önerdiği algoritmanın sezgisellerle birleştirilen iki değişik versiyonunu sunmuştur. Bunlardan biri GSAT diğeri de rasgele ilerleme (random walk) sezgiselleridir. Bu modellerin 3-SAT problemleri üzerinde karşılaştırmalı analizleri yapılmıştır [83]. Abbass tarafından MBO algoritması üzerinde farklı bir modifikasyon daha yapılmıştır [84]. Bu değişiklik koloninin çoklu işçiye sahip bir kraliçesinin olmasıdır. Yeni algoritma her sınırlamasının 3 değişkene sahip olduğu 3-SAT problemlerine uygulanmıştır [84]. Teo ve Abbass algoritmanın tam olarak bir tavlama yaklaşımına sahip olmayışından kaynaklanan zayıf araştırma sorununu gidermek amacıyla çiftleşme uçuşu süreci için başka tavlama yaklaşımlarının etkisini incelemişlerdir [85]. Bu yaklaşımlar rasgele üretilen 3-SAT problemleri kullanılarak test edilmiş ve performans karşılaştırmaları yapılmıştır. Abbass ve Teo, klasik tavlama yaklaşımını erkek arıların genetik havuzunun belirlenmesinde kullanmışlardır [86]. Bu yaklaşım Benatchba ve ark. tarafından veri madenciliği problemine [87], Koudil ve ark. tarafından da bölme (partitioning) ve çizelgeleme problemlerine [88] uygulanmıştır. Curkovic ve Jerbic, MBO'yu lineer olmayan Diophantine eşitliği test problemlerine uygulamış ve sonuçlarını GA'nın sonuçları ile karşılaştırmıştır. Aynı çalışmada farklı şekilde ve dağıtık engellerin bulunduğu bir uzayda hareket eden robotun yönlendirilmesine uygulanmıştır [89].

Chang, MBO algoritmasını modifiye ederek kombinasyonel problemleri çözmek amacıyla kullanmış ve çözüme uygulamıştır ve stokastik dinamik programlama problemlerinin çözümü için MBO'yu Honey-Bees Policy Iteration (HBPI) adında bir algoritma haline dönüştürmüştür [90]. MBO, optimum rezervuar işletimi [91–93], su dağıtım sistemleri [94] gibi su kaynakları alanındaki bazı çalışmalarda da kullanılmıştır. Amiri ve Fathian, kendi kendine organize olan özellik haritaları (self-organizing maps, SOM) sinir ağı yapısını k-ortalama algoritmasına dayalı MBO'ya entegre etmiştir [95]. SOM küme sayısını belirlerken, MBO da bu küme sayısını kullanarak optimum kümeleme çözümü üretmektedir. Yapılan çalışmalarda Monte Carlo ile benzetilmiş verinin sonuçlarından önerilen metodun çalışmada yer alan diğer iki metottan daha iyi sonuç verdiği kanısına varılmıştır. Algoritma bir internet üzerinden kitap satışı yapan mağazanın verilerinin bölütlenmesine uygulanmıştır. Fathian ve ark. bu iki aşamalı modeli kümelemedeki bölgesel minima problemini gidermek için kullanmışlar ve kümelemede kullanılan diğer sezgisel algoritmalarla performansını karşılaştırmışlardır [96,97]. Yang ve ark., MBO algoritmasının hesaplama işlemlerinin karmaşık ve yavaş bir algoritma olmasından dolayı global yakınsama kabiliyetini artıracak şekilde modifiye etmişler ve bu algoritmaya Hızlı MBO (Fast Marriage in Honey Bees Optimization, FMBO) ismini vermişlerdir [98]. Başlangıç durumunda rasgele erkek arılar üretilmiş ve ilerleme durumları kısıtlanarak hesaplama süreci basitleştirilmiş ve hızlandırılmıştır. Yang ve ark.'nın başka bir çalışmasında FMBO algoritması Nelder-Mead metoduyla birleştirilerek (NMFMBO) Nelder-Mead metodunun bölgesel yakınsama karakteristiği algoritmaya kazandırılmış ve optimizasyon başarımı artırılmıştır [99]. Önerilen NMFMBO algoritması Gezgin Satıcı Problemine (TSP) ve bazı test problemlerine uygulanmıştır .

Marinakis ve ark., Bal Arısı Üreme Optimizasyon (Honey Bees Mating Optimization, HBMO) algoritmasına dayalı Çok Fazlı Komşuluk Araması (Multiple Phase Neighborhood Search - Greedy Randomized Adaptive Search Procedure, MPNS-GRASP) ile bütünleşik bir karma algoritma geliştirmişler ve taşıt yönlendirme probleminin çözümünde kullanmışlardır [100]. Marinakis ve ark., karma HBMO algoritmasını N objeyi K kümeye optimum şekilde ayırma problemini çözmek

amacıyla kullanmışlardır [101]. Başka bir karma algoritma da Niknam ve ark. tarafından sunulmuş ve çok amaçlı dağıtım besleyici yapılandırması (multi-objective distribution feeder reconfiguration) için kullanmışlardır. Bu model bulanık çok amaçlı (fuzzy multi-objective) yaklaşım ile HBMO algoritmasını birleştirmektedir [102]. Yang ve ark. Wolf Pack Search (WPS) algoritmasını MBO algoritmasının bölgesel arama sürecinde kullanmışlardır (WPS-MBO) ve WPS-MBO algoritmasını popüler kompleks test fonksiyonlarına ve TSP problemine uygulamışlardır [103]. Niknam, dağıtım şebekelerindeki durum değişkenlerinin kestirimi amacıyla HBMO algoritmasına dayalı bir yaklaşım sunmuştur. Metodun, 34-bus radyal test beslemesi ve 80-bus 20 kV şebekesi problemleri üzerinde yapay sinir ağları (YSA), karınca kolonisi optimizasyon algoritması ve genetik algoritma ile karşılaştırılması yapılmıştır [104]. Armamentarii, MBO'nun gelişmiş bir versiyonunu önererek karadan uçak savar silah sistemlerine uygulamıştır [105].

2.2.6. Yiyecek Arama Davranışı Benzetimleri

Sumpter ve Broomhead, arıların nektar toplama esnasında haberleşme süreçlerini tanımlamışlardır. Bir kolonideki mantıksal özellikleri biraraya getirerek dinamik süreçleri modellemişlerdir [106]. Karınca sisteminin karmaşık mühendislik sistemlerine başarılı uygulamaları Lucic ve Teodorivic'i zor kombinasyonel problemlerin çözmek amacıyla arıların yiyecek arama davranışını modelleyen Bee System (BS) adını verdikleri bir yaklaşım geliştirmeye sevk etmiştir [107]. BS'de yiyecek ararken kaşif arıların herhangi bir şekilde yönlendirilmeleri söz konusu değildir. Bu şekilde çalışan kaşif arıların düşük arama maliyetlerine karşın bulunan yiyeceklerin kalite ortalaması da düşüktür. BS, TSP problemlerinin bir çoğu üzerinde denenmiştir [108–111]. Lucic ve Teodorovic, BS ve bulanık sistemi birleştirerek bazı kompleks ulaşım problemlerinde oluşan belirsizlikleri de ele almışlardır. BS ve bulanık karınca sisteminin trafik ve ulaşım mühendisliği alanındaki potansiyel uygulamaları [112]'de tanıtılan çalışmada tartışılmıştır. Teodorovic ve Dellorco, Bee Colony Optimization (BCO) adını verdikleri bir sezgisel yaklaşımı geliştirerek gezinti eşleme (ride matching) problemine [113], Markovic ve ark. da BCO'yu optik ağlardaki yönlendirme ve dalga boyu atama (RWA)

problemlerine uygulamışlardır [114]. Vassiliadis ve Dounias BCO'yu sınırlamalı portföy optimizasyon problemine kaliteli çözüm arayışı için uygulamışlardır [115]. Banarjee ve ark., BCO'yu kaba küme (rough set) yaklaşımı ile birleştirerek üretim ve tedarik zinciri çizelgeleme probleminde kullanmışlardır [116]. Wong ve ark., BCO yu TSP'ye uygulamışlardır [117]. Teodorovic ve ark., arılara karşılık gelen bireylerin yaklaşık muhakeme (approximate reasoning) yaptıkları ve haberleşmelerinde ve işlevlerinde bulanık kuralların kullanıldığı Bulanık Arı Sistemi (Fuzzy Bee System, FBS)'ni tanımlamışlardır [118]. Teodorovic ve Dellorco, FBS'yi gezinti eşleme probleminin ve genel olarak kombinasyonel problemlerin çözümünde seyehat istek yönetimi tekniği olarak kullanmışlardır [119]. Teodorovic, arı sistemleri de dahil olmak üzere ulaşım mühendisliğinde kullanılan sistemleri inceleyen bir derleme yapmıştır [120].

Tereshko, arı kolonisinin yiyecek arama davranışını reaksiyon-difüzyon denklemlerine dayalı bir modelle tanımlamış ve kovanda yiyecek arama davranışında rol alan haberleşmenin nasıl gerçekleştiğini incelemiştir [121]. Tereshko ve Lee keşfedilen ortamla ilgili bilgilerin kovanda bireyler arasında nasıl aktarıldığı ile ilgili çalışma yapmışlardır [122]. Bu model iki önemli moddan oluşmaktadır: bir yiyeceğe yönlenilmesi ve kullanılan bir yiyeceğin bırakılmasıdır. Tereshko ve Loengarov, arı kolonisini çevreden bilgi alan ve davranışını buna göre ayarlayan dinamik bir sistem olarak değerlendirerek geliştirdikleri modellerinde bireyler bölgesel ve küresel olarak bilgi sahibidirler. Küresel olarak bilgi kollektif zekayı doğurmaktadır [123]. Loengarov ve Tereshko, faz geçişlerine ve kararlılığa sahip bal arısı modeli önermişlerdir. Modelde yiyecek arayıcı arıların sayısı arı yoğunluğuna ve kaşif arı oranına bağlı olarak karmaşık bir yöntemle belirlenmektedir [124]. Ghosh ve Marshall, yiyecek aramada rol alan öğrenme ve kollektif karar verme sürecini modellemişler ve bir robot grubu için kullanmışlardır [125]. Walker, özelleştirilmiş rotalandırmayı (customized routing) sağlamak ve internet servislerindeki yoğunluğu önlemek için arıların yiyecek arama davranışını temel almıştır [126] .

Wedde va Farooq tarafından önerilen diğer bir algoritma da mobil ad-hoc ağlarda enerji bakımından verimli rotalama sağlamakta kullanılan BeeAdHoc algoritmasıdır [127]. BeeAdHoc algoritması arıların yiyecek arama prensibinden esinlenilerek

ortaya çıkmıştır. Algoritma iki tip arı kullanmaktadır: kaşif arılar ve yiyecek getiriciler [128]. Mazhar ve Farooq, sistematik şekilde BeeAdHoc algoritmasının güvenlik açıklarını analiz ederek bir güvenlik çatısı önermişlerdir. BeeAdHoc için MANET ağlardaki güvenilir olmayan zararlı noktaların çıkarılmasını sağlayan bu sisteme BeeSec adı verilmiştir [129].

Saleem ve Farooq, MANET ağlardaki güvenlik sorununa yönelik AIS temelli bir güvenlik çatısı geliştirerek BeeAdHoc'un yanlış davranışlarını tespit etmeye çalışmışlar ve bu çalışmaya BeeAIS adını vermişlerdir. Aynı zamanda BeeAIS'ı kriptolanmış bir güvenlik sistemi ile karşılaştırmışlardır [130]. Saleem ve ark., BeeAdHoc protokolünün başarımını değerlendirmek için iki metrik ve bunların matematiksel modelini ortaya koymuşlardır. Bu metrikler rotalandırma maliyeti (routing overhead) ve rotalandırma kalitesidir [131]. Mazhar ve Farooq, dağıtık yanlış davranış tespit sistemine dayalı bir dentrik hücre önermişler ve bunu BeeAIS-DC olarak isimlendirmişlerdir [132]. Saleem ve Farooq BeeHive ve BeeAdHoc'un özelliklerinden esinlenerek BeeSensor adında bir algoritma geliştirmişlerdir [130]. BeeHive sabit ağlarda daha iyi performans sergilerken BeeAdHoc diğer adhoc ağlarda daha az enerji maliyetiyle daha iyi yada benzer performans sergilemektedir. BeeSensor ise hem daha performanslı hem de daha az enerji maliyetine sahiptir [130].

Karaboga arıların yiyecek arama davranışını modelleyen Yapay Arı Kolonisi (Artificial Bee Colony, ABC) algoritması adı verilen algoritmayı geliştirmiştir [22]. ABC algoritmasının performansı GA ile [133], PSO ve PS-EA ile [134], DE, PSO ve EA ile [135, 136]'de tanıtılan çalışmalarında çeşitli test fonksiyonları üzerinde karşılaştırılmıştır. ABC algoritmasının performansının, parametrelerin başlangıç aralıklarına olan duyarlılığı [137]'de sunulan çalışmada test edilmiştir. Karaboga ve Basturk tarafından ABC algoritması geliştirilerek sınırlamalı problemleri de çözecek hale getirilmiştir [138]. Karaboga ve ark., ABC algoritmasını yapay sinir ağlarının eğitiminde kullanmışlardır [139,140]. Karaboga ve Ozturk tarafından ABC algoritması medikal dataların sınıflandırılmasında ve kümelenmesinde kullanılmıştır [141, 142]. Fenglei, ABC algoritmasını TSP problemine uygulayarak bölgesel optimumun kontrol mekanizmasını incelemiştir. TSPLIB kütüphanesindeki

problemler üzerinde çalışmalarını gerçekleştirmiş ve ABC algoritmasının küresel arama kabiliyetini artırma üzerinde çalışmıştır [143]. Singh, ABC algorithmasını yaprak sınırlamalı en az kapsayan ağaç (leaf-constrained minimum spanning tree, LCMST) problemine uygulamış ve algoritmanın bu versiyonuna ABC-LSMST adını vermiştir. ABC-LCMST algoritmasının performansını GA, ACO ve tabu arama (TS) algoritmaları ile kıyaslamış ve ABC-LCMST'nin bulunan en iyi çözüm ve ortalama çözüm kalitesi ve hesaplama maliyeti dikkate alındığında daha iyi bir performans sergilediğini rapor etmiştir [144]. Rao ve ark., ABC algoritmasını radyal dağılımlı sistemin ağ yapılandırmasında güç kaybını minimize etmek, voltaj profilini geliştirmek ve besleyici yükünü desteklemek için kullanmışlardır. Simulasyonlarda 14, 33 ve 119 bus yapılı sistemler kullanılmış ve sonuçlar GA, DE ve ısıl işlem (SA) algoritmalarınınki ile karşılaştırılarak ABC'nin çözüm kalitesi ve hesaplama maliyeti açısından daha iyi olduğu raporlanmıştır [145]. Bendes ve Ozkan, ABC algoritmasını 3B koordinatların 2B resim düzlemine lineer olarak izdüşürülmesi arasında bir ilişki kurarak kamera kalibrasyonu yapan Direk Lineer Transformasyon yönteminde kullanmışlar ve sonuçları DE algoritması ile kıyaslamışlardır [146]. Karaboga, ABC algoritmasını sinyal işleme alanında IIR filtrelerin tasarımında kullanmıştır [147]. Qingxian ve Haijun ABC algoritmasının başlangıç gruplarını simetrik yaparak ve seleksiyon biriminde rulet tekerleği yerine Boltzmann seleksiyon yöntemini kullanarak ABC algoritmasının yakınsama kabiliyetini artırmaya yönelik çalışma yapmışlardır [148]. Hemamalini ve Simon, ABC algoritmasını kullanarak valf noktası etkili ekonomik yük sevkiyatı çözümü önermişlerdir [149]. Quan ve Shi, ABC algoritmasının yakınsama hızını artırmak amacıyla Banach uzayında ters haritalamanın (Contractive Mapping in Banach Spaces) sabit nokta teoremine dayalı bir arama operatörü geliştirmişlerdir [150]. Pawar ve ark. ABC algoritmasını makine mühendisliği alanındaki çok amaçlı optimizasyon problemleri olan elektro-kimayasal makina süreçlerinin, aşındırıcı akış ve öğütme süreçlerinin parametrelerinin optimizasyonunda kullanmışlardır [151–153]. Gözcü arı birimindeki bilginin kullanım kapasitesinin artırılması için, Tsai ve ark. rulet tekerleğine göre seleksiyonun gerçekleştiği gözcü arı biriminde Newton'un evrensel çekim yasasını kullanmışlardır (Interaktif ABC, IABC) [154]. Baykasoglu ve ark,. kaymalı komşu arama ve GRASP sezgisellerini ABC algoritmasında kullanarak

genelleşirilmiş atama problemlerinin çözümü için ABC algoritmasını kullanmışlardır [155].

Yang, iki boyutlu nümerik fonksiyonların optimizasyonu için sanal arı algoritmasını (Virtual Bee Algorithm, VBA) geliştirmiştir [156]. Modelde sanal arı sürüsü oluşturularak faz uzayında rasgele araştırma yapmaları sağlanmaktadır. Bir nektar kaynağı buldukları zaman diğer arılarla etkileşime girmektedir. Nektar kaynakları fonksiyonun değerine karşılık gelmektedir. Optimizasyon probleminin çözümü arıların etkileşiminin yoğunluğundan çıkarılmaktadır [156].

Pham ve ark., arıların yiyecek arama davranışını modelleyerek Bees Algorithm (BA) adını verdikleri bir algoritma geliştirmişlerdir [157]. Algoritma rasgele araştırma ile komşuluk arama mekanizmasına dayalı olup kombinasyonel ve nümerik problemlere uygulanmıştır [157]. Komşuluk arama sürecinde en yüksek uygunluk değerine sahip çözümler seçilmekte ve arılar uygunluklara göre ilgili bölgelere yönlendirilmektedir. En iyi kaynağa giden elit arı bir sonraki popülasyonun oluşturulmasında görev almaktadır. BA kompleks optimizasyon problemlerinin çözümünde [158], kerestelerdeki hataların belirlenmesine yönelik yapay sinir ağlarının eğitiminde [159], çok katmanlı algılayıcı (multi layer perceptron, MLP) ağlarının ağırlıklarının belirlenmesinde [160], örüntü tanıma uygulamalarına yönelik radyal ağların (RBF) eğitilmesinde [161], öğrenen vektör quantalama (LVQ) ağlarının eğitiminde [162], kaynatılmış kiriş tasarımının optimizasyonunda [163], üretim ünitesi oluşumunda [164], bir makinaya ait işlerin çizelgelenmesinde [165], bulanık mantık kontrolörlü robot jimnastikçinin parametrelerinin ayarlanmasında [166], veri kümelemesinde [167], kereste hatalarının sınıflandırılmasında destek vektör makinalarının (support vector machine, SVM) optimizasyonunda [168], ön tasarımda (preliminary design) [169], bazı mühendislik problemlerinin optimizasyonunda [170], proteinin üç boyutlu yapısının araştırlamasında [171], çoklu anten dizilerinin sentezinde [172] kullanılmıştır. Lee ve Darwish, ağırlıklı toplamlı BA'yı yakıt maliyetinin ve emisyonun aynı anda minimize edildiği çevresel/ekonomik güç dağıtım problemine uygulamışlardır [173].

Drias ve ark., Arı Sürüsü Optimizasyonu (Bee Swarm Optimization, BSO) adında bir sezgisel geliştirmişler ve maksimum ağırlıklı sağlanabilirlik (MAX-W-SAT) problemine uygulamışlardır [174]. Johnson test seti üzerinde karınca koloni ve diğer gelişime dayalı algoritmalar ile karşılaştırma yapılmıştır. Sadeg ve Drias, BSO'nun paralel versiyonunu geliştirerek MAX-W-SAT problemi üzerinde seri ve paralel yaklaşımlarla ilgili çalışma yapmışlardır [175].

Chong ve ark., yiyecek arama işlemi ve kuyruk dansını modelleyerek Arı Kolonisi Optimizasyon (bee colony optimization, BCO) algoritmasını tanımlamışlardır. Algoritma atölye üretim çizelgeleme problemine uygulanmıştır [176]. Chong ve ark. kabul edilebilir çözümlerin araştırılması ve daha önceki çözümlerinde geliştirilebilmesi gayesiyle verimli bir komşuluk yapısı kullanmışlardır. Algoritmada başlangıç çözümleri öncelik dağıtım kurallarına göre oluşturulmaktadır. BCO algoritması bir takım atölye üretim problemleri üzerinde karınca koloni, tabu araştırma gibi mevcut tekniklerle karşılaştırıldığında diğerlerine göre kıyaslanabilir performans sergilemiştir [177].

Quijano ve Passino, optimal kaynak tahsisi problemlerinin çözümünde kullanılabilecek bir bal arısı yiyecek arama modeli geliştirmişlerdir [178, 179].

Baig ve Rashid, bal arılarının yiyecek arama davranışını modelleyerek potansiyel komşuluklarda kaliteli kaynak için kollektif işlem yapan diğer alanlarda ise bireysel keşif yapan bal arısı yiyecek arama (Honey Bee Foraging, HBF) algoritmasını önermişlerdir [180] . Araştırma esnasında potansiyel bir kaynak bölgesi bulunduğunda algoritma dinamik olarak kaşif ve yiyeyek arayıcı arıların tahsisini gerçekleştirmektedir [181].

Lu ve Zhou, arıların polen toplama sürecinin benzetimini yaparak Polen Toplayan Arı Algoritmasını (Bee Collection Pollen Algorithm, BCPA) geliştirmişler ve TSP'nin çözümünde kullanmışlardır [182].

Ko ve ark., HoneyAdapt isminde arıların adaptif yiyecek arama davranışına dayalı kendi kendine uyarlanabilir grid hesaplama protokolü geliştirmiş ve grid uygulamalarında kullanmışlardır. Ayrıca, HoneySort isminde bu algoritmanın

başka versiyonunu geliştirerek işveren-işçi paradigmasını paralelleştirilmiş gride uygulamışlardır [183] .

2.2.7. Floral, Feromen Bırakma Benzetimleri

Ashlock ve Oftelie, sanal arılarda belli bir tip çiçekten nektar toplama yatkınlığı anlamına gelen floral sadakat olup olmadığının analizini yapmışlardır [184]. Hemen hemen aynı miktarda nektara sahip çiçek popülasyonlarında ayrım yapılmıyor ancak nektar miktarında farklılık olan popülasyonlarda daha fazla nektara sahip çiçek önem kazanıyor şeklinde bir hipotez öne sürmüşlerdir [184]. Purnamadjaja ve Russell, feromen haberleşmesi ile bir robot sürüsünde nekroforik arı davranışını geliştirmişlerdir [185]. Ölü arılar tarafından salgılanan nekroforik feromen işçi arılardaki ceset kaldırma davranışını tetiklemektedir. Benzer şekilde feromen robotlar arasındaki iletişimide sağlayabilir düşüncesinden hareketle aktivasyonu kaybolan robotların bir sinyal göndererek yerinin tespiti, kurtarılması ve yeniden aktifleştirilmesi için uygulamalar geliştirilmiştir [185]. Bu çalışmanın devamında da kraliçe arının kovandaki bazı işlevlerde rol oynayan feromen maddesinden esinlenerek, robotik sistemde bir robot liderin gönderdiği sinyallerle diğer robotların davranışlarını yönlendirmesi amaçlanmıştır [186].

2.2.8. Navigasyon Benzetimleri

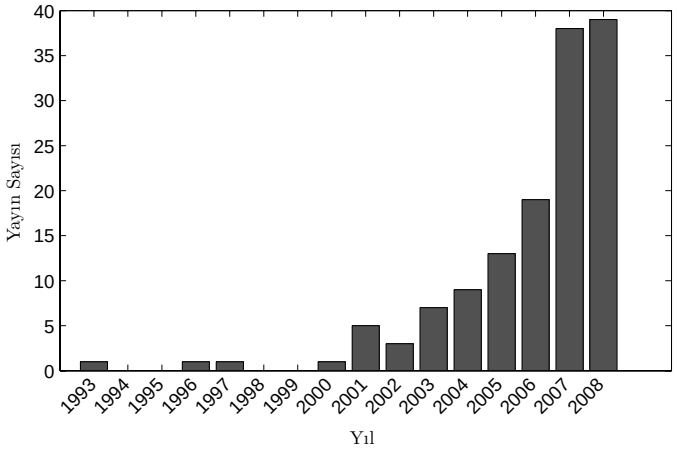
Arıların büyük ölçekli navigasyon davranışları Bianco'yu bir haritalama yaklaşımı geliştirmeye teşvik etmiştir. Navigasyon iki ayrık yer imi setinin kullanımı ile sağlanmaktadır: global yer imleri ve son hedefe varmak için hassas adımlar atılmasını sağlayan bölgesel yerimleri. Bu bağlamda harita iki seviyede oluşmaktadır: global potansiyel fonksiyonun takip edilerek bir yerden bir yere hareket (topolojik navigasyon), bir yere yakın olunduğunda özel bir noktanın daha iyi haritalanması için bölgesel potansiyel fonksiyonun takip edilmesi [187]. Lemmens ve ark., feromen tabanlı olmayan arıların tahsis ve navigasyon stratejilerinden esinlenen bir algoritma geliştirerek phoromone tabanlı olanlarla kıyaslamışlardır [188, 189]. Ayrıca, kendi yaklaşımlarının feromen tabanlı versiyonunu da geliştirmişlerdir [190]. Walker ve ark.'da arıların yiyecek arama durumundaki navigasyonel davranışların uzaysal

olarak hafızaya alınmasına dayalı robotik kontrol sistemi modellemişler [191] .

2.3. Sonuç

Arı kolonisinin zeki özelliklerine dayalı algoritmalar ve bunların uygulamaları Tablo 2.1'de gösterilmektedir. Bu çalışmaların yıllara göre dağılımı da Şekil 2.2'de verilmektedir. Tablo 2.1'den, arı sürü zekasının benzetimini yapan algoritmaların değişik alanlarda farklı problemlere uygulandığı görülmektedir. Bu tablodan kraliçe arıdan esinlenen çalışmaların, kraliçe arı o popülayondaki en iyi birey olduğundan genellikle ebeveyn ve elit birey seçme mekanizmalarında kullanıldığı görülmektedir. Görev dağılımı özelliğini temel alan modeller, mekanizmanın doğal işleyişi gereği daha ziyade kombinasyonel problemlere uygulansa da araştırmacının geliştirdiği modele bağlı olduğundan uygulama alanları bu tür problemlerle sınırlı değildir. Ayrıca algoritma nümerik problemleri çözmek amacıyla geliştirilmiş olsa dahi (VBA, ABC, BA gibi), uygun modifikasyonların gerçekleştirilmesiyle kombinasyonel problemlere de uygulanabilir hale getirilebilir.

Şekil 2.2'den, araştırmacıların arı kolonisine olan ilgilerinin ve bu ilgilerinin sonucu olarak arı kolonisine dayalı uygulamaların her yıl biraz daha arttığı görülmektedir. Bunun sonucunda bu alanda literatüre sunulan yayın sayısı zamana göre üstel olarak artmaktadır.



Şekil 2.2. Arı kolonilerinin zeki davranışları üzerine yapılan yayın sayılarının yıllara göre dağılımı

Tablo 2.1. Algoritmaların ve uygulamalarının kategorisel olarak listesi

Algoritma	**Uygulaması**	**Yayın**
Kraliçe Arı Gelişim	GA üzerinde bir gelişim	[49]
	Ekonomik güç sevkiyatı	[50]
	2 sistem için Fuzzy Kontrolör parametrelerinin ayarlanması	[51]
	4 sistem için Fuzzy Kontrolör parametrelerinin ayarlanması	[52]
GA'da kraliçe arı çaprazlamasının uygulanması	GA'nın çaprazlamasında ebeveyn olarak kraliçe arının seçilmesi	[53]
	Etiket sınırlamalı minimum kapsamalı ağaç problemi	[56]
Genetik Algoritma Tabanlı MBGA	Nümerik Problemler	[55]
BS	GA'nın gelişimi	[57]
Arıların bilgi paylaşım ve süreç modeli	LAN, WAN, Internetteki bilgi paylaşımı	[58]
DBD	Bir gridde örüntü oluşumu	[59]
BeeHive	Ağlarda yönlendirme	[60]
	Ağlarda yönlendirme	[61]
	Ağlarda yönlendirme	[62]
	Qos tekli yayın yönlendirme şeması	[66]
BeeHiveGuard	BeeHive algoritmasının güvenlik açıklarını gidermek	[64]
BeeHiveAIS	Güvenlik	[65]
BeeJAMa	Yönlendirme [67]	
BHM	Örün arama	[69]
	Online arama	[71]
HBS	Aralıklı yeniden yapılandırma	[72]
Ecological Algorithm	Optimal Sıralama	[79]
Systems Biology		[80]
Quorum Sensing	Software fault tolerant system	[81]
Decentralized HB	İnternet hosting merkezlerindeki sunucuların dinamik tahsisi	[73]
HB	Internet hosting merkezlerindeki sunucuların otonomik orkestrasyonu	[74]
HB	Hosting merkezleri	[75]
Swan	IP ağların yönetiminde	[76]

Tablo 2.1. (devamı)

Algoritma	**Uygulaması**	**Yayın**
HBT	Yazılım ajanları	[77]
MBO	SAT problemleri	[82] [83] [84] [85] [86]
	Veri madenciliği	[87]
	Parçalama ve çizelgeleme problemleri	[88]
	Doğrusal olmayan diyofant denklemi problemi, farklı şekilde ve dağıtık engellere sahip bir uzayda bir robotun yönlendirilmesi	[89]
	Kombinasyonel optimizasyon problemleri, Stokastik dinamik programlama	[90]
	Stokastik dinamik programlama problemleri	[90]
	Optimal rezervuar işlemi	[91–93]
	Su dağıtım şebekeleri	[94]
	Kümeleme, internet üzerinden satış yapan kitapçı verilerinin segmentasyonu	[95]
	Kümelemede bölgesel optima problemi	[96,97]
	Nümerik problemler	[98]
	TSP	[99]
	Araç yönlendirme problemi	[100]
	Kümeleme	[101]
	Çok-amaçlı optimizasyon	[102]
	Karmaşık değerlendirme fonksiyonları ve TSP	[103]
	Yerden uçaksavar silah sistemleri ağı	[105]
	Dağıtık jenaratörlü dağıtık ağların durum değişkenlerinin kestirimi	[104]
BS	Algoritma temelleri	[107]
	TSP	[108–111]
BS + FAS	Trafik ve ulaşım mühendisliği	[112]
BCO	Gezinti Eşleme problemi(ride-matching)	[113]

Tablo 2.1. (devamı)

Algoritma	**Uygulaması**	**Yayın**
	Tamamen optik ağlarda Yönlendirme ve dalga boyu atama (routing and wavelengtth assignment (RWA))	[114]
	TSP	[117]
	Sınırlamalı pörtföy optimizasyonu	[115]
	Üretim ve tedarik zinciri çizelgelemesi	[116]
	Atölye üretim çizelgeleme	[176]
	Atölye üretim çizelgeleme	[177]
FS	[118]	
Reaksiyon-difüzyon modeli	Model temelleri	[121]
Bilgi-Haritalama Örüntüleri		[122]
Dinamik sistem modeli		[123]
Faz geçişleri ve çift kararlılık modeli		[124]
Yiyecek arama modeli		[125]
Bal arısı arama stratejileri	Internet hizmetlerinde yönlendirme ve tıkanıklıkların önlenmesi	[126]
BeeAdhoc	Mobil ve adhoc ağların yönlendirilmesi	[127, 128]
BeeSec	Güvenilir olmayan MANET ağlardaki zararlı noktaların giderilmesi	[129]
BeeAIS	MANET ağlardaki güvenlik sorunu	[130]
BeeAIS-DC	MANET ağların güvenliği	[132]
BeeAdHoc	BeeAdHoc için iki performans metriği	[131]
BeeSensor=BeeAdhoc + BeeHive	Ağlarda yönlendirm	[130]
ABC	Nümerik problemler	[22]
	Nümerik problemler üzerinde performans karşılaştırması	[133–137]
	Sınırlamalı optimizasyon problemleri	[138]
	Yapay sinir ağlarının eğitimi	[139, 140]
	Medikal örüntü sınıflandırma ve kümeleme problemleri	[141, 142].

Tablo 2.1. (devamı)

Algoritma	**Uygulaması**	**Yayın**
	TSP	[143]
	Yaprak sınırlamalı en az kapsaayan ağaç (LCMST) problemi	[144]
	Ağ yapılandırma problemi	[145]
	Kamera kalibrasyonu	[146]
	IIF filtre tasarımı	[147]
	Nümerik problemler	[148]
	Valf noktası etkili ekonomik yük dağıtımı	[149]
	Nümerik Problemler	[150]
	Elektro-kimyasal işleme süreç parametrelerinin çok amaçlı optimizasyonu	[151]
	Aşındırmalı akış işleme sürecinin optimizasyonu	[152]
	Öğütme süreç parametrelerinin optimizasyonu	[153]
	Nümerik problemler	[154]
	Genelleştirilmiş atama problemi	[155]
VBA	Nümerik problemler	[156]
BA	Algoritma temelleri	[157]
	Nümerik problemler	[158]
	Kereste hatalarının tespiti için YSA'ların optimizasyonunda	[159]
	Çok katmanlı algılayıcıların ağırlıklarının optimizasyonu	[160]
	Örüntü tanıma için RBF ağların eğitilmesi	[161]
	Örüntü tanım için LVQ ağlarının eğitilmesi	[162]
	Kaynak kiriş problemi	[163]
	Üretim alanı oluşturulması	[164]
	Tek makina için işlerin çizelgelenmesi	[165]
	Robot jimnastikçi bulanık mantık kontrolörünün ayarlanması	[166]
	Veri kümeleme	[167]
	Kereste hatalarının tespiti için SVM'nin optimizasyonu	[168]

Tablo 2.1. (devamı)

Algoritma	**Uygulaması**	**Yayın**
	Ön Tasarım	[169]
	Mühendislik tasarım problemleri	[170]
	Yapısal protein araması	[171]
	Anten dizisi sentezi	[172]
	Çok amaçlı çevresel/ekonomik dağıtım	[173]
BSO	MAX-W-SAT problemi	[174]
Paralel BSO	MAX-W-SAT problem,	[175]
HBSF	Optimal kaynak tahsisi problemleri	[178, 179]
HBF	Nümerik problemler	[180, 181]
BCPA	TSP	[182]
Honeyadapt, Honeysort	Grid computing	[183]
Feromen tabanlı haerleşme	Model temelleri	[184]
Feromen haberleşmesi	Bir robot sürüsünde nekroforik davranış benzetimi	[185]
Kraliçe arı feromen'i	Robotların yönlendirilmesi	[186]
Navigasyon	Geniş ölçekli görsel hassas navigasyon	[187]
Bee Algorithm	Çok ajanlı sistemler	[188–190]
Uzaysal hafıza	Robotik kontrol sistemi	[191]

3. BÖLÜM

YAPAY ARI KOLONİSİ (ARTIFICIAL BEE COLONY, ABC) ALGORİTMASI

3.1. Gerçek Arıların Yiyecek Arama Davranışları

Gerçek bir kolonide yapılacak işler o iş için özelleşmiş arılar tarafından yapılır. Yani, yapılacak işlere göre arılar arasında bir iş bölümü vardır ve herhangi bir merkezi otorite olmadan bu iş dağılımını gerçekleştirdikleri için kendi kendilerine organize olabilmektedirler. İş bölümü yapabilme ve kendi kendine organize önceki bölümde belirtildiği gibi sürü zekasının iki önemli bileşenidir. Tereshko'nun reaktif difüzyon denklemlerine dayalı olarak önerdiği kollektif zekanın ortaya çıkmasını sağlayan minimal yiyecek arama modelinde temel üç bileşen vardır: yiyecek kaynakları, görevi belirli işçi arılar ve görevi belirsiz işçi arılar. Bu minimal model iki modda çalışmaktadır: bir yiyecek kaynağına yönlenme ve kaynağı bırakma [121]. Bileşenleri şu şekilde açıklayabiliriz:

i) Yiyecek Kaynakları: Arıların nektar, polen veya bal elde etmek için gittikleri kaynaklardır. Bir yiyecek kaynağının değeri, çeşidi, yuvaya yakınlığı, nektar konsantrasyonu veya nektarın çıkarılmasının kolaylığı gibi birçok faktöre bağlı olmasına rağmen basitlik açısından sadece kaynağın zenginligi tek bir kriter olarak alınabilir.

ii) Görevi Belirli İşçi Arılar: İşçi arılar, daha önceden keşfedilen belli kaynaklara ait nektarın kovana getirilmesinden sorumludurlar. İşçi arılar aynı zamanda ziyaret ettikleri kaynağın kalitesi ve yeriyle ilgili bilgiyi kovanda bekleyen diğer arılarla paylaşırlar. Bundan sonra görevi belirli işçi arılar için görevli arılar ifadesi kullanılacaktır.

iii) Görevi Belirsiz İşçi Arılar: Bu arılar nektarını toplayabilecekleri kaynak arayışı içerisindedirler. Görevi belirsiz iki çeşit işçi arı bulunmaktadır: içsel bir dürtüye

veya bir dış etmene bağlı olarak rastgele kaynak arayışında olan kaşif arılar ve kovanda bekleyen ve görevli arıları izleyerek bu arılar tarafından paylaşılan bilgiyi kullanarak yeni bir kaynağa yönelen gözcü arılar. Kaşif arıların sayısının tüm koloniye oranı ortalama % 5-10 arasındadır.

Arılar arasında bilgi paylaşımı kollektif bilginin oluşumunda en önemli husustur. Bir kovan göz önüne alındığında, tüm kovanlarda ortak olan bazı parçalara ayırmak mümkündür ve bu parçalardan en önemlisi ise dans alanıdır. Yiyecek kaynağının kalitesi ve yeri ilgili bilgi paylaşımı dans alanında olmaktadır. Bir arı dans ederken diğer arılarda ona antenleri ile dokunurlar ve bulduğu kaynağın tadı ve kokusu ile ilgili de bilgi alırlar. Ziyaret ettikleri kaynağa daha fazla arı yönlendirebilmek için kovandaki çeşitli alanlarda bu dansı gerçekleştirir ve kaynağına geri döner.

Kesin çizgileri olmamakla birlikte genelde kovandaki uçuş yapılan yere yakın bir yerde olan dans alanında yapılan danslar, taşınan bilgiye göre ve canlılığa göre değişmektedir. En önemlisi nektarın tatlılığıdır. Bu üstünlük dans eşik değerini belirler. Ancak bu uyarıcı yeterli değildir. Bunun yanında nektarın çıkarılmasının kolaylığı da bir etkendir. Kovandan olan uzaklık, çiçek nektarının kıvamı, besinin genel durumu, kalitedeki göreceli değişimler, hava koşulları ve günün hangi vaktinin olduğu dansı etkileyen diğer etmenlerdir.

Yiyecek getiricilerin, diğer yiyecek getiricileri uzak noktadaki bir kaynağa yönlendirmek için hedefe ait yön bilgisini vermeleri gerekir. Yön bilgisi alındıktan sonra hedefe ulaşmada güneşten faydalanılır. Bileşik gözleri ile arılar, kendi yörüngeleri ile güneş arasındaki açıyı hesaplayabilmektedirler. Güneşin önü kapanmış olsa dahi polarize gün ışığından yine güneşin konumunu tayin edebilmektedirler. Arılar, bir noktanın uzaklığını enerji tüketimine bağlı olarak hesaplamakta ve yüklerine göre farklı yükseklikte uçarak enerji tüketimlerini ayarlamaktadırlar.

Kaynağın kovana olan mesafesine göre çeşitli danslar mevcuttur: dairesel dans (round dance), kuyruk dansı (waggle dans) ve titreme dansı (tremble dance).

Daire ve kuyruk dansları yiyecek getiricilerin yeniden aktivasyonunda etkilidir. Bu iki dans farklı uzaklıktaki bölgelerin ayrımında kullanılır. Daire dansı ile belirtilen yiyecek kaynağının kovana olan uzaklığı maksimum 50-100 metre civarında olduğundan bu dans yön ve uzaklık bilgisi vermemektedir.

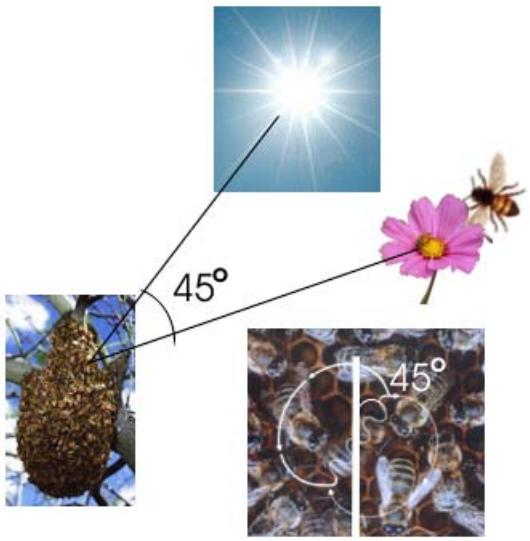
Titreme dansında, arıların petek üzerinde düzensiz tarzda ve yavaş tempoda bacaklarını titreterek ileri, geri, sağa ve sola hareketleri söz konusudur. Arı, zengin bir nektar kaynağı bulduğunu, ancak kovana işlenebileceğinden fazla nektar geldiğini ve bundan dolayı nektarı işleme görevine geçmek istediğini belirtmektedir. Sadece dans alanında değil kovanın başka bölümlerinde de bu dans gerçekleştirilmektedir. Bu dansın amacı kovan kapasitesi ve yiyecek getirme aktivitesi arasındaki dengeyi sağlamaktır.

100 metreden 10 kilometreye kadar olan geniş bir alan içerisinde bulunan kaynaklarla ilgili bilgi aktarımında kuyruk dansı kullanılmaktadır. Bu dans 8 rakamına benzeyen bir dans çeşididir. Dansı izleyen arıların bir titreşim oluşturması ile bu dansı yapan arı, dansına son verir. Her 15 saniyede dansın tekrarlanma sayısı, nektar kaynağının uzaklığı hakkında bilgi vermektedir. Daha az tekrarlanma sayısı daha uzak bölgeleri ifade etmektedir. Yön bilgisi, Şekil 3.1'deki gibi 8 rakamı şeklindeki dansın açı bilgisinden elde edilir. Şekilde verilen örnekte dansı izleyen arılar, danstan güneşle yiyecek arasındaki açının 45° olduğunu anlamaktadırlar. Arılar arasında uzak mesafede bulunan kaynakla ilgili bilgi paylaşımı sırasında daha hızlı dans edilir [192, 193].

Tüm zengin kaynaklarla ilgili bilgiler dans alanında gözcü arılara iletildiğinden, gözcü arılar bir kaç dansı izledikten sonra hangisini tercih edeceğine karar verir. Zengin kaynaklarla ilgili daha fazla bilgi aktarımı olduğundan bu kaynakların seçilme olasılığı daha fazladır.

Yiyecek arayıcıların davranışlarının daha iyi anlaşılabilmesi için Şekil 3.2'de verilen modelin incelenmesi faydalı olacaktır.

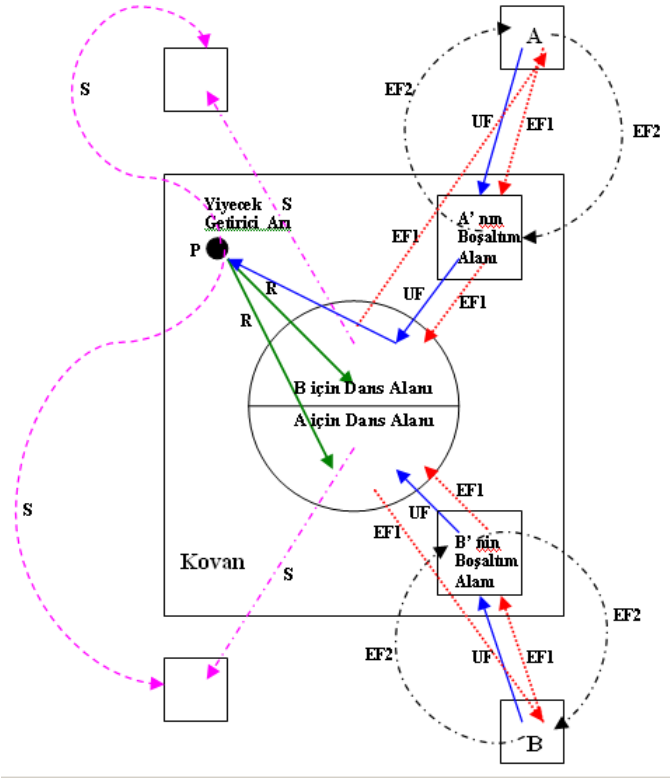
A ve B isminde iki keşfedilmiş kaynak olduğunu varsayalım. Araştırmanın başlangıcında potansiyel bir yiyecek arayıcı görevi belirsiz işçi arı olarak araştırmaya



Şekil 3.1. Arılarda Dans

başlayacak ve bu arı kovan etrafındaki kaynakların yerinden haberdar değildir. Bu durumdaki bir arı için iki olası seçenek söz konusudur:
i) Bu arı kaşif arı olabilir ve içsel ve dışsal etkenlere bağlı olarak yiyecek aramaya başlayabilir (Şekil 3.2'de S ile gösterilmektedir).
ii) Bu arı, kuyruk danslarını izleyen bir gözcü arı olabilir ve izlediği dansla anlatılan kaynağa gidebilir (Şekil 3.2'de R ile gösterilmektedir).

Bir kaynak bulunduktan sonra arı imkanları dahilinde bu kaynağın yerini hafızasına alır ve hemen nektar toplamaya başlar. Böylece bu arı artık görevli arı haline gelmektedir. İşçi arı kaynaktan nektarı aldıktan sonra kovana döner ve bunu yiyecek depolayıcılara aktarır. Nektarını aktardıktan sonra üç seçenek ortaya çıkar:
i) Gittiği kaynağı bırakarak bağımsız izleyici olabilir (Şekil 3.2'de UF ile gösterilmektedir).
ii) Gittiği kaynağa dönmeden önce dans eder ve kovandaki arkadaşlarını da aynı kaynağa yönlendirebilir (Şekil 3.2'de EF1 ile gösterilmektedir).
iii) Diğer arıları yönlendirmeden kaynağına gidebilir (Şekil 3.2'de EF2 ile gösterilmektedir).
Tüm arıların eş zamanlı olarak yiyecek arama sürecinde olmadıklarını belirtmek gerekir. Yeni arıların yiyecek aramaya katılma olasılıklarının toplam arı sayısı ile o anda yiyecek arama sürecinde olan arıların sayılarının farkıyla orantılı olduğu



Şekil 3.2. Yiyecek Arama Çevrimi

çalışmalarla doğrulanmıştır [108].

Bu durumda arılardaki kendi kendine organize olabilme şu şekilde açıklanabilir: i) *Pozitif geribesleme:* kaynağın nektar miktarı arttıkça bu kaynağı seçen gözcü arı sayısı da artmaktadır. ii) *Negatif geribesleme:* tükenen kaynak bırakılmaktadır. iii) *Salınımlar:* kaşif arılar yiyecek kaynağı bulabilmek için rastgele arama yapmaktadırlar. iv) *Çoklu etkileşimler:* arılar yiyecek kaynakları ile ilgili olarak dans alanında bilgi paylaşımı gerçekleştirirler.

3.2. Yapay Arı Kolonisi Algoritması

Doğada var olan zeki davranışlar içeren süreçlerin incelenmesi araştırmacıları yeni optimizasyon metotları geliştirmeye sevketmiştir. Karaboğa, arıların yiyecek arama davranışını modelleyerek Yapay Arı Kolonisi (ABC) algoritmasını geliştirmiştir [22].

Karaboğa'nın ABC algoritmasının temel aldığı model de bazı kabuller yapılmaktadır. Bunlardan birincisi her bir kaynağın nektarının sadece bir görevli

arı tarafından alınıyor olmasıdır. Yani görevli arıların sayısı toplam yiyecek kaynağı sayısına eşittir. İşçi arıların sayısı aynı zamanda gözcü arıların sayısına da eşittir. Nektarı tükenmiş kaynağın görevli arısı artık kaşif arı haline dönüşmektedir. Yiyecek kaynaklarının yerleri optimizasyon problemine ait olası çözümlere ve kaynakların nektar miktarları ise o kaynaklarla ilgili çözümlerin kalitesine (uygunluk) karşılık gelmektedir. Dolayısıyla ABC optimizasyon algoritması en fazla nektara sahip kaynağın yerini bulmaya çalışarak uzaydaki çözümlerden problemin minimumunu yada maksimumunu veren noktayı (çözümü) bulmaya çalışmaktadır.

Bu modele ait süreç adımları aşağıdaki gibi verilebilir:

1. Yiyecek arama sürecinin başlangıcında, kaşif arılar çevrede rastgele arama yaparak yiyecek aramaya başlarlar.

2. Yiyecek kaynakları bulunduktan sonra, kaşif arılar artık görevli arı olurlar ve buldukları kaynaklardan kovana nektar taşımaya başlarlar. Her bir görevli arı kovana dönüp getirdiği nektarı boşaltır ve bu noktadan sonra ya bulduğu kaynağa geri döner yada kaynakla ilgili bilgiyi dans alanında sergilediği dans aracılığıyla kovanda bekleyen gözcü arılara iletir. Eğer faydalandığı kaynak tükenmiş ise görevli arı kaşif arı haline gelir ve yeni kaynak arayışına yönelir.

3. Kovanda bekleyen gözcü arılar zengin kaynakları işaret eden dansları izlerler ve yiyeceğin kalitesi ile orantılı olan dans frekansına bağlı olarak bir kaynağı tercih ederler.

ABC algoritmasının bu süreçleri ve temel adımları Algoritma 1'de verilmektedir.

Algoritma 1 ABC algoritmasının temel adımları

1: Başlangıç yiyecek kaynağı bölgelerinin üretilmesi
2: **repeat**
3: İşçi arıların yiyecek kaynağı bölgelerine gönderilmesi
4: Olasılıksal seleksiyonda kullanılacak olasılık değerlerinin görevli arılardan gelen bilgiye göre hesaplanması
5: Gözcü arıların olasılık değerlerine göre yiyecek kaynağı bölgesi seçmeleri
6: Kaynağı bırakma kriteri: limit ve kaşif arı üretimi
7: **until** çevrim sayısı=Maksimum çevrim sayısı

Yiyecek arayan arılarda görülen zeki davranış ile bu davranışı simüle eden ABC algoritmasının temel birimleri takip eden alt bölümlerde açıklanmaktadır.

3.2.1. Başlangıç Yiyecek Kaynağı Bölgelerinin Üretilmesi

Arama uzayını yiyecek kaynaklarını içeren kovan çevresi olarak düşünürsek, algoritma arama uzayındaki çözümlere karşılık gelen rastgele yiyecek kaynağı yerleri üreterek çalışmaya başlamaktadır. Rastgele yer üretme süreci her bir parametrenin alt ve üst sınırları arasında rastgele değer üreterek gerçeklenir(Eşitlik 3.1):

$$x_{ij} = x_j^{min} + rand(0,1)(x_j^{max} - x_j^{min}) \tag{3.1}$$

Burada $i = 1 \ldots SN, j = 1 \ldots D$ ve SN yiyecek kaynağı sayısı ve D ise optimize edilecek parametre sayısıdır. x_j^{min}, j. parametrenin alt sınırı, x_j^{max} ise j. parametrenin üst sınırıdır. Aynı zamanda başlangıç aşamasında her kaynağın geliştirilememe sayısını ifade eden $failure_i$ (i. kaynağın geliştirilememe sayısı) sayaçları da sıfırlanmaktadır.

Başlangıç aşamasından sonra yiyecek kaynaklarının görevli arı, gözcü arı ve kaşif arı süreçlerinden geçirilerek, daha iyisi bulunmaya çalışılır. ABC algoritması için durdurma kriteri olarak maksimum çevrim sayısı (MCN) ve kabul edilebilir bir hata değeri (ϵ) veya diğer optimizasyon algoritmaları için kullanılan standart bir durdurma kriteri tanımlanabilir.

3.2.2. İşçi Arıların Yiyecek Kaynağı Bölgelerine Gönderilmesi

Daha önce de belirtildiği gibi her bir kaynağın bir görevli arısı vardır. Dolayısıyla yiyecek kaynaklarının sayısı görevli arıların sayısına eşittir. İşçi arı çalıştığı yiyecek kaynağı komşuluğunda yeni bir yiyecek kaynağı belirler ve bunun kalitesini değerlendirir. Yeni kaynak daha iyi ise bu yeni kaynağı hafızasına alır. Yeni kaynağın mevcut kaynak komşuluğunda belirlenmesinin benzetimi Eşitlik 3.2 ile tanımlanmaktadır:

$$\upsilon_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) \tag{3.2}$$

x_i ile gösterilen her bir kaynak için bu kaynağın yani çözümün tek bir parametresi

(rastgele seçilen parametresi, j) değiştirilerek x_i komşuluğunda v_i kaynağı bulunur. Eşitlik 3.2'de j, $[1, D]$ aralığında rastgele üretilen bir tamsayıdır. Rastgele seçilen j parametresi değiştirilirken, yine rastgele seçilen x_k komşu çözümünün ($k \in \{1, 2, \ldots SN\}$) j. parametresi ile mevcut kaynağın j. parametresinin farkları alınıp $[-1, 1]$ arasında rastgele değer alan ϕ_{ij} sayısı ile ağırlıklandırıldıktan sonra mevcut kaynağın j. parametresine eklenmektedir.

Eşitlik 3.2'den de görüldüğü gibi $x_{i,j}$ ve $x_{k,j}$ arasındaki fark azaldıkça, yani çözümler birbirine benzedikçe, $x_{i,j}$ parametresindeki değişim miktarı da azalacaktır. Böylece bölgesel optimal çözüme yaklaştıkça değişim miktarı da adaptif olarak azalmaktadır.

Bu işlem sonucunda üretilen v_{ij}'nin daha önceden belli olan parametre sınırlarını aşması durumunda j. parametreye ait olan alt veya üst sınır değerlerine ötelenmektedir (Eşitlik 3.3):

$$v_{ij} = \begin{cases} x_j^{\min}, & v_{ij} < x_j^{\min} \\ v_{ij}, & x_j^{\min} \leq v_{ij} \leq x_j^{\max} \\ x_j^{\max}, & v_{ij} > x_j^{\max} \end{cases} \tag{3.3}$$

Sınırlar dahilinde üretilen v_i parametre vektörü yeni bir kaynağı temsil etmekte ve bunun kalitesi hesaplanarak bir uygunluk değeri atanmaktadır (Eşitlik 5.9):

$$fitness_i = \left\{ \begin{array}{ll} 1/(1+f_i) & f_i \geq 0 \\ \\ 1 + abs(f_i) & f_i < 0 \end{array} \right\} \tag{3.4}$$

Burada f_i, v_i kaynağının yani çözümünün maliyet değeridir. x_i ile v_i arasında nektar miktarlarına yani uygunluk değerlerine göre bir aç gözlü (greedy) seçme işlemi uygulanır. Yeni bulunan v_i çözümü daha iyi ise görevli arı hafızasından eski kaynağın yerini silerek v_i kaynağının yerini hafızaya alır. Aksi takdirde görevli arı x_i kaynağına gitmeye devam eder ve x_i çözümü geliştiremediği için x_i kaynağı ile ilgili geliştirememe sayacı ($failure_i$) bir artar, geliştirdiği durumda ise sayaç sıfırlanır.

3.2.3. Gözcü Arıların Seleksiyonda Kullanacakları Olasılık Değerlerinin Hesaplanması (Dans Benzetimi)

Tüm görevli arılar bir çevrimde araştırmalarını tamamladıktan sonra kovana dönüp buldukları kaynakların nektar miktarları ile ilgili gözcü arılara bilgi aktarırlar. Bir gözcü arı dans aracılığıyla paylaşılan bilgiden faydalanarak yiyecek kaynaklarının nektar miktarları ile orantılı bir olasılıkla bir bölge (kaynak) seçer. Bu ABC'nin çoklu etkileşim sergilediğinin bir örneğidir. Olasılıksal seçme işlemi, algoritmada nektar miktarlarına karşılık gelen uygunluk değerleri kullanılarak yapılmaktadır. Uygunluk değerine bağlı seçme işlemi rulet tekerleği, sıralamaya dayalı, stokastik örnekleme, turnuva yöntemi yada diğer seleksiyon şemalarından herhangi biri ile gerçeklenebilir. Temel ABC algoritmasında bu seleksiyon işlemi rulet tekerleği kullanılarak yapılmıştır. Tekerlekteki her bir dilimin açısı uygunluk değeri ile orantılıdır. Yani bir kaynağın uygunluk değerinin tüm kaynakların uygunluk değeri toplamına oranı o kaynağın diğer kaynaklara göre nisbi seçilme olasılığı olduğunu vermektedir (Eşitlik 3.5).

$$p_i = \frac{fitness_i}{\sum_{i=1}^{SN} fitness_i} \tag{3.5}$$

Burada $fitness_i$, i. kaynağın kalitesini, SN görevli arı sayısını göstermektedir. Bu olasılık hesaplama işlemine göre bir kaynağın nektar miktarı arttıkça (uygunluk değeri arttıkça) bu kaynak bölgesini seçecek gözcü arı sayısı da artacaktır. Bu özellik ABC'nin pozitif geribesleme özelliğine karşılık gelmektedir.

3.2.4. Gözcü Arıların Yiyecek Kaynağı Bölgesi Seçmeleri

Algoritmada olasılık değerleri hesaplandıktan sonra bu değerler kullanılarak rulet tekerleğine göre seçim işleminde her bir kaynak için [0,1] aralığında rastgele sayı üretilir ve p_i değeri bu üretilen sayıdan büyükse görevli arılar gibi gözcü arı da eşitlik 3.2'yi kullanarak bu kaynak bölgesinde yeni bir çözüm üretir. Yeni

çözüm değerlendirilir ve kalitesi hesaplanır. Sonra, yeni çözümle eski çözümün uygunluklarının karşılaştırıldığı ve iyi olanın seçildiği aç gözlü seleksiyon işlemine tabi tutulur. Yeni çözüm daha iyi ise eski çözüm yerine bu çözüm alınır ve çözüm geliştirememe sayacı ($failure_i$) sıfırlanır. Eski çözümün uygunluğu daha iyi ise bu çözüm muhafaza edilir ve geliştirememe sayacı ($failure_i$) bir artırılır. Bu süreç tüm gözcü arılar yiyecek kaynağı bölgelerine dağılana kadar devam eder.

3.2.5. Kaynağı Bırakma Kriteri: Limit ve Kaşif Arı Üretimi

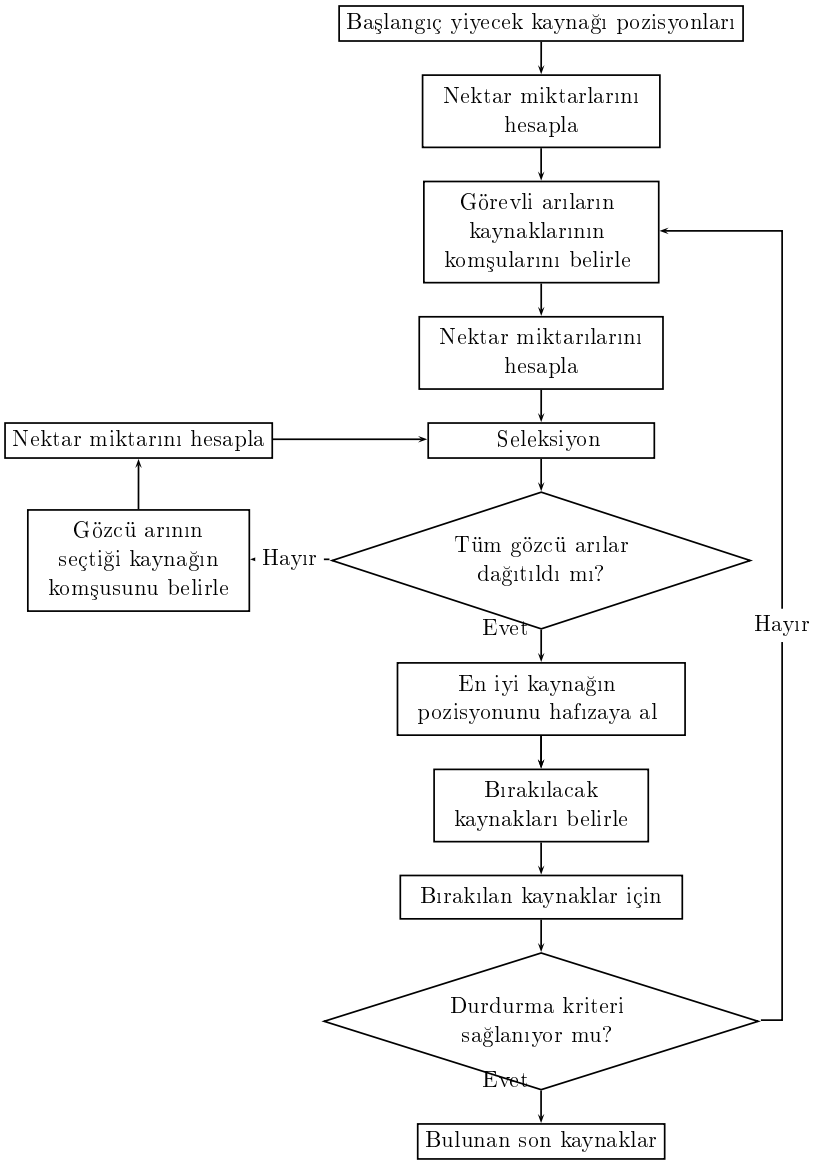
Bir çevrim sonunda tüm görevli ve gözcü arılar arama süreçlerini tamamladıktan sonra çözüm geliştirememe sayaçları ($failure_i$) kontrol edilir. Bir arının bir kaynaktan faydalanıp faydalanamadığı, yani gidip geldiği kaynağın nektarının tükenip tükenmediği çözüm geliştirememe sayaçları aracılığıyla bilinir. Bir kaynak için çözüm geliştirememe sayacı belli bir eşik değerinin üzerindeyse, artık bu kaynağın görevli arısının tükenmiş olan o çözümü bırakıp kendisi için başka bir çözüm araması gerekmektedir. Bu da biten kaynakla ilişkili olan görevli arının kaşif arı olması anlamına gelmektedir. Kaşif arı haline geldikten sonra, bu arı için rastgele çözüm arama süreci başlar (Eşitlik 3.1). Kaynağın tükendiğinin belirlenmesi için kullanılan eşik değeri, ABC algoritmasının önemli bir kontrol parametresidir ve "*limit*" olarak adlandırılmaktadır. Temel ABC algoritmasında her çevrimde sadece bir kaşif arının çıkmasına izin verilir.

Tüm bu birimler arasındaki ilişki ve döngü Şekil 3.3'deki gibi bir akış diyagramı ile şematize edilebilir.

3.2.6. Seleksiyon Mekanizmaları

ABC algoritması 4 farklı seleksiyon işlemi kullanmaktadır. Bunlar; 1) potansiyel iyi kaynakların belirlenmesine yönelik eşitlik 3.5 ile olasılık değerlerinin hesaplandığı global olasılık temelli seleksiyon süreci 2) görevli ve gözcü arıların renk, şekil, koku gibi nektar kaynağının türünü belirlemesini sağlayan görsel bilgiyi kullanarak bir bölgede kaynağın bulunmasına vesile olan bölgesel olasılık tabanlı seleksiyon işlemi (eşitlik 3.2) 3) İşçi ve gözcü arıların daha iyi olan kaynağı belirlemek amacıyla

kullandıkları aç gözlü seleksiyon 4) kaşif arılar tarafından eşitlik 3.1 aracılığla gerçekleştirilen rastgele seleksiyon. Bütün bu seleksiyon metotlarının bir arada kullanılmasıyla ABC algoritması hem iyi bir global araştırma hemde bölgesel araştırma yapabilmektedir.



Şekil 3.3. ABC algoritmasının akış diyagramı

3.2.7. ABC Algoritmasının Adımları

Önceki bölümlerde genel hatları ile ABC algoritmasının adımları ve her bir adımda yapılan işlemler tarif edilmişti ancak bu adımları sözde kod şeklinde baştan aşağı yazmak faydalı olacaktır.

1: Eşitlik 3.1 aracılığıyla tüm $x_{i,j}, i = 1 \ldots SN, j = 1 \ldots D$, çözümlerine başlangıç değerlerinin atanması ve çözüm geliştirememe sayaçlarının sıfırlanması $(failure_i = 0)$
2: $f_i = f(x_i)$ fonksiyon değerlerinin ve bu değerlere karşılık gelen uygunluk değerlerinin, $fitness_i$, hesaplanması.
3: **repeat**
4: **for** $i = 1$ to SN **do**
5: Eşitlik 3.2'i kullanarak x_i çözümünün görevli arısı için yeni bir kaynak üret, υ_i, ve $f(\upsilon_i)$'yi (5.9) eşitliğinde yerine koyarak bu çözümün uygunluk değerini hesapla.
6: υ_i ve x_i arasında aç gözlü seleksiyon işlemi uygula ve daha iyi olanı seç.
7: x_i çözümü gelişememişse çözüm geliştirememe sayacını bir artır, $failure_i = failure_i + 1$, gelişmişse sıfırla, $failure_i = 0$.
8: **end for**
9: Eşitlik 3.5 ile gözcü arıların seçim yaparken kullanacakları uygunluk değerine dayalı olasılık değerlerlerini, p_i, hesapla.
10: $t = 0, i = 1$
11: **repeat**
12: **if** $random < p_i$ **then**
13: Eşitlik 3.2'i kullanarak gözcü arı için yeni bir kaynak, υ_i üret
14: υ_i ve x_i arasında aç gözlü seleksiyon işlemi uygula ve daha iyi olanı seç.
15: x_i çözümü gelişememişse çözüm geliştirememe sayacını bir artır, $failure_i = failure_i + 1$, gelişmişse sıfırla, $failure_i = 0$.
16: $t = t + 1$
17: **end if**

18: **until** $t = SN$

19: **if** $max(failure_i) > limit$ **then**

20: x_i'yi eşitlik 3.1 ile üretilen rastgele bir çözümle değiştir.

21: **end if**

22: En iyi çözümü hafızada tut.

23: **until** Durma kriteri

3.2.8. ABC'nin Temel Özellikleri

ABC algoritması

i) oldukça basit ve esnektir.

ii) gerçek yiyecek arayıcı arıların davranışlarına oldukça yakın şekilde simüle eder.

iii) sürü zekasına dayalı bir algoritmadır.

iv) nümerik problemler için geliştirilmiştir ama ayrık problemler için de kullanılabilir.

v) oldukça az kontrol parametresine sahiptir.

vi) kaşif arılar tarafından gerçekleştirilen küresel ve görevli ve gözcü arılar tarafından gerçekleştirilen bölgesel araştırma kabiliyetine sahiptir ve ikisi paralel yürütülmektedir.

4. BÖLÜM

SINIRLAMASIZ OPTİMİZASYON PROBLEMLERİ ÜZERİNDE YAPAY ARI KOLONİSİ ALGORİTMASININ PERFORMANS ANALİZİ

Bu bölümde sınırlamasız optimizasyon problemlemleri üzerinde bir önceki bölümde anlatılan ABC algoritmasının performansı incelenecek ve literatürde bilinen algoritmalarla kıyaslanacaktır. Ancak öncesinde ABC algoritmasının kontrol parametrelerinin algoritmanın sonuçları üzerindeki etkisi incelenerek algoritma kendi durumları içerisinde kıyaslanacaktır.

4.1. Basit Sınırlamasız Fonksiyonlar Üzerinde Çalışmalar

Evrimsel hesaplama alanında özelliklede nümerik optimizasyon konusunda farklı algoritmaları geniş test seti üzerinde kıyaslamak oldukça yaygındır. Bir algoritmanın fonksiyonlar üzerinde diğer algoritmaya göre daha iyi olup olmadığını belirlemek için her türden fonksiyonu içerecek şekilde oldukça geniş ve iyi bir set oluşturmak oldukça zor bir iştir. Bunun için algoritmanın her türden problem üzerinde iyi olmasını beklemek yerine iyi performans sergilediği türlerin belirlenerek hangi türden problemlerde hangi algoritmanın iyi olduğunun cevabını aramak daha makuldur [194].

Bir fonksiyonun birden fazla yerel optimumu varsa bu tür bir fonksiyona çok modlu denmektedir. Çok modlu fonksiyonlar algoritmaların yerel minimalardan kurtulabilmesini test etmektedirler. Bir algoritmanın keşif (exploration) yeteneği zayıf ise tüm uzayı verimli bir şekilde araştıramaz ve yerel minimalara takılabilir. Düz bir yüzeye sahip olan fonksiyonlarda araştırmanın yönü hakkında bilgi vermedikleri için çok modlu fonksiyonlar gibi algoritmalar için zorluk arz etmektedirler (Stepint, Matyas, PowerSum fonksiyonları gibi). Diğer bir grup

test problemleri ise ayrıştırılabilir veya ayrıştırılamaz niteliktedir. p- değişkenli bir fonksiyon bir tek değişkenli p tane fonksiyon cinsinden ifade edilebiliyorsa bu ayrıştırılabilir bir fonksiyondur. Ayrıştırılamayan fonksiyonlar bu formda yazılamazlar ve değişkenleri arasında birbirine bağlılık söz konusudur. Bundan dolayı ayrıştırılamayan fonksiyonlar ayrıştırılabilir olanlara nazaran daha zordur. Ayrıca problemlerin boyutuda problemle ilgili önemli bir husustur [194]. Bazı fonksiyonlarda, global minimumun olduğu bölge araştırma uzayına göre çok küçük bir noktadadır (Easom, Michalewicz ($m = 10$), Powell) yada global minimum yerel minimuma oldukça yakındır (Perm, Kowalik, Schaffer). Çok modlu fonksiyonlarda olduğu gibi, bir algoritma araştırma uzayını yeterince araştıramıyorsa ve dar kavisli vadilere sahip fonksiyonlardaki (Beale, Colville, Rosenbrock) yön değişimlerini takip edemiyorsa algoritma bu tür problemlerde başarılı olamayacaktır. Algoritmaların zorlandıkları başka bir problem de problem domeni ve fonksiyon yüzeyi arasındaki genlik derecelerindeki ölçekleme problemidir (Goldstein-Price, Trid) [195]. Fletcher-Powell ve Langerman fonksiyonları simetrik olmayan ve optimalarının rasgele dağılımlı olduğu fonksiyonlardır. Bu durumda problem parametrelerinin kopyalanmasını kullanan algoritmaların simetriden kaynaklı avantajı ortadan kalkmış olacaktır. Gürültülü fonksiyonlarda beklenen uygunluk değeri uniform yada gaussian tipteki rasgele gürültü üretecine bağlıdır (Quartic). Bu fonksiyonda algoritmanın başarımının düşük olması gürültülü verilerde performansının düşük olacağı anlamına gelir. Step fonksiyonu ise süreksiz fonksiyonlara bir örnektir. Penalized fonksiyonları sinüs fonksiyonunun farklı peryodlarının kombinasyonundan kaynaklanan bir zorluğa sahiptir [196].

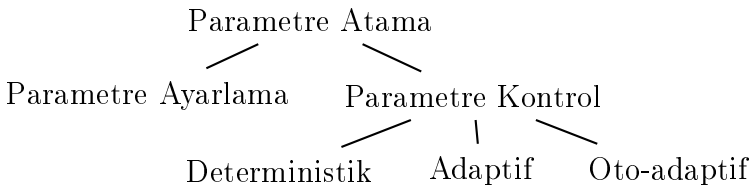
4.1.1. Kontrol Parametrelerinin Etkisi

Optimizasyon problemlerinin çözümü ile kullanılan algoritmalar karakteristiklerine göre bölgesel, küresel, sezgisel, stokastik, deterministik, iteratif, popülasyon tabanlı, doğa temelli gibi gruplara ayrılabilir. Algoritmaların performansı da içinde bulunduğu grubun özelliklerine veya benzetimini yaptığı olayın doğasına göre değişebilmektedir. Algoritmanın sahip olduğu operatörler benzetimini yaptığı olayın süreçlerini modelleyecek şekilde çalışır ve araştırmanın başarımını doğrudan

etkiler. Operatörlerin ortak özelliği popülasyondaki bireylerde kontrollü bir değişiklik yapmaya veya onlarla ilgili karar süreci oluşturmaya yöneliktir. Bu işlemler sırasında algoritma, operatörlerinin çalışmalarında rol alan popülasyon büyüklüğü, değişim oranı gibi bazı kontrol parametrelerinin belirlenmesine ihtiyaç duymaktadır. Ancak bir parametre seti belirli bir problem yüzeyinde optimal sonuca ulaşmayı sağlayabilirken, başka bir problemde başarısız olabilmektedir. Bir diğer hususta araştırmanın başında iyi olduğu söylenebilen bir parametre setinin araştırmanın ileriki aşamalarında iyi sonuç üretememesidir. Bundan dolayı bir parametre setinden genel olarak optimal olması ve iyi performans üretmesi beklenir [197]. Algoritmaların performansını kontrol parametrelerinin nasıl etkilediğini belirlemek amacıyla analiz yapılmasına ihtiyaç duyulmakta ve problem yüzeyinin karakteristiğine göre parametre setini dinamik ve kendi-kendine uyarlabilir kılmak amacıyla incelemeler yapılması gerekmektedir. Gelişime dayalı algoritmaların parametre kontrol tekniklerinin sınıflandırılmasında Eiben ve ark. şu hususları dikkate almışlardır [198, 199]:

- Ne değişecek?
- Bu değişiklik nasıl yapılacak? Parametrelerin ayarlanmasında kullanılan metotların sınıflandırılması ile ilgili sınıflandırma Şekil 4.1'de gösterilmiştir. Parametre ayarlama (parametre tunning) kontrol parametrelerinin araştırma süreci başlamadan değerlerinin atanması iken, parametre kontrolü (parameter control) belirli bir kurala yada fonksiyona bağlı olarak araştırma sürecinde bunlara değer atanması ile ilgilidir.
- Ne tür belirtiler değişikliği bildirir?
- Değişikliğin miktarı nedir?

Gelişim algoritmaları için yapılan bu sınıflandırma ABC algoritması için de kullanılabilir. Eiben ve ark. gelişim algoritmasında dikkate aldıkları hususlar göz önüne alınarak ve bir anoloji kurularak "Ne Değişecek?" başlığı altında ABC algoritmasının bazı bileşenleri gösterilebilir. Bunlar değişim operatörü, seleksiyon operatörü, popülasyon büyüklüğü ve limit değeridir. Daha önceden de belirtildiği



Şekil 4.1. Gelişim algoritmalarının parametre ayarlanması ile ilgili sınıflandırması

gibi ABC algoritmasının da kendine has araştırma öncesinde veya araştırma sürecinde ayarlanması gereken bazı kontrol parametreleri bulunmaktadır. ABC algoritmasının kontrol parametreleri çalışma öncesinde ayarlanarak performans üzerindeki etkisi literatürde bilinen dokuz test fonksiyonu üzerinde incelenecek ve ABC'nin kontrol parametrelerin ayarlanması ile ilgili dikkat edilmesi gereken noktalar belirlenecektir. Ayrıca, ABC algoritmasının kontrol parametrelerinin çalışma süresince bir kurala yada fonksiyona bağlı olarak değiştirilmesi de incelenecektir.

4.1.1.1. Koloni Büyüklüğü ve Limitin Etkisi

Temel ABC algoritması üç kontrol parametresine sahiptir: maksimum çevrim sayısı, koloni büyüklüğü ve limit. Bunlardan parametre ayarlamasında dikkate alınacak kontrol parametreleri koloni büyüklüğü ve limit parametresidir. Ancak problemlerin boyutunun artmasıyla algoritma performansının nasıl değiştiğinin ve algoritma performansının parametrelerin başlangıç aralığına olan hassasiyetini göstermek amacıyla farklı problem boyutları ve farklı başlangıç aralıkları için incelemeler yapılmıştır.

Parametre analizi için yapılan çalışmalarda maksimum çevrim sayısı 10000 olarak alınmış ve tablolarda 30 koşmaya ait ortalama sonuçlar ve standart sapmaları sunulmuştur.

Koloni büyüklüğünün (popülasyon büyüklüğü) algoritmanın performansını nasıl etkilediğini analiz etmek veya minimaya ulaşabilmek için yeterli koloni büyüklüğünün ne olması gerektiğine karar verebilmek amacıyla çalışmalar

yapılmıştır. İlk olarak Tablo 4.1'deki foksiyonlar üzerinde koloni büyüklüğü sabit (SN=100) tutularak problem boyutu (D) 10, 100 ve 1000 olacak şekilde değiştirerek sonuçlar alınmıştır. Aynı parametre değerleri kullanılarak deneyler diferansiyel gelişim (DE) ve parçacık sürüsü optimizasyon (PSO) algoritmaları için de tekrarlanarak elde edilen sonuçlar kaydedilmiştir. Değişen problem boyutları için yeterli popülasyon boyutunu inceleyecen deneylerin sonuçları Tablo 4.2'de sunulmaktadır. Tablo 4.2'deki sonuçlara göre sabit popülasyon büyüklüğü için PSO algoritmasnın performansı problemin boyutu arttıkça kötüleşmektedir. DE algoritması sadece Rosenbrock fonksiyonu üzerinde en iyi sonucu düşük boyutta üretmişken problem boyutu arttıkça DE'nin performansı kötüleşmektedir. Ancak, ABC algoritması, problem boyutuna göre kararlı performansının devam etmesi nedeniyle DE algoritmasının önüne geçmektedir. D=10 durumunda sekiz fonksiyon (Sphere, Rastrigin, Griewank, Schwefel, Ackley, Step, Penalized, Dixon-Price) üzerinde ABC algoritması daha iyi performans sergilemiş ve D=1000'e çıkarıldığında kabul edilebilir sonuçlar üretmiştir.

İkinci deneyde sabit bir boyut için hangi popülasyon büyüklüğünün yeterli olduğu analiz edilmektedir. Yine Tablo 4.1'de verilen fonksiyonlar üzerinde DE ve PSO algoritmaları için de sonuçlar alınarak ABC algoritması ile kıyaslamaları gerçekleştirilmiştir. Sonuçlar Tablo 4.3'de sunulmaktadır. Bu tablodan ABC algoritmasının daha küçük veya daha büyük popülasyon büyüklüklerinde DE ve PSO algoritmalarına kıyasla gürbüzlüğünü koruduğu görülmektedir. Buradan yola çıkarak ABC algoritması uygun popülasyon büyüklüğünün belirlenmesi için ince ayar yapmaya fazla gerek duymamaktadır denilebilir. Nispeten düşük popülasyon büyüklüklerin de iyi sonuçlar üretmektedir. Belli bir problem boyutu için popülasyon büyüklüğünün artmasının kuşkusuzki performans üzerinde olumlu etkisi vardır. Ancak belli bir aşamadan sonra işlem karmaşıklığı artmakta ve tesadüfen sonuç bulunması ortaya çıkabilmektedir. Yani algoritmanın araştırma yeteneğinden istifade edilememekte demektir.

Limit, ABC algoritmasında tüketilmiş olan terkedilecek yiyecek kaynağının belirlenmesinde kullanılan kontrol parametresidir. Limit parametresinin analizi ile kaşif arı biriminin analizi yapılmış olmaktadır. Kaşif arı çıkması ve çıkmaması

durumunun performans üzerindeki etkisini incelemek yani limit parametresinin etkisini analiz etmek amacıyla farklı limit değerleri durumunda sonuçlar alınarak Tablo 4.4-4.6'da verilmiştir. Tablo 4.4'de her bir koşmada problem boyutu (D) 100, popülasyon büyüklüğü (SN) 100 olarak alınmıştır. Bu sonuçlardan da görüldüğü gibi, düşük limit değerleri kullanıldığında yakınsama sağlanamadan popülasyondaki bilgi yok olmakta ve sonucunda algoritmanın performansı düşük olmaktadır. Büyük limit değerleri ise performans üzerinde önemli bir değişiklik yapmamaktadır. Bunun sebebi de popülasyonun büyük olmasından dolayı yeterli derecede farklılığın (diversity) mevcut olmasıdır. Yani algoritma, popülasyonda yeterince farklılık olduğundan yeni yerlerin (çözümlerin) keşfedilmesi için kaşif arı üretilmesine gerek duymamaktadır. Popülasyonun büyük seçilmiş olması limitin üst sınırı hakkında fikir vermediği için popülasyon büyüklüğünü 100'den 10'a indirerek çalışma tekrarlanmıştır. Popülasyon büyüklüğünün 10 olması durumunda elde edilen sonuçlar Tablo 4.5'de sunulmaktadır. Tablo 4.5'den bazı fonksiyonlarda ABC algoritmasının limit'in 500 değeri civarında iyi sonuçlar ürettiği görülmektedir. Bu sonuç kabul edilebilir ve beklenen bir sonuçtur. Nedeni ise 4. Bölümde detayları verilen ABC algoritmasının her bir çevrimde sadece bir parametre değiştirmesinden dolayı tüm parametrelerin değişebilmesi için en az D kadarlık çevrim gerektirmesidir. Dolayısıyla SN tane arının bir kaynak üzerinde değişiklik yaptığı düşünülürse $SNxD$ civarı bir değerin limit parametresi için uygun olduğu değerlendirilebilir. Ancak daha düz yüzeylere sahip olan fonksiyonlarda (Sphere, Ackley, Step, Dixon-Price) limit değerinin artması fonksiyonun başarımı üzerinde olumlu etki yapmamaktadır. Araştırma uzayını daha da zorlaştırarak kaşif arıların etkisini daha net görebiliriz. Parametrelerin başlangıç değerlerini optimum çözümü içermeyen bir aralıktan seçtiğimizde, kaşif arılar sayesinde araştırmanın bu yöne doğru nasıl kaydığını göstermek amacıyla parametrelerin uzayın sol çeyreğinde değer alacağı şekilde başlatma yapılmıştır. Deneyler problem boyutu 100, popülasyon büyüklüğü 10 olacak şekilde gerçekleştirilerek sonuçlar Tablo 4.6'da ve Şekil 4.2(a)-4.4(c)'de verilmiştir. Tablo 4.6'dan düşük limit değerlerinde ihtiyaçdan çok sayıda kaşif arı üretilmesine sebep olunurken büyük limit değerlerinde ise kaşif arıların az sayıda çıkması durumu ortaya çıktığı ve algoritmanın keşif yeteneğinin zayıflamasına sebep olduğu anlaşılmaktadır. Ancak araştırma zorlaştıkça $SNxD$ civarı olarak önerilen

limit değerinin daha düşük olması gerektiği görülmektedir. Şekil 4.2(a)-4.4(c)'de bunu açıkça göstermektedir.

Tablo 4.1. Parametre analizinde kullanılan test fonksiyonları

	Fonksiyon	Aralık	Formülasyon	Minimum
1	Sphere	$[-100, 100]^n$	$f(x) = \sum_{i=1}^{n} x_i^2$	0
2	Rosenbrock	$[-100, 100]^n$	$f(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	0
3	Rastrigin	$[-5.12, 5.12]^n$	$f(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	0
4	Griewank	$[-600, 600]^n$	$f(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	0
5	Schwefel	$[-500, 500]^n$	$f(x) = \sum_{=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert})$	-418.983*n
6	Ackley	$[-32, 32]^n$	$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	0
7	Step	$[-100, 100]^n$	$f(x) = \sum_{i=1}^{n} (\lfloor x_i + 0.5 \rfloor)^2$	0
8	Penalized	$[-50, 50]^n$	$f(x) = \frac{\pi}{n}\left\{10\sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\right\} + \sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{u}(\mathrm{x_i}, 10, 100, 4)$ $yi = 1 + \frac{1}{4}(x_i + 1)$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & , \quad x_i > a \\ 0 & , \quad -a \le x_i \le a \\ k(-x_i - a)^m & , \quad x_i < -a \end{cases}$	0
9	Dixon-Price	$[-10, 10]^n$	$f(x) = (x_1 - 1)^2 + \sum_{i=2}^{n} i(2x_i^2 - x_{i-1})^2$	0

Tablo 4.2. PSO, DE ve ABC algoritmalarının farklı problem boyutları için deneysel sonuçları, $SN = 100$, Ort:Ortalama, SD: Standar sapma.

		PSO			DE			ABC		
	D	10	100	1000	10	100	1000	10	100	1000
Sphere	Ort	4.13E-17	5.14E-16	9723.034942	4.41E-17	8.84E-17	329214.6744	4.88E-17	1.08E-15	0.058275686
	SD	7.71E-18	3.12E-16	3920.944041	8.09E-18	4.29E-17	917847.3604	5.21E-18	1.04E-16	0.021093306
Rosenbrock	Ort	0.425645397	113.143751	1679629.019	4.22E-17	132.3488752	14373397912	0.013107593	0.054865327	2603.968539
	SD	1.187984439	48.99432331	648462.4744	1.09E-17	41.72265261	361340776.6	0.008658828	0.045566135	599.4022496
Rastrigin	Ort	7.362692992	148.2486456	2722.799729	0.099495906	133.1138439	1674.782779	4.76E-17	1.08E-15	735.8480014
	SD	2.485404145	17.76489083	83.14754621	0.298487717	106.6728854	96.86409615	4.40E-18	8.99E-17	24.75231998
Griewank	Ort	0.059270504	0.048643996	86.03568115	0.008127282	0.000739604	266.1639753	5.10E-19	4.92E-17	0.10290266
	SD	0.03371245	0.063166266	29.1502045	0.009476456	0.002218812	335.3504904	1.93E-19	4.25E-18	0.068217103
Schwefel	Ort	-2654.033431	-20100.36156	-187704.1438	-4166.141206	-31182.49983	-252854.5198	-4189.828873	-41898.28873	-350890.8062
	SD	246.5263242	1763.156655	11097.95553	47.37533385	2078.47339	17724.02042	9.09E-13	3.30E-10	2279.801625
Ackley	Ort	4.67E-17	0.732022399	8.741445965	4.86E-17	2.14E-16	17.47129372	1.71E-16	4.21E-15	3.200412604
	SD	8.06E-18	0.755456829	0.784830594	6.55E-18	4.53E-17	3.815946124	3.57E-17	3.09E-16	0.133628837
Step	Ort	0	1.7	13538.3	0	0	47653.1	0	0	0
	SD	0	2.60959767	3112.434162	0	0	27093.57104	0	0	0
Penalized	Ort	4.13E-17	0.184850201	9706.265611	3.65E-17	0.01243899	35667997246	4.57E-17	1.07E-15	0.000332724
	SD	9.11E-18	0.314058054	6077.475023	9.90E-18	0.028500601	891866689.6	7.28E-18	6.72E-17	0.00093282
Dixon Price	Ort	0.666666667	2.07647225	615801.4332	0.666666667	0.666666667	3246360171	4.07E-16	1.26E-06	2704.747482
	SD	1.67E-14	4.229416739	642782.3094	4.97E-17	3.51E-17	1082157842	1.22E-16	7.21E-07	360.341568

Tablo 4.3. PSO, DE ve ABC algoritmalarının farklı popülasyon büyüklükleri için deneysel sonuçları, $D = 100$, Ort:Ortalama, SD: Standart sapma.

		PSO			DE			ABC		
	SN	50	100	200	50	100	200	50	100	200
Sphere	Ort	6.16E-16	5.14E-16	3.11E-16	1.54E-16	8.84E-17	1.11E-16	1.12E-15	1.08E-15	1.02E-15
	SD	2.86E-16	3.12E-16	5.02E-17	1.08E-17	4.29E-17	4.87E-17	8.78E-17	1.04E-16	7.56E-17
Rosenbrock	Ort	152.6300403	113.143751	129.8648065	244.4621408	132.3488752	65.57549295	0.334214887	0.054865327	0.033840879
	SD	55.22881247	48.99432331	72.04063248	167.3699563	41.72265261	2.539552946	0.579373464	0.045566135	0.058330247
Rastrigin	Ort	161.6805719	148.2486456	159.3921715	90.740202	133.1138439	666.5000853	1.14E-15	1.08E-15	1.04E-15
	SD	30.0214669	17.76489083	25.03970003	11.60140207	106.6728854	53.08918163	6.26E-17	8.99E-17	1.13E-16
Griewank	Ort	0.066057787	0.048643996	0.00320331	0.034770873	0.000739604	2.52E-17	8.33E-17	4.92E-17	4.93E-17
	SD	0.073721225	0.063166266	0.00505551	0.038642093	0.002218812	2.86E-18	9.96E-17	4.25E-18	2.90E-18
Schwefel	Ort	-18931.71298	-20100.36156	-21200.01235	-30783.55835	-31182.49983	-16666.82437	-41898.21377	-41898.28873	-41898.28873
	SD	1453.154666	1763.156655	2155.846443	930.3749272	2078.47339	1663.709053	0.223900743	3.30E-10	3.99E-12
Ackley	Ort	1.857181245	0.732022399	6.80E-16	2.666357404	2.14E-16	1.74E-14	4.48E-15	4.21E-15	4.11E-15
	SD	0.739376943	0.755456829	9.64E-17	0.607483624	4.53E-17	5.61E-15	2.14E-16	3.09E-16	1.59E-16
Step	Ort	38.5	1.7	0.2	0	0	0	0	0	0
	SD	56.90210892	2.60959767	0.4	0	0	0	0	0	0
Penalized	Ort	0.196111794	0.184850201	0.049804791	7000.171624	0.01243899	0.003110071	1.17E-15	1.07E-15	1.00E-15
	SD	0.54742191	0.314058054	0.077774663	12773.11331	0.028500601	0.009330212	9.12E-17	6.72E-17	6.86E-17
Dixon Price	Ort	1.697096319	2.07647225	0.666666667	12.31123978	0.666666667	0.666666667	1.64E-06	1.26E-06	6.23E-07
	SD	3.090995081	4.229416739	2.81E-16	8.511926958	3.51E-17	3.51E-17	1.25E-06	7.21E-07	4.45E-07

Tablo 4.4. ABC algoritmasının farklı limit değerleri için sonuçları, $D = 100$, $SN = 100$, Ort:Ortalama, SD: Standart sapma.

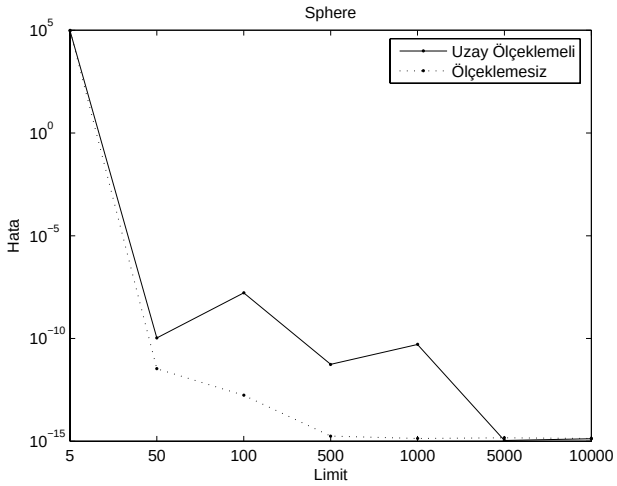
Fonksiyon		Limit						
		5	50	100	500	1000	5000	10000
Sphere	Ort	40172.619	1.30E-14	2.21E-15	1.27E-15	1.11E-15	1.09E-15	1.05E-15
	SD	4636.9765	1.28E-14	3.23E-16	1.05E-16	8.94E-17	1.00E-16	9.55E-17
Rosenbrock	Ort	75479633	0.3687739	0.1066848	0.0577759	0.0305455	0.0400428	0.0903778
	SD	16643440	0.2485734	0.1231579	0.0384831	0.0295326	0.048434	0.1663324
Rastrigin	Ort	722.41549	4.13E-08	5.27E-11	1.24E-15	1.12E-15	1.09E-15	1.07E-15
	SD	30.956511	8.44E-08	1.55E-10	1.95E-16	1.14E-16	1.04E-16	1.04E-16
Griewank	Ort	387.05656	2.36E-13	6.56E-15	5.94E-17	4.98E-17	4.76E-17	4.92E-17
	SD	44.678379	3.23E-13	4.18E-15	3.33E-17	4.13E-18	4.93E-18	3.69E-18
Schwefel	Ort	-20848.434	-41347.734	-41612.546	-41898.289	-41898.289	-41898.289	-41898.289
	SD	626.53929	106.94234	106.48415	6.75E-08	0.0003771	3.88E-08	2.97E-06
Ackley	Ort	18.376048	2.62E-13	1.01E-14	4.75E-15	4.35E-15	4.31E-15	4.29E-15
	SD	0.390422	1.56E-13	1.57E-15	2.12E-16	2.27E-16	1.72E-16	2.68E-16
Step	Ort	46155.1	13.5	0.7	0	0	0	0
	SD	8465.072	5.4267854	0.9	0	0	0	0
Penalized	Ort	86851655	1.64E-14	2.48E-15	1.15E-15	1.10E-15	1.06E-15	1.05E-15
	SD	42828908	5.48E-15	5.41E-16	8.45E-17	9.15E-17	1.12E-16	8.59E-17
Dixon Price	Ort	1565358.2	0.0031539	1.07E-06	1.47E-06	1.96E-06	9.17E-07	1.01E-06
	SD	674594.38	0.0017079	5.76E-07	7.28E-07	2.26E-06	3.82E-07	4.45E-07

Tablo 4.5. ABC algoritmasının farklı limit değerleri için sonuçları, $D = 100$, $SN = 10$, Ort:Ortalama, SD: Standart sapma.

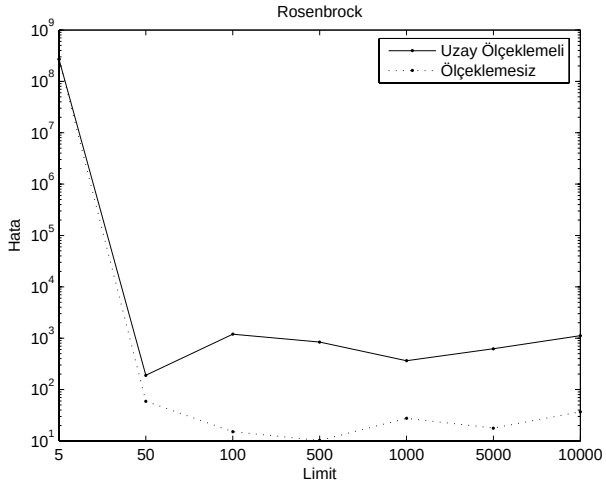
Fonksiyon		Limit						
		5	50	100	500	1000	5000	10000
Sphere	Ort	98095.07116	3.37E-12	1.72E-13	1.77E-15	1.36E-15	1.46E-15	1.36E-15
	Std	12959.09176	1.02E-11	3.58E-13	3.05E-15	1.64E-16	6.73E-16	1.90E-16
Rosenbrock	Ort	267422928	59.2079779	15.18515019	10.23990112	27.60392216	17.75898437	36.92995963
	Std	51812195.52	218.0286192	43.30865642	23.98853007	56.06136362	47.47330859	114.2961123
Rastrigin	Ort	942.1587016	8.565987682	8.172342827	7.604827352	10.70040148	9.658915175	9.883292388
	Std	59.80687384	3.194695616	4.648093126	3.697719518	6.798092714	4.932444925	4.839869249
Griewank	Ort	869.2096345	0.003233202	0.002041184	0.002975775	0.028078594	0.00810774	0.010963982
	Std	134.3235844	0.010931443	0.005640278	0.008799799	0.085084827	0.017752733	0.028803579
Schwefel	Ort	-17790.24013	-39738.06939	-40051.94037	-40339.09933	-40219.75069	-40230.36359	-40297.86110
	Std	1015.674426	336.6597611	503.2839908	407.1759947	422.4181709	461.630346	367.496268
Ackley	Ort	19.52803902	0.212870212	0.255519831	0.255407435	0.40063048	0.319323308	0.412635114
	Std	0.22340354	0.296340225	0.340478681	0.381650479	0.511608274	0.483124854	0.454188474
Step	Ort	106992.4	55.66666667	7.866666667	0	0	0	0
	Std	14232.35432	22.35073949	4.030991055	0	0	0	0
Penalized	Ort	576542253.4	9.44E-10	6.10E-14	3.02E-14	1.55E-12	1.61E-06	0.00207338
	Std	156824154.4	4.68E-09	1.11E-13	8.81E-14	8.36E-12	8.67E-06	0.007757879
Dixon Price	Ort	6197348.721	1.919382041	3.109303975	2.467438828	1.862482692	1.686539242	3.110083529
	Std	1661787.691	4.344332443	5.509126939	5.304917011	4.863661272	3.869273525	5.447932166

Tablo 4.6. Popülasyonun araştırma uzayının sol çeyrek bölümünde oluşturulması durumunda ve farklı limit değerleri kullanılarak ABC algoritması ile elde edilen sonuçlar, $D = 100$, $SN = 10$, Ort:Ortalama, SD: Standart sapma.

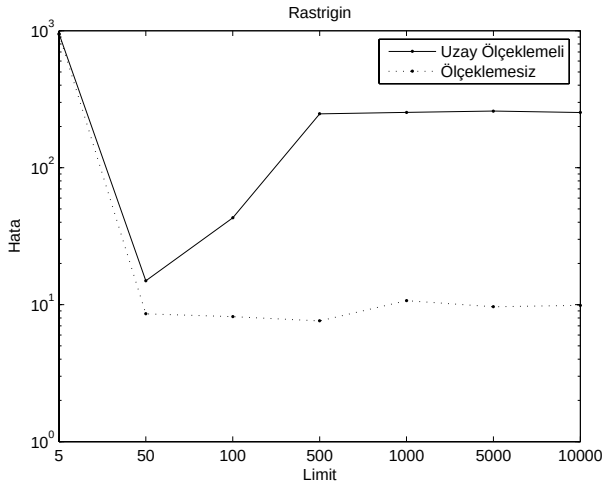
Fonksiyon		Limit						
		5	50	100	500	1000	5000	10000
Sphere	Ort	95105.3328	1.07E-10	1.67E-08	5.54E-12	5.11E-11	1.09E-15	1.33E-15
	Std	14661.92645	4.28E-10	8.28E-08	2.89E-11	2.75E-10	1.92E-16	5.25E-16
Rosenbrock	Ort	269505615.3	190.3554476	1199.876802	838.9745817	363.3571176	624.7432321	1121.62295
	Std	68373898.46	322.9181696	4998.402971	2472.735357	571.2721707	849.2494919	4397.815502
Rastrigin	Ort	950.5869637	14.94996831	43.07122725	247.1658087	253.5954414	259.1623807	253.0372257
	Std	56.02863864	9.560372871	22.61337802	43.24912076	36.28455454	42.23473145	39.77043485
Griewank	Ort	859.7045978	0.008499821	0.004393589	0.032915998	0.043504983	1.4181218	0.566366173
	Std	141.3025005	0.018106107	0.014055128	0.048141907	0.061808438	4.657248824	1.952489982
Schwefel	Ort	-17107.32873	-39327.94192	-39491.03497	-36680.47542	-35317.9738	-32379.39516	-29995.05183
	Std	805.7904857	474.3960215	344.835107	1299.968374	1191.019816	2034.384921	128.6856694
Ackley	Ort	19.61754314	0.392724383	9.043197628	10.16441257	10.27369945	10.71427402	10.27599214
	Std	0.225559909	0.463505056	3.231658614	2.571042729	2.953414239	1.46314876	2.926421499
Step	Ort	101259.8667	61.6	9.466666667	0	0	5.1	182.4666667
	Std	15550.22787	24.55415783	5.155795014	0	0	27.46434052	601.9868622
Penalized	Ort	587043270.1	7.38E-06	3.66E-08	0.014249608	0.004113517	0.001008872	0.001617202
	Std	160656052.9	3.98E-05	1.13E-07	0.055924082	0.022151949	0.005432943	0.006300558
Dixon Price	Ort	6185260.626	19.0649427	34.04251776	21.89179569	20.64629783	20.92190235	34.0333849
	Std	1514959.996	15.58836441	57.80617854	13.03460008	20.90865202	10.89836716	59.58264989



(a) Sphere

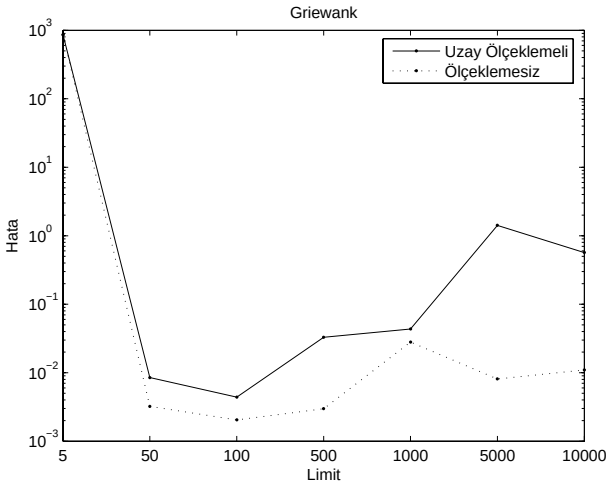


(b) Rosenbrock

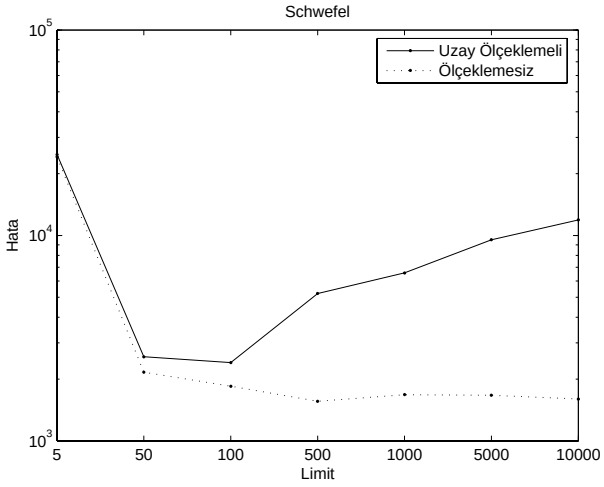


(c) Rastrigin

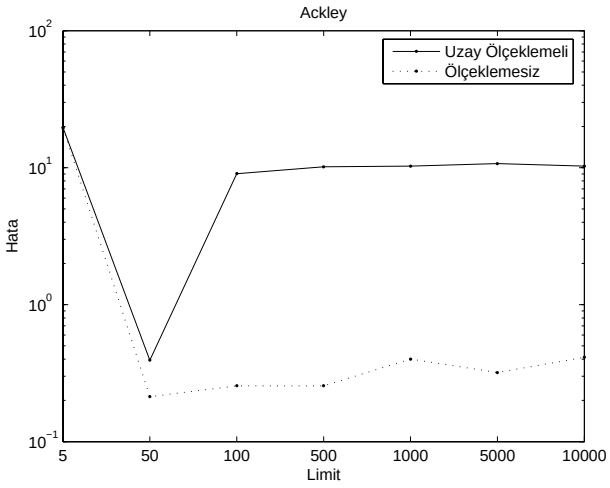
Şekil 4.2. Sphere, Rosenbrock ve Rastrigin fonksiyonları için başlangıç popülasyonun tüm uzayda ve sol çeyrek uzayda oluşturulması durumlarında limit parametresinin etkisinin analizi, $D = 100$, $SN = 10$.



(a) Griewank

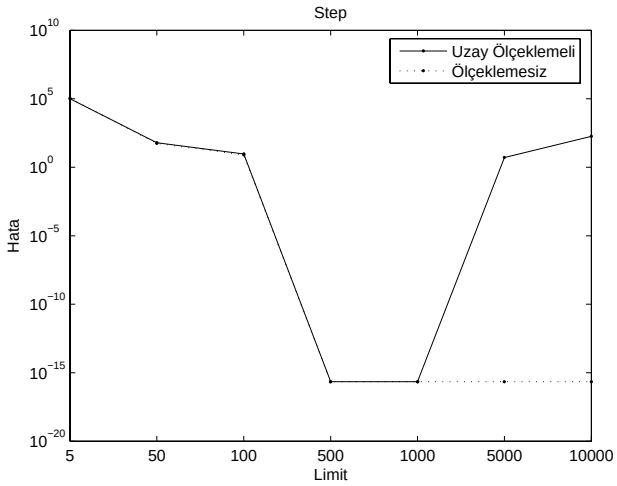


(b) Schwefel

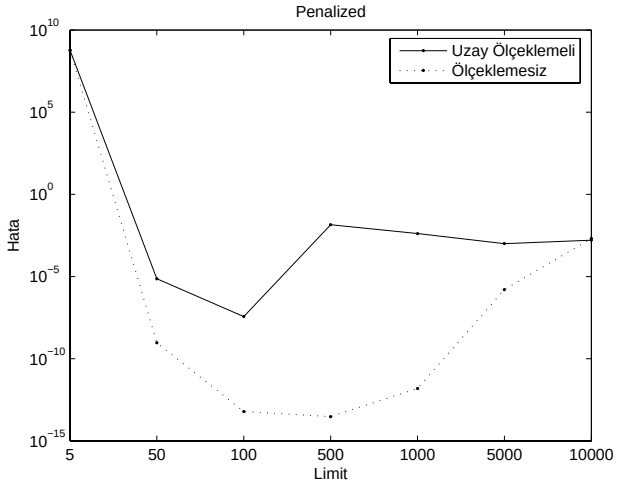


(c) Ackley

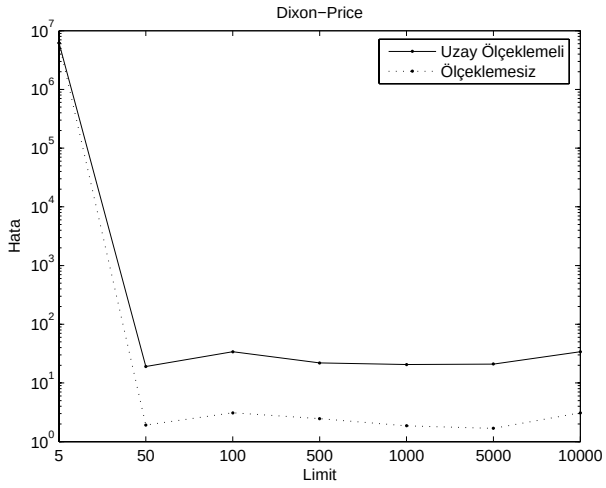
Şekil 4.3. Griewank, Schwefel ve Ackley fonksiyonları için başlangıç popülasyonun tüm uzayda ve sol çeyrek uzayda oluşturulması durumlarında limit parametresinin etkisinin analizi, $D = 100$, $SN = 10$.



(a) Step



(b) Penalized



(c) Dixon-Price

Şekil 4.4. Step, Penalized ve Dixon-Price fonksiyonları için başlangıç popülasyonun tüm uzayda ve sol çeyrek uzayda oluşturulması durumlarında limit parametresinin etkisinin analizi, $D = 100$, $SN = 10$.

4.1.1.2. Değişim Oranı ve Ölçekleme Faktörünün Etkisi

Temel ve standart ABC algoritmasında yeni bir çözüm üretilirken mevcut çözümün sadece bir parametresi değiştirilmektedir. Ancak hızlı yakınsamanın istendiği durumlarda değiştirilecek parametre sayısı bir kontrol parametresine bağlı olarak artırılabilir. Bunun için değişim oranı (Modification rate, MR) adı verilen yeni bir kontrol parametresi kullanılmaktadır. Algoritmanın (3.2) ile verilen yeni çözüm üretme ifadesi (4.1) ile değiştirilmiştir. Bu ifadeye göre her j parametresi için uniform dağılımlı rasgele [0,1] aralığında bir rasgele sayı üretilmekte ve bu sayı kontrol parametresi MR değerinden küçük ise j. parametresi değişime uğramaktadır. Mevcut çözümün tüm parametreleri için bu kontrol yapıldığında çözüme ait herhangi bir parametre değişim için seçilmemişse en azından bir parametrenin değişimini garanti etmek amacıyla rasgele seçilen bir parametre ile (3.2) ifadesine bağlı olarak değişime uğratılır:

$$\upsilon_{ij} = \begin{cases} x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) & , \quad if \quad R_j < MR \\ x_{ij} & , \quad \text{otherwise} \end{cases} \tag{4.1}$$

(3.2) ve (4.1) ifadesindeki ϕ_{ij} terimi, adım büyüklüğü olan $(x_{ij} - x_{kj})$ değerini ölçeklemekte ve [-1,1] arasında değer almaktadır. Adım büyüklüğünün [-1,1] arasında olan değeri SF (scaling factor) olarak adlandırılan bir parametre ile ölçeklenecek olursa (4.1) eşitliği (4.2) haline gelir.

$$\upsilon_{ij} = \begin{cases} x_{ij} + SF * \phi_{ij}(x_{ij} - x_{kj}) & , \quad if \quad R_j < MR \\ x_{ij} & , \quad \text{otherwise} \end{cases} \tag{4.2}$$

(4.1) ve (4.2) eşitliklerindeki MR ile (4.2) eşitliğindeki SF parametrelerinin sınırlamasız fonksiyonlar üzerinde algoritmanın performansına yaptığı etkiyi incelemek amacıyla [200] çalışmasında kullanılan Tablo 4.7'deki fonksiyonları kullanarak MR ve SF'nin değişik değerleri ile sonuçlar elde edilmiştir. Bu denemelerde MR'nin {0,0.1,0.3,0.5,0.7,0.9,1} ve SF'nin ise {0.7,0.5,0.3,0.1,ASF} değerleri dikkate alınmıştır. Adaptif SF (ASF) ifadesi SF'nin otomatik olarak

algoritmanın çalışması sırasında değiştiğini göstermektedir. Bu otomatik değişim Rechenberg'in 1/5 kuralı kullanılarak gerçekleştirilmiştir. Rechenberg'in 1/5 kuralına göre başarılı değişimlerin tüm değişimlere oranı 1/5 olmalıdır [201]. Her m çevrimde 1/5 kuralına göre adım büyüklüğü (4.3) uyarınca ayarlanmaktadır:

$$SF(t+1) = \begin{cases} SF(t) * 0.85 & if\ \varphi(m) < 1/5 \\ SF(t)/0.85 & if\ \varphi(m) > 1/5 \\ SF(t) & if\ \varphi(m) = 1/5 \end{cases} \tag{4.3}$$

Eğer algoritmanın m çevrim boyunca gerçekleştirdiği başarılı değişimleri tüm değişimlere oranı 1/5'den küçük ise ince ayar yapmak amacıyla SF azaltılır, bu oran 1/5'den büyük ise araştırmanın hızını arttırmak gayesiyle SF arttırılır.

Tablo 4.7'de tek modlu ve çok modlu olmak üzere iki grup test fonksiyonu bulunmaktadır. Tek modlu olan fonksiyonlar bir bölgesel minimumunun olduğu Sphere ve Rosenbrock fonksiyonlarıdır. Sphere, konveks yapılı ve sürekli bir fonksiyondur. Rosenbrock fonksiyonu ise uzun, dar, parabolik yapıda ve vadi şeklindedir. Fonksiyonun parametreleri arasında ilişki bulunduğundan gradyent bilgisi araştırmayı optimum çözüme doğru götürmez; dolayısıyla global optimuma yakınsaması güç olan bir fonksiyondur. İkinci grup sürekli olan Ackley, Griewank, Weierstrass, Rastrigin, Schwefel ve sürekli olmayan Rastrigin ve fonksiyonlarından oluşan çok modlu fonksiyonlar grubudur. Ackley fonksiyonu exponansiyel bir terim içermesinden dolayı çok sayıda yerel minimaları olan araştırma yüzeyine sahiptir. Griewank fonksiyonu çarpım terimi içerdiğinden parametreleri arasında ilişki olan bir fonksiyondur ve ($n > 30$) şartını sağlayan problem boyutlarında fonksiyonun çok modluluğu ortadan kalkmakta ve tek modlu olarak görünmektedir. Weierstrass fonksiyonu her noktada sürekli ama hiç bir noktada türevlenebilir değildir. Sürekli olan ve olmayan Rastrigin fonksiyonları Sphere fonksiyonuna yerel minimalar gelmesi amacıyla cosinüs modulasyonunun eklenmesiyle oluşur. Schwefel fonksiyonunun yüzeyinde çok sayıda vadi ve tepeler bulunmaktadır. İkinci en iyi minimum global minimumdan uzak ve global minimum araştırma uzayının sınırlarında bulunmaktadır.

Deneylerde 10 boyutlu durum için koloni büyüklüğü 10 ve maksimum değerlendirme

sayısı 30000 olarak alınmıştır. Her deney her fonksiyon için 30 kere bağımsız olarak tekrarlanmıştır. ABC algoritmasının sonuçları kendi içinde kıyaslanırken ayrıca [200] çalışmasında verilen PSO'nun türevlerinin sonuçları ile de kıyaslanmıştır. PSO-w (PSO with inertia factor), PSO-cf (PSO with constriction factor), PSO-w-local (PSO with inertia factor using local neighbourhood), PSO-cf-local (PSO with constriction factor using local neighbourhood), UPSO (Unified PSO combining local and global neighbourhood topologies), FDR-PSO (Fitness Distance Ratio based PSO uses the neighbour with higher fitness), FIPS (Fully Informed Particle Swarm), CPSO-H (Cooperative PSO), CLPSO (Comprehensive Learning PSO) [200]'da sunulan çalışmada kullanılan PSO'nun türevleridir. Deneysel sonuçlar kıyaslanırken hata oranı 10^{-7}'den küçükse pratik açıdan anlamlı bir farkın olmadığı kabul edilir [202]. PSO varyantlarının sonuçları ve ABC algoritmasının farklı MR, SFF ve *limit* değerleri için alınan sonuçları Sphere, Rosenbrock, Ackley ve Griewank fonksiyonları için Tablo 4.8'de, Weierstrass, Rastrigin, Sürekli olmayan Rastrigin ve Schwefel fonksiyonları için Tablo 4.9'da verilmektedir. En iyi olan sonucu, diğer sonuçlardan ayırmak amacıyla koyu renkte tabloya yazılmıştır.

Sonuçlardan, Rosenbrock fonksiyonu için ABC algoritmasının adım büyüklüğünü otomatik olarak belirlediği durumda en iyi sonucu verdiği görülmektedir. Griewank ve Schwefel fonksiyonlarında, CLPSO en iyi sonucu vermektedir. Diğer fonksiyonlarda (Sphere, Ackley, Rastrigin, Sürekli olmayan Rastrigin) CLPSO ve temel ABC (SF=1, MR=0) algoritmaları benzer performans sergilemişlerdir. Rosenbrock fonksiyonu haricindeki diğer fonksiyonlarda başlangıç çözüm aralıkları araştırma uzayı aralığından farklıdır. Başlangıç aralıkları ve araştırma uzayı aralıkları Tablo 4.7'de sunulmuştur. ABC algoritması, kaşif arı biriminin araştırma uzayının verimli şeklide küresel olarak araştırılmasını sağlamasından dolayı başlatma koşullarına karşı oldukça dayanıklı ve bu yüzden gürbüz bir algoritmadır.

Değişim Oranı (MR) ile ölçekleme oranı (SF) ABC algoritmasının yakınsama hızını ve performansını artırmak amacıyla önerilen parametrelerdir. Bu parametrelere manuel olarak algoritma çalışmaya başlamadan önce değişik değerler verilerek ve ayrıca algoritmanın çalışması esnasında araştırmanın gelişime bağlı olarak SF parametresinin otomatik olarak değiştirildiği durumlar dikkate alınmıştır. ABC

algoritmasının bu parametreler için ürettiği sonuçları daha görsel hale getirmek amacıyla Tablo 4.8-4.9'da verilen değerler Şekil 4.5(a)-4.5(f)'de grafiksel olarak gösterilmiştir. Bu grafiklerden MR'nin 0 ve SF'nin 1 değerini aldığı yani ϕ parametresinin [-1,1] arasında değer aldığı temel ABC algoritmasının tek modlu ve çok modlu sınırlamasız basit test problemleri üzerinde diğer dikkate alınan durumlara göre daha iyi sonuçlar ürettiği gözlemlenmiştir. MR parametresi, sınırlamalı problemlerde ile değişik tipteki problemlerin dönme ve kayma gibi dönüşümlere tabi tutulmasıyla elde edilen yeni tip fonksiyonlara dönüştürüldüğü karma (hibrid) fonksiyonlarda daha önemlidir. ABC algoritmasının tek modlu fonksiyonlar için yakınsama hızlarına ait grafikler Şekil 4.6(a)'da, çok modlu fonksiyonlar için de Şekil 4.6(b)'de verilmektedir.

Tablo 4.7. Değişim oranı (MR) ve ölçekleme faktörünün (SF) analizinde kullanılan tek modlu ve çok modlu basit sınırlamasız test fonksiyonları

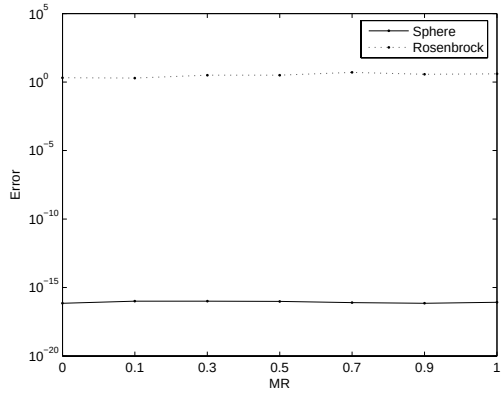
f	$f(x^*)$	Uzay Aralığı	Başlangıç Aralığı	Formül
Sphere	$f(\vec{0}) = 0$	$[-100, 100]^D$	$[-100, 50]^D$	$f(x) = \sum_{i=1}^{D} x_i^2$
Rosenbrock	$f(\vec{1}) = 0$	$[-2.048, 2.048]^D$	$[-2.048, 2.048]^D$	$f(x) = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$
Ackley	$f(\vec{0}) = 0$	$[-32.768, 32.768]^D$	$[-32.768, 16]^D$	$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + e$
Griewank	$f(\vec{0}) = 0$	$[-600, 600]^D$	$[-600, 200]^D$	$f(x) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
Weierstrass	$f(\vec{0}) = 0$	$[-0.5, 0.5]^D$	$[-0.5, 0.2]^D$	$f(x) = \sum_{i=1}^{D}\left(\sum_{k=0}^{k\max}[a^k\cos(2\pi b^k(x_i + 0.5))]\right) - D\sum_{k=0}^{k\max}[a^k\cos(2\pi b^k 0.5)],$ $a = 0.5, b = 3, k\max = 20$
Rastrigin	$f(\vec{0}) = 0$	$[-5.12, 5.12]^D$	$[-5.12, 2]^D$	$f(x) = \sum_{i=1}^{D}[x_i^2 - 10\cos(2\pi x_i) + 10]$
Süreksiz Rastrigin	$f(\vec{0}) = 0$	$[-5.12, 5.12]^D$	$[-5.12, 2]^D$	$f(x) = \sum_{i=1}^{D}(y_i^2 - 10\cos(2\pi y_i) + 10)$ $y_i = \begin{cases} x_i & \lvert x_i\rvert < \frac{1}{2} \\ \frac{round(2x_i)}{2} & \lvert x_i\rvert \geq \frac{1}{2} \end{cases}$
Schwefel	$f(\vec{4}20.96) = 0$	$[-500, 500]^D$	$[-500, 500]^D$	$f(x) = 418.9829xD - \sum_{i=1}^{D} -x_i\sin(\sqrt{\lvert x_i\rvert})$

Tablo 4.8. Farklı MR ve SF değerleri ile koşulan ABC algoritmasının ve PSO'nun türevlerinin ilk dört fonksiyon için sonuçları, MM:Çok modlu, UM:Tek modlu

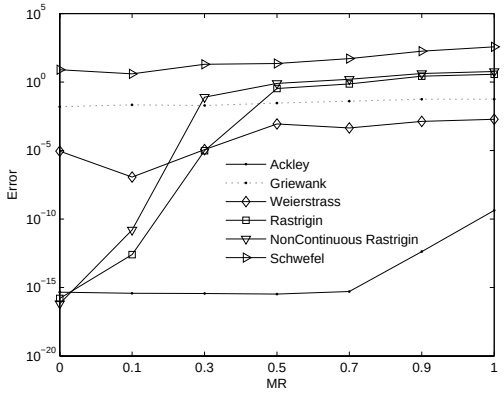
			UM	UM	MM	MM
			1 Sphere	Rosenbrock	Ackley	Griewank
PSO-w			7.96e-051±3.56e-050	3.08e+000±7.69e-001	1.58e-014±1.60e-014	9.69e-002±5.01e-002
PSO-cf			9.84e-105±4.21e-104	6.98e-001±1.46e+000	9.18e-001±1.01e+000	1.19e-001±7.11e-002
PSO-w-local			2.13e-035±6.17e-035	3.92e+000±1.19e+000	6.04e-015±1.67e-015	7.80e-002±3.79e-002
PSO-cf-local			1.37e-079±5.60e-079	8.60e-001±1.56e+000	5.78e-002±2.58e-001	2.80e-002/6.34e-002
UPSO			9.84e-118±3.56e-117	1.40e+000±1.88e+000	1.33e+000±1.48e+000	1.04e-001±7.10e-002
FDR			2.21e-090±9.88e-090	8.67e-001±1.63e+000	3.18e-014±6.40e-014	9.24e002±5.61e-002
FIPS			3.15e-030±4.56e-030	2.78e+000±2.26e-001	3.75e-015±2.13e-014	1.31e-001±9.32e-002
CPSO-H			4.98e-045±1.00e-044	1.53e+000±1.70e+000	1.49e-014±6.97e-015	4.07e-002±2.80e-002
CLPSO			5.15e-029±2.16e-028	2.46e+000±1.70e+000	4.32e-10±2.55e-014	**4.56e-003±4.81e-003**
ABC	MR, SF:1,Limit=200	0 (basic)	7.09E-017±4.11E-017	2.08E+000±2.44E+000	4.58E-016±1.76E-016	1.57E-002±9.06E-003
		0.1	1.00E-016±4.88E-017	1.96E+000±2.22E+000	3.79E-016±9.68E-017	2.17E-002±1.78E-002
		0.3	1.01E-016±5.29E-017	3.16E+000±2.35E+000	3.65E-016±1.84E-016	1.93E-002±1.30E-002
		0.5	9.63E-017±5.01E-017	3.20E+000±1.81E+000	3.32E-016±1.84E-016	2.94E-002±2.47E-002
		0.7	7.92E-017±4.87E-017	5.06E+000±1.69E+000	5.13E-016±6.56E-016	4.00E-002±3.52E-002
		0.9	7.04E-017±4.55E-017	3.66E+000±1.97E+000	4.21E-013±2.04E-012	5.65E-002±3.05E-002
		1	8.28E-017±4.95E-017	3.97E+000±2.24E+000	4.29E-010±2.31E-009	5.61E-002±3.26E-002
	SF, MR:0, Limit=200	0.7	1.05E-016±5.26E-017	2.77E+000±2.26E+000	3.41E-014±1.12E-013	2.00E-002±1.59E-002
		0.5	1.44E-016±3.72E-017	3.22E+000±2.05E+000	2.93E-008±1.53E-007	3.87E-002±2.64E-002
		0.3	1.77E-016±5.48E-017	3.79E+000±1.99E+000	4.59E-002±2.10E-001	7.15E-002±5.92E-002
		0.1	1.89E-010±6.60E-010	3.89E+000±1.49E+000	3.05E+000±4.29E+000	6.81E-001±8.39E-001
		ASF	3.00E-012±5.48E-012	**4.42E-001±8.67E-00**1	2.70E-006±1.46E-005	1.14E-001±1.25E-001
	Limit, MR:, SF	10	2.00E-001±3.24E-001	5.28E+000±1.49E+000	1.82E+000±5.62E-001	4.25E-001±1.43E-001
		200	9.68E-017±5.08E-017	1.61E+000±1.93E+000	4.03E-016±1.25E-016	1.52E-002±1.28E-002
		500	8.33E-017±4.99E-017	2.38E+000±2.59E+000	3.92E-016±1.07E-016	2.10E-002±1.32E-002
		1000	9.04E-017±4.81E-017	2.15E+000±2.46E+000	4.39E-016±2.22E-016	2.36E-002±1.98E-002
		3000	9.08E-017±4.99E-017	2.60E+000±2.61E+000	5.11E-016±1.78E-016	2.50E-002±1.46E-002
		5000	1.21E-016±6.66E-017	2.08E+000±2.54E+000	5.87E-001±1.36E+000	2.59E-002±2.01E-002

Tablo 4.9. Farklı MR ve SF değerleri ile koşulan ABC algoritmasının ve PSO'nun türevlerinin son dört fonksiyon için sonuçları, MM:Çok modlu, UM:Tek modlu

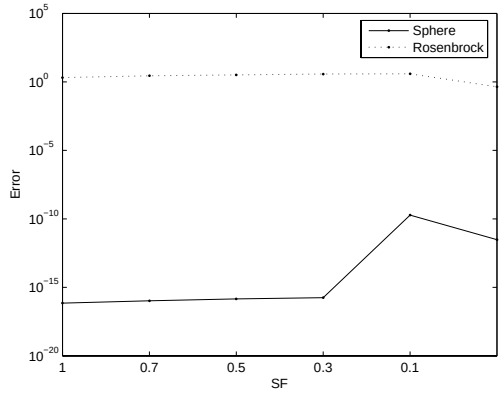
			MM	MM	MM	MM
			5 Weierstrass	6Rastrigin	7 NCRastrigin	8 schwefel
PSO-w			2.28e-003±7.04e-003	5.82e+000±2.96e+000	4.05e+000/±2.58e+000	3.20e+002±1.85e+002
PSO-cf			6.69e-001±7.17e-001	1.25e+001±5.17e+000	1.20e+001±4.99e+000	9.87e+002±2.76e+002
PSO-w-local			1.41e-006±6.31e-006	3.88e+000±2.30e+000	4.77e+000±2.84e+000	3.26e+002±1.32e+002
PSO-cf-local			7.85e-002±5.16e-002	9.05e+000±3.48e+000	5.95e+000±2.60e+000	8.78e+002±2.93e+002
UPSO			1.14e+000±1.17e+000	1.17e+001±6.11e+000	5.85e+000±3.15e+000	1.08e+003±2.68e+002
FDR			3.01e-003±7.20e-003	7.51e+000±3.05e+000	3.35e+000±2.01e+000	8.51e+002±2.76e+002
FIPS			2.02e-003±6.40e-003	2.12e+000±1.33e+000	4.35e+000±2.80e+000	7.10e+001±1.50e+002
CPSO-H			1.07e-015±1.67e-015	0±0	2.00e-001±4.10e-001	2.13e+002±1.41e+002
CLPSO			0±0	0±0	0±0	**0±0**
ABC	MR, SF:1,Limit=200	0 (basic)	9.01E-006±4.61E-005	1.61E-016±5.20E-016	6.64E-017±3.96E-017	7.91E+000±2.95E+001
		0.1	1.15E-007±6.17E-007	2.54E-013±1.37E-012	1.58E-011±7.62E-011	3.96E+000±2.13E+001
		0.3	1.17E-005±4.90E-005	9.61E-006±5.17E-005	7.84E-002±2.54E-001	1.97E+001±4.41E+001
		0.5	8.80E-004±2.94E-003	3.38E-001±6.44E-001	8.00E-001±7.02E-001	2.25E+001±4.45E+001
		0.7	4.45E-004±1.69E-003	7.31E-001±7.23E-001	1.59E+000±9.59E-001	5.13E+001±8.31E+001
		0.9	1.34E-003±5.62E-003	2.68E+000±1.95E+000	4.21E+000±1.37E+000	1.78E+002±1.25E+002
		1	1.92E-003±6.63E-003	3.71E+000±1.58E+000	5.89E+000±1.69E+000	3.69E+002±1.52E+002
	SF, MR:0, Limit=200	0.7	1.18E-016±6.38E-016	1.29E+000±8.95E-001	9.00E-001±7.00E-001	3.20E+002±1.36E+002
		0.5	6.33E-001±7.00E-001	1.73E+000±1.18E+000	1.37E+000±6.57E-001	3.59E+002±1.16E+002
		0.3	2.89E+000±1.26E+000	1.17E+001±4.71E+000	3.17E+000±1.42E+000	9.49E+002±2.19E+002
		0.1	6.08E+000±1.46E+000	3.94E+001±1.41E+001	2.77E+001±8.45E+000	1.40E+003±2.61E+002
		ASF	3.24E-008±3.06E-008	1.94E-001±3.85E-001	6.80E-001±7.80E-001	1.09E+002±8.49E+001
		10	3.10E-001±1.54E-001	6.97E+000±1.69E+000	7.33E+000±1.65E+000	5.16E+002±8.75E+001
	Limit, MR:1, SF:1	200	9.41E-007±5.07E-006	1.14E-007±6.16E-007	1.12E-005±6.04E-005	1.65E+001±4.01E+001
		500	0.00E+000±0.00E+000	3.33E-002±1.79E-001	1.17E-006±5.19E-006	8.39E+000±2.95E+001
		1000	1.88E-006±1.01E-005	3.32E-002±1.79E-001	7.22E-004±3.89E-003	1.30E+001±3.55E+001
		3000	2.37E-016±1.28E-015	6.63E-002±2.48E-001	1.67E-001±4.53E-001	4.08E+001±6.42E+001
		5000	1.15E-001±3.79E-001	2.02E+000±3.35E+000	1.17E+000±3.01E+000	1.53E+002±1.20E+002



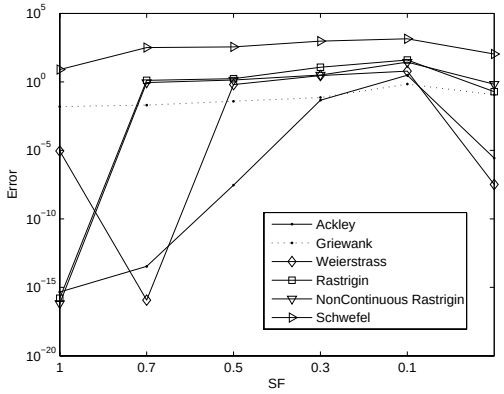
(a) MR'nin tek modlu fonksiyonlar üzerindeki Etkisi



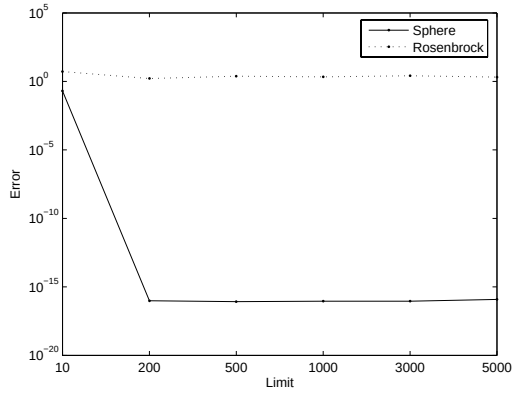
(b) MR'nin çok modlu fonksiyonlar üzerindeki Etkisi



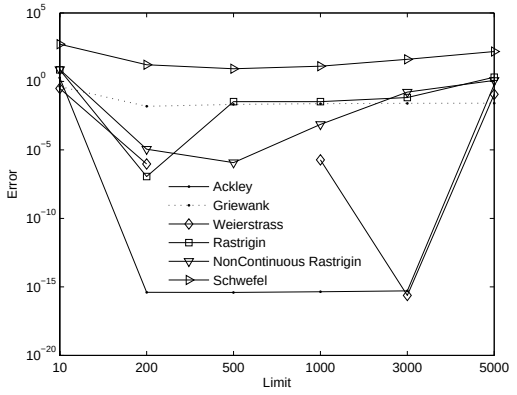
(c) SF'nin tek modlu fonksiyonlar üzerindeki Etkisi



(d) SF'nin çok modlu fonksiyonlar üzerindeki Etkisi

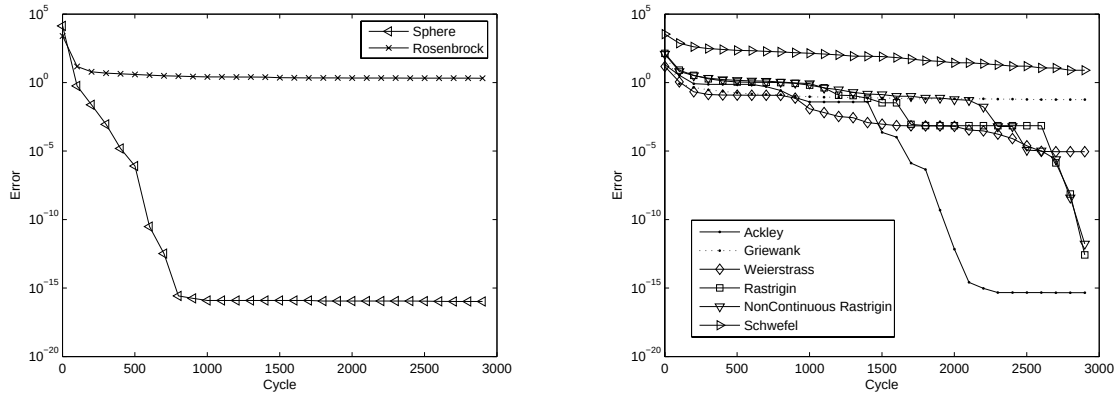


(e) Limit'in tek modlu fonksiyonlar üzerindeki Etkisi



(f) Limit'in çok modlu fonksiyonlar üzerindeki Etkisi

Şekil 4.5. Farklı kontrol parametreleri ile bulunan en iyi sonuçların ortalaması



(a) Tek modlu fonksiyonlarda yakınsama grafiği (b) Çok modlu fonksiyonlarda yakınsama grafiği

Şekil 4.6. ABC algoritmasının yakınsama grafikleri

4.1.1.3. Başlangıç Popülasyonunun Oluşturulma Aralığının Etkisi

Algoritmaların başlatılması algoritmanın performansını etkileyen önemli bir adımdır. Birçok algoritmada başlatılma süreci hesaplama karmaşıklığını indirgemek adına oldukça basit tutulmaktadır. Başlangıç popülasyonunun probleme özgü sezgisel ön işlemlerle yüksek uygunluklu bireylerden oluşması sağlanabilir. Ancak bu işlem hesaplama karmaşıklığı ve çözüm kalitesi arasında bir ödünleşimdir (trade-off). Başlangıç popülasyonunun oluşturulmasında kullanılan parametre aralıkları bireylerin kalite değerlerini etkilediğinden algoritma performansını da etkilemektedir. Küresel optimum araştırma uzayının merkezinde yer alıyorsa ve simetrik bir başlatma uygulanmışsa, algoritmanın optimumu bulması nispeten kolay olabilir. Bir algoritma bölgesel minimalara takılabiliyorsa, araştırma uzayının yeni başka bölgelerini keşfetme özelliğine sahip değilse, optimum çözümü içermeyen bir bölgede başlangıç popülasyonu oluşturulduğunda başarısız olacaktır. Böyle bir algoritmanın başlatma şartlarına karşı gürbüz olmadığı söylenir. Önceki bölümde araştırmayı zorlaştırmak adına ABC algoritması uzayın sol çeyrek bölümünde başlatıldığında farklı limit değerleri için kaşif arının etkisi dokuz fonksiyon üzerinde incelenmişti. Bu bölümde çok daha geniş bir set üzerinde, ABC algoritması ile birlikte DE ve PSO algoritmalarının da başlangıç koşullarına olan bağlılığı ANOVA istatistiksel testi kullanılarak analiz edilecektir. Bu analizin gerçekleştirilmesinde

Tablo 4.11'deki çok boyutlu, tek ve çok modlu 50 fonksiyon kullanılmıştır. Tabloda her fonksiyona ait karakteristik özelliklerde verilmiştir. C isimli kolon fonksiyonun karakteristiğini özetlemektedir. U tek modlu ve M çok modlu olduğunu, S ayrıştırılabilir ve N ayrıştırılamayan türden bir fonksiyon olduğunu göstermektedir. D kolonu ise problemin boyutunu göstermektedir. Boyutun artması problemi zorlaştıran bir faktördür. Aralık kolonunda da fonksiyona ait araştırma uzayının alt ve üst sınırları verilmektedir. Tabloda, 33 çok modlu, 17 tek modlu, 14 tane ayrıştırılabilir, 36 tane ayrıştırılamayan olmak üzere toplam 50 fonksiyon bulunmaktadır. Bu set çok sayıda test fonksiyon türünü içerecek kadar geniş bir settir. Tablolarda kullanılan fonksiyonların formülasyonlarında geçen bazı sabitler bulunmaktadır. Langerman fonksiyonunun a ve c parametrelerine Ekler bölümündeki Tablo EK-2.1'den, Fletcher-Powell fonksiyonunun a, b ve α değerlerine sırasıyla Tablo EK-2.2, Tablo EK-2.3 and Tablo EK-2.4'den, Foxholes, Kowalik, Shekel, Hartman3, Hartman6 fonksiyonlarının parametrelerine de sırasıyla Tablo EK-2.5, Tablo EK-2.6, Tablo EK-2.7, Tablo EK-2.8, Tablo EK-2.9'dan erişilebilir.

Algoritma araştırma uzayının tümünde, sağ yarısında, sol yarısında, sağ çeyreğinde ve sol çeyreğinde başlatılarak bu fonksiyonlar için sonuçlar elde edilmiştir. Bu 5 farklı durum arasında istatistiksel olarak fark olup olmadığını görmek amacıyla ANOVA testi uygulanmıştır. Algoritmalara bu testin uygulanmasından sonra DE, PSO ve ABC algoritmalarının başlatma aralığına olan bağlılıkları yorumlanmıştır. Tüm algoritmalar 1000000 fonksiyon değerlendirmesi boyunca çalıştırılmışlar ve testler 30 kez tekrarlanmıştır. DE, PSO ve ABC algoritmalarının kontrol parametrelerine atanan değerler Tablo 4.10'da sunulmaktadır. PSO, DE ve ABC algoritmalarının 30 koşmasına ait ortalama değerler Tablo 4.12-4.14'de verilmektedir. ANOVA testi için her fonksiyonun 5 durumuyla ilgili F değeri hesaplanarak bu değer α güvenlik seviyesi ile serbestlik derecesine bağlı F kümülatif dağılım fonksiyonunun F kritik değeriyle kıyaslanmıştır. Null hipotez, tüm grup örneklerinin aynı popülasyondan geldiği ve grup ortalamaları arasında α güven seviyesinde anlamlı bir fark olmadığıdır. Eğer hesaplanan F değeri, F kritik değerinden küçükse null hipotez doğru kabul edilir, aksi takdirde örnekler

arasında farklılık olduğu ve başlatma koşullarına bağlılık olduğu ifade edilir. PSO, DE ve ABC algoritmaları için hesaplanan F kritik değerleri ve farklılık olup olmadığı ile ilgili sonuçlar Table 4.15'de verilmektedir. Tablodaki sonuçlardan PSO algoritmasının 50 fonksiyonun 38'inde, DE algoritmasının 26'sında ve ABC algoritmasının 22'sinde başlatma koşullarına bağlılık olduğu görülmüştür. Fonksiyon seti yeterince büyük olduğu için başlatma koşullarına PSO algoritmasının en bağımlı, ABC algoritmasının da en az bağımlı olduğu söylenebilir. ABC algoritması çok dar bir bölgede oluşturulan popülasyonla başaltılmış olsa dahi keşif yeteneğinin etkili olmasından olayı tüm uzayı verimli şekilde araştırabilmektedir.

Tablo 4.10. Popülasyon başlangıç aralığının etkisi analizinde kullanılan DE, PSO ve ABC algoritmalarının kontrol parametrelerinin değerleri, SN: Koloni büyüklüğü, PS: Popülasyon büyüklüğü, SS: Sürü büyüklüğü, MCN: Maksimum Çevrim Sayısı, MGN: Maksimum jenerasyon sayısı, CR: Çaprazlama oranı, F: Ölçekleme Faktörü, ϕ_1: Bilişsel bileşen, ϕ_2: Sosyal bileşen, ω: Atalet(inertia)

ABC		DE		PSO	
SN	100	PS	100	SS	100
MCN	10000	MGN	10000	MGN	10000
limit	$SNxD * 0.1$	CR	0.9	ϕ_1	1.8
		F	0.5	ϕ_2	1.8
				ω	0.6

Tablo 4.11. Çalışmalarda kullanılan kullanılan test fonksiyonları, D: Problemin Boyutu, C: Karakteristiği, U: Tek modlu, M: Çok Modlu, S: Ayrıştırılabilir, N: Ayrıştırılamaz

No	**Aralık**	**D**	**C**	**Fonksiyon**	**Formül**
1	[-5.12,5.12]	5	US	Stepint	$f(x) = 25 + \sum_{i=1}^{5} \lfloor x_i \rfloor$
2	[-100,100]	30	US	Step	$f(x) = \sum_{i=1}^{n} (\lfloor x_i + 0.5 \rfloor)^2$
3	[-100,100]	30	US	Sphere	$f(x) = \sum_{i=1}^{n} x_i^2$
4	[-10,10]	30	US	SumSquares	$f(x) = \sum_{i=1}^{n} i x_i^2$
5	[-1.28,1.28]	30	US	Quartic	$f(x) = \sum_{i=1}^{n} i x_i^4 + random[0,1)$
6	[-4.5,4.5]	5	UN	Beale	$f(x) = (1.5 - x_1 + x_1 x_2)^2$ $+(2.25 - x_1 + x_1 x_2^2)^2 +$ $(2.625 - x_1 + x_1 x_2^3))^2$
7	[-100,100]	2	UN	Easom	$f(x) = -cos(x_1)cos(x_2)exp(-(x_1 - \pi)^2 - (x_2 - \pi)^2)$
8	[-10,10]	2	UN	Matyas	$f(x) = 0.26(x_1^2 + x_2^2) - 0.48 x_1 x_2$
9	[-10,10]	4	UN	Colville	$f(x) = 100(x_1^2 - x_2)^2 + (x_1 - 1)^2 + (x_3 - 1)^2 +$ $90(x_3^2 - x_4)^2 + 10.1((x_2 - 1)^2 + (x_4 - 1)^2)$ $+19.8(x_2 - 1)(x_4 - 1)$
10	[-D^2,D^2]	6	UN	Trid6	$f(x) = \sum_{i=1}^{n} (x_i - 1)^2 - \sum_{i=2}^{n} x_i x_{i-1}$
11	[-D^2,D^2]	10	UN	Trid10	$f(x) = \sum_{i=1}^{n} (x_i - 1)^2 - \sum_{i=2}^{n} x_i x_{i-1}$
12	[-5,10]	10	UN	Zakharov	$f(x) = \sum_{i=1}^{n} x_i^2 + \left(\sum_{i=1}^{n} 0.5 i x_i\right)^2 + \left(\sum_{i=1}^{n} 0.5 i x_i\right)^4$

Tablo 4.11. (devamı)

No	**Aralık**	**D**	**C**	**Fonksiyon**	**Formül**
13	[-4,5]	24	UN	Powell	$f(x) = \sum_{i=1}^{n/k} (x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4$
14	[-10,10]	30	UN	Schwefel 2.22	$f(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$
15	[-100,100]	30	UN	Schwefel 1.2	$f(x) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$
16	[-30,30]	30	UN	Rosenbrock	$f(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$
17	[-10,10]	30	UN	Dixon-Price	$f(x) = (x_1 - 1)^2 + \sum_{i=2}^{n} i(2x_i^2 - x_{i-1})^2$
18	[-65.536,65.536]	2	MS	Foxholes	$f(x) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})6} \right]^{-1}$
19	[-5,10]x[0,15]	2	MS	Branin	$f(x) = \left(x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right) cos x_1 + 10$
20	[-100,100]	2	MS	Bohachevsky1	$f(x) = x_1^2 + 2x_2^2 - 0.3cos(3\pi x_1) - 0.4cos(4\pi x_2) + 0.7$
21	[-10,10]	2	MS	Booth	$f(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$
22	[-5.12,5.12]	30	MS	Rastrigin	$f(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$
23	[-500,500]	30	MS	Schwefel	$f(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert})$
24	[0,π]	2	MS	Michalewicz2	$f(x) = -\sum_{i=1}^{n} sin(x_i)(sin(ix_i^2/\pi))^{2m}$ $m = 10$

Tablo 4.11. (devamı)

No	**Aralık**	**D**	**C**	**Fonksiyon**	**Formül**
25	[0,π]	5	MS	Michalewicz5	$f(x) = -\sum_{i=1}^{n} sin(x_i)(sin(ix_i^2/\pi))^{2m}$ $m = 10$
26	[0,π]	10	MS	Michalewicz10	$f(x) = -\sum_{i=1}^{n} sin(x_i)(sin(ix_i^2/\pi))^{2m}$ $m = 10$
27	[-100,100]	2	MN	Schaffer	$f(x) = 0.5 + \frac{\sin^2(\sqrt{x_1^2+x_2^2})-0.5}{(1+0.001(x_1^2+x_2^2))^2}$
28	[-5,5]	2	MN	Six Hump Camel Back	$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x1x2 - 4x_2^2 + 4x_2^4$
29	[-100,100]	2	MN	Bohachevsky2	$f(x) = x_1^2 + 2x_2^2 - 0.3cos(3\pi x_1)(4\pi x_2) + 0.3$
30	[-100,100]	2	MN	Bohachevsky3	$f(x) = x_1^2 + 2x_2^2 - 0.3cos(3\pi x_1 + 4\pi x_2) + 0.3$
31	[-10,10]	2	MN	Shubert	$f(x) = \left(\sum_{i=1}^{5} i\cos((i+1)x_1 + i)\right)\left(\sum_{i=1}^{5} i\cos((i+1)x_2 + i)\right)$
32	[-2,2]	2	MN	GoldStein-Price	$f(x) = \left[1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right]$ $\left[30 + (2x_1 - 3x_2)^2 (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right]$
33	[-5,5]	4	MN	Kowalik	$f(x) = \sum_{i=1}^{11} (a_i - \frac{x_1(b_i^2+b_ix_2)}{b_i^2+b_ix_3+x_4})^2$
34	[0,10]	4	MN	Shekel5	$f(x) = -\sum_{i=1}^{5} \left[(x-a_i)(x-a_i)^T + c_i\right]^{-1}$
35	[0,10]	4	MN	Shekel7	$f(x) = -\sum_{i=1}^{7} \left[(x-a_i)(x-a_i)^T + c_i\right]^{-1}$
36	[0,10]	4	MN	Shekel10	$f(x) = -\sum_{i=1}^{10} \left[(x-a_i)(x-a_i)^T + c_i\right]^{-1}$

Tablo 4.11. (devamı)

No	**Aralık**	**D**	**C**	**Fonksiyon**	**Formül**
37	[-D,D]	4	MN	Perm	$f(x) = \sum_{k=1}^{n} \left[\sum_{i=1}^{n} (i^k + \beta) \left((x_i/i)^k - 1 \right) \right]^2$
38	[0,D]	4	MN	PowerSum	$f(x) = \sum_{k=1}^{n} \left[\left(\sum_{i=1}^{n} x_i^k \right) - b_k \right]^2$
39	[0,1]	3	MN	Hartman3	$f(x) = -\sum_{i=1}^{4} c_i \exp \left[-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2 \right]$
40	[0,1]	6	MN	Hartman6	$f(x) = -\sum_{i=1}^{4} c_i \exp \left[-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2 \right]$
41	[-600,600]	30	MN	Griewank	$f(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
42	[-32,32]	30	MN	Ackley	$f(x) = -20 \exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i)\right) + 20 + e$
43	[-50,50]	30	MN	Penalized	$f(x) = \frac{\pi}{n} \left\{ \begin{array}{l} 10\sin^2(\pi y_1) \\ + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10\sin^2(\pi y_{i+1})] \\ + (y_n - 1)^2 \end{array} \right\}$ $+ \sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{u}(\mathrm{x_i}, 10, 100, 4)$ $y_i = 1 + \frac{1}{4}(x_i + 1)$ $u(x_i, a, k, m) = \left\{ \begin{array}{rcl} k(x_i - a)^m & , & x_i > a \\ 0 & , & -a \le x_i \le a \\ k(-x_i - a)^m & , & x_i < -a \end{array} \right.$

Tablo 4.11. (devamı)

No	**Aralık**	**D**	**C**	**Fonksiyon**	**Formül**
44	[-50,50]	30	MN	Penalized2	$f(x) = 0.1 \left\{ \begin{array}{l} \sin^2(\pi x_1) + \\ \sum_{i=1}^{n-1} (x_i - 1)^2[1 + \sin^2(3\pi x_{i+1})] + \\ (x_n - 1)^2[1 + \sin^2(2\pi x_n)] \end{array} \right\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$
45	[0,10]	2	MN	Langerman2	$f(x) = -\sum_{i=1}^{m} c_i \left(\exp\left(-\frac{1}{\pi} \sum_{j=1}^{n} (x_j - a_{ij})^2 \right) \cos\left(\pi \sum_{j=1}^{n} (x_j - a_{ij})^2 \right) \right)$
46	[0,10]	5	MN	Langerman5	$f(x) = -\sum_{i=1}^{m} c_i \left(\exp\left(-\frac{1}{\pi} \sum_{j=1}^{n} (x_j - a_{ij})^2 \right) \cos\left(\pi \sum_{j=1}^{n} (x_j - a_{ij})^2 \right) \right)$
47	[0,10]	10	MN	Langerman10	$f(x) = -\sum_{i=1}^{m} c_i \left(\exp\left(-\frac{1}{\pi} \sum_{j=1}^{n} (x_j - a_{ij})^2 \right) \cos\left(\pi \sum_{j=1}^{n} (x_j - a_{ij})^2 \right) \right)$
48	[-π,π]	2	MN	FletcherPowell2	$f(x) = \sum_{i=1}^{n} (A_i - B_i)^2$ $A_i = \sum_{j=1}^{n} (a_{ij} \sin \alpha_j + b_{ij} \cos \alpha_j)$ $B_i = \sum_{j=1}^{n} (a_{ij} \sin x_j + b_{ij} \cos x_j)$
49	[-π,π]	5	MN	FletcherPowell5	$f(x) = \sum_{i=1}^{n} (A_i - B_i)^2$ $A_i = \sum_{j=1}^{n} (a_{ij} \sin \alpha_j + b_{ij} \cos \alpha_j)$ $B_i = \sum_{j=1}^{n} (a_{ij} \sin x_j + b_{ij} \cos x_j)$

Tablo 4.11. (devamı)

No	**Aralık**	**D**	**C**	**Fonksiyon**	**Formül**
50	[-π,π]	10	MN	FletcherPowell10	$f(x) = \sum_{i=1}^{n} (A_i - B_i)^2$ $A_i = \sum_{j=1}^{n} (a_{ij} \sin \alpha_j + b_{ij} \cos \alpha_j)$ $B_i = \sum_{j=1}^{n} (a_{ij} \sin x_j + b_{ij} \cos x_j)$

Tablo 4.12. PSO algoritmasının farklı başlatma aralıkları ile elde edilen 30 koşmasının ortalaması.

	Tüm	Sağ yarım	Sol yarım	Sağ çeyrek	Sol Çeyrek
Foxholes	0.998003931	0.998003931	0.9980039	0.998003931	0.998003931
Schaffer	2.84E-17	2.56E-17	3.12E-17	3.80E-17	2.82E-17
sphere	1.56E-16	8333.333333	6666.6667	107333.3333	102000
Griewank	0.017391178	48.22452098	96.283032	1002.997296	906.9885194
Rastrigin	43.97713689	627.565342	506.53184	1148.112322	1030.483019
Rosenbrock	15.08861705	562.3023433	15237.084	275991207	401646846.8
Ackley	0.164622363	8.866574057	7.3595312	19.84563386	19.83473317
Schwefel	-20877.6481	-27048.77671	-24495.91	-38813.06805	-24792.20846
Stepint	0	1.2	0	1.466666667	0.066666667
Schwefel 2.22	1.91E-16	4.88E-17	4.98E-17	2.333333333	1.666666667
Schwefel 1.2	1.33E-16	4.21E-17	1000	12666.66667	16666.66667
Step	0	9343.533333	12418.933	111334.1333	103256.9
quartic	0.001156594	8.53E-05	7.51E-05	6.79E-05	7.16E-05
penalized	0.020733804	0.110611269	0.0851934	315733340.8	349866673.4
penalized2	0.007675353	4.78E-17	4.33E-17	4.30E-17	4.35E-17
Kowalik	0.000490623	0.000307486	0.0012355	0.003882033	0.027602996
SixHumpCamelBack	-1.03162845	-1.031628453	-1.0316285	-1.004422966	-1.031628453
Branin	0.397887358	2.551557506	0.3978874	2.705391088	0.397887358
GoldSteinPrice	3	84	3	184.8	3
Shekel5	-2.08700789	-2.034819972	-3.525066	-2.62321198	-3.03936009
Shekel7	-1.98987126	-2.122727249	-3.813889	-2.801989496	-2.687066125
Shekel10	-1.87967526	-2.565851161	-3.434902	-2.553802154	-2.861266308
Hartman6	-1.85912978	-0.19409054	-2.605726	-0.007608101	-2.285369651
Hartman3	-3.63335232	-3.663461474	-3.753365	-2.366418968	-0.959439339
Beale	3.57E-17	1.11E-16	2.96E-17	2.66E-17	2.94E-17
Bohachecsky1	3.58E-17	7.29E-16	2.70E-17	2.70E-17	2.66E-17
Bohachecsky2	5.44E-17	2.04E-17	2.74E-17	2.18E-17	2.87E-17
Bohachecsky3	4.12E-17	3.06E-17	2.68E-17	2.24E-17	2.48E-17
Booth	1.91E-17	2.77E-17	3.33E-17	2.98E-17	2.66E-17
DixonPrice	0.666666667	94678.85227	110451.25	6752582.611	6557624.868
Easom	-1	-0.933333333	-0.6666667	0	0
Matyas	3.66E-17	2.92E-17	3.18E-17	5.63E-17	3.40E-17
Michalewics2	-1.57286915	-1.477315463	-1.2921075	-0.686158856	-1.023136335
Michalewics5	-2.49087284	-2.286173657	-2.706265	-1.142420675	-2.04960771
Michalewics10	-4.00718033	-3.43863307	-4.300333	-1.914121676	-3.80670397
Perm	0.036051575	0.013658593	3332.3562	0.072891093	12763.36245
Powell	0.000110044	82.38158386	3.99E-05	753.176221	40.58540954
PowerSum	11.39044786	603.4002329	27.306269	87111.11172	17.72790624
Shubert	-186.730909	-186.7309088	-186.73091	-186.7309088	-186.7309088
SumSquares	1.48E-16	4.74E-17	4.78E-17	56.66666667	13.33333333
Trid6	-50	-50	-50	-50	-50
Trid10	-210	-210	-210	-150.1111111	-102.7238095
Zakharov	4.57E-17	37.26360536	4.80E-17	57.6216524	4.41E-17
Colville	4.76E-16	2.54E-16	1.72E-16	238.8379228	2.01E-16
Langerman2	-0.67926802	-0.014219775	-0.472572	-2.44E-05	-0.522885457
Langerman5	-0.50485788	-0.307150893	-0.1740855	-0.002704026	-0.158629751
Langerman10	-0.0025656	-1.59E-07	-0.0129824	0	-9.80E-05
FletcherPowell2	2.95E-17	2904.191554	1023.2846	2904.191554	1023.284585
FletcherPowell5	1457.88344	6844.973812	381.28739	21011.30239	114.9322707
FletcherPowell10	1364.455554	21332.12799	28793.27	91230.69963	40633.5406

Tablo 4.13. DE algoritmasının farklı başlatma aralıkları ile elde edilen 30 koşmasının ortalaması.

	Tüm	Sağ yarım	Sol yarım	Sağ çeyrek	Sol Çeyrek
Foxholes	0.99800393	0.998003931	0.99800393	0.99800393	0.99800393
Schaffer	2.30E-17	2.29E-17	2.09E-17	2.36E-17	1.83E-17
sphere	4.61E-17	1.31E-16	1.42E-16	1.20E-16	1.22E-16
Griewank	0.00147921	0.000328822	0.00139545	2.37E-17	1.64E-03
Rastrigin	11.7167285	104.0600376	121.971847	1.38E+02	1.08E+02
Rosenbrock	18.2039377	140.1742555	91.8800269	124.379157	94.5856227
Ackley	1.48E-16	2.22E-16	2.17E-16	1.89E+01	1.99E+01
Schwefel	-31384.158	-39747.9039	-28743.561	-41584.423	-27091.063
Stepint	0	0	0	0	0
Schwefel 2.22	1.53E-16	4.73E-17	4.62E-17	4.71E-17	4.77E-17
Schwefel 1.2	4.63E-17	4.40E-17	3.90E-17	3.67E-17	4.00E-17
Step	0	0	0	0	0
quartic	0.00136331	0.000138626	0.00015434	0.0001396	0.00015309
penalized	4.53E-17	0.025222763	0.01977518	3.17E-02	3.84E+02
penalized2	0.00219747	3.73E-17	4.16E-17	4.22E-17	3.76E-17
Kowalik	0.00042659	0.000307486	0.00118998	0.0003075	0.00122317
SixHumpCamelBack	-1.0316285	-1.03162845	-1.0316285	-1.004423	-1.0316285
Branin	0.39788736	0.397887358	0.39788736	2.53159802	0.39788736
GoldSteinPrice	3	3	3	73.2	3
Shekel5	-10.1532	-8.97429992	-10.1532	-5.1007721	-5.0551977
Shekel7	-10.402941	-10.0513327	-10.402941	-5.1288228	-5.0876718
Shekel10	-10.53641	-10.5364098	-10.53641	-5.1756467	-5.1284808
Hartman6	-3.2268807	-3.21102826	-3.3219952	-3.3101059	-3.3219952
Hartman3	-3.8627821	-3.86278215	-3.8627821	-3.8627821	-1.0008169
Beale	3.02E-17	1.99E-17	2.05E-17	2.35E-17	2.15E-17
Bohachecsky1	1.60E-17	2.28E-17	1.61E-17	2.12E-17	2.36E-17
Bohachecsky2	1.91E-17	2.08E-17	2.48E-17	2.18E-17	1.80E-17
Bohachecsky3	2.98E-17	2.19E-17	1.93E-17	2.21E-17	2.16E-17
Booth	2.17E-17	2.06E-17	2.37E-17	2.43E-17	2.10E-17
DixonPrice	0.66666667	1.835609093	1.05690798	3.28267747	0.66666667
Easom	-1	-1	-1	-1	-1
Matyas	3.61E-17	1.88E-17	2.40E-17	2.64E-17	2.37E-17
Michalewics2	-1.8013034	-1.80130341	-1.8013034	-1.2336343	-1.8013034
Michalewics5	-4.6834819	-4.68765818	-4.6743055	-4.4147834	-4.6039917
Michalewics10	-9.5911509	-9.64803274	-9.6350128	-9.526387	-9.595123
Perm	0.02400688	0.007682311	0.02119421	0.0007849	508.553773
Powell	2.17E-07	4.85E-15	1.10E-13	1.24E-14	1.38E-14
PowerSum	0.00014252	4.14E-11	1.24E-10	2.80E-10	1.74E-10
Shubert	-186.73091	-186.730909	-186.73091	-186.73091	-186.73091
SumSquares	4.39E-17	3.74E-17	4.10E-17	3.97E-17	3.93E-17
Trid6	-50	-50	-50	-50	-50
Trid10	-210	-210	-210	-210	-210
Zakharov	3.39E-17	4.15E-17	4.05E-17	3.78E-17	3.98E-17
Colville	0.04091221	3.14E-17	3.06E-17	3.14E-17	7.65E-04
Langerman2	-1.0809384	-1.08093844	-1.0809384	-1.0719177	-1.0809384
Langerman5	-1.4999992	-1.49999922	-1.362927	-1.4999992	-1.1314317
Langerman10	-1.0528	-1.43866667	-0.8754333	-1.2293667	-0.8576
FletcherPowell2	1.87E-17	96.80638512	579.861265	2.90E+03	1.02E+03
FletcherPowell5	5.98878303	2043.947853	93.5873849	5544.48911	107.4096
FletcherPowell10	781.550281	308.3056792	654.012945	339.36658	7814.54157

Tablo 4.14. ABC algoritmasının farklı başlatma aralıkları ile elde edilen 30 koşmasının ortalaması.

	Tüm	Sağ yarım	Sol yarım	Sağ çeyrek	Sol Çeyrek
Foxholes	0.998003931	0.998003931	0.998003931	0.9980039	0.99800393
Schaffer	3.34E-14	4.73E-09	2.65E-09	3.19E-09	4.03E-09
sphere	2.59E-16	1.14E-15	1.13E-15	1.13E-15	1.12E-15
Griewank	4.90E-18	2.51E-16	4.88E-17	4.74E-17	4.81E-17
Rastrigin	2.32E-16	1.13E-15	1.14E-15	1.08E-15	1.10E-15
Rosenbrock	0.08877074	1.811351577	65.93643288	9.4027884	47.5745853
Ackley	1.02E-15	4.37E-15	4.46E-15	8.08E-15	7.90E-15
Schwefel	-41898.2887	-41898.2887	-40085.76299	-41898.289	-38534.063
Stepint	0	0	0	0	0
Schwefel 2.22	5.96E-16	1.51E-16	1.54E-16	1.50E-16	1.55E-16
Schwefel 1.2	1.98E-16	4.66E-17	4.81E-17	4.64E-17	4.90E-17
Step	0	0	0	0	0
quartic	0.030016551	0.00532695	0.006487761	0.0058434	0.0064553
penalized	2.34E-16	1.11E-15	1.10E-15	1.12E-15	1.12E-15
penalized2	2.48E-16	4.83E-17	4.84E-17	4.60E-17	4.84E-17
Kowalik	0.000427442	0.000451513	0.000452229	0.0004292	0.00044757
SixHumpCamelBack	-1.03162845	-1.03162845	-1.031628453	-1.0316285	-1.0316285
Branin	0.397887358	0.397887358	0.397887358	0.3978874	0.39788736
GoldSteinPrice	3	3.000004148	3.000002959	3.0000041	3.00000832
Shekel5	-10.1531997	-10.1316739	-10.15319968	-10.13307	-10.122343
Shekel7	-10.4029406	-10.3810774	-10.40292996	-10.383404	-10.363464
Shekel10	-10.5364098	-10.5117408	-10.53640852	-10.52424	-10.512935
Hartman6	-3.32199517	-3.32199517	-3.321995172	-3.3219952	-3.3219952
Hartman3	-3.86278215	-3.86278215	-3.862675163	-3.8627821	-3.8626565
Beale	2.84E-15	7.30E-07	8.19E-07	8.53E-07	6.94E-07
Bohachecsky1	2.58E-17	2.96E-17	3.21E-17	3.23E-17	2.57E-17
Bohachecsky2	3.35E-17	2.90E-17	2.71E-17	3.16E-17	3.25E-17
Bohachecsky3	5.52E-16	4.93E-07	4.31E-07	4.10E-07	4.24E-07
Booth	2.90E-17	1.70E-10	3.16E-10	1.60E-10	1.69E-10
DixonPrice	1.50E-15	6.43E-06	1.452068394	1.41E-05	2.03959156
Easom	-0.99999982	-1	-1	-1	-1
Matyas	4.48E-16	4.39E-07	8.04E-07	3.31E-07	4.54E-07
Michalewics2	-1.80130341	-1.80130341	-1.80130341	-1.8013034	-1.8013034
Michalewics5	-4.68765818	-4.68765818	-4.687657788	-4.6876582	-4.687657
Michalewics10	-9.66015172	-9.66015172	-9.660144584	-9.6601517	-9.6601505
Perm	0.041105228	0.01909997	0.020539085	0.0210687	0.0195922
Powell	0.003134448	0.00136249	1.40E-03	1.39E-03	1.34E-03
PowerSum	0.002946808	0.001439968	1.35E-03	1.51E-03	1.77E-03
Shubert	-186.730909	-186.730908	-186.7309079	-186.73091	-186.73091
SumSquares	2.44E-16	4.75E-17	4.64E-17	4.60E-17	4.59E-17
Trid6	-50	-50	-50	-50	-50
Trid10	-210	-210	-210	-210	-210
Zakharov	0.000247647	0.008280686	6.69E-03	7.88E-03	6.01E-03
Colville	0.09296742	0.08852807	7.98E-02	0.0669047	9.78E-02
Langerman2	-1.08093844	-1.08093844	-1.080938442	-1.0809384	-1.0809384
Langerman5	-0.93815026	-0.9379182	-0.937855769	-0.9378946	-0.9379186
Langerman10	-0.44609247	-0.42189814	-0.415158778	-0.4557773	-0.4084353
FletcherPowell2	1.52E-17	3.25E-17	3.22E-17	3.53E-17	2.91E-17
FletcherPowell5	0.173549512	0.291348002	0.242649021	0.2896778	0.27988991
FletcherPowell10	8.233440105	15.4378727	12.25554196	18.888982	16.628682

Tablo 4.15. PSO, DE ve ABC algoritmalarının ANOVA testi ile elde edilen anlamlılık sonuçları, ♯ anlamlı farklılık olan fonksiyonların sayısını göstermektedir.

	PSO			DE			ABC		
	F Level	F Stat.	Farklılık	F Level	F Stat.	Farklılık	F Level	F Stat.	Farklılık
Foxholes	2.434065136	2.08E-15	YOK	2.434065136	2.08E-15	YOK	2.434065136	2.08E-15	YOK
Schaffer	2.434065136	3.05E-18	YOK	2.434065136	6.39E-19	YOK	2.434065136	4.45E-01	YOK
sphere	2.434065136	3.15E+02	VAR	2.434065136	1.95E-16	YOK	2.434065136	2.05E-14	YOK
Griewank	2.434065136	3.52E+02	VAR	2.434065136	1.57E+00	YOK	2.434065136	1.28E-15	YOK
Rastrigin	2.434065136	1.32E+03	VAR	2.434065136	1.19E+01	VAR	2.434065136	2.10E-14	YOK
Rosenbrock	2.434065136	1.11E+02	VAR	2.434065136	6.29E+01	VAR	2.434065136	6.77E+01	VAR
Ackley	2.434065136	9.78E+01	VAR	2.434065136	1.19E+03	VAR	2.434065136	1.16E-12	YOK
Schwefel	2.434065136	1.70E+03	VAR	2.434065136	3.03E+03	VAR	2.434065136	9.53E+02	VAR
Stepint	2.434065136	6.49E+00	VAR	2.434065136	0.00E+00	YOK	2.434065136	0.00E+00	YOK
Schwefel 2.22	2.434065136	5.93E+00	VAR	2.434065136	3.03E-16	YOK	2.434065136	5.32E-15	YOK
Schwefel 1.2	2.434065136	5.36E+00	VAR	2.434065136	2.04E-18	YOK	2.434065136	6.12E-16	YOK
Step	2.434065136	2.75E+02	VAR	2.434065136	0.00E+00	YOK	2.434065136	0.00E+00	YOK
quartic	2.434065136	4.17E+02	VAR	2.434065136	2.36E+02	VAR	2.434065136	1.89E+01	VAR
penalized	2.434065136	3.56E+01	VAR	2.434065136	1.71E+00	YOK	2.434065136	2.09E-14	YOK
penalized2	2.434065136	6.66E+00	VAR	2.434065136	7.50E+00	VAR	2.434065136	1.08E-15	YOK
Kowalik	2.434065136	2.36E+01	VAR	2.434065136	3.30E+02	VAR	2.434065136	1.39E+00	YOK
SixHumpCamelBack	2.434065136	1.03E+00	YOK	2.434065136	1.03E+00	YOK	2.434065136	0.00E+00	YOK
Branin	2.434065136	6.77E+02	VAR	2.434065136	4.06E+02	VAR	2.434065136	0.00E+00	YOK
GoldSteinPrice	2.434065136	1.46E+01	VAR	2.434065136	1.95E+02	VAR	2.434065136	5.61E+00	VAR
Shekel5	2.434065136	1.04E+01	VAR	2.434065136	2.24E+02	VAR	2.434065136	1.97E+01	VAR
Shekel7	2.434065136	1.14E+01	VAR	2.434065136	6.99E+02	VAR	2.434065136	2.53E+01	VAR
Shekel10	2.434065136	1.26E+01	VAR	2.434065136	1.18E+18	VAR	2.434065136	1.80E+01	VAR
Hartman6	2.434065136	7.52E+02	VAR	2.434065136	1.02E+02	VAR	2.434065136	0.00E+00	YOK

Tablo 4.15. (devamı)

	PSO			DE			ABC		
	F Level	F Stat.	Farklılık	F Level	F Stat.	Farklılık	F Level	F Stat.	Farklılık
Hartman3	2.434065136	8.16E+03	VAR	2.434065136	2.21E+17	VAR	2.434065136	2.28E+01	VAR
Beale	2.434065136	1.77E-16	YOK	2.434065136	2.37E-18	YOK	2.434065136	8.25E+00	VAR
Bohachecsky1	2.434065136	1.32E-14	YOK	2.434065136	1.80E-18	YOK	2.434065136	1.42E-18	YOK
Bohachecsky2	2.434065136	2.57E-17	YOK	2.434065136	9.35E-19	YOK	2.434065136	9.37E-19	YOK
Bohachecsky3	2.434065136	7.34E-18	YOK	2.434065136	2.16E-18	YOK	2.434065136	5.36E+03	VAR
Booth	2.434065136	3.72E-18	YOK	2.434065136	3.68E-19	YOK	2.434065136	1.69E-03	YOK
DixonPrice	2.434065136	1.55E+02	VAR	2.434065136	4.79E+00	VAR	2.434065136	1.29E+01	VAR
Easom	2.434065136	1.27E+02	VAR	2.434065136	0.00E+00	YOK	2.434065136	1.05E+01	VAR
Matyas	2.434065136	1.58E-17	YOK	2.434065136	5.51E-18	YOK	2.434065136	1.12E+04	VAR
Michalewics2	2.434065136	7.49E+01	VAR	2.434065136	8.70E+02	VAR	2.434065136	0.00E+00	YOK
Michalewics5	2.434065136	1.31E+02	VAR	2.434065136	2.10E+01	VAR	2.434065136	1.81E+00	YOK
Michalewics10	2.434065136	9.33E+01	VAR	2.434065136	4.59E+00	VAR	2.434065136	1.65E+00	YOK
Perm	2.434065136	8.33E+01	VAR	2.434065136	1.03E+00	YOK	2.434065136	1.00E+01	VAR
Powell	2.434065136	6.57E+01	VAR	2.434065136	1.27E+03	VAR	2.434065136	3.22E+02	VAR
PowerSum	2.434065136	1.08E+03	VAR	2.434065136	2.89E+01	VAR	2.434065136	7.57E+00	VAR
Shubert	2.434065136	1.36E-10	YOK	2.434065136	1.36E-10	YOK	2.434065136	4.10E+00	VAR
SumSquares	2.434065136	5.55E+00	VAR	2.434065136	7.84E-19	YOK	2.434065136	1.05E-15	YOK
Trid6	2.434065136	0.00E+00	YOK	2.434065136	0.00E+00	YOK	2.434065136	0.00E+00	YOK
Trid10	2.434065136	9.28E-01	YOK	2.434065136	0.00E+00	YOK	2.434065136	4.95E+00	VAR
Zakharov	2.434065136	1.03E+01	VAR	2.434065136	1.22E-18	YOK	2.434065136	1.75E+01	VAR
Colville	2.434065136	2.00E+01	VAR	2.434065136	7.39E+00	VAR	2.434065136	1.21E+00	YOK
Langerman2	2.434065136	8.13E+01	VAR	2.434065136	2.14E+00	YOK	2.434065136	8.33E-15	YOK
Langerman5	2.434065136	3.86E+01	VAR	2.434065136	2.76E+01	VAR	2.434065136	3.84E+00	VAR
Langerman10	2.434065136	8.01E+00	VAR	2.434065136	2.23E+01	VAR	2.434065136	1.66E+00	YOK

Tablo 4.15. (devamı)

	PSO			DE			ABC		
	F Level	F Stat.	Farklılık	F Level	F Stat.	Farklılık	F Level	F Stat.	Farklılık
FletcherPowell2	2.434065136	2.24E+23	VAR	2.434065136	3.95E+02	VAR	2.434065136	8.53E-18	YOK
FletcherPowell5	2.434065136	5.79E+01	VAR	2.434065136	1.88E+02	VAR	2.434065136	3.37E+00	VAR
FletcherPowell10	2.434065136	3.29E+01	VAR	2.434065136	9.68E+00	VAR	2.434065136	8.60E+00	VAR
♯			38			26			22

4.1.2. Diğer Algoritmalarla ABC Algoritmasının Kıyaslanması

ABC algoritmasının literatürdeki mevcut diğer algoritmalarla kıyaslanması performansının değerlendirilmesinde önemli bir husustur. Bazı algoritmalar sahip oldukları davranışsal özelliklerinden dolayı bazı fonksiyonlarda iyi sonuç verirken başka fonksiyonlar üzerinde başarımı düşük olabilmektedir. Her tür problem üzerinde en iyi performansı sergilemek ideal durum olsa da pek mümkün değildir. Algoritmalardan beklenen genel olarak her tür fonksiyon üzerinde kabul edilebilir iyi bir performans sergilemesidir. Bu bölümde, ABC algoritması popüler olan başka algoritmalarla kıyaslanarak ne tür fonksiyonlar üzerinde hangi algoritmalara göre daha iyi, benzer veya düşük başarım göstermektedir sorusunun cevabı aranacaktır. Bu bölümde 3 farklı kıyaslama çalışması yapılmıştır. Birinci çalışmada ABC algoritması GA, DE ve PSO algoritamalarıyla çok geniş bir test fonksiyon seti üzerinde kıyaslanmıştır. İkinci ve üçüncü çalışmalarda ABC algoritması ES'nin türevleri ile kıyaslanmıştır. İkinci ve üçüncü çalışmalarda kullanılan fonksiyon sayısı ve türlerinin az oluşunun nedeni literatürdeki çalışmaların referans alınmasından kaynaklanmaktadır.

4.1.2.1. Çalışma 1: ABC, GA, DE ve PSO Algoritmalarının Kıyaslanması

Bu çalışmada GA, DE, PSO ve ABC algoritmalarının başarımınlarının karşılaştırılması için Tablo 4.11'de verilen 50 test fonksiyonu kullanılmıştır. Daha öncede belirtildiği gibi tabloda tek modlu, çok modlu, düzenli, düzensiz, ayrıştırılabilir, ayrıştırılamayan çok boyutlu olmak üzere çok sayıda fonksiyon bulunmaktadır. Çalışma 1'de yapılan tüm deneylerde algoritmaların popülasyon büyüklüğü ve maksimum değerlendirme sayısı gibi ortak parametrelerin değerleri aynı alınmıştır. GA, DE, PSO ve ABC algoritmalarının hepsi için popülasyon büyüklüğü 50 ve maksimum değerlendirme sayısı da 500000 olarak alınmıştır. Algoritmalara has diğer kontrol parametrelerinin değerleri aşağıda verilmektedir.

***GA Algoritmasının Parametre Değerleri*:** Kıyaslamalarda kullanılan GA gösterim olarak ikili kodlama kullanan değerlendirme, ölçekleme, seleksiyon, çaprazlama, mutasyon ve elit birimlerinden oluşan standart GA'dır. Tek noktalı çaprazlama operatörü kullanılmış ve çaprazlama oranı 0.8 olarak seçilmiştir. Popülasyondaki kaybolan farklılığı yeniden sağlayan mutasyon işlemininin uygulanma oranı 0.01 olarak alınmıştır. Seleksiyon metodu olarak stokastik uniform örnekleme tekniği kullanılmıştır. Değişecek popülasyonun oranını belirleyen kuşak boşluğu (generation gap) değeri 0.9 olarak alınmıştır.

***DE Algoritmasının Parametre Değerleri* :** DE algoritmasında rasgele seçilen iki çözümün farkının ölçeklenerek üçüncü bir rasgele çözüme eklendiği strateji kullanılmıştır. İki çözümün farkını ölçekleyen F parametresinin değeri 0.5, çaprazlama oranı 0.9 olarak alınmıştır [203].

***PSO Algoritmasının Parametre Değerleri*:** PSO algoritmasında parçacığın kendisinin ve tüm popülasyon bilgisinin yeni çözüm üzerindeki ağırlığını belirleyen (EK-1.5) eşitliğindeki bilişsel ve sosyal bileşenlerin (ϕ_1 and ϕ_2) değerleri 1.8, daha önceki hız değerinin yeni hız değerini ne kadar etkileyeceğini belirleyen inertia parametresi değeri 0.6 olarak alınmıştır. Bunlar [204] referansında önerilen değerlerdir.

***ABC Algoritmasının Parametre Değerleri*:** Algoritmaların ortak kontrol parametreleri olan popülasyon büyüklüğü ve maksimum değerlerdirme sayısı haricinde, temel ABC algoritmasının tek kontrol parametresi bir yiyecek kaynağının bırakılıp bırakılmayacağına karar verilmesini sağlayan *limit* parametresidir. Bir çözüm önceden belirlenen deneme sayısınca geliştirilememişse o çözümün yerine rasgele çözüm üretilir. Bu belirlenen sayı *limit* değerine eşittir. *limit* parametresinin değeri popülasyon büyüklüğü (SN) ile parametre sayısına (D) bağlı olarak aşağıdaki formda yazılabilir:

$$limit = (SN * D) * 0.5 \tag{4.4}$$

Çalışma 1 başlığı altında yapılan çalışmalarda, Tablo 4.11'de listesi verilen çok geniş bir fonksiyon seti üzerinde GA, PSO, DE ve ABC algoritmaları karşılaştırılmıştır. Her algoritma her bir fonksiyon üzerinde farklı rasgele başlangıç popülasyonları ile 30 kez koşturulmuş ve algoritmaların ürettiği en iyi değerlerin ortalaması kaydedilmiştir. Rakamların daha anlaşılır olması için 10^{-12}'den daha küçük değerler 0 olarak kabul edilmiştir. Ortalaması en iyi değer (mean best), standart sapması (standart deviation) ve ortalamaların standart hataları (standart error of means) Tablo 6.8'de verilmektedir.

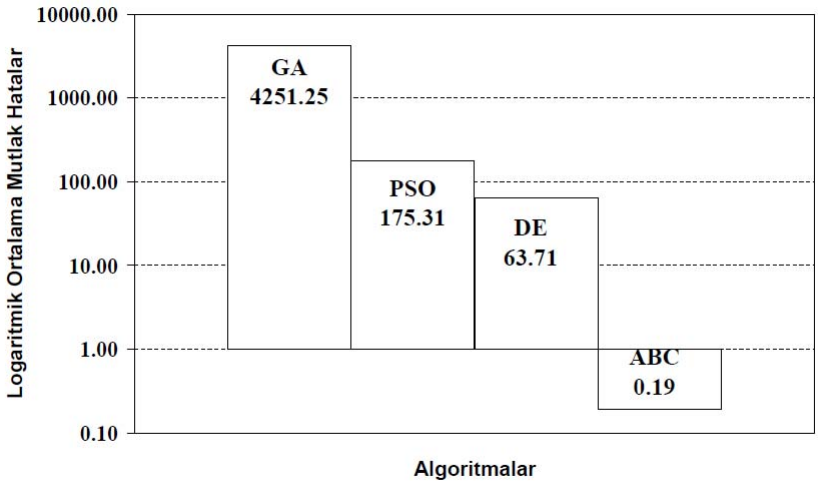
Tablo 6.8'de verilen sonuçlar arasında anlamlı bir fark olup olmadığının analizini yapmak amacıyla ABC ve diğer algoritmaların her biri arasında Student t-testi yapılmıştır. Her bir veri setinden hesaplanan p kritik değeri, α güvenilirlik değeri ile doğrudan kıyaslanmamıştır. Bunun sebebi rasgelelik içeren algoritmaların aynı dağılımdan gelen rasgele sayılarla başlatılsalar bile farklı sonuçlar üretebiliyor olmalarıdır [205]. Bunun için t-testinden sonra Değiştirilmiş Bonferroni Düzeltmesi (Modified Bonferroni Correction, MBC) yapılmıştır. MBC düzeltmesi ile t-testi ile hesaplanan p değerleri azalan şekilde sıralanır ve 0.05 olan güvenilirlik seviyesi α sırasının tersine (inverse rank) bölünerek yeni α seviyesi bulunur. Eğer $p > \alpha$ ise algoritmaları o fonksiyon üzerinde anlamlı derecede farklı sonuçlar üretmiş olarak kabul edilir. Aksi takdirde sonuçlar arasında anlamlı bir farklılık yoktur demektir. Hesaplanan p değerleri, sıralar(rank), ters sıralar (inverse rank, IR) ve anlamlı fark olup olmadığına dair sonuçlar ABC-GA çifti için Tablo 4.17'de, ABC-PSO çifti için Tablo 4.18'de ve ABC-DE çifti için Tablo 4.19'da verilmiştir.

Sonuçlar incelendiğinde, tüm algoritmaların farklılık sağlayan operatörlerinin ve değişken adım büyüklüğüne sahip operatörlerinin olmasından dolayı Stepint ve Matyas gibi düz yüzeylere sahip fonksiyonlarda aynı düzeyde performans gösterdikleri görülür. DE algoritması mutant vektör oluşturmak için rasgele seçilmiş çözümlerin farkını (eşitlik EK-1.1), PSO rasgele ağırlıklandırılan fark vektörlerini (eşitlik EK-1.5), GA ise çaprazlama operatörünü kullanmaktadır. ABC algoritması ise değişim sağlamak için (3.2) veya (4.1) ile tanımlı

operatörünü kullanmaktadır. Düz yüzeye sahip fonksiyonlar arasında PowerSum'da DE algoritması diğer algoritmalara nazaran daha iyi performans göstermiştir. Goldstein-Price, Stepint, Beale, Easom, Matyas, Foxholes, Branin, Bohachevsky1, Booth, Six Hump Camel Back, Bohachevsky3, Shubert, Fletcher-Powell2 fonksiyonlarında algoritmaların sonuçları arasında anlamlı fark görülmemiştir. Tablo 4.17'den GA ve ABC algoritmaları arasında 20 fonksiyon üzerinde anlamlı bir fark yokken, 28 fonksiyonda ABC algoritması GA'dan daha iyi, 2 fonksiyon üzerinde ise GA'nın ABC algoritmasından daha iyi olduğu görülmektedir. PSO ve ABC algoritmalarının karşılaştırılmasında, Tablo 4.18'den 22 fonksiyonda PSO ve ABC algoritmaları eşit performans gösterirken kalan 28 fonksiyonunun 4 ünde PSO, 24'ünde ise ABC algoritması daha iyi performans göstermiştir. DE ve ABC kıyaslamasında ise Tablo 4.19'dan görüldüğü gibi 37 fonksiyonda eşit performans, 5'inde DE'nin üstünlüğü, 8'inde ABC'nin üstünlüğü görülmektedir.

Tüm algoritmaların eşit performans gösterdiği Beale fonksiyonu minimumun yakınlarında kıvrık yapılı şekle sahiptir. Ancak bu tür bir fonksiyon olan Colville fonksiyonunda sadece PSO algoritması minimuma ulaşabilmiştir. 30 parametreli olacak şekilde genişletilmiş Rosenbrock fonksiyonunda ABC algoritması diğer algoritmalara kıyasla en iyi sonucu üretmiştir. Diğer taraftan bu fonksiyonlar Griewank fonksiyonu gibi parametreleri arasında ilişki olan, yani ayrıştırılamayan türden fonksiyonlardır. Simetrik olmayan Langerman fonksiyonlarında, 2 parametreli durum için ABC ve DE eşit performans gösterirken 5 ve 10 parametreli durumda DE daha iyi performans göstermiştir.

Tablolardan çıkarılabilecek diğer bir sonuçta ABC algoritmasının problemlerin parametre sayılarının artmasıyla performansını yeterli düzeyde koruyabildiğidir. Tablo 4.11'den görüldüğü gibi kıyaslamalarda kullanılan fonksiyonların 14 tanesinin parametre sayısı 30'dur. Bu 14 fonksiyonun 6'sında ABC algoritması DE'den, 8'inde PSO'dan, 14'ünde ise GA'dan daha iyi performans sergilemiştir. Tek modlu olan 4 fonksiyonda (Colville, Zakharov, Powell ve Quartic'de ABC, Colville, Quartic, Rosenbrock ve Dixon-Price'da DE) ABC ve DE'nin her ikiside yeterli başarımı gösterememişlerdir. Çok modlu olan 5 fonksiyonda ise ABC, 9 fonksiyonda DE başarısız olmuştur. Kullanılan çok modlu fonksiyonlar üzerinde ABC algoritması



Şekil 4.7. Algoritmaların ortalama mutlak hata değerleri

DE, PSO ve GA'ya nazaran daha başarılı ve daha gürbüz (robust) olarak değerlendirilebilir.

Daha önce de belirtildiği gibi Çalışma 1 başlığı altında yapılan çalışmalarda kullanılan GA ikili gösterim kullanmaktadır. Reel gösterim kullanan GA'nın daha hızlı ve koşmadan koşmaya göre daha tutarlı olduğu ve bazı özel problemlerde daha yüksek hassasiyet sunduğu belirtilmelidir [206].

Şekil 4.7'den görüldüğü gibi, tüm fonksiyonlar üzerinde algoritmaları birarada değerlendirilmek istenirse ortalama mutlak hataları kullanılabilir. Ortalama mutlak hata hesaplanırken algoritmaların buldukları mutlak hatalar hesaplanmış, olması gereken değere bölünerek normalize edilmiştir ve sonra bunların tüm fonksiyonlar için toplamı alınmıştır. Langerman10 fonksiyonunun minimumu bilinmediği için bu değerlendirme yapılırken 49 fonksiyon esas alınmıştır. Farklar oldukça büyük olduğundan Şekil 4.7 logaritmik ölçekte çizdirilmiştir. Görüldüğü üzere ABC algoritması en az ortalama mutlak hataya sahip algoritmadır.

4.1.2.2. Çalışma 2: ABC ve ES-1 Algoritmalarının Kıyaslanması

Çalışma 2 başlığı altında yapılan çalışmalarda ABC algoritması [3] numaralı referansta kullanılan ES'nin türevleri ile kıyaslanmıştır. Sonuçların alındığı

[3] referansında ES türevlerinden CES, FES, CMA-ES ve ESLAT algoritmaları kullanılmıştır. Bu algoritmalarla ilgili bilgiye EK-1 bölümünden ulaşılabilir. CES, FES, CMA-ES ve ESLAT algoritmalarının kontrol parametresine ait değerler [3, 4] çalışmalarında sunulmaktadır. Her fonksiyon için ABC algoritması 50 kez koşulmuş ve hata değeri ile optimum arasındaki fark 10^{-3} olunca algoritmanın koşturulması durdurulmuştur. ABC algoritmasının tüm fonksiyonlar için koloni büyüklüğü 20 ve maksimum değerlendirme sayısı 100000 olarak alınmıştır. Kontrol parametresi *limit*'in değeri yine Çalışma 1'de olduğu gibi 4.4 ifadesi ile belirlenmiştir. Çalışma 2'de kullanılan CES, FES, CMA-ES ve ESLAT algoritmalarına ait çözüm kalitesi ve hesaplama maliyeti gibi sonuçlar [3] numaralı referanstan alınmıştır. Kıyaslamada kullanılan fonksiyonlar Tablo 4.20'de listelenmiştir. Tablodaki fonksiyonlardan Sphere, Schwefel 1.2, Schwefel 2.21, Schwefel 2.22, Rosenbrock, Step ve Quartic fonksiyonları tek modlu ve çok boyutlu problemlerdir. Schwefel, Rastrigin, Ackley, Griewank, Penalized and Penalized2 fonksiyonları ise çok modlu ve çok boyutlu problemlerdir. Foxholes, Kowalik, Six Hump Camel Back, Branin, Goldstein-Price, Hartman family ve Shekel Family fonksiyonları ise çok modlu ancak problem boyutunun düşük olduğu fonksiyonlardır.

Algoritmaların belirtilen fonksiyonlar için belirtilen parametre setleri ile 50 kez koşulmaları ile bulunan ortalama hata değerleri ve hataların standart sapmaları Tablo 4.21'de sunulmaktadır. Ayrıca algoritmaların ürettikleri hata değerlerinin 10^{-3}'e düşmesi için geçen değerlendirme sayıları; düşmediği durumda da maksimum değerlendirme sayısı kaydedilerek algoritmaların hesaplama maliyeti ile ilgili analiz yapılmış ve bu değerler Tablo 4.22'de verilmiştir. Bu tablodaki sonuçlara göre CMA-ES 12 fonksiyonda, CES ve ESLAT 2 fonksiyonda ABC ise 8 fonksiyonda değerlendirme sayısına göre daha az maliyete sahiptir. Genel olarak söylemek gerekirse, CMA-ES'in tek modlu problemlerde ESLAT, CES ve ABC' ye göre maliyeti daha azdır. Bunun sebebi, CMA-ES algoritmasının bölgesel bilginin kullanılması esasına dayanarak çözüm arayan bir lokal araştırma algoritma olmasıdır [5]. Tablo 4.23'da verilen başarı oranlarına göz atılacak olursak CMA-ES algoritmasının çok modlu problemlerdeki başarısızlığı görülebilir. ESLAT algoritması CMA-ES'e nazaran düşük boyutlu problemlerde daha iyi olmasına

rağmen, boyutun artmasıyla birlikte çok modlu problemlerde CMA-ES gibi performansı kötüleşmektedir. ABC algoritması, 20 fonksiyon üzerinde en iyi performansı gösteren algoritma olmuştur. 20 fonksiyonun 14'ü üzerinde 100% başarı oranına sahiptir. Performansı hem tek modlu hem çok modlu fonksiyonlar üzerinde oldukça yüksektir. CMA-ES ve ESLAT 9 fonksiyon üzerinde en iyi başarımı yakalamışlardır. 2 fonksiyon (Rosenbrock, Schewfel 2.21) üzerinde CMA-ES ABC'den daha iyidir. ABC'nin CMA-ES'e göre daha iyi olduğu fonksiyon sayısı ise 13'tür (Step, Schewfel, Rastrigin, Griewank, Penalized, Penalized 2, Foxholes, Kowalik, Goldstein-Price, Hartman 6, Shekel 5, Shekel 7, Shekel 10). Fonksiyonların 7'sinde ise eşit performansa sahiptirler. ESLAT algoritmasının ABC'ye göre daha iyi olduğu fonksiyon sayısı ise sadece 1'dir. ABC algoritması ESLAT'a göre 11 fonksiyonda (Step, Schwefel, Rastrigin, Griewank, Penalized 2, Foxholes, Kowalik, Hartman 6, Shekel 5, Shekel 7, Shekel 10) daha iyi ve 9 fonksiyonda da eşit performanslıdır. ABC algoritmasının keşif (exploration) ve faydalanma (exploitation) süreçlerinin ikisini birlikte etkili biçimde kullanıyor olmasından dolayı CMA-ES ve ESLAT algoritmalarına kıyasla daha yüksek başarı oranı bulunmaktadır.

4.1.2.3. Çalışma 3: ABC ve ES-2 Algoritmalarının Kıyaslanması

Çalışma 3 başlığı altında ABC algoritması yine ES türevlerinden olan SOM-ES, NG-ES ve CMA-ES algoritmaları ile kıyaslanmıştır. SOM-ES, NG-ES ve CMA-ES'a ait olan sonuçlar [5] numaralı referanstan alınmıştır. Bu kıyaslamada kullanılan fonksiyonlar Tablo 4.24'de verilmektedir. Fonksiyonlardan 1, 2 ve 3 numaralı olanların yeterli yakınsama için eşik değerleri sırasıyla 40, 0.001 ve 0.001'dir. Her bir koşmada izin verilen maksimum değerlendirme sayısı 5000'dir ve algoritmalar da 10000 kez koşulmuştur. Adil bir karşılaştırma olması açısından parametre değerleri [5] numaralı çalışmadaki gibi ayarlanmıştır. Tablo 4.24'deki fonksiyonlardan Rastrigin ve Griewank fonksiyonları Çalışma 1 ve Çalışma 2'deki gibidir. Ancak Rosenbrock fonksiyonuna (-1,1) aralığında Gaussian terim eklenmiştir. Bu eklenti neticesinde fonksiyona (1,1) noktasında bölgesel minimum katılmakla ve global minimum (-1,-1)'e kaymaktadır. Bu işlem, bölgesel minimumun global minimuma

göre daha geniş bir alana yayılmasından dolayı fonksiyonu daha zor hale getirmiştir [5].

Tablo 4.25'de, belirlenen eşik değerine ulaşlması için koşmalardan elde edilen gerekli değerlendirme sayılarının ortalama ve standart sapma değerleri ile başarı oranlarının ortalama ve standart sapma değerleri verilmektedir. Sonuçlardan görüldüğü gibi Rosenbrock fonksiyonu üzerinde SOM-ES ve NG-ES algoritmalarının hem çözüm kalitesi hem de maliyeti açısından daha başarılı oldukları görülmektedir. Ancak çok modlu fonksiyonlar olan Griewank ve Rastrigin'de ise ABC'nin SOM-ES, NG-ES ve CMA-ES algoritmalarını geride bıraktığı görülmektedir. Çözüm kalitesi açısından 3 fonksiyonda da ABC, CMA-ES'den daha iyidir. Ancak Çalışma 2'de de olduğu gibi ABC'nin yakınsama hızı CMA-ES'e göre daha yavaştır. Bunun sebebi CMA-ES'in bölgesel bilgiyi kullanmasıdır. Küresel optimizasyon açısından düşünürsek ABC daha iyi performans sergilemektedir.

Buraya kadarki bölümde ABC algoritmasının performansı GA, DE, PSO ve ES türevleri ile kıyaslanmıştır. ABC algoritmasının performansı algoritmaya başka sezgisellerin katılmasıyla artırılabilir olmasına rağmen buradaki amacımız ABC'nin standart versiyonu ile diğer literatürdeki algoritmaları kıyaslamak olduğundan temel ABC algoritması kullanılmıştır. Çalışma 1'de aynı popülasyon büyüklüğü ve maksimum değerlendirme sayısı kullanılarak ve kullanıcının parametre değerleri hakkında bilgisi olmadığını varsayarak çalışmalar gerçekleştirilmiştir.

GA ve DE yeni bir aday çözüm üretirken çaprazlama operatörü kullanırken basit ABC algoritmasının böyle bir operatörü bulunmamaktadır. Yeni çözüm üretme aşamasında ABC rasgele seçtiği bir çözümle mevcut çözümün bazı kısımlarının farklarını alan basit bir işlem kullanmaktadır. Bu işlem bölgesel minimaya doğru araştırmayı hızlandırmaktadır. GA, DE ve PSO da popülasyonda en iyi çözüm daima tutulmakta ve yeni çözüm üretiminde kullanılabilmektedir. Ancak basit ABC algoritmasında bulunan en iyi çözümün bırakılması yani rasgele bir çözümle yerdeğiştirmesi söz konusudur. Bu nedenle, en iyi çözüm sonraki aday çözümlerin üretimine katkı sağlamamaktadır. DE ve ABC algoritmalarının her ikiside mevcut çözümle aday çözüm arasında aç gözlü seleksiyon işlemi uygulamaktadırlar. DE

algoritması gibi ABC algoritması da görevli arı biriminde çözümlerin kalite değerine bakmaksızın her bir çözüme karşılık bir aday çözüm üretir. Ancak gözcü arı biriminde yüksek uygunluk değerine sahip çözümlere karşılık daha fazla aday çözüm üretilme olasılığı vardır. Böylece çözümü içerebilecek potansiyel bölgeler daha kısa zamanda ve detaylıca araştırılmış olur. Bu seleksiyon işlemi GA'daki seleksiyon işlemine benzemektedir.

GA'da veya DE'de çözümlerde gerekli değişimi yapmak için bir çözümün rasgele seçilen bir bölgesinde veya tamamında değişiklik yaparlar. ABC'de popülasyondaki farklılığı kontrol eden iki mekanizma bulunmaktadır: a) DE ve GA'da olduğu gibi mevcut çözümün rasgele seçilen bir bölümünü rasgele belirlenen genlik değeri ile modifiye etmek. Bu değişiklik, bölgesel arama ve ince ayar için oldukça faydalıdır. b) Mevcut çözümün bir bölümünün değiştirilmesindense mevcut çözümün popülasyondan tamamen atılarak kaşif arı tarafından rasgele keşfedilen yeni bir çözümün eklenmesi. Bu mekanizma ABC'nin küresel araştırma yeteneğini arttırmakta ve araştırmanın erken yakınsama sıkıntısını engellemektedir. Bu özellik aynı zamanda algoritmaların performansının popülasyon büyüklüğüne olan bağımlılığını da azaltmaktadır. Yani görevli ve gözcü arılar tarafından gerçekleştirilen bölgesel araştırma ile kaşif arılar tarafından gerçekleştirilen küresel araştırma arasında iyi bir denge kurulmuştur. Dolayısıyla çok modlu ve çok boyutlu problemler üzerinde ABC algoritması bu çalışmada kıyaslanan diğer algoritmlara göre daha iyi başarım sergilemektedir.

Algoritmaların ortak kontrol parametreleri olan maksimum değerlendirme sayısı ve popülasyon büyüklüğü dışında, GA üç (çaprazlama oranı, mutasyon oranı ve kuşak boşluğu), DE en az iki (çaprazlama oranı ve ölçekleme faktörü) ve temel PSO algoritması üç (inertia, bilişsel ve sosyal bileşenler) kontrol parametresine daha sahiptir. Ayrıca PSO'nun performansında, hızın alabileceği maksimum limit olan υ_{max}'da oldukça etkilidir. ABC algoritmasının maksimum çevrim sayısı ve koloni büyüklüğü dışında tek kontrol parametresi "limit" parametresidir. "limit" parametresinin değerinin de (4.4) eşitliği ile koloni büyüklüğü ve parametre sayısına bağlı olarak belirlendiği göz önüne alınırsa ABC'nin tüm algoritmalarda ortak olan iki parametresi dışında değerinin ayarlanması gereken başka bir parametresi

bulunmamaktadır. Sonuç olarak ABC algoritması DE ve PSO kadar basit ve esnek ve ayrıca daha az kontrol parametresine sahiptir.

Çalışma 2 ve 3'de yer verilen ES'lerde yeni bir bireyin oluşturulması aşamasında yeniden oluşum (recombination) ve mutasyon birimleri kullanılırken, ABC algoritması yalnız mutasyon birimi ile yeni bir birey oluşturmaktadır. ES algoritmasının temel versiyonları ABC kadar basit yapıda iken, bu çalışmada kullanılan ileri versiyonlar oldukça karmaşık yapıdadırlar. Ayrıca hepside ABC'den daha fazla kontrol parametresine sahiptirler.

Tüm bu sonuçlara dayanarak ABC algoritmasının basit sınırlamasız fonksiyonlar üzerindeki performansının çalışmada kıyaslanan diğer algoritmalarınkine benzer yahut daha iyi olduğu; ve yine daha az kontrol parametresine sahip ABC algoritmasının çok modlu ve çok parametreli optimizasyon problemlerinin çözümünde oldukça iyi sonuçlar ürettiği söylenebilir.

Tablo 4.16. Çalışma 1'de yapılan GA, PSO, DE ve ABC algoritmalarının 30 koşmalarına ait istatistiksel sonuçlar. D:Problemin Boyutu, Ort.: En iyi değerlerin ortalamaları, Std.Sapma: En iyi değerlerin standart sapmaları, SEM: Ortalamaların standart hataları.

No	**Aralık**	**D**	**Fonksiyon**	**Min.**		**GA**	**PSO**	**DE**	**ABC**
1	[-5.12,5.12]	5	Stepint	0	**Ort.**	0	0	0	0
					Std.Sap.	0	0	0	0
					SEM	0	0	0	0
2	[-100,100]	30	Step	0	**Ort.**	1.17E+03	0	0	0
					Std.Sap.	76.561450	0	0	0
					SEM	13.978144	0	0	0
3	[-100,100]	30	Sphere	0	**Ort.**	1.11E+03	0	0	0
					Std.Sap.	74.214474	0	0	0
					SEM	13.549647	0	0	0
4	[-10,10]	30	SumSquares	0	**Ort.**	1.48E+02	0	0	0
					Std.Sap.	12.4092893	0	0	0
					SEM	2.265616	0	0	0
5	[-1.28,1.28]	30	Quartic	0	**Ort.**	0.1807	0.00115659	0.0013633	0.0300166
					Std.Sap.	0.027116	0.000276	0.000417	0.004866
					SEM	0.004951	5.04E-05	7.61E-05	0.000888
6	[-4.5,4.5]	2	Beale	0	**Ort.**	0	0	0	0
					Std.Sap.	0	0	0	0
					SEM	0	0	0	0
7	[-100,100]	2	Easom	-1	**Ort.**	-1	-1	-1	-1
					Std.Sap.	0	0	0	0
					Std.Sap.	0	0	0	0
8	[-10,10]	2	Matyas	0	**Ort.**	0	0	0	0
					Std.Sap.	0	0	0	0
					SEM	0	0	0	0

Tablo 4.16. (devamı)

No	**Aralık**	**D**	**Fonksiyon**	**Min.**		**GA**	**PSO**	**DE**	**ABC**
9	[-10,10]	4	Colville	0	**Ort.**	0.014938	0	0.0409122	0.0929674
					Std.Sap.	0.007364	0	0.081979	0.066277
					SEM	0.001344	0	0.014967	0.012100
10	[-D^2,D^2]	6	Trid6	-50	**Ort.**	-49.9999	-50	-50	-50
					Std.Sap.	2.25E-5	0	0	0
					Std.Sap.	4.11E-06	0	0	0
11	[-D^2,D^2]	10	Trid10	-210	**Ort.**	-209.476	-210	-210	-210
					Std.Sap.	0.193417	0	0	0
					SEM	0.035313	0	0	0
12	[-5,10]	10	Zakharov	0	**Ort.**	0.013355	0	0	0.0002476
					Std.Sap.	0.004532	0	0	0.000183
					SEM	0.000827	0	0	3.34E-05
13	[-4,5]	24	Powell	0	**Ort.**	9.703771	0.00011004	2.17E-07	0.0031344
					Std.Sap.	1.547983	0.000160	1.36E-7	0.000503
					SEM	0.282622	2.92E-05	2.48E-08	9.18E-05
14	[-10,10]	30	Schwefel 2.22	0	**Ort.**	11.0214	0	0	0
					Std.Sap.	1.386856	0	0	0
					SEM	0.253204	0	0	0
15	[-100,100]	30	Schwefel 1.2	0	**Ort.**	7.40E+03	0	0	0
					Std.Sap.	1.14E+03	0	0	0
					SEM	208.1346	0	0	0
16	[-30,30]	30	Rosenbrock	0	**Ort.**	1.96E+05	15.088617	18.203938	0.0887707
					Std.Sap.	3.85E+04	24.170196	5.036187	0.077390
					SEM	7029.106155	4.412854	0.033333	0.014129

Tablo 4.16. (devamı)

No	Aralık	D	Fonksiyon	Min.		GA	PSO	DE	ABC
17	[-10,10]	30	Dixon-Price	0	**Ort.**	1.22E+03	0.66666667	0.6666667	0
					Std.Sap.	2.66E+02	E-8	E-9	0
					SEM	48.564733	1.8257e-009	1.8257e-0010	0
18	[-65.536,65.536]	2	Foxholes	0.998	**Ort.**	0.998004	0.99800393	0.9980039	0.9980039
					Std.Sap.	0	0	0	0
					SEM	0	0	0	0
19	[-5,10]x[0,15]	2	Branin	0.398	**Ort.**	0.397887	0.39788736	0.3978874	0.3978874
					Std.Sap.	0	0	0	0
					SEM	0	0	0	0
20	[-100,100]	2	Bohachevsky1	0	**Ort.**	0	0	0	0
					Std.Sap.	0	0	0	0
					SEM	0	0	0	0
21	[-10,10]	2	Booth	0	**Ort.**	0	0	0	0
					Std.Sap.	0	0	0	0
					SEM	0	0	0	0
22	[-5.12,5.12]	30	Rastrigin	0	**Ort.**	52.92259	43.9771369	11.716728	0
					Std.Sap.	4.564860	11.728676	2.538172	0
					SEM	0.833426	2.141353	0.463405	0
23	[-500,500]	30	Schwefel	-12569.5	**Ort.**	-11593.4	-6909.1359	-10266	-12569.487
					Std.Sap.	93.254240	457.957783	521.849292	0
					SEM	17.025816	83.611269	95.276209	0
24	[0,π]	2	Michalewicz2	-1.8013	**Ort.**	-1.8013	-1.5728692	-1.801303	-1.8013034
					Std.Sap.	0	0.119860	0	0
					SEM	0	0.021883	0	0

Tablo 4.16. (devamı)

No	Aralık	D	Fonksiyon	Min.		GA	PSO	DE	ABC
25	[0,π]	5	Michalewicz5	-4.6877	**Ort.**	-4.64483	-2.4908728	-4.683482	-4.6876582
					Std.Sap.	0.097850	0.256952	0.012529	0
					SEM	0.017865	0.046913	0.002287	0
26	[0,π]	10	Michalewicz10	-9.6602	**Ort.**	-9.49683	-4.0071803	-9.591151	-9.6601517
					Std.Sap.	0.141116	0.502628	0.064205	0
					SEM	0.025764	0.091767	0.011722	0
27	[-100,100]	2	Schaffer	0	**Ort.**	0.004239	0	0	0
					Std.Sap.	0.004763	0	0	0
					SEM	0.000870	0	0	0
28	[-5,5]	2	6HumpCamelBack	-1.03163	**Ort.**	-1.03163	-1.0316285	-1.031628	-1.0316285
					Std.Sap.	0	0	0	0
					SEM	0	0	0	0
29	[-100,100]	2	Bohachevsky2	0	**Ort.**	0.06829	0	0	0
					Std.Sap.	0.078216	0	0	0
					SEM	0.014280	0	0	0
30	[-100,100]	2	Bohachevsky3	0	**Ort.**	0	0	0	0
					Std.Sap.	0	0	0	0
					SEM	0	0	0	0
31	[-10,10]	2	Shubert	-186.73	**Ort.**	-186.731	-186.73091	-186.7309	-186.73091
					Std.Sap.	0	0	0	0
					SEM	0	0	0	0
32	[-2,2]	2	GoldStein-Price	3	**Ort.**	5.250611	3	3	3
					Std.Sap.	5.870093	0	0	0
					SEM	1.071727	0	0	0

Tablo 4.16. (devamı)

No	Aralık	D	Fonksiyon	Min.		GA	PSO	DE	ABC
33	[-5,5]	4	Kowalik	0.00031	**Ort.**	0.005615	0.00049062	0.0004266	0.0004266
					Std.Sap.	0.008171	0.000366	0.000273	6.04E-5
					SEM	0.001492	6.68E-05	4.98E-05	1.10E-05
34	[0,10]	4	Shekel5	-10.15	**Ort.**	-5.66052	-2.0870079	-10.1532	-10.1532
					Std.Sap.	3.866737	1.178460	0	0
					SEM	0.705966	0.215156	0	0
35	[0,10]	4	Shekel7	-10.4	**Ort.**	-5.34409	-1.9898713	-10.40294	-10.402941
					Std.Sap.	3.517134	1.420602	0	0
					SEM	0.642138	0.259365	0	0
36	[0,10]	4	Shekel10	-10.53	**Ort.**	-3.82984	-1.8796753	-10.53641	-10.53641
					Std.Sap.	2.451956	0.432476	0	0
					SEM	0.447664	0.078959	0	0
37	[-D,D]	4	Perm	0	**Ort.**	0.302671	0.03605158	0.0240069	0.0411052
					Std.Sap.	0.193254	0.048927	0.046032	0.023056
					SEM	0.035283	0.008933	0.008404	0.004209
38	[0,D]	4	PowerSum	0	**Ort.**	0.010405	11.3904479	0.0001425	0.0029468
					Std.Sap.	0.009077	7.355800	0.000145	0.002289
					SEM	0.001657	1.342979	2.65E-05	0.000418
39	[0,1]	3	Hartman3	-3.86	**Ort.**	-3.86278	-3.6333523	-3.862782	-3.8627821
					Std.Sap.	0	0.116937	0	0
					SEM	0	0.021350	0	0
40	[0,1]	6	Hartman6	-3.32	**Ort.**	-3.29822	-1.8591298	-3.226881	-3.3219952
					Std.Sap.	0.050130	0.439958	0.047557	0
					SEM	0.009152	0.080325	0.008683	0

Tablo 4.16. (devamı)

No	Aralık	D	Fonksiyon	Min.		GA	PSO	DE	ABC
41	[-600,600]	30	Griewank	0	**Ort.**	10.63346	0.01739118	0.0014792	0
					Std.Sap.	1.161455	0.020808	0.002958	0
					SEM	0.212052	0.003799	0.000540	0
42	[-32,32]	30	Ackley	0	**Ort.**	14.67178	0.16462236	0	0
					Std.Sap.	0.178141	0.493867	0	0
					SEM	0.032524	0.090167	0	0
43	[-50,50]	30	Penalized	0	**Ort.**	13.3772	0.0207338	0	0
					Std.Sap.	1.448726	0.041468	0	0
					SEM	0.2645	0.007571	0	0
44	[-50,50]	30	Penalized2	0	**Ort.**	125.0613	0.00767535	0.0021975	0
					Std.Sap.	12.001204	0.016288	0.004395	0
					SEM	2.191110	0.002974	0.000802	0
45	[0,10]	2	Langerman2	-1.08	**Ort.**	-1.08094	-0.679268	-1.080938	-1.0809384
					Std.Sap.	0	0.274621	0	0
					SEM	0	0.050139	0	0
46	[0,10]	5	Langerman5	-1.5	**Ort.**	-0.96842	-0.5048579	-1.499999	-0.938150
					Std.Sap.	0.287548	0.213626	0	0.000208
					SEM	0.052499	0.039003	0	3.80E-05
47	[0,10]	10	Langerman10	NA	**Ort.**	-0.63644	-0.0025656	-1.0528	-0.4460925
					Std.Sap.	0.374682	0.003523	0.302257	0.133958
					SEM	0.068407	0.000643	0.055184	0.024457

Tablo 4.16. (devamı)

No	**Aralık**	**D**	**Fonksiyon**	**Min.**		**GA**	**PSO**	**DE**	**ABC**
48	[-π,π]	2	FletcherPowell2	0	**Ort.**	0	0	0	0
					Std.Sap.	0	0	0	0
					SEM	0	0	0	0
49	[-π,π]	5	FletcherPowell5	0	**Ort.**	0.004303	1457.88344	5.988783	0.1735495
					Std.Sap.	0.009469	1269.362389	7.334731	0.068175
					SEM	0.001729	231.752805	1.339133	0.012447
50	[-π,π]	10	FletcherPowell10	0	**Ort.**	29.57348	1364.45555	781.55028	8.2334401
					Std.Sap.	16.021078	1325.379655	1048.813487	8.092742
					SEM	2.925035	1325.379655	241.980111	1.477526

Tablo 4.17. GA ve ABC algoritması için anlamlı farklılık testi t: student t-testine ait t-değeri, SED: Farkların standart hatası, p: t-değerine karşılık gelen p-değeri, R: p-değerinin rankı, I.R.:p-değerinin invers rankı, Sign: Farklılık.

No	**Fonksiyon**	**t**	**SED**	**p**	**R.**	**I.R.**	**Yeni α**	**Sign.**
42	Ackley	451.107	0.033	0	1	50	0.001	ABC
2	Step	83.7021	13.978	0	2	49	0.00102	ABC
3	sphere	81.921	13.55	0	3	48	0.001042	ABC
4	SumSquares	65.3244	2.266	0	4	47	0.001064	ABC
22	Rastrigin	63.5001	0.833	0	5	46	0.001087	ABC
23	Schwefel	57.3298	17.026	0	6	45	0.001111	ABC
44	Penalized2	57.0767	2.191	0	7	44	0.001136	ABC
43	Penalized	50.5754	0.264	0	8	43	0.001163	ABC
41	Griewank	50.1456	0.212	0	9	42	0.00119	ABC
14	Schwefel2.22	43.5277	0.253	0	10	41	0.00122	ABC
15	Schwefel1.2	35.5539	208.135	0	11	40	0.00125	ABC
13	Powell	34.3237	0.283	0	12	39	0.001282	ABC
5	Quartic	29.9584	0.005	0	13	38	0.001316	ABC
16	Rosenbrock	27.884	7029.106	0	14	37	0.001351	ABC
17	Dixon-Price	25.1211	48.565	0	15	36	0.001389	ABC
10	Trid6	24.3432	0	0	16	35	0.001429	ABC
12	Zakharov	15.8047	0.001	0	17	34	0.001471	ABC
36	Shekel10	14.9813	0.448	0	18	33	0.001515	ABC
11	Trid10	14.8387	0.035	0	19	32	0.001563	ABC
49	FletcherPowell5	13.4681	0.013	0	20	31	0.001613	GA
35	Shekel7	7.8781	0.642	0	21	30	0.001667	ABC
37	Perm	7.683	0.036	0	22	29	0.001724	ABC
50	FletcherPowell10	6.512	3.277	0	23	28	0.001786	ABC
34	Shekel5	6.3639	0.706	0	24	27	0.001852	ABC
26	Michalewicz10	6.3391	0.026	0	25	26	0.001923	ABC
27	Schaffer	5.1869	0.008	0.000003	26	25	0.002	ABC
29	Bohachevsky2	4.7821	0.014	0.000001	27	24	0.002083	ABC
38	PowerSum	4.3638	0.002	0.000053	28	23	0.002174	ABC
9	Colville	4.3138	0.018	0.000063	29	22	0.002273	GA
33	Kowalik	3.4778	0.001	0.000965	30	21	0.002381	ABC
47	Langerman10	2.6201	0.073	0.011202	31	20	0.0025	–
40	Hartman6	2.5977	0.009	0.011876	32	19	0.002632	–
25	Michalewicz5	2.3973	0.018	0.019758	33	18	0.002778	–
32	GoldStein-Price	2.1	1.072	0.040089	34	17	0.002941	–
46	Langerman5	0.5766	0.052	0.5644	35	16	0.003125	–
1	Stepint	-Inf	0	1	36	15	0.003333	–
6	Beale	-Inf	0	1	37	14	0.003571	–
7	Easom	-Inf	0	1	38	13	0.003846	–
8	Matyas	-Inf	0	1	39	12	0.004167	–
18	Foxholes	-Inf	0	1	40	11	0.004545	–
19	Branin	-Inf	0	1	41	10	0.005	–
20	Bohachevsky1	-Inf	0	1	42	9	0.005556	–
21	Booth	-Inf	0	1	43	8	0.00625	–
24	Michalewicz2	-Inf	0	1	44	7	0.007143	–
28	6HumpCamelBack	-Inf	0	1	45	6	0.008333	–
30	Bohachevsky3	-Inf	0	1	46	5	0.01	–
31	Shubert	-Inf	0	1	47	4	0.0125	–
39	Hartman3	-Inf	0	1	48	3	0.016667	–
45	Langerman	-Inf	0	1	49	2	0.025	–
48	FletcherPowell2	-Inf	0	1	50	1	0.05	–

Tablo 4.18. PSO ve ABC algoritmaları için anlamlı farklılık testi t: student t-testine ait t-değeri, SED: Farkların standart hatası, p: t-değerine karşılık gelen p-değeri, R: p-değerinin rankı, I.R.:p-değerinin invers rankı, Sign: Farklılık.

No	**Fonksiyon**	**t**	**SED**	**p**	**R.**	**I.R.**	**Yeni α**	**Sign.**
17	Dixon-Price	3.65E+08	0	0	1	50	0.001	ABC
23	Schwefel	67.6984	83.611	0	2	49	0.00102	ABC
26	Michalewicz10	61.6014	0.092	0	3	48	0.001042	ABC
25	Michalewicz5	46.827	0.047	0	4	47	0.001064	ABC
34	Shekel5	37.4899	0.215	0	5	46	0.001087	ABC
36	Shekel10	33.3766	0.259	0	6	45	0.001111	ABC
35	Shekel7	32.4372	0.259	0	7	44	0.001136	ABC
5	Quartic	32.433	0.001	0	8	43	0.001163	PSO
13	Powell	31.3832	0	0	9	42	0.00119	PSO
22	Rastrigin	20.5371	2.141	0	10	41	0.00122	ABC
40	Hartman6	18.2118	0.08	0	11	40	0.00125	ABC
47	Langerman10	18.1285	0.024	0	12	39	0.001282	ABC
46	Langerman5	11.1093	0.03	0	13	38	0.001316	ABC
39	Hartman3	10.7463	0.021	0	14	37	0.001351	ABC
24	Michalewicz2	10.4387	0.022	0	15	36	0.001389	ABC
38	PowerSum	8.4793	1.343	0	16	35	0.001429	ABC
45	Langerman2	8.0112	0.05	0	17	34	0.001471	ABC
9	Colville	7.683	0.012	0	18	33	0.001515	PSO
12	Zakharov	7.4107	0	0	19	32	0.001563	PSO
49	FletcherPowell5	6.2899	231.753	0	20	31	0.001613	ABC
50	FletcherPowell10	5.6046	241.985	0	21	30	0.001667	ABC
41	Griewank	4.5778	0.004	0.000025	22	29	0.001724	ABC
16	Rosenbrock	3.3991	4.413	0.001228	23	28	0.001786	ABC
43	Penalized	2.7386	0.008	0.008182	24	27	0.001852	–
44	Penalized2	2.581	0.003	0.012403	25	26	0.001923	–
42	Ackley	1.8257	0.09	0.073045	26	25	0.002	–
33	Kowalik	0.9453	0	0.34843	27	24	0.002083	–
37	Perm	0.5118	0.001	0.61073	28	23	0.002174	–
1	Stepint	-Inf	0	1	29	22	0.002273	–
2	Step	-Inf	0	1	30	21	0.002381	–
3	sphere	-Inf	0	1	31	20	0.0025	–
4	SumSquares	-Inf	0	1	32	19	0.002632	–
6	Beale	-Inf	0	1	33	18	0.002778	–
7	Easom	-Inf	0	1	34	17	0.002941	–
8	Matyas	-Inf	0	1	35	16	0.003125	–
10	Trid6	-Inf	0	1	36	15	0.003333	–
11	Trid10	-Inf	0	1	37	14	0.003571	–
14	Schwefel2.22	-Inf	0	1	38	13	0.003846	–
15	Schwefel1.2	-Inf	0	1	39	12	0.004167	–
18	Foxholes	-Inf	0	1	40	11	0.004545	–
19	Branin	-Inf	0	1	41	10	0.005	–
20	Bohachevsky1	-Inf	0	1	42	9	0.005556	–
21	Booth	-Inf	0	1	43	8	0.00625	–
27	Schaffer	-Inf	0	1	44	7	0.007143	–
28	6HumpCamelBack	-Inf	0	1	45	6	0.008333	–
29	Bohachevsky2	-Inf	0	1	46	5	0.01	–
30	Bohachevsky3	-Inf	0	1	47	4	0.0125	–
31	Shubert	-Inf	0	1	48	3	0.016667	–
32	GoldStein-Price	-Inf	0	1	49	2	0.025	–
48	FletcherPowell2	-Inf	0	1	50	1	0.05	–

Tablo 4.19. DE ve ABC algoritmaları için anlamlı farklılık testi t: student t-testine ait t-değeri, SED: Farkların standart hatası, p: t-değerine karşılık gelen p-değeri, R: p-değerinin rankı, I.R.:p-değerinin invers rankı, Sign: Farklılık.

No	**Fonksiyon**	**t**	**SED**	**p**	**R.**	**I.R.**	**Yeni α**	**Sign.**
17	Dixon-Price	3651483719	0	0	1	50	0.001	ABC
46	Langerman5	14795.0659	0	0	2	49	0.00102	DE
13	Powell	34.1285	0	0	3	48	0.001042	DE
5	Quartic	32.1347	0.001	0	4	47	0.001064	DE
22	Rastrigin	25.284	0.463	0	5	46	0.001087	ABC
23	Schwefel	24.1769	95.276	0	6	45	0.001111	ABC
16	Rosenbrock	19.6993	0.92	0	7	44	0.001136	ABC
40	Hartman6	10.9545	0.009	0	8	43	0.001163	ABC
47	Langerman10	10.0513	0.06	0	9	42	0.00119	DE
38	PowerSum	6.6968	0	0	10	41	0.00122	DE
26	Michalewicz10	5.8863	0.012	0	11	40	0.00125	ABC
49	FletcherPowell5	4.3424	1.339	0.000057	12	39	0.001282	ABC
50	FletcherPowell10	4.0384	191.492	0.00016	13	38	0.001316	ABC
41	Griewank	2.739	0.001	0.008173	14	37	0.001351	–
44	Penalized2	2.7386	0.001	0.008182	15	36	0.001389	–
9	Colville	2.7046	0.019	0.008961	16	35	0.001429	–
25	Michalewicz5	1.8257	0.002	0.073045	17	34	0.001471	–
37	Perm	1.8191	0.009	0.074059	18	33	0.001515	–
33	Kowalik	0	0	1	19	32	0.001563	–
1	Stepint	-Inf	0	1	20	31	0.001613	–
2	Step	-Inf	0	1	21	30	0.001667	–
3	sphere	-Inf	0	1	22	29	0.001724	–
4	SumSquares	-Inf	0	1	23	28	0.001786	–
6	Beale	-Inf	0	1	24	27	0.001852	–
7	Easom	-Inf	0	1	25	26	0.001923	–
8	Matyas	-Inf	0	1	26	25	0.002	–
10	Trid6	-Inf	0	1	27	24	0.002083	–
11	Trid10	-Inf	0	1	28	23	0.002174	–
12	Zakharov	-Inf	0	1	29	22	0.002273	–
14	Schwefel2.22	-Inf	0	1	30	21	0.002381	–
15	Schwefel1.2	-Inf	0	1	31	20	0.0025	–
18	Foxholes	-Inf	0	1	32	19	0.002632	–
19	Branin	-Inf	0	1	33	18	0.002778	–
20	Bohachevsky1	-Inf	0	1	34	17	0.002941	–
21	Booth	-Inf	0	1	35	16	0.003125	–
24	Michalewicz2	-Inf	0	1	36	15	0.003333	–
27	Schaffer	-Inf	0	1	37	14	0.003571	–
28	SixHumpCamelBack	-Inf	0	1	38	13	0.003846	–
29	Bohachevsky2	-Inf	0	1	39	12	0.004167	–
30	Bohachevsky3	-Inf	0	1	40	11	0.004545	–
31	Shubert	-Inf	0	1	41	10	0.005	–
32	GoldStein-Price	-Inf	0	1	42	9	0.005556	–
34	Shekel5	-Inf	0	1	43	8	0.00625	–
35	Shekel7	-Inf	0	1	44	7	0.007143	–
36	Shekel10	-Inf	0	1	45	6	0.008333	–
39	Hartman3	-Inf	0	1	46	5	0.01	–
42	Ackley	-Inf	0	1	47	4	0.0125	–
43	Penalized	-Inf	0	1	48	3	0.016667	–
45	Langerman2	-Inf	0	1	49	2	0.025	–
48	FletcherPowell2	-Inf	0	1	50	1	0.05	–

Tablo 4.20. Çalışma 2'de kullanılan test fonksiyonları, D:Problemin boyutu, C:Karakteristik, U:Tek modlu, M:Çok modlu, S:Ayrıştırılabilir,N:Ayrıştırılamaz.

No	**Aralık**	**D**	**C**	**Fonksiyon**	**Formul**
1	[-100,100]	30	US	Sphere	$f(x) = \sum_{i=1}^{n} x_i^2$
2	[-10,10]	30	UN	Schwefel 2.22	$f(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$
3	[-100,100]	30	UN	Schwefel 1.2	$f(x) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$
4	[-100,100]	30	UN	Schwefel 2.21	$f(x) =$
5	[-30,30]	30	UN	Rosenbrock	$f(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$
6	[-100,100]	30	US	Step	$f(x) = \sum_{i=1}^{n} (\lfloor x_i + 0.5 \rfloor)^2$
7	[-1.28,1.28]	30	US	Quartic	$f(x) = \sum_{i=1}^{n} i x_i^4 + random[0,1)$
8	[-500,500]	30	MS	Schwefel	$f(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert})$
9	[-5.12,5.12]	30	MS	Rastrigin	$f(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$
10	[-32,32]	30	MN	Ackley	$f(x) = -20 \exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i)\right) +20 + e$
11	[-600,600]	30	MN	Griewank	$f(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$

Tablo 4.20. (devamı)

No	**Aralık**	**D**	**C**	**Fonksiyon**	**Formul**
12	[-50,50]	30	MN	Penalized	$f(x) = \frac{\pi}{n}\left\{\begin{array}{l} 10\sin^2(\pi y_1) \\ +\sum_{i=1}^{n-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})] \\ +(y_n-1)^2 \end{array}\right\}$ $+\sum_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{u(x_i,10,100,4)}$ $y_i = 1+\frac{1}{4}(x_i+1)$ $u(x_i,a,k,m) = \left\{\begin{array}{ccc} k(x_i-a)^m & , & x_i > a \\ 0 & , & -a \le x_i \le a \\ k(-x_i-a)^m & , & x_i < -a \end{array}\right.$
13	[-50,50]	30	MN	Penalized2	$f(x) = 0.1\left\{\begin{array}{l} \sin^2(\pi x_1)+ \\ \sum_{i=1}^{n-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})]+ \\ (x_n-1)^2[1+\sin^2(2\pi x_n)] \end{array}\right\}+$ $\sum_{i=1}^{n} u(x_i,5,100,4)$
14	[-65.536,65.536]	2	MS	Foxholes	$f(x) = \left[\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})6}\right]^{-1}$
15	[-5,5]	4	MN	Kowalik	$f(x) = \sum_{i=1}^{11}\left(a_i - \frac{x_1(b_i^2+b_i x_2)}{b_i^2+b_i x_3+x_4}\right)^2$
16	[-5,5]	2	MN	Six Hump Camel Back	$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x1x2 - 4x_2^2 + 4x_2^4$
17	[-5,10]x[0,15]	2	MS	Branin	$f(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2$ $+10\left(1-\frac{1}{8\pi}\right)cosx_1 + 10$

Tablo 4.20. (devamı)

No	**Aralık**	**D**	**C**	**Fonksiyon**	**Formul**
18	[-2,2]	2	MN	GoldStein-Price	$f(x) = \left[1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right] \left[30 + (2x_1 - 3x_2)^2 (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right]$
19	[0,1]	3	MN	Hartman 3	$f(x) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right]$
20	[0,1]	6	MN	Hartman 6	$f(x) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right]$
21	[0,10]	4	MN	Shekel 5	$f(x) = -\sum_{i=1}^{5} \left[(x - a_i)(x - a_i)^T + c_i\right]^{-1}$
22	[0,10]	4	MN	Shekel 7	$f(x) = -\sum_{i=1}^{7} \left[(x - a_i)(x - a_i)^T + c_i\right]^{-1}$
23	[0,10]	4	MN	Shekel 10	$f(x) = -\sum_{i=1}^{10} \left[(x - a_i)(x - a_i)^T + c_i\right]^{-1}$

Tablo 4.21. CES [3], FES [4], ESLAT [3], CMA-ES [3] ve ABC Algoritmalarının sonuçları. D:Boyut, Ort:Ortalama, SD:Standart Sapma

No	**Fonksiyon**	**Aralık**	**D**	**CES**		**FES**		**ESLAT**		**CMA-ES**		**ABC**	
				Ort	**SD**	**Ort**	**SD**	**Ort**	**SD**	**Ort**	**SD**	**Ort**	**SD**
1	Sphere	-100,100	30	1.7e-26	1.1e-25	2.5e-4	6.8e-5	2.0e-17	2.9e-17	9.7e-23	3.8e-23	7.57E-04	2.48E-04
2	Schwefel 2.22	-10,10	30	8.1e-20	3.6e-19	6.0e-2	9.6e-3	3.8e-5	1.6e-5	4.2e-11	7.1e-23	8.95E-04	1.27E-04
3	Schwefel 1.2	-100,100	30	337.62	117.14	1.4e-3	5.3e-4	6.1e-6	7.5e-6	7.1e-23	2.9e-23	7.01E-04	2.78E-04
4	Schwefel 2.21	-100,100	30	2.41	2.15	5.5e-3	6.5e-4	0.78	1.64	5.4e-12	1.5e-12	2.72E+00	1.18E+00
5	Rosenbrock	-30,30	30	27.65	0.51	33.28	43.13	1.93	3.35	0.4	1.2	9.36E-01	1.76E+00
6	Step	-100,100	30	0	0	0	0	2.0e-2	0.14	1.44	1.77	0.00E+00	0.00E+00
7	Quartic	-1.28,1.28	30	4.7e-2	0.12	1.2e-2	5.8e-3	0.39	0.22	0.23	8.7e-2	9.06E-02	1.89E-02
8	Schwefel	-500,500	30	-8.00E+93	4.9e+94	-12556.4	32.53	-2.3e+15	5.70E+15	-7637.14	895.6	-12563.67335	2.36E+01
9	Rastrigin	-5.12,5.12	30	13.38	43.15	0.16	0.33	4.65	5.67	51.78	13.56	4.66E-04	3.44E-04
10	Ackley	-32,32	30	6.0e-13	1.7e-12	1.2e-2	1.8e-3	1.8e-8	5.4e-9	6.9e-12	1.3e-12	7.81E-04	1.83E-04
11	Griewank	-600,600	30	6.0e-14	4.2e-13	3.7e-2	5.0e-2	1.4e-3	4.7e-3	7.4e-4	2.7e-3	8.37E-04	1.38E-03
12	Penalized	-50,50	30	1.46	3.17	2.8e-6	8.1e-7	1.5e-12	2.0e-12	1.2e-4	3.4e-2	6.98E-04	2.78E-04
13	Penalized 2	-50,50	30	2.40	0.13	4.7e-5	1.5e-5	6.4e-3	8.9e-3	1.7e-3	4.5e-3	7.98E-04	2.13E-04
14	Foxholes	-65.536,65.536	2	2.20	2.43	1.20	0.63	1.77	1.37	10.44	6.87	9.98E-01	3.21E-04
15	Kowalik	-5,5	4	1.3e-3	6.3e-4	9.7e-4	4.2e-4	8.1e-4	4.1e-4	1.5e-3	4.2e-3	1.18E-03	1.45E-04
16	SixHump	-5,5	2	-1.0310	1.2e-3	-1.0316	6.0e-7	-1.0316	9.7e-14	-1.0316	7.7e-16	-1.031	3.04E-04
17	Branin	(-5,10),(0,15)	2	0.401	3.6e-3	0.398	6.0e-8	0.398	1.0e-13	0.398	1.4e-15	0.3985	3.27E-04
18	Goldstein-Price	-2,2	2	3.007	1.2e-2	3.0	0	3	5.8e-14	14.34	25.05	3.000E+00	3.09E-04
19	Hartman 3	0,1	3	-3.8613	1.2e-3	-3.86	4.0e-3	-3.8628	2.9e-13	-3.8628	4.8e-16	-3.862E+00	2.77E-04
20	Hartman 6	0,1	6	-3.24	5.8e-2	-3.23	0.12	-3.31	3.3e-2	-3.28	5.8e-2	-3.322E+00	1.35E-04
21	Shekel 5	0,10	4	-5.72	2.62	-5.54	1.82	-8.49	2.76	-5.86	3.60	-10.151	1.17E-02
22	Shekel 7	0,10	4	-6.09	2.63	-6.76	3.01	-8.79	2.64	-6.58	3.74	-10.402	3.11E-04
23	Shekel 10	0,10	4	-6.42	2.67	-7.63	3.27	-9.65	2.06	-7.03	3.74	-10.535	2.02E-03

Tablo 4.22. CES [3], FES [4], ESLAT [3], CMA-ES [3] ve ABC Algoritmalarının değerlendirme sayısına göre hesaplanan çözüm maliyetleri

No	**Fonksiyon**	**CES**	**FES**	**ESLAT**	**CMA-ES**	**ABC**	
		Ort.	Ort.	Ort.	Ort.	Ort.	StdFE
1	Sphere	69724	150000	69724	10721	9264	1481
2	Schwefel 2.22	60859	200000	60859	12145	12991	673
3	Schwefel 1.2	72141	500000	72141	21248	12255	1390
4	Schwefel 2.21	69821	500000	69821	20813	100000	0
5	Rosenbrock	66609	1500000	66609	55821	100000	0
6	Step	57064	150000	57064	2184	4853	1044
7	Quartic	50962	300000	50962	667131	100000	0
8	Schwefel	61704	900000	61704	6621	64632	23897
9	Rastrigin	53880	500000	53880	10079	26731	9311
10	Ackley	58909	150000	58909	10654	16616	1201
11	Griewank	71044	200000	71044	10522	36151	17128
12	Penalized	63030	150000	63030	13981	7340	2020
13	Penalized 2	65655	150000	65655	13756	8454	1719
14	Foxholes	1305	10000	1305	540	1046	637
15	Kowalik	2869	400000	2869	13434	6120	4564
16	SixHump	1306	10000	1306	619	342	109
17	Branin	1257	10000	1257	594	530	284
18	Goldstein-Price	1201	10000	1201	2052	15186	13500
19	Hartman 3	1734	10000	1734	996	4747	16011
20	Hartman 6	3816	20000	3816	2293	1583	457
21	Shekel 5	2338	10000	2338	1246	6069	13477
22	Shekel 7	2468	10000	2468	1267	7173	9022
23	Shekel 10	2410	10000	2410	1275	15392	24413

Tablo 4.23. ESLAT [3], CMA-ES [3] and ABC Algoritmalarının başarı oranları(%). Koyu renkle yazılanlar en iyi sonucu göstermektedir.

No	**Fonksiyon**	**ESLAT**	**CMA-ES**	**ABC**
1	Sphere	**100**	**100**	**100**
2	Schwefel 2.22	**100**	**100**	**100**
3	Schwefel 1.2	**100**	**100**	**100**
4	Schwefel 2.21	0	**100**	0
5	Rosenbrock	70	**90**	0
6	Step	98	36	**100**
7	Quartic	0	0	0
8	Schwefel	0	0	**86**
9	Rastrigin	40	0	**100**
10	Ackley	**100**	**100**	**100**
11	Griewank	90	92	**96**
12	Penalized	**100**	88	**100**
13	Penalized 2	60	86	**100**
14	Foxholes	60	0	**100**
15	Kowalik	94	88	**100**
16	SixHump	**100**	**100**	**100**
17	Branin	**100**	**100**	**100**
18	Goldstein-Price	**100**	78	**100**
19	Hartman 3	**100**	**100**	**100**
20	Hartman 6	94	48	**100**
21	Shekel 5	72	40	**98**
22	Shekel 7	72	48	**100**
23	Shekel 10	84	52	**96**
	Toplam:	9	9	20

Tablo 4.24. Çalışma 3'de kullanılan test fonksiyonları. D:Boyut, C:Karakteristik, U:Tek modlu, M:Çok modlu, S:Ayrıştırılabilir, N:Ayrıştırılamaz

No	**Aralık**	**D**	**C**	**Fonksiyon**	**Formülasyon**
1	[-2,2]	2	UN	Modified Rosenbrock	$f(x,y) = 74 + 100(y-x^2)^2 + (1-x)^2 - 400e^{-((x+1)^2+(y+1)^2/0.1)}$
2	[-100,100]	2	MN	Griewank	$f(x,y) = 1 + \frac{1}{200}(x^2+y^2) - \cos(x)\cos(\frac{y}{\sqrt{2}})$
3	[-5.12,5.12]	2	MS	Rastrigin	$f(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$

Tablo 4.25. SOM-ES [5], NG-ES [5], CMA-ES [5] ve ABC Algoritmalarının istatistiksel sonuçları, Ort. Değ.: Ortalama Değerlendirme Sayısı

	Modified Rosenbrock		**Griewank**		**Rastrigin**	
Metot	**Ort. Değ.**	**% Başarı**	**Ort. Değ.**	**% Başarı.**	**Ort. Değ.**	**% Başarı**
5x5 SOM-ES	1600±200	%70±8	130±40	%90±7	180±50	%90±7
7x7 SOM-ES	750±90	%90±5	100±40	%90±8	200±50	%90±8
NG-ES (m=10)	1700±200	%90±7	180±50	%90±10	210±50	%90±8
NG-ES (m=20)	780±80	%90±9	150±40	%90±8	180±40	%90±7
CMA-ES	70±40	%30±10	210±50	%70±10	100±40	%80±9
ABC	1371±2678	%52±5	1121±960	%99±1	1169±446	%100±0

4.2. Tam Sayılı Test Fonksiyonları Üzerinde Çalışmalar

Tam sayı programlama, tüm optimizasyon parametre değerlerinin tam sayı olması gerektiği bir çeşit lineer programlama problemidir. Bütçe analizleri, portföy analizleri, ağ yapılandırması, VLSI devre tasarımı, üretim sistemleri gibi bir çok uygulamada tam sayı programlama karşımıza çıkmaktadır [7, 207].

Tam sayı programlama problemi Z tam sayı kümesi ve $\mathcal{S}$ kabul edilebilir (feasible) çözüm bölgesi olmak üzere, değişkenlerin (4.5)'de tanımlı olduğu gibi kabul edilebilir bölgede tam sayı değerler alabildiği problemdir:

$$min\ f(x),\ x \in \mathcal{S} \subseteq Z^n \tag{4.5}$$

Bir problemin bazı parametreleri tam sayı, bazıları değil ise bu tür probleme karışık tam sayı programlama problemi (mixed integer programming problem) denir. Parametrelerin sadece 0 ve 1 tam sayı değerlerini alabildiği problemlere ise 0-1 programlama problemleri yada ikili tam sayı programlama problemleri denilmektedir.

Tam sayı programlama teknikleri kesin, yakınsak ve sezgisel teknikler olmak üzere üç ana kategoride incelenebilir [208]. Yakınsak tekniklerde parametreler tam sayı olmayan değerler cinsinden bulunur ve tam sayı değerlere yuvarlanırlar. Kesin tekniklerde; dal ve sınır (branch and bound, BB), dinamik programlama gibi, kabul edilebilir bölge alt bölgelere bölünür ve problem alt problemlere parçalanır.

Ancak ağaç yapısı üzerinde yüzlerce, binlerce noktanın durumu analiz edildiğinden hesaplama maliyetleri oldukça fazladır [208, 209]. Dal ve sınır metodu, lineer olmak zorunda olmayan amaç fonksiyonu ve lineer kısıtlamalara tabidir [208]. Bu metodun dezavantajı kabul edilebilir uzayın uygun şekilde bölümlenmesinin gerekliliğidir. Sezgisel teknikler ise kabul edilebilir zaman içerisinde optimum çözüme yakın çözümler üretmektedirler. Bu tür metodlar da her adımda üretilen parametrelerin tam sayıya yuvarlanması veya kesilmesi ile çözümler üretmektedirler.

Bu kısımda yapılan çalışmalarda, PSO algoritması ve türevi Quantum davranışlı PSO (QPSO), BB metodu ve ABC algoritması tam sayı programlama problemlerine uygulanarak performansları incelenecektir. PSO ve ABC algoritmaları üretilen çözüm değelerini tamsayıya yuvarlayarak çalışan sezgisel teknikler iken BB metodu kesin teknikler kategorisindedir. PSO, QPSO ve BB metotları ile ilgili açıklamalar EK-1 bölümünde verilmektedir. Çalışmada kullanılan test problemleri literatürde oldukça sık kullanılan tam sayı problemleridir. Bunlar aşağıda listelenmektedir:

Test Problemi 1 [210]:

$F_1(x) = \|x\|_1 = |x_1| + \ldots + |x_D|$.

$x = (x_1, \ldots, x_D) \in [-100, 100]^D$ Çözüm: $x_i^* = 0$, $i = 1, \ldots, D$ ve $F_1(x^*) = 0$.

Test Problemi 2 [210]:

$$F_2(x) = x^\top x = (x_1 \quad \ldots \quad x_D) \begin{pmatrix} x_1 \\ \vdots \\ x_D \end{pmatrix}.$$

$x = (x_1, \ldots, x_D)^\top \in [-100, 100]^D$ Çözüm: $x_i^* = 0$, $i = 1, \ldots, D$ ve $F_2(x^*) = 0$.

Test Problemi 3 [211]:

$$F_3(x) = -(\ 15 \quad 27 \quad 36 \quad 18 \quad 12\ )x + x^\top \begin{pmatrix} 35 & -20 & -10 & 32 & -10 \\ -20 & 40 & -6 & -31 & 32 \\ -10 & -6 & 11 & -6 & -10 \\ 32 & -31 & -6 & 38 & -10 \\ -10 & 32 & -10 & -20 & 31 \end{pmatrix} x.$$

Bilinen en iyi çözümler: $x^* = (0, 11, 22, 16, 6)^\top$ ve $x^* = (0, 12, 23, 17, 6)^\top$ ve $F_3(x) = -737$.

Test Problemi 4 [211]:

$F_4(x) = (9x_1^2 + 2x_2^2 - 11)^2 + (3x_1 + 4x_2^2 - 7)^2$. Çözüm: $x^* = (1, 1)^\top$ ve $F_4(x^*) = 0$.

Test Problemi 5 [211]:

$F_5(x) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4$. Çözüm: $x^* = (0, 0, 0, 0)^\top$ ve $F_5(x^*) = 0$.

Test Problemi 6 [212]:

$F_6(x) = 2x_1^2 + 3x_2^2 + 4x_1x_2 - 6x_1 - 3x_2$. Çözüm: $x^* = (2, -1)^\top$ ve $F_6(x^*) = -6$.

Test Problemi 7 [211]:

$F_7(x) = -3803.84 - 138.08x_1 - 232.92x_2 + 123.08x_1^2 + 203.64x_2^2 + 182.25x_1x_2$. Çözüm: $x^* = (0, 1)^\top$ ve $F_7(x^*) = -3833.12$.

Test Problemi 8 [211]:

$F_8(x) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$. Çözüm: $x^* = (3, 2)^\top$ ve $F_8(x^*) = 0$.

Test Problemi 9 [211]:

$F_9(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$. Çözüm: $x^* = (1, 1)^\top$ ve $F_9(x^*) = 0$.

4.2.1. Çalışma 1: Tam sayı programlama problemleri için QPSO, PSO-In ve ABC

Çalışma 1'de ABC algoritmasının performansı [6] numaralı referansdan alınan Quantum davranışlı PSO (QPSO) ve PSO-In algoritmasının sonuçları ile kıyaslanmıştır. PSO-In kısaltması PSO algoritmasının intertia parametreli ve

konstriksiyon parametresinin olmadığı versiyonu için kullanılmıştır. Inertia parametresi, ω, araştırma esnasında 1 den 0.4'e kadar azaltılmıştır. Yine, QPSO algoritmasına özgü kontrol parametresi β değeri de 1.2'den 0.4'e kadar azaltılmıştır [6]. ABC algoritmasının limit parametresi $SN * D * 5$, MR parametresi de 0.9 olarak alınmıştır. Her problem için seçilen maksimum iterasyon sayısı ve sürüdeki eleman sayısı Tablo 4.26'da verilmektedir. ABC ve QPSO algoritmaları ile tam sayı programlama problemleri için elde edilen sonuçları Tablo 4.26'da verilmektedir. Bu sonuçlar farklı rasgele değerler ile üretilen popülasyonla koşulan algoritmaların 50 koşmalarına ait başarı oranları (SR) ve arzu edilen sonucu yakalayabildiği iterasyon sayılarının ortalamaları (MI)'dır. [6] numaralı referansta doğruluk hassasiyeti verilmemiş ancak bu çalışmada hassasiyet değeri 10^{-6} alınmıştır. Bu oldukça yeterli bir hassasiyettir. Tablo 4.26'daki sonuçlardan, PSO, QPSO ve ABC algoritmalarının tüm fonksiyonlarda istenen hassasiyette verilen maksimum iterasyon sayısı içerisinde sonuca ulaştığı görülmektedir. Algoritmalar çözüme ulaştıkları ortalama iterasyon sayısı bakımından değerlendirilirse düşük boyutlu fonksiyonlar için $(F_2, F_3, F_4, F_5, F_8, F_9)$ benzer sonuçlar ürettikleri görülmektedir. Ancak yüksek boyutlu durumlarda ABC algoritmasının diğer iki algoritmaya göre daha iyi sonuçlar ürettiği gözlemlenmektedir. Buradan yola çıkarak tam sayı programlama problemleri üzerinde ABC'nin QPSO'ya benzer veya daha iyi, PSO-In algoritmasına göre de çok daha iyi sonuçlar verdiği sonucuna varılabilir. ABC algoritmasının yakınsama grafikleri Şekil 4.8 ve 4.9'de verilmiştir. Şekil 4.8'da, ABC algoritmasının Test Problem 1'in farklı boyutlarında yakınsama grafiği görülmektedir. Tüm durumlarda ABC algoritması yaklaşık 100 çevrim civarında optimuma erişmektedir. Figure 4.9'de Test Problemleri 2, 3, 4, 5, 8, 9'a ait yakınsama grafikleri verilmiştir.

4.2.2. Çalışma 2: Tam sayı programlama problemleri için PSO Türevleri, BB ve ABC

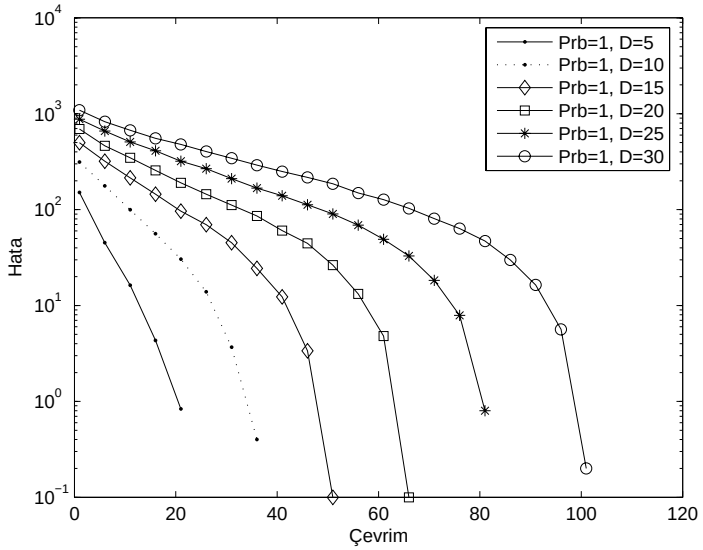
Tam sayı programlama problemleri için yapılan ikinci çalışmada ABC algoritmasının sonuçları [7] numaralı çalışmadan alınan PSO-In, PSO-Co, PSO-Bo ve BB tekniklerinin sonuçlarıyla kıyaslanmıştır. PSO-In konstriksiyon parametresi

olmayan ve inertia'nın olduğu, PSO-Co inertia olmayan ama konstriksiyon faktörü olan, PSO-Bo da hem inertia hem de konstriksiyon faktörü olan PSO varyantlarıdır. Bu kıyaslamalarda kullanılan algoritmalara ait sürülerdeki eleman sayıları ve problem boyutları Tablo 4.27'de verilmektedir. Sonlandırma kriteri olarak maksimum değerlendirme sayısının 25000'e ulaşması veya hatanın 10^{-6} değerine düşmesi kabul edilmiştir. Inertia ağırlığı ω=1 den 0.1'e dereceli olarak azaltılmış; konstriksiyon faktörünün değeri χ=0.729; $\phi_1 = \phi_2 = 2$; ve $\upsilon_{max} = 4$ olarak alınmıştır [7]. Algoritmalar 30 kez koşulmuş ABC'nin kontrol parametreleri Çalışma 1'de olduğu gibi seçilmiştir. PSO varyantlarının, BB'nin ve ABC'nin başarı oranları, değerlendirme sayılarının ortalaması, medianı ve standard sapması ile ilgili sonuçlardan Test Problem 1 için Tablo 4.28'de, $F_2 - F_7$ problemleri için Tablo 4.29'da verilmiştir. Tablo 4.28'de verilen sonuçlardan F_1 fonksiyonu üzerinde ABC algoritmasının ortalama, median ve standard sapma açısından diğer algoritmalara göre tüm boyutlarda daha iyi olduğu görülmektedir. Tablo 4.29'dan F_2 fonksiyonu üzerinde BB tekniğinin; F_3 üzerinde PSO-Co'nun; F_6'da PSO-In'in;F_4, F_5 ve F_7 fonksiyonlarında da ABC'nin en iyi performansa sahip olduğu oldukça açıktır.

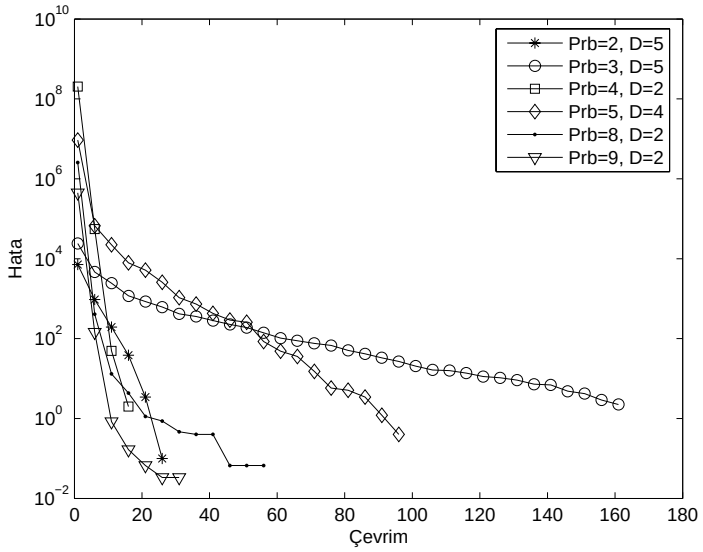
Bir çok ayrık problemin dönüştürülebileceği bir türden olan tam sayı programlama problemleri optimizasyon alanında oldukça önemli problemlerdir. ABC algoritması da bu çalışmada dikkate alınan tam sayı programlama problemleri üzerinde, PSO türevleri ve BB tekniğine nazaran kıyaslanabilir veya daha iyi performans sergilemiştir. ABC algoritması kesin teknikler kadar başarılı sonuçlar üretirken, onların bir dezavantajı olan hesaplama maliyetlerinin yüksekliği problemini ortadan kaldırmaktadır. ABC kendisinin de içinde bulunduğu sezgisel teknikler kategorisindeki diğer metotlara göre de yüksek boyutlu problemler üzerinde düşük hesaplama maliyetli yüksek performans sergilemektedir.

Tablo 4.26. PSO-In [6], QPSO [6] ve ABC algoritmalarının $F_1 - F_5$, F_8, F_9 fonksiyonları için maksimum iterasyon sayıları (MNI), sürüdeki eleman sayıları (SS), başarı oranları (SR), ortalama iterasyon sayıları (MI). Koyu renkle yazılanlar en iyi sonucu göstermektedir. D:Problemin boyutu

				PSO-In		QPSO		ABC	
SS	MNI	F	D	SR	MI	SR	MI	SR	MI
20	1000	F_1	5	100	72.8	100	27.2	100	**20.03**
50	1000	F_1	10	100	90.26	100	48.26	100	**34.63**
100	1000	F_1	15	100	94.36	100	64.46	100	**48.93**
200	1000	F_1	20	100	96.3	100	72.24	100	**64.43**
250	1500	F_1	25	100	99.44	100	83.86	100	**81.33**
300	2000	F_1	30	100	103.14	100	**92.56**	100	99.76
20	1000	F_2	5	100	77.82	100	**21.2**	100	21.93
150	1000	F_3	5	100	**125.03**	100	166.9	100	165.43
20	1000	F_4	2	100	32.9	100	**8.95**	100	11.5
40	1000	F_5	4	100	79.6	100	65.7	100	**60.1**
20	1000	F_8	2	100	81.24	100	**19.35**	100	19.4
50	1000	F_9	2	100	41.4	100	14.9	100	**11.93**



Şekil 4.8. ABC algoritmasının Test Problem 1'in farklı boyutları için logaritmik ölçekli yakınsama grafikleri



Şekil 4.9. ABC algoritmasının Test Problemleri 2-5,8,9'un farklı boyutları için logaritmik ölçekli yakınsama grafikleri

Tablo 4.27. Çalışma 2'de kullanılan fonksiyonların boyutları ve bu fonksiyonlar için sürü boyutları

Fonksiyon	Boyut	Sürü boyutu
F1	5	20
F1	10	20
F1	15	50
F1	20	50
F1	25	100
F1	30	100
F2	5	10
F3	5	70
F4	2	20
F5	4	20
F6	2	10
F7	2	20

Tablo 4.28. PSO varyantları [7], BB Tekniği [7] ve ABC algoritmalarının F_1 fonksiyonu için başarı oranları (SR), değerlendirme sayılarının ortalama, median ve standard sapmaları (Std). Koyu renkle yazılan sonuçlar en iyi sonucu göstermektedir.

Fonksiyon	Metod	SR	Ort	Std	Median
$F_1,\ D=5$	PSO-In	30/30	1646.0	661.5	1420
	PSO-Co	30/30	744.0	89.8	730
	PSO-Bo	30/30	692.6	97.2	680
	BB	30/30	1167.83	659.8	1166
	ABC	30/30	**376**	64.6	380
$F_1,\ D=10$	PSO-In	30/30	4652.0	483.2	4610
	PSO-Co	30/30	1362.6	254.7	1360
	PSO-Bo	30/30	1208.6	162.7	1230
	BB	30/30	5495.8	1676.3	5154
	ABC	30/30	**727.3**	64.4	740
$F_1,\ D=15$	PSO-In	30/30	7916.6	624.1	7950
	PSO-Co	30/30	3538.3	526.6	3500
	PSO-Bo	30/30	2860.0	220.2	2850
	BB	30/30	10177.1	2393.4	10011
	ABC	30/30	**974**	60.5	960
$F_1,\ D=20$	PSO-In	30/30	8991.6	673.3	9050
	PSO-Co	30/30	4871.6	743.3	4700
	PSO-Bo	29/30	4408.3	3919.4	3650
	BB	30/30	16291.3	3797.9	14550
	ABC	30/30	**1275.3**	97.7	1260
$F_1,\ D=25$	PSO-In	30/30	11886.6	543.7	11900
	PSO-Co	30/30	9686.6	960.1	9450
	PSO-Bo	25/30	9553.3	7098.6	6500
	BB	20/30	23689.7	2574.2	25043
	ABC	30/30	**1554.7**	108.6	1540
$F_1,\ D=30$	PSO-In	30/30	13186.6	667.8	13050
	PSO-Co	30/30	12586.6	1734.9	12500
	PSO-Bo	19/30	13660.0	8863.9	7500
	BB	14/30	25908.6	755.5	26078
	ABC	30/30	**1906**	129.9	1900

Tablo 4.29. PSO varyantları [7], BB Tekniği [7] ve ABC algoritmalarının F_2-F_7 fonksiyonları için başarı oranları (SR), değerlendirme sayılarının ortalama, median ve standard sapmaları (Std). Koyu renkle yazılan sonuçlar en iyi sonucu göstermektedir.

Fonksiyon	Metot	SR	Ort	Std	Median
F_2 D=5	PSO-In	30/30	1655.6	618.4	1650
	PSO-Co	30/30	428.0	57.9	430
	PSO-Bo	30/30	418.3	83.9	395
	BB	30/30	**139.7**	102.6	93
	ABC	30/30	449.3	56.7	440
F_3 D=5	PSO-In	30/30	4111.3	1186.7	3850
	PSO-Co	30/30	**2972.6**	536.4	2940
	PSO-Bo	30/30	3171.0	493.6	3080
	BB	30/30	4185.5	32.8	4191
	ABC	24/30	13850	6711.3	11550
F_4 D=2	PSO-In	30/30	304.0	101.6	320
	PSO-Co	30/30	297.3	50.8	290
	PSO-Bo	30/30	302.0	80.5	320
	BB	30/30	316.9	125.4	386
	ABC	30/30	**240.7**	79.4	240
F_5 D=4	PSO-In	30/30	1728.6	518.9	1760
	PSO-Co	30/30	1100.6	229.2	1090
	PSO-Bo	30/30	1082.0	295.6	1090
	BB	30/30	2754.0	1030.1	2714
	ABC	30/30	**193.3**	53.5	180
F_6 D=2	PSO-In	30/30	**178.0**	41.9	180
	PSO-Co	30/30	198.6	59.2	195
	PSO-Bo	30/30	191.0	65.9	190
	BB	30/30	211.1	15.0	209
	ABC	30/30	258.7	113.6	240
F_7 D=2	PSO-In	30/30	334.6	95.5	340
	PSO-Co	30/30	324.0	78.5	320
	PSO-Bo	30/30	306.6	96.7	300
	BB	30/30	358.6	14.7	355
	ABC	30/30	**106.7**	44.8	100

4.3. Karma Test Fonksiyonları

Bu bölümde ABC algoritması [213]'de tanımlanan gerçek-parametre optimizasyon problemlerine uygulanmıştır. Algoritmaların sahip oldukları kendilerine has bir takım özelliklerden dolayı bazı fonksiyonlar üzerinde iyi sonuç üretirken bazılarında istenen başarıyı yakalayamayabilirler. Şayet algoritma parametre kopyalanması yöntemiyle komşu çözüm üretiyorsa ve global optimum simetrik olarak eksenlere yerleşmişse araştırma hızlı bir şekilde yakınsar. Başka bir özel durum ise global optimumun orjinde olmasıdır. Bir algoritmanın bölgesel araştırma kabiliyeti iyi ise bu tür problemlerde araştırmanın yakınsaması kolaylaşacaktır [214]. Bu nedenle CEC 2005 konferansında gerçek parametre optimizasyonu için özel oturum düzenlenmiş ve bu oturumda algoritmaların kıyaslanması için özel fonksiyonlar tanımlanmıştır. Bu fonksiyonlar basit sınırlamasız fonksiyonlara bazı işlemlerin uygulanması ile elde edilmiştir. Basit fonksiyonlar Gaussian fonksiyonu ile biraraya getirilerek birleşik fonksiyonlar tanımlanmıştır. Bu fonksiyonlar asimetrik ve rasgele ötelenerek ve uzayın rasgele bir noktasına taşınmaktadırlar. Önerilen fonksiyonlar değişik kategoriler altında sınıflandırılmışlardır. Bunlar tek modlu, çok modlu, kaydırılmış, genişletilmiş, döndürülmüş ve karma fonksiyonlar olmak üzere toplam 25 tanedir. Bu fonksiyonlar EK-3 bölümünde verilmektedir. Döndürülmemiş kaydırılmış fonksiyonlar (4.6) eşitliği ile hesaplanmaktadır:

$$F(x) = \sum_{i=1}^{D} f_i(x - o) + f_{bias} \tag{4.6}$$

$o = [o_1, o_2, ..., o_D]$ kaymış global minimum noktalarıdır ve f_{bias} probleme özgü bias değeridir. Birden fazla temel fonksiyonun toplanmasıyla birleşik fonksiyonlar elde edilmektedir.

Döndürülmüş ve kaydırılmış fonksiyonlar M ortagonal döndürme matrisi olmak üzere şu şekilde hesaplanmaktadır:

$$F(x) = \sum_{i=1}^{D} f_i((x - o) * M) + f_{bias} \tag{4.7}$$

Genişletilmiş fonksiyonlar $EF(x)$, $F(x, y)$ şeklindeki iki boyutlu bir fonksiyonun D boyutlu olacak şekilde açılmış halidir:

$$EF(x_1, x_2, ..., x_D) = F(x_1, x_2) + F(x_2, x_3) + ... + F(x_{D-1}, x_D) + F(x_D, x_1) \tag{4.8}$$

Karma fonksiyonlar (4.9) ile hesaplanmaktadır:

$$F(x) = \sum_{i=1}^{n} \{w_i * [f_i' + bias_i]\} + f\_bias \tag{4.9}$$

$F(x)$ karma fonksiyonu göstermekte, $bias = [0, 100, 200, 300, 400, 500, 600, 700, 800, 900]$, n temel fonksiyonların sayısına, f_{bias} ise fonksiyonun ürettiği optimum değere karşılık gelmektedir. $f_i'(x)$, karma fonksiyonu oluşturan i. temel fonksiyonun ($f_i(x)$) ötelenmiş ve dönderilmiş hali olup (4.10) eşitliği ile tanımlanmaktadır.

$$f_i' = f_i((x - o_i)/\lambda_i) * M_i) \tag{4.10}$$

Burada o_i kaymış global ve bölgesel optimum parametrelerdir, λ_i, $f_i(x)$ fonksiyonunun ölçeklenmesinde kullanılmaktadır. $\lambda_i > 1$ olması fonksiyonu esnekleştirirken $\lambda_i < 1$ olması sıkılaştırır. Döndürme için kullanılan doğrusal dönüşüm matrisi, M_i, (4.11) bağıntısı ile oluşturulmaktadır.

$$M = P * N * Q \tag{4.11}$$

P ve Q, rasgele değerli matrislerin klasik Gram-Schmidt metodu kullanılarak QR faktörizasyonlarının yapılmasıyla hesaplanan ortogonal matrislerdir. $u = rand(1, D), dii = c^{\frac{ui - \min(u)}{\max(u) - \min(u)}}$ ifadesinin diagonal elemanları N matrisidir. Buradaki c terimi M matrisinin durum numarasıdır (condition number).

f_i değerleri, (4.13) eşitliği ile hesaplanan $fmax_i$ değeriyle ölçeklenir ve önceden tanımlı C değeriyle çarpılır (4.12).

$$f_i'(x) = C * f_i(x) / |fmax_i| \tag{4.12}$$

$fmax_i$ önceden tanımlı $y = [5, \ldots, 5]$ değerlerinin λ_i ile ölçeklenip M_i matrisiyle döndürülerek temel f_i fonksiyonunda yerine konmasıyla hesaplanır (4.13):

$$fmax_i = f_i((y/\lambda_i) * M_i) \tag{4.13}$$

Eşitlik (4.9)'da kullanılan diğer bir terim de her fonksiyonun ağırlığını belirten w_i terimidir. Bu terim (4.14,4.15,4.16,4.17,4.18) eşitlikleri ile aracılığıyla hesaplanmaktadır:

$$w_i = \exp\left(-\frac{\sum_{k=1}^{D} (x_k - o_{ik})^2}{2D\sigma_i^2} \right) \tag{4.14}$$

σ_i, $f_i(x)$'nin kapsama alanını kontrol eder; küçük σ_i değeri $f_i(x)$'nin dar bir alana sahip olması anlamına gelir. D problem boyutudur.

$$SW = \sum_{i=1}^{n} w_i \tag{4.15}$$

$$MaxW = \max(w_i) \tag{4.16}$$

$$w_i = \begin{cases} w_i & w_i == MaxW \\ w_i * (1 - \max(w_i).\ ^{10}) & w_i \neq MaxW \end{cases} \tag{4.17}$$

Her bir ağırlık ağırlıkların toplamı ile normalize edilir:

$$w_i = w_i / SW \tag{4.18}$$

4.3.1. ABC Algoritmasının Karma Fonksiyonlara Uygulanması

ABC algoritmasının uygulanmasında başlangıç popülasyonu oluşturulurken 7 ve 25. problemler hariç diğer problemlerde araştırma uzayından uniform olarak rasgele çözümler üretilmiştir. Bu iki problemin başlangıç popülasyon oluşturma aralığına EK-3 bölümünden ulaşılabilir. 7. ve 25. problemler dışındaki diğer tüm problemlerde global optimum sınırlar içerisindedir. D problem boyutu olmak üzere maksimum değerlendirme sayısı (MaxFES) 10000*D olarak alınmıştır. Çalışmada D değeri için 10, 30 ve 50 boyutlu durumlar incelenmiştir. Algoritma MaxFES değerlendirme sayısına ulaştığında veya hata değeri 10^{-8} değerinin altına düştüğünde koşma sonlandırılmıştır. Her bir fonksiyon için algoritma 25 kez koşularak bu 25 koşmanın en iyisi (1.), en kötüsü (25.), medianı (13.), 7. ve 19. en iyi değerleri 10 boyutlu durum için Tablo 4.30-4.32'de, 30 boyutlu durum için Tablo 4.33-4.35'de, 50 boyutlu durum için ise 4.36-4.38'de verilmiştir. 30 boyutlu durum için koşmaların median değerine ait logaritmik ölçekli yakınsama grafikleri Şekil 4.10(a)-4.10(e)'de verilmektedir.

Algoritmanın karmaşıklık analizi için farklı problem boyutlarında (10, 30, 50) sonuçlar alınmıştır. Öncelikle aşağıda verilen kod parçası için uygulamanın koşulduğu makinanın kod çalıştırma zamanının ölçütü olan T_0 hesaplanmıştır:

```
for i=1:1000000
    x= (double) 5.55;
    x=x + x;
    x=x./2;
    x=x*x;
    x=sqrt(x);
    x=ln(x);
    x=exp(x);
    y=x/x;
end
```

Daha sonra 3. problemin 200000 değerlendirmesi yapılarak platformun çalıştırma zamanının ölçütü olan T_1 değeri hesaplanmaktadır. Algoritmanın 200000 değerlendirme ile birlikte 3. probleme çözüm üretme zamanı olan T_2 bulunur. Algoritma içerisinde rasgele süreçler bulunduğundan T_2 değeri çoklu koşma (5 kez) için tekrarlanır ve ortalaması $\hat{T}_2$ alınır . Bu değerler vasıtasıyla algoritma karmaşıklığının (4.19) ile hesaplanır:

$$(\hat{T}_2 - T_1)/T_0 \tag{4.19}$$

D=10, 30, 50 için algoritmanın karmaşıklık değerleri Tablo 4.39'de verilmektedir. ABC algoritmasının koşulduğu platform, Windows XP (SP3), Pentium (R) M 1.60 GHz ve Ram: 1 GB'lık konfigürasyona sahip ve programlama dili Delphi 7'dir.

ABC algoritmasının kontrol parametresi MR'ye farklı değerler atanarak farklı fonksiyonlar için sonuçlar elde edilmiştir. Bazı fonksiyonlarda (7, 9, 12, 13 nolu fonksiyonlar) MR'nin düşük değerleri daha iyi sonuç üretirken bazı fonksiyonlarda (4, 16 nolu fonksiyonlar) büyük değerler daha iyi sonuç vermiştir. Her fonksiyonda en iyi parametre değerlerini alarak algoritmayı koşmak yerine ortalama bir MR değeri (0.4) kullanılarak sonuçlar elde edilmiştir. Limit parametresinin değeride tüm fonksiyonlar için 200 olarak alınmıştır. Yiyecek kaynağı (çözüm) sayısı D=10, 30 ve 50 durumları için sırasıyla 10, 30 ve 50 olarak alınmıştır. Tablo 4.30-4.32'de verilen sonuçlardan D=10 boyutu için 1, 2, 4 ve 9 numaralı fonksiyonlarda ABC algoritmasının istenen hassasiyete ulaştığı görülmektedir. D=30 boyutlu durumda ise 1, 2 ve 4 numaralı problemlerde ulaşmıştır. D=50 boyutlu durumda da sadece 1 ve 2 numaralı problemlerde istenen hassasiyete ulaşmıştır.

4.3.2. Literatürdeki Diğer Algoritmalarla ABC Algoritmasının Kıyaslanması

Tablo 4.30-4.38'de verilen sonuçlar elde edilirken ABC algoritması en iyi parametre değerleri ile koşulmamıştır. ABC algoritması farklı MR, SF ve limit değerleri için koşularak bu değerlerle üretilen sonuçlar arasındaki farklılıkları ve literatürde mevcut diğer algoritmalarla performans farklılıklarını görmek amacıyla kıyaslama yapılmıştır. Kıyaslama yapılan algoritmalar Recombination with Dynamic Linkage Discovery in Particle Swarm Optimization (PSO-RDL) [8], Dynamic Multi-swarm

Particle Swarm Optimizer with local search (DMS-PSO) [9], SPC-PNX [10], Differential Evolution [11], Self Adaptive Differential Evolution (SADE) [12], Restart Covariance Matrix Adaptation Evolution Strategy (RestartCMA-ES) [13]dır. Kıyaslamalarda problem boyutu D, 10 olarak seçilmiş, koloni büyüklüğü SN 20, maksimum değerlendirme sayısıda MaxFES 100000 olarak alınmıştır. MR'nin 1, 0.8, 0.6, 0.4, 0.2 değerleri ile SF'nin 1, 0.7, 0.5, 0.3 değerleri ve otomatik ölçekleme (ASF) durumu dikkate alınarak sonuçlar incelenmiştir.

Karşılaştırma sonuçları Tablo 4.40-4.44'de sunulmaktadır. Bu sonuçlardan 1 ve 2 numaralı problemlerde tüm algoritmaların benzer sonuçlar ürettikleri görülmektedir. 3 numaralı problemde ABC algoritmasının performansı MR'nin artmasıyla ve otomatik ölçekleme faktörü kullanılmasıyla artmaktadır. Bu fonksiyon ayrıştırılamayan türde bir fonksiyon olduğundan, ABC algoritmasının komşu çözüm ararken daha fazla parametre değiştirerek yeni çözüm üretmesi gerektiğinin bir göstergesidir. 4 numaralı fonksiyonda ise ölçekleme faktörünün (SF) azaltılması performansı olumsuz yönde etkilemektedir. Diğer algoritmalar benzer sonuçlar üretirken DMS-PSO algoritması bu fonksiyon üzerinde en kötü performansı sergilemektedir. 5 numaralı fonksiyon üzerinde ABC algoritması en iyi sonucu MR'nin yüksek değerleri ile üretmişken 6 numaralı fonksiyonda diğer algoritmalar ABC algoritmasından daha iyi sonuçlar bulmuştur. 8 numaralı fonksiyonda DMS-PSO, PSO-RDL, restartCMA-ES ve otomatik ölçekleme faktörlü ABC benzer başarı göstermişlerdir. 9 numaralı fonksiyon üzerinde MR'nin azalması ABC üzerinde olumlu etki yapmakta ve SADE ile DMS-PSO, ABC'ye benzer sonuçlar üretmişlerdir. 11 numaralı fonksiyon üzerinde restartCMAES ve DE algoritmaları daha iyi iken, 12 ve 14 numaralılarda SADE diğerlerinden daha iyi performans sergilemiştir. 13, 15, 23 ve 25 numaralı fonksiyonlarda temel ABC algoritması (SF=1); 7, 10, 16, 18, 19 ve 20 numaralı fonksiyonlarda restart-CMAES ve 17, 22 numaralı fonksiyonlarda DMS-PSO iyi olan algoritmalardır. 24 numaralı fonksiyonda SPC-PNX, DE, SADE ve restart-SMAEs eşit başarım göstermişlerdir. 21 numaralı fonksiyonda ise ABC en iyi sonucu üretmiş ancak SF'nin küçük değerlerinin daha iyi sonuçlar ürettiği görülmüştür.

Tablo 4.30. 1-8 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 10, Koloni Büyüklüğü: 10, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 100000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması

FES	Prob	1	2	3	4	5	6	7	8
1.00E+03	En iyi	1.76E+00	3.92E+01	2.98E+04	2.12E+02	3.64E+02	5.93E+03	5.22E+00	2.04E+01
	7.	1.54E+01	1.47E+02	5.66E+04	5.55E+02	1.55E+03	4.37E+04	1.02E+01	2.07E+01
	Median	4.44E+01	2.08E+02	8.52E+04	8.49E+02	2.34E+03	1.04E+05	1.82E+01	2.08E+01
	19.	7.29E+01	3.11E+02	1.03E+05	1.37E+03	2.86E+03	1.88E+05	2.89E+01	2.08E+01
	En kötü	1.03E+02	6.94E+02	1.71E+05	3.01E+03	6.79E+03	8.13E+05	6.54E+01	2.09E+01
	Ortalama	4.43E+01	2.41E+02	8.51E+04	1.07E+03	2.66E+03	1.73E+05	2.39E+01	2.07E+01
	Std	2.91E+01	1.54E+02	3.74E+04	7.28E+02	1.66E+03	1.97E+05	1.70E+01	1.30E-01
1.00E+04	En iyi	0.00E+00	0.00E+00	6.33E+03	0.00E+00	4.82E-02	5.67E+00	4.15E-01	2.03E+01
	7.	0.00E+00	0.00E+00	1.51E+04	0.00E+00	3.36E-01	1.24E+01	5.58E-01	2.04E+01
	Median	0.00E+00	0.00E+00	2.02E+04	0.00E+00	1.22E+00	1.62E+01	6.72E-01	2.05E+01
	19.	0.00E+00	0.00E+00	2.24E+04	0.00E+00	9.11E+00	2.93E+01	8.56E-01	2.06E+01
	En kötü	0.00E+00	0.00E+00	3.72E+04	0.00E+00	6.73E+01	1.23E+02	1.16E+00	2.07E+01
	Ortalama	0.00E+00	0.00E+00	1.96E+04	0.00E+00	9.13E+00	2.68E+01	7.17E-01	2.05E+01
	Std	0.00E+00	0.00E+00	6.69E+03	0.00E+00	1.57E+01	2.77E+01	2.02E-01	1.09E-01
1.00E+05	En iyi	0.00E+00	0.00E+00	7.83E+02	0.00E+00	0.00E+00	1.07E+00	2.38E-01	2.02E+01
	7.	0.00E+00	0.00E+00	3.82E+03	0.00E+00	0.00E+00	3.23E+00	3.05E-01	2.03E+01
	Median	0.00E+00	0.00E+00	6.33E+03	0.00E+00	0.00E+00	4.44E+00	3.37E-01	2.04E+01
	19.	0.00E+00	0.00E+00	8.05E+03	0.00E+00	0.00E+00	6.50E+00	3.90E-01	2.04E+01
	En kötü	0.00E+00	0.00E+00	1.42E+04	0.00E+00	0.00E+00	8.84E+00	4.95E-01	2.05E+01
	Ortalama	0.00E+00	0.00E+00	6.27E+03	0.00E+00	0.00E+00	4.69E+00	3.46E-01	2.04E+01
	Std	0.00E+00	0.00E+00	2.83E+03	0.00E+00	0.00E+00	2.24E+00	6.43E-02	6.52E-02
	MTE	8.63E-09	7.65E-09	6.27E+03	1.74E-09	1.15E-03	4.69E+00	3.46E-01	2.04E+01
	STE	1.31E-09	1.40E-09	2.83E+03	7.95E-09	2.33E-03	2.24E+00	6.43E-02	6.52E-02

Tablo 4.31. 9-17 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 10, Koloni Büyüklüğü: 10, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 100000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması

FES	Prob	9	10	11	12	13	14	15	16	17
1.00E+03	En iyi	1.95E+01	4.52E+01	7.70E+00	2.81E+03	2.04E+00	3.58E+00	2.57E+02	2.69E+02	2.68E+02
	7.	2.72E+01	6.47E+01	9.81E+00	6.98E+03	3.17E+00	4.12E+00	3.95E+02	3.18E+02	3.18E+02
	Median	3.04E+01	6.99E+01	1.04E+01	9.63E+03	3.64E+00	4.24E+00	4.56E+02	3.40E+02	3.35E+02
	19.	3.71E+01	7.73E+01	1.07E+01	1.42E+04	4.47E+00	4.32E+00	5.14E+02	3.63E+02	3.74E+02
	En kötü	4.74E+01	9.35E+01	1.14E+01	2.45E+04	7.20E+00	4.51E+00	5.88E+02	3.88E+02	5.27E+02
	Ortalama	3.19E+01	7.02E+01	1.02E+01	1.11E+04	3.92E+00	4.22E+00	4.49E+02	3.37E+02	3.44E+02
	Std	6.72E+00	1.12E+01	8.16E-01	5.53E+03	1.20E+00	1.80E-01	7.53E+01	3.44E+01	5.06E+01
1.00E+04	En iyi	1.28E-02	1.91E+01	6.62E+00	6.01E+01	8.00E-01	3.40E+00	1.31E+02	1.77E+02	2.02E+02
	7.	1.05E+00	3.10E+01	7.50E+00	6.76E+02	1.44E+00	3.66E+00	1.99E+02	2.14E+02	2.48E+02
	Median	2.35E+00	3.58E+01	7.86E+00	1.64E+03	1.58E+00	3.81E+00	2.32E+02	2.32E+02	2.63E+02
	19.	4.91E+00	4.10E+01	8.55E+00	2.25E+03	1.84E+00	3.97E+00	2.66E+02	2.71E+02	2.78E+02
	En kötü	7.28E+00	4.90E+01	8.83E+00	3.07E+03	2.18E+00	4.10E+00	3.49E+02	3.05E+02	3.64E+02
	Ortalama	2.91E+00	3.52E+01	7.90E+00	1.42E+03	1.60E+00	3.81E+00	2.30E+02	2.42E+02	2.66E+02
	Std	2.35E+00	7.45E+00	6.38E-01	8.30E+02	3.59E-01	1.86E-01	5.45E+01	3.32E+01	2.96E+01
1.00E+05	En iyi	2.02E+01	1.47E+01	4.37E+00	6.01E+01	1.19E-01	3.07E+00	1.03E+02	1.75E+02	1.84E+02
	7.	2.03E+01	1.91E+01	5.69E+00	2.97E+02	3.79E-01	3.43E+00	1.31E+02	1.85E+02	2.04E+02
	Median	2.04E+01	2.27E+01	6.12E+00	3.43E+02	5.03E-01	3.51E+00	1.49E+02	1.94E+02	2.15E+02
	19.	2.04E+01	2.70E+01	6.66E+00	4.99E+02	1.92E-01	3.65E+00	1.68E+02	2.11E+02	2.29E+02
	En kötü	2.05E+01	2.96E+01	7.20E+00	1.02E+03	8.43E-01	3.78E+00	1.97E+02	2.22E+02	2.52E+02
	Ortalama	2.04E+01	2.27E+01	6.13E+00	3.99E+02	4.90E-01	3.51E+00	1.48E+02	1.98E+02	2.16E+02
	Std	6.52E-02	4.24E+00	6.65E-01	2.11E+02	1.92E-01	1.55E-01	2.40E+01	1.46E+01	1.61E+01
	MTE	7.89E-09	2.27E+01	6.13E+00	3.99E+02	4.90E-01	3.51E+00	1.48E+02	1.98E+02	2.16E+02
	STE	1.82E-09	4.24E+00	6.65E-01	2.11E+02	1.92E-01	1.55E-01	2.40E+01	1.46E+01	1.61E+01

Tablo 4.32. 18-25 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 10, Koloni Büyüklüğü: 10, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 100000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması

FES	Prob	18	19	20	21	22	23	24	25
1.00E+03	En iyi	5.87E+02	5.49E+02	5.22E+02	7.02E+02	8.79E+02	7.49E+02	2.01E+02	2.00E+02
	7.	6.57E+02	6.29E+02	6.54E+02	9.55E+02	9.23E+02	9.66E+02	2.01E+02	2.00E+02
	Median	7.00E+02	6.71E+02	7.05E+02	9.96E+02	9.53E+02	1.02E+03	2.02E+02	2.00E+02
	19.	9.44E+02	7.19E+02	8.14E+02	1.03E+03	9.70E+02	1.04E+03	2.02E+02	2.00E+02
	En kötü	1.02E+03	1.07E+03	9.96E+02	1.13E+03	1.09E+03	1.15E+03	5.56E+02	2.01E+02
	Ortalama	7.78E+02	7.04E+02	7.47E+02	9.91E+02	9.54E+02	1.00E+03	2.16E+02	2.00E+02
	Std	1.47E+02	1.29E+02	1.31E+02	8.80E+01	4.88E+01	8.19E+01	6.94E+01	2.77E-01
1.00E+04	En iyi	4.14E+02	3.95E+02	4.38E+02	5.20E+02	8.49E+02	5.49E+02	2.01E+02	2.00E+02
	7.	5.38E+02	5.15E+02	5.09E+02	5.82E+02	8.58E+02	5.64E+02	2.01E+02	2.00E+02
	Median	5.67E+02	5.71E+02	5.46E+02	6.19E+02	8.64E+02	6.38E+02	2.01E+02	2.00E+02
	19.	5.87E+02	6.18E+02	5.77E+02	8.00E+02	8.70E+02	8.73E+02	2.01E+02	2.00E+02
	En kötü	8.15E+02	6.83E+02	6.49E+02	8.03E+02	8.91E+02	8.84E+02	2.02E+02	2.00E+02
	Ortalama	5.66E+02	5.62E+02	5.49E+02	6.67E+02	8.66E+02	6.82E+02	2.01E+02	2.00E+02
	Std	7.06E+01	7.06E+01	4.80E+01	1.13E+02	9.70E+00	1.32E+02	3.88E-01	6.65E-03
1.00E+05	En iyi	3.65E+02	3.58E+02	3.88E+02	3.97E+02	8.79E+02	7.49E+02	2.00E+02	2.00E+02
	7.	4.66E+02	4.51E+02	4.40E+02	5.29E+02	9.23E+02	9.66E+02	2.00E+02	2.00E+02
	Median	4.87E+02	4.78E+02	5.00E+02	5.82E+02	9.53E+02	1.02E+03	2.01E+02	2.00E+02
	19.	5.04E+02	4.94E+02	5.05E+02	6.29E+02	9.70E+02	1.04E+03	2.01E+02	2.00E+02
	En kötü	5.42E+02	5.24E+02	5.42E+02	8.00E+02	1.09E+03	1.15E+03	2.01E+02	2.00E+02
	Ortalama	4.77E+02	4.68E+02	4.77E+02	6.00E+02	9.54E+02	1.00E+03	2.01E+02	2.00E+02
	Std	4.44E+01	3.81E+01	4.60E+01	9.76E+01	4.88E+01	8.19E+01	9.01E-02	1.95E-03
	MTE	4.77E+02	4.68E+02	4.77E+02	6.00E+02	9.54E+02	1.00E+03	2.01E+02	2.00E+02
	STE	4.44E+01	3.81E+01	4.60E+01	9.76E+01	4.88E+01	8.19E+01	9.01E-02	1.95E-03

Tablo 4.33. 1-8 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 30, Koloni Büyüklüğü: 30, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 300000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması

FES	Prob	1	2	3	4	5	6	7	8
1.00E+03	En iyi	6.59E+03	1.18E+05	1.02E+06	1.34E+05	1.86E+04	6.87E+08	1.31E+03	2.11E+01
	7.	1.51E+04	1.83E+05	1.41E+06	3.97E+05	2.36E+04	1.47E+09	1.76E+03	2.12E+01
	Median	1.73E+04	2.11E+05	1.55E+06	4.73E+05	2.51E+04	2.22E+09	1.94E+03	2.12E+01
	19.	1.82E+04	2.42E+05	1.83E+06	5.86E+05	2.79E+04	4.61E+09	2.27E+03	2.13E+01
	En kötü	2.04E+04	2.88E+05	2.18E+06	8.62E+05	3.03E+04	7.29E+09	2.54E+03	2.13E+01
	Ortalama	1.59E+04	2.11E+05	1.60E+06	4.83E+05	2.55E+04	3.04E+09	2.00E+03	2.12E+01
	Std	3.51E+03	4.38E+04	2.89E+05	1.52E+05	2.73E+03	1.91E+09	3.63E+02	5.03E-02
1.00E+04	En iyi	2.34E-01	1.59E+00	3.35E+05	5.70E+02	8.40E+03	1.35E+04	5.82E+00	2.10E+01
	7.	3.33E-01	4.15E+00	4.59E+05	7.30E+02	1.14E+04	2.03E+04	1.35E+01	2.11E+01
	Median	4.33E-01	5.77E+00	4.97E+05	9.02E+02	1.19E+04	3.09E+04	1.65E+01	2.11E+01
	19.	6.82E-01	8.42E+00	5.48E+05	1.74E+03	1.32E+04	4.36E+04	2.00E+01	2.11E+01
	En kötü	9.78E-01	1.56E+01	7.29E+05	2.37E+03	1.55E+04	1.63E+05	2.38E+01	2.12E+01
	Ortalama	5.18E-01	6.61E+00	5.00E+05	1.19E+03	1.20E+04	3.64E+04	1.60E+01	2.11E+01
	Std	2.24E-01	3.40E+00	8.34E+04	5.83E+02	1.64E+03	2.84E+04	4.40E+00	5.44E-02
1.00E+05	En iyi	0.00E+00	0.00E+00	1.86E+05	1.86E+05	5.20E+03	5.93E+01	1.09E-03	2.08E+01
	7.	0.00E+00	0.00E+00	2.41E+05	2.41E+05	6.32E+03	1.44E+02	4.11E-03	2.10E+01
	Median	0.00E+00	0.00E+00	2.67E+05	2.67E+05	7.06E+03	1.82E+02	5.03E-02	2.10E+01
	19.	0.00E+00	0.00E+00	2.98E+05	2.98E+05	7.61E+03	2.93E+02	2.41E-01	2.10E+01
	En kötü	0.00E+00	0.00E+00	3.39E+05	3.39E+05	8.74E+03	3.56E+02	7.56E-01	2.11E+01
	Ortalama	0.00E+00	0.00E+00	2.65E+05	2.65E+05	6.91E+03	2.09E+02	1.92E-01	2.10E+01
	Std	0.00E+00	0.00E+00	3.95E+04	3.95E+04	9.13E+02	8.59E+01	2.62E-01	6.15E-02
3.00E+05	En iyi	0.00E+00	0.00E+00	1.76E+05	1.76E+05	4.53E+03	3.35E+01	2.68E-08	2.08E+01
	7.	0.00E+00	0.00E+00	1.99E+05	1.99E+05	5.54E+03	9.37E+01	3.69E-01	2.09E+01
	Median	0.00E+00	0.00E+00	2.16E+05	2.16E+05	6.13E+03	1.43E+02	1.17E+02	2.10E+01
	19.	0.00E+00	0.00E+00	2.39E+05	2.39E+05	6.43E+03	1.82E+02	1.72E+02	2.10E+01
	En kötü	0.00E+00	0.00E+00	2.72E+05	2.72E+05	7.68E+03	2.60E+02	2.60E+02	2.10E+01
	Ortalama	0.00E+00	0.00E+00	2.20E+05	2.20E+05	6.02E+03	1.38E+02	1.05E+02	2.09E+01
	Std	0.00E+00	0.00E+00	2.53E+04	2.53E+04	7.16E+02	5.81E+01	8.20E+01	5.63E-02
	MTE	9.35E-09	9.06E-09	2.20E+05	9.01E-09	6.02E+03	1.38E+02	1.49E-02	2.09E+01
	STE	6.21E-10	5.80E-10	2.53E+04	8.89E-10	7.16E+02	5.81E+01	7.23E-02	5.63E-02

Tablo 4.34. 9-17 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 30, Koloni Büyüklüğü: 30, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 300000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması

FES	Prob	9	10	11	12	13	14	15	16	17
1.00E+03	En iyi	2.13E+02	3.44E+02	4.06E+01	5.63E+05	3.79E+01	1.39E+01	6.21E+02	4.39E+02	4.47E+02
	7.	2.74E+02	4.13E+02	4.22E+01	7.42E+05	5.10E+01	1.40E+01	7.39E+02	4.96E+02	5.37E+02
	Median	2.95E+02	4.54E+02	4.33E+01	7.85E+05	7.35E+01		8.31E+02	5.29E+02	5.84E+02
	19.	3.03E+02	4.79E+02	4.46E+01	8.77E+05	1.06E+02	1.41E+01	8.63E+02	5.58E+02	6.48E+02
	En kötü	3.28E+02	5.19E+02	4.66E+01	9.66E+05	1.31E+02	1.42E+01	9.03E+02	6.55E+02	8.43E+02
	Ortalama	2.88E+02	4.44E+02	4.34E+01	8.04E+05	7.82E+01	1.44E+01	8.05E+02	5.25E+02	6.08E+02
	Std	2.39E+01	4.56E+01	1.80E+00	9.42E+04	2.87E+01	1.41E+01	7.60E+01	4.77E+01	9.69E+01
1.00E+04	En iyi	1.01E+02	2.27E+02	3.65E+01	1.10E+05	1.45E+01	1.42E-01	4.74E+02	2.86E+02	3.00E+02
	7.	1.18E+02	2.52E+02	3.83E+01	2.17E+05	1.69E+01	1.35E+01	5.26E+02	3.33E+02	3.43E+02
	Median	1.23E+02	2.67E+02	4.00E+01	2.26E+05	1.75E+01	1.37E+01	5.36E+02	3.52E+02	3.69E+02
	19.	1.32E+02	2.73E+02	4.06E+01	2.49E+05	1.85E+01	1.38E+01	5.41E+02	3.70E+02	3.88E+02
	En kötü	1.50E+02	2.85E+02	4.19E+01	3.36E+05	2.02E+01	1.39E+01	6.22E+02	4.21E+02	4.87E+02
	Ortalama	1.24E+02	2.62E+02	3.97E+01	2.35E+05	1.75E+01	1.40E+01	5.36E+02	3.54E+02	3.73E+02
	Std	1.22E+01	1.61E+01	1.49E+00	4.81E+04	1.54E+00	1.38E+01	2.98E+01	3.68E+01	4.58E+01
1.00E+05	En iyi	6.07E+01	1.62E+02	3.43E+01	8.86E+04	9.13E+00	1.21E-01	2.00E+02	2.77E+02	2.67E+02
	7.	7.12E+01	2.06E+02	3.59E+01	1.03E+05	1.15E+01	1.31E+01	3.00E+02	2.99E+02	3.10E+02
	Median	7.72E+01	2.16E+02	3.68E+01	1.23E+05	1.20E+01	1.35E+01	3.00E+02	3.16E+02	3.27E+02
	19.	8.14E+01	2.25E+02	3.72E+01	1.41E+05	1.28E+01	1.35E+01	3.03E+02	3.33E+02	3.38E+02
	En kötü	9.04E+01	2.42E+02	3.84E+01	1.74E+05	1.35E+01	1.36E+01	4.84E+02	4.00E+02	3.84E+02
	Ortalama	7.67E+01	2.14E+02	3.65E+01	1.22E+05	1.19E+01	1.37E+01	3.15E+02	3.21E+02	3.27E+02
	Std	7.60E+00	1.58E+01	1.11E+00	2.27E+04	1.09E+00	1.35E+01	6.80E+01	3.09E+01	2.89E+01
3.00E+05	En iyi	5.49E+01	1.60E+02	3.36E+01	5.09E+04	8.21E+00	1.41E-01	2.00E+02	2.69E+02	2.46E+02
	7.	6.15E+01	1.99E+02	3.48E+01	8.65E+04	1.02E+01	1.28E+01	3.00E+02	2.90E+02	2.88E+02
	Median	6.53E+01	2.02E+02	3.59E+01	9.52E+04	1.08E+01	1.32E+01	3.00E+02	3.02E+02	3.04E+02
	19.	7.27E+01	2.10E+02	3.64E+01	1.05E+05	1.14E+01	1.34E+01	3.00E+02	3.18E+02	3.17E+02
	En kötü	7.95E+01	2.24E+02	3.68E+01	1.24E+05	1.21E+01	1.34E+01	3.00E+02	3.52E+02	3.29E+02
	Ortalama	6.60E+01	2.01E+02	3.56E+01	9.55E+04	1.07E+01	1.36E+01	2.88E+02	3.06E+02	3.01E+02
	Std	6.74E+00	1.44E+01	8.84E-01	1.75E+04	9.32E-01	1.33E+01	3.25E+01	2.18E+01	2.00E+01
	MTE	6.60E+01	2.01E+02	3.56E+01	9.55E+04	1.07E+01	1.88E-01	2.88E+02	3.06E+02	3.01E+02
	STE	6.74E+00	1.44E+01	8.84E-01	1.75E+04	9.32E-01	1.33E+01	3.25E+01	2.18E+01	2.00E+01

Tablo 4.35. 18-25 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 30, Koloni Büyüklüğü: 30, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 300000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması

FES	Prob	18	19	20	21	22	23	24	25
1.00E+03	En iyi	1.21E+03	1.17E+03	1.21E+03	1.20E+03	1.13E+03	1.22E+03	2.03E+02	2.01E+02
	7.	1.28E+03	1.26E+03	1.29E+03	1.23E+03	1.24E+03	1.26E+03	2.04E+02	2.05E+02
	Median	1.32E+03	1.31E+03	1.31E+03	1.25E+03	1.36E+03	1.27E+03	2.05E+02	2.14E+02
	19.	1.36E+03	1.34E+03	1.35E+03	1.27E+03	1.40E+03	1.29E+03	2.08E+02	2.19E+02
	En kötü	1.41E+03	1.41E+03	1.41E+03	1.29E+03	1.46E+03	1.32E+03	2.20E+02	7.05E+02
	Ortalama	1.31E+03	1.30E+03	1.32E+03	1.25E+03	1.33E+03	1.28E+03	2.07E+02	2.36E+02
	Std	5.11E+01	6.07E+01	4.58E+01	2.34E+01	9.25E+01	2.57E+01	4.42E+00	9.71E+01
1.00E+04	En iyi	1.01E+03	9.95E+02	1.00E+03	5.00E+02	9.15E+02	7.38E+02	2.01E+02	2.00E+02
	7.	1.03E+03	1.03E+03	1.04E+03	5.63E+02	9.40E+02	8.79E+02	2.01E+02	2.00E+02
	Median	1.04E+03	1.05E+03	1.06E+03	7.69E+02	9.67E+02	9.15E+02	2.01E+02	2.00E+02
	19.	1.06E+03	1.07E+03	1.07E+03	8.28E+02	9.81E+02	9.70E+02	2.01E+02	2.00E+02
	En kötü	1.08E+03	1.13E+03	1.11E+03	9.52E+02	1.01E+03	1.03E+03	2.01E+02	2.00E+02
	Ortalama	1.04E+03	1.05E+03	1.06E+03	7.24E+02	9.62E+02	9.18E+02	2.01E+02	2.00E+02
	Std	2.13E+01	3.48E+01	2.66E+01	1.46E+02	2.56E+01	6.50E+01	6.52E-02	4.92E-02
1.00E+05	En iyi	8.00E+02	8.00E+02	8.00E+02	5.00E+02	8.91E+02	5.45E+02	2.01E+02	2.00E+02
	7.	8.43E+02	8.03E+02	8.59E+02	5.00E+02	9.06E+02	8.44E+02	2.01E+02	2.00E+02
	Median	9.70E+02	9.58E+02	9.57E+02	6.45E+02	9.12E+02	8.46E+02	2.01E+02	2.00E+02
	19.	9.77E+02	9.69E+02	9.74E+02	8.00E+02	9.19E+02	8.47E+02	2.01E+02	2.00E+02
	En kötü	9.93E+02	9.90E+02	1.01E+03	8.00E+02	9.28E+02	8.49E+02	2.01E+02	2.00E+02
	Ortalama	9.26E+02	9.13E+02	9.22E+02	6.42E+02	9.12E+02	8.22E+02	2.01E+02	2.00E+02
	Std	7.31E+01	7.68E+01	6.80E+01	1.41E+02	9.31E+00	7.86E+01	5.05E-02	4.62E-03
3.00E+05	En iyi	8.00E+02	8.00E+02	8.00E+02	5.00E+02	8.91E+02	5.42E+02	2.01E+02	2.00E+02
	7.	8.00E+02	8.00E+02	8.00E+02	5.00E+02	8.99E+02	8.44E+02	2.01E+02	2.00E+02
	Median	8.00E+02	8.00E+02	8.00E+02	6.45E+02	9.04E+02	8.44E+02	2.01E+02	2.00E+02
	19.	8.00E+02	8.00E+02	8.00E+02	8.00E+02	9.07E+02	8.44E+02	2.01E+02	2.00E+02
	En kötü	9.71E+02	9.45E+02	9.80E+02	8.00E+02	9.17E+02	8.46E+02	2.01E+02	2.00E+02
	Ortalama	8.12E+02	8.17E+02	8.23E+02	6.42E+02	9.04E+02	8.20E+02	2.01E+02	2.00E+02
	Std	4.07E+01	4.50E+01	5.31E+01	1.41E+02	6.01E+00	8.13E+01	5.45E-02	2.23E-03
	MTE	8.12E+02	8.17E+02	8.23E+02	6.42E+02	9.04E+02	8.20E+02	2.01E+02	2.00E+02
	STE	4.07E+01	4.50E+01	5.31E+01	1.41E+02	6.01E+00	8.13E+01	5.45E-02	2.23E-03

Tablo 4.36. 1-8 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 50, Koloni Büyüklüğü: 50, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 500000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması

FES	Prob	1	2	3	4	5	6	7	8
1.00E+03	En iyi	5.97E+04	1.15E+06	4.73E+06	2.51E+06	5.12E+04	3.16E+10	7.03E+03	2.12E+01
	7.	7.98E+04	1.70E+06	5.81E+06	3.02E+06	5.92E+04	4.21E+10	8.11E+03	2.13E+01
	Median	8.91E+04	1.85E+06	6.71E+06	3.44E+06	6.22E+04	5.49E+10	9.00E+03	2.13E+01
	19.	1.00E+05	2.09E+06	7.39E+06	3.90E+06	6.52E+04	5.85E+10	9.77E+03	2.14E+01
	En kötü	1.18E+05	2.37E+06	8.65E+06	4.57E+06	7.18E+04	7.53E+10	1.06E+04	2.14E+01
	Ortalama	8.90E+04	1.85E+06	6.62E+06	3.43E+06	6.23E+04	5.20E+10	8.94E+03	2.13E+01
	Std	1.43E+04	3.01E+05	1.09E+06	5.49E+05	5.08E+03	1.10E+10	9.65E+02	4.73E-02
1.00E+04	En iyi	1.01E+03	1.85E+04	1.79E+06	2.41E+05	2.19E+04	1.16E+08	2.14E+02	2.12E+01
	7.	1.46E+03	2.87E+04	2.17E+06	2.93E+05	2.74E+04	1.95E+08	2.79E+02	2.12E+01
	Median	1.57E+03	3.15E+04	2.34E+06	3.25E+05	2.94E+04	2.58E+08	3.16E+02	2.12E+01
	19.	1.78E+03	3.57E+04	2.49E+06	3.37E+05	3.14E+04	3.26E+08	3.50E+02	2.13E+01
	En kötü	2.58E+03	4.37E+04	2.95E+06	4.33E+05	3.36E+04	4.87E+08	4.36E+02	2.13E+01
	Ortalama	1.65E+03	3.19E+04	2.35E+06	3.23E+05	2.90E+04	2.66E+08	3.20E+02	2.12E+01
	Std	3.48E+02	5.46E+03	2.45E+05	5.11E+04	3.19E+03	9.09E+07	4.84E+01	4.29E-02
1.00E+05	En iyi	0.00E+00	0.00E+00	1.01E+06	1.31E+02	1.05E+04	7.40E+02	7.95E-01	2.11E+01
	7.	0.00E+00	0.00E+00	1.19E+06	7.13E+02	1.22E+04	4.55E+03	8.76E-01	2.12E+01
	Median	0.00E+00	0.00E+00	1.38E+06	1.44E+03	1.31E+04	6.45E+03	9.19E-01	2.12E+01
	19.	0.00E+00	0.00E+00	1.41E+06	2.63E+03	1.42E+04	1.28E+04	9.36E-01	2.12E+01
	En kötü	0.00E+00	0.00E+00	1.54E+06	9.84E+03	1.55E+04	2.38E+04	9.73E-01	2.12E+01
	Ortalama	0.00E+00	0.00E+00	1.32E+06	2.13E+03	1.32E+04	9.37E+03	9.05E-01	2.12E+01
	Std	0.00E+00	0.00E+00	1.40E+05	2.23E+03	1.39E+03	6.53E+03	4.46E-02	3.37E-02
5.00E+05	En iyi	0.00E+00	0.00E+00	9.76E+05	4.49E+00	7.93E+03	7.40E+02	7.18E-01	2.11E+01
	7.	0.00E+00	0.00E+00	1.07E+06	1.57E+02	9.67E+03	1.69E+03	7.81E-01	2.11E+01
	Median	0.00E+00	0.00E+00	1.10E+06	2.27E+02	1.03E+04	2.11E+03	8.03E-01	2.11E+01
	19.	0.00E+00	0.00E+00	1.22E+06	4.59E+02	1.09E+04	3.40E+03	8.54E-01	2.12E+01
	En kötü	0.00E+00	0.00E+00	1.31E+06	1.81E+03	1.20E+04	4.55E+03	8.79E-01	2.12E+01
	Ortalama	0.00E+00	0.00E+00	1.13E+06	3.82E+02	1.03E+04	2.47E+03	8.10E-01	2.11E+01
	Std	0.00E+00	0.00E+00	9.38E+04	3.98E+02	9.13E+02	1.10E+03	4.70E-02	2.65E-02
	MTE	9.34E-09	9.28E-09	1.13E+06	3.82E+02	1.03E+04	2.47E+03	8.10E-01	2.11E+01
	STE	6.05E-10	6.19E-10	9.38E+04	3.98E+02	9.13E+02	1.10E+03	4.70E-02	2.65E-02

Tablo 4.37. 9-17 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 50, Koloni Büyüklüğü: 50, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 500000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması

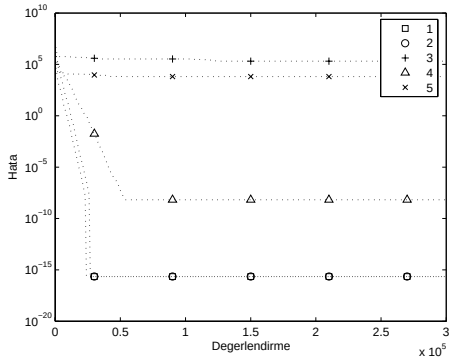
FES	Prob	9	10	11	12	13	14	15	16	17
1.00E+03	En iyi	6.10E+02	9.08E+02	7.71E+01	4.09E+06	5.91E+02	2.35E+01	8.11E+02	5.30E+02	6.17E+02
	7.	6.77E+02	1.03E+03	7.89E+01	4.44E+06	9.04E+02	2.40E+01	9.78E+02	6.27E+02	7.30E+02
	Median	7.26E+02	1.11E+03	8.08E+01	4.77E+06	1.07E+03	2.40E+01	1.03E+03	6.70E+02	7.68E+02
	19.	7.58E+02	1.18E+03	8.15E+01	5.24E+06	1.36E+03	2.41E+01	1.05E+03	7.56E+02	8.27E+02
	En kötü	8.02E+02	1.30E+03	8.24E+01	5.78E+06	2.19E+03	2.43E+01	1.08E+03	8.20E+02	9.87E+02
	Ortalama	7.17E+02	1.11E+03	8.02E+01	4.86E+06	1.15E+03	2.40E+01	1.01E+03	6.82E+02	7.91E+02
	Std	4.83E+01	1.04E+02	1.58E+00	4.79E+05	4.50E+02	1.84E-01	7.04E+01	8.14E+01	9.44E+01
1.00E+04	En iyi	3.41E+02	4.65E+02	7.26E+01	1.18E+06	4.73E+01	2.33E+01	5.68E+02	3.35E+02	3.53E+02
	7.	3.50E+02	5.61E+02	7.51E+01	1.50E+06	6.00E+01	2.36E+01	5.80E+02	3.55E+02	3.99E+02
	Median	3.61E+02	5.82E+02	7.58E+01	1.68E+06	6.32E+01	2.37E+01	5.92E+02	3.83E+02	4.16E+02
	19.	3.68E+02	5.97E+02	7.67E+01	1.80E+06	6.86E+01	2.38E+01	6.40E+02	3.97E+02	4.29E+02
	En kötü	3.98E+02	6.13E+02	7.86E+01	1.94E+06	7.61E+01	2.39E+01	7.19E+02	4.40E+02	4.75E+02
	Ortalama	3.62E+02	5.74E+02	7.58E+01	1.65E+06	6.35E+01	2.37E+01	6.15E+02	3.80E+02	4.17E+02
	Std	1.60E+01	3.14E+01	1.43E+00	1.89E+05	7.23E+00	1.54E-01	4.49E+01	2.88E+01	2.80E+01
1.00E+05	En iyi	2.36E+02	4.49E+02	6.95E+01	8.33E+05	3.21E+01	2.30E+01	2.01E+02	3.00E+02	2.98E+02
	7.	2.74E+02	4.81E+02	7.11E+01	9.70E+05	3.66E+01	2.33E+01	2.18E+02	3.24E+02	3.26E+02
	Median	2.82E+02	4.93E+02	7.23E+01	1.07E+06	3.81E+01	2.34E+01	3.00E+02	3.46E+02	3.39E+02
	19.	2.86E+02	5.02E+02	7.31E+01	1.14E+06	3.91E+01	2.34E+01	3.05E+02	3.58E+02	3.54E+02
	En kötü	2.98E+02	5.06E+02	7.43E+01	1.21E+06	4.28E+01	2.36E+01	5.12E+02	3.86E+02	4.10E+02
	Ortalama	2.77E+02	4.88E+02	7.21E+01	1.04E+06	3.81E+01	2.34E+01	2.97E+02	3.44E+02	3.39E+02
	Std	1.39E+01	1.60E+01	1.30E+00	1.10E+05	2.41E+00	1.41E-01	8.91E+01	2.39E+01	2.43E+01
5.00E+05	En iyi	2.36E+02	4.31E+02	6.62E+01	7.31E+05	2.91E+01	2.29E+01	2.00E+02	2.68E+02	2.81E+02
	7.	2.51E+02	4.49E+02	6.99E+01	8.33E+05	3.42E+01	2.31E+01	2.00E+02	3.24E+02	3.01E+02
	Median	2.61E+02	4.57E+02	7.03E+01	8.91E+05	3.54E+01	2.32E+01	2.00E+02	3.44E+02	3.10E+02
	19.	2.64E+02	4.68E+02	7.09E+01	9.44E+05	3.66E+01	2.33E+01	3.00E+02	3.57E+02	3.17E+02
	En kötü	2.76E+02	4.88E+02	7.25E+01	1.05E+06	3.75E+01	2.33E+01	4.00E+02	3.86E+02	3.30E+02
	Ortalama	2.59E+02	4.58E+02	7.03E+01	8.88E+05	3.50E+01	2.32E+01	2.52E+02	3.39E+02	3.08E+02
	Std	9.91E+00	1.41E+01	1.44E+00	8.77E+04	1.98E+00	1.13E-01	5.74E+01	3.16E+01	1.24E+01
	MTE	2.59E+02	4.58E+02	7.03E+01	8.88E+05	3.50E+01	2.32E+01	2.52E+02	3.39E+02	3.08E+02
	STE	9.91E+00	1.41E+01	1.44E+00	8.77E+04	1.98E+00	1.13E-01	5.74E+01	3.16E+01	1.24E+01

Tablo 4.38. 18-25 numaralı problemlerde ABC algoritmasının istatistiksel sonuçları, D: 50, Koloni Büyüklüğü: 50, Limit: 200, Çevrim Sayısı: 10000, MaxFES: 500000, MR=0.4, MTE: 25 koşmanın son hatalarının ortalaması, STE: 25 koşmanın son hatalarının standard sapması

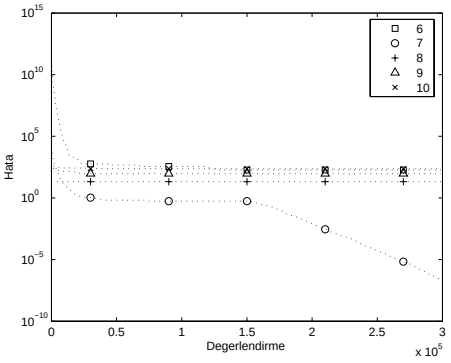
FES	Prob	18	19	20	21	22	23	24	25
1.00E+03	En iyi	1.24E+03	1.30E+03	1.29E+03	1.33E+03	1.19E+03	1.36E+03	2.10E+02	4.55E+02
	7.	1.29E+03	1.31E+03	1.33E+03	1.37E+03	1.23E+03	1.39E+03	2.39E+02	8.13E+02
	Median	1.33E+03	1.36E+03	1.36E+03	1.40E+03	1.26E+03	1.41E+03	2.75E+02	1.24E+03
	19.	1.34E+03	1.37E+03	1.38E+03	1.42E+03	1.32E+03	1.43E+03	3.44E+02	1.44E+03
	En kötü	1.38E+03	1.42E+03	1.40E+03	1.46E+03	1.35E+03	1.48E+03	4.21E+02	1.58E+03
	Ortalama	1.32E+03	1.36E+03	1.36E+03	1.40E+03	1.27E+03	1.41E+03	2.90E+02	1.16E+03
	Std	3.75E+01	3.66E+01	2.89E+01	3.39E+01	5.05E+01	2.78E+01	6.09E+01	3.56E+02
1.00E+04	En iyi	1.04E+03	1.04E+03	1.03E+03	9.47E+02	9.08E+02	1.00E+03	2.01E+02	2.01E+02
	7.	1.07E+03	1.05E+03	1.06E+03	1.03E+03	9.41E+02	1.06E+03	2.01E+02	2.02E+02
	Median	1.09E+03	1.05E+03	1.07E+03	1.04E+03	9.54E+02	1.09E+03	2.01E+02	2.02E+02
	19.	1.09E+03	1.07E+03	1.09E+03	1.05E+03	9.65E+02	1.11E+03	2.01E+02	2.03E+02
	En kötü	1.12E+03	1.10E+03	1.10E+03	1.06E+03	9.84E+02	1.16E+03	2.02E+02	2.05E+02
	Ortalama	1.08E+03	1.06E+03	1.07E+03	1.04E+03	9.52E+02	1.09E+03	2.01E+02	2.02E+02
	Std	1.71E+01	1.54E+01	1.89E+01	2.47E+01	1.69E+01	3.72E+01	2.92E-01	8.16E-01
1.00E+05	En iyi	9.88E+02	9.77E+02	9.76E+02	5.00E+02	8.77E+02	5.82E+02	2.01E+02	2.00E+02
	7.	9.90E+02	9.80E+02	9.80E+02	5.00E+02	8.86E+02	5.89E+02	2.01E+02	2.01E+02
	Median	9.91E+02	9.82E+02	9.81E+02	9.92E+02	8.89E+02	5.98E+02	2.01E+02	2.01E+02
	19.	9.93E+02	9.84E+02	9.85E+02	1.01E+03	8.92E+02	6.07E+02	2.01E+02	2.01E+02
	En kötü	1.00E+03	9.88E+02	9.89E+02	1.01E+03	9.06E+02	6.12E+02	2.01E+02	2.02E+02
	Ortalama	9.92E+02	9.82E+02	9.82E+02	8.35E+02	8.90E+02	5.97E+02	2.01E+02	2.01E+02
	Std	3.00E+00	3.07E+00	3.70E+00	2.32E+02	5.95E+00	1.01E+01	2.19E-02	4.54E-01
5.00E+05	En iyi	9.80E+02	9.66E+02	9.67E+02	5.00E+02	8.59E+02	5.58E+02	2.00E+02	2.00E+02
	7.	9.84E+02	9.69E+02	9.69E+02	5.00E+02	8.74E+02	5.66E+02	2.01E+02	2.01E+02
	Median	9.86E+02	9.70E+02	9.69E+02	9.92E+02	8.76E+02	5.70E+02	2.01E+02	2.01E+02
	19.	9.87E+02	9.71E+02	9.70E+02	1.01E+03	8.78E+02	5.74E+02	2.01E+02	2.01E+02
	En kötü	9.90E+02	9.73E+02	9.72E+02	1.01E+03	8.84E+02	5.86E+02	2.01E+02	2.02E+02
	Ortalama	9.85E+02	9.70E+02	9.70E+02	8.34E+02	8.75E+02	5.71E+02	2.01E+02	2.01E+02
	Std	2.12E+00	1.38E+00	1.22E+00	2.31E+02	4.83E+00	7.12E+00	3.27E-02	4.54E-01
	MTE	9.85E+02	9.70E+02	9.70E+02	8.34E+02	8.75E+02	5.71E+02	2.01E+02	2.01E+02
	STE	2.12E+00	1.38E+00	1.22E+00	2.31E+02	4.83E+00	7.12E+00	3.27E-02	4.54E-01

Tablo 4.39. ABC algoritmasının karmaşıklığı

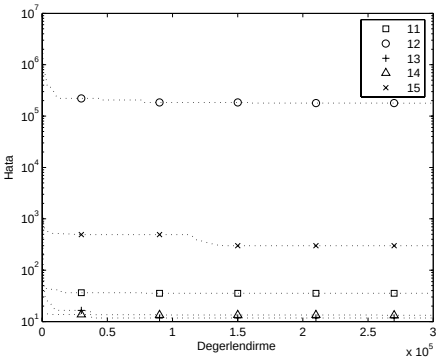
	T0	T1	$\hat{T}2$	Complexity
D=10	0.37499985191971	0.84299985319376	7.62500013224781	18.0853412190309
D=30		0.890999869443476	22.5620004348457	57.7893576609787
D=50		0.938000343739986	45.5779997399077	117.08004553051



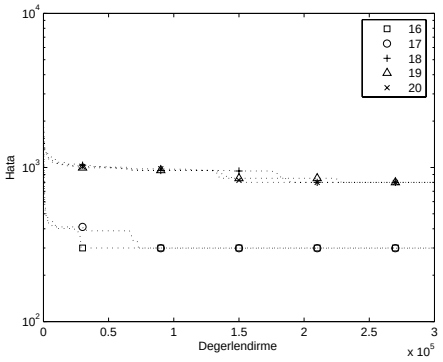
(a) 1-5 numaralı problemler



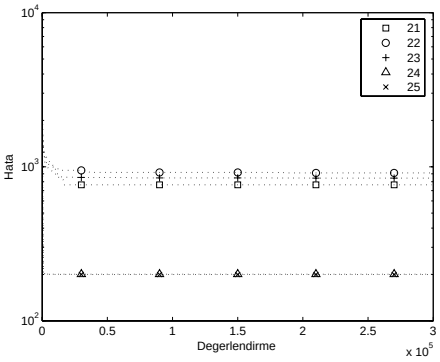
(b) 6-10 numaralı problemler



(c) 11-15 numaralı problemler



(d) 16-20 numaralı problemler



(e) 21-25 numaralı problemler

Şekil 4.10. 30 boyutlu durum için 25 koşmanın median değerine ait koşmanın logaritmik ölçekli yakınsama grafikleri

Tablo 4.40. 1-5 numaralı karma problemler için tüm algoritmaların sonuçları, D:10, Koloni Büyüklüğü: 20, Çevrim Sayısı: 5000, PSO-RDL: [8], DMS-PSO: [9], SPC-PNX [10], DE: [11], SADE [12], restartCMA-ES: [13]

		F1	F2	F3	F4	F5
PSO-RDL	Ort.	2.50E-14	1.77E-13	9.68E-02	2.47E-07	2.09E-07
	std.	2.88E-14	2.03E-13	3.34E-01	7.07E-07	7.22E-07
DMS-PSO	Ort.	0.00E+00	1.30E-13	7.01E-09	1.89E-03	1.14E-06
	std.	0.00E+00	1.56E-13	2.66E-09	1.89E-03	2.18E-06
SPC-PNX	Ort.	8.90E-09	9.63E-09	1.08E+05	9.38E-09	9.15E-09
	std.	9.39E-10	3.30E-10	8.72E+04	6.33E-10	6.32E-10
DE	Ort.	0.00E+00	0.00E+00	1.94E-06	9.09E-15	0.00E+00
	std.	0.00E+00	0.00E+00	4.63E-06	3.15E-14	0.00E+00
SaDE	Ort.	0.00E+00	1.05E-13	1.67E-05	1.42E-05	1.23E-02
	std.	0.00E+00	5.11E-13	3.12E-05	7.09E-05	1.46E-02
restartCMA-ES	Ort.	5.20E-09	4.70E-09	5.60E-09	5.02E-09	6.58E-09
	std.	1.94E-09	1.56E-09	1.93E-09	1.71E-09	2.17E-09
ABC	Ort.	4.89E-17	4.81E-17	2.50E+03	1.50E-16	5.82E+01
(SF=1)	std.	7.23E-18	5.89E-18	8.68E+02	5.47E-17	4.11E+01
ABC (SF=0.7)	Ort.	8.85E-17	1.04E-16	1.37E+03	1.29E-04	1.32E+02
	std.	5.36E-17	4.90E-17	5.54E+02	3.05E-04	1.09E+02
ABC (SF=0.5)	Ort.	1.16E-16	1.14E-16	9.90E+02	1.25E-02	3.26E+02
	std.	4.91E-17	4.60E-17	4.62E+02	1.25E-02	3.30E+02
ABC (SF=0.3)	Ort.	1.52E-16	1.83E-16	5.88E+02	1.29E-01	8.07E+02
	std.	3.43E-17	8.43E-17	2.79E+02	9.46E-02	8.12E+02
ABC (MR=1)	Ort.	4.64E-17	4.62E-17	1.52E+03	4.61E-17	7.28E-13
	std.	6.91E-18	9.16E-18	7.67E+02	6.95E-18	1.15E-12
ABC (MR=0.8)	Ort.	8.80E-17	1.10E-16	3.57E+03	8.24E-17	1.82E-12
	std.	4.84E-17	4.66E-17	1.11E+03	4.73E-17	1.46E-12
ABC (MR=0.6)	Ort.	4.75E-17	4.58E-17	3.72E+03	4.82E-17	1.89E-12
	std.	6.79E-18	7.75E-18	1.29E+03	6.45E-18	1.74E-12
ABC (MR=0.4)	Ort.	4.48E-17	4.30E-17	3.58E+03	4.48E-17	6.18E-12
	std.	8.91E-18	9.77E-18	1.60E+03	7.31E-18	5.52E-12
ABC (MR=0.2)	Ort.	4.80E-17	4.89E-17	3.22E+03	6.98E-17	1.66E+00
	std.	5.46E-18	4.62E-18	2.14E+03	4.05E-17	1.61E+00
ABC(ASF-MR:0.9)	Ort.	4.94E-17	4.76E-17	5.23E-04	4.41E-12	4.12E-04
	std.	4.29E-18	8.58E-18	5.23E-04	1.32E-11	6.31E-04

Tablo 4.41. 6-10 numaralı karma problemler için tüm algoritmaların sonuçları, D:10, Koloni Büyüklüğü: 20, Çevrim Sayısı: 5000, PSO-RDL: [8], DMS-PSO: [9], SPC-PNX [10], DE: [11], SADE [12], restartCMA-ES: [13]

		F6	F7	F8	F9	F10
PSO-RDL	Ort.	9.57E-01	5.73E-02	2.00E+01	1.25E+01	3.86E+01
	std.	1.74E+00	4.66E-02	2.17E-04	8.17E+00	1.80E+01
DMS-PSO	Ort.	6.89E-08	4.52E-02	2.00E+01	0.00E+00	3.62E+00
	std.	3.19E-07	3.26E-02	5.54E-09	0.00E+00	8.55E-01
SPC-PNX	Ort.	1.89E+01	8.26E-02	2.10E+01	4.02E+00	7.30E+00
	std.	4.00E+01	6.24E-02	5.79E-02	2.27E+00	5.21E+00
DE	Ort.	1.59E-01	1.46E-01	2.04E+01	9.55E-01	1.25E+01
	std.	7.97E-01	1.38E-01	7.58E-02	9.73E-01	7.96E+00
SaDE	Ort.	1.20E-08	1.99E-02	2.00E+01	0.00E+00	4.97E+00
	std.	1.93E-08	1.07E-02	5.39E-08	0.00E+00	1.69E+00
restartCMA-ES	Ort.	4.87E-09	3.31E-09	2.00E+01	2.39E-01	7.96E-02
	std.	1.66E-09	2.02E-09	3.89E-03	4.34E-01	2.75E-01
ABC (SF=1)	Ort.	3.31E+00	2.52E-01	2.03E+01	4.87E-17	2.22E+01
	std.	5.18E+00	9.29E-02	6.07E-02	1.79E-17	7.32E+00
ABC (SF=0.7)	Ort.	8.89E+00	2.28E-01	2.03E+01	4.00E-01	3.77E+01
	std.	1.40E+01	7.54E-02	5.67E-02	4.86E-01	8.98E+00
ABC (SF=0.5)	Ort.	9.48E+00	2.27E-02	2.03E+01	3.21E-01	5.13E+01
	std.	9.25E+00	8.10E-02	7.24E-02	4.62E-01	8.56E+00
ABC (SF=0.3)	Ort.	1.37E+01	2.49E-01	2.03E+01	5.28E+00	7.50E+01
	std.	1.36E+01	1.08E-01	5.88E-01	2.32E+00	1.65E+01
ABC (MR=1)	Ort.	1.27E+00	4.84E-01	2.04E+01	1.20E+01	2.92E+01
	std.	1.38E+00	9.07E-02	7.50E-02	3.56E+00	4.33E+00
ABC (MR=0.8)	Ort.	6.02E+00	5.20E-01	2.03E+01	2.35E+01	3.85E+01
	std.	1.62E+01	8.40E-02	6.33E-02	3.59E+00	4.09E+00
ABC (MR=0.6)	Ort.	4.95E+00	3.01E-01	2.04E+01	1.83E-01	2.41E+01
	std.	4.00E+00	7.15E-02	7.40E-02	4.31E-01	4.37E+00
ABC (MR=0.4)	Ort.	2.76E+00	1.27E-02	2.04E+01	5.29E-17	1.43E+01
	std.	3.62E+00	2.91E-01	7.67E-02	2.33E-17	2.61E+00
ABC (MR=0.2)	Ort.	2.51E+00	9.48E-02	2.04E+01	5.00E-17	1.69E+00
	std.	5.61E+00	3.73E-02	6.79E-02	1.93E-17	4.62E+00
ABC(ASF-MR:0.9)	Ort.	3.36E+00	6.52E-02	2.00E+01	7.65E+00	1.25E+01
	std.	2.49E+00	1.44E-02	1.28E-02	2.15E+00	2.93E+00

Tablo 4.42. 11-15 numaralı karma problemler için tüm algoritmaların sonuçları, D:10, Koloni Büyüklüğü: 20, Çevrim Sayısı: 5000, PSO-RDL: [8], DMS-PSO: [9], SPC-PNX [10], DE: [11], SADE [12], restartCMA-ES: [13]

		F11	F12	F13	F14	F15
PSO-RDL	Ort.	5.58E+00	1.31E+02	8.87E-01	3.78E+00	2.71E+02
	std.	1.42E+00	4.50E+02	4.06E-01	3.44E-01	1.59E+02
DMS-PSO	Ort.	4.62E+00	2.40E+00	3.69E-01	2.36E+00	4.85E+00
	std.	5.84E-01	4.36E+00	5.64E-02	3.38E-01	1.34E+01
SPC-PNX	Ort.	1.91E+00	2.60E+02	8.38E-01	3.05E+00	2.54E+02
	std.	1.16E+00	4.89E+02	2.69E-01	4.37E-01	1.51E+02
DE	Ort.	8.47E-01	3.17E+01	9.77E-01	3.45E+00	2.59E+02
	std.	1.40E+00	4.20E+01	4.67E-01	4.40E-01	1.83E+02
SaDE	Ort.	4.89E+00	4.50E-07	2.20E-01	2.92E+00	3.20E+01
	std.	6.62E-01	8.51E-07	4.11E-02	2.06E-01	1.11E+02
restartCMA-ES	Ort.	9.34E-01	2.93E+01	6.96E-01	3.01E+00	2.28E+02
	std.	9.00E-01	1.42E+02	1.50E-01	3.49E-01	6.80E+01
ABC (SF=1)	Ort.	5.46E+00	9.85E+01	2.96E-02	3.41E+00	1.53E-01
	std.	5.84E-01	6.16E+01	2.12E-02	1.53E-01	3.34E-01
ABC (SF=0.7)	Ort.	5.67E+00	1.16E+02	5.87E-02	3.44E+00	2.25E+00
	std.	7.24E-01	1.02E+02	2.80E-02	1.73E-01	9.89E+00
ABC (SF=0.5)	Ort.	5.75E+00	1.10E+02	1.03E-01	3.36E+00	5.29E-01
	std.	7.21E-01	8.68E+01	5.83E-02	1.45E-01	2.50E+00
ABC (SF=0.3)	Ort.	6.09E+00	1.52E+02	1.64E-01	3.53E+00	1.17E+01
	std.	9.60E-01	3.41E+02	8.43E-02	1.59E-01	2.08E+01
ABC (MR=1)	Ort.	8.38E+00	1.30E+02	1.77E+00	3.66E+00	3.01E+02
	std.	6.64E-01	1.76E+02	3.52E-01	1.45E-01	7.66E+01
ABC (MR=0.8)	Ort.	6.88E+00	1.72E+03	1.48E+00	3.59E+00	2.87E+02
	std.	5.20E-01	5.35E+02	3.36E-01	1.12E-01	3.49E+01
ABC (MR=0.6)	Ort.	6.79E+00	4.22E+02	7.52E-01	3.56E+00	1.99E+02
	std.	5.19E-01	2.10E+01	3.17E-01	1.44E-01	4.73E+01
ABC (MR=0.4)	Ort.	5.58E+00	2.22E+02	2.05E-01	3.49E+00	1.08E+02
	std.	6.38E-01	1.80E+02	9.97E-02	1.27E-01	4.34E+01
ABC (MR=0.2)	Ort.	5.26E+00	1.55E+02	6.91E-02	3.35E+00	1.71E+01
	std.	5.84E-01	6.58E+01	5.28E-02	2.12E-01	3.15E+01
ABC(ASF-MR:0.9)	Ort.	2.39E+00	1.71E+01	6.32E-01	3.24E+00	2.41E+02
	std.	1.04E+00	1.46E+01	1.74E-01	2.95E-01	8.64E+01

Tablo 4.43. 16-20 numaralı karma problemler için tüm algoritmaların sonuçları, D:10, Koloni Büyüklüğü: 20, Çevrim Sayısı: 5000, PSO-RDL: [8], DMS-PSO: [9], SPC-PNX [10], DE: [11], SADE [12], restartCMA-ES: [13]

		F16	F17	F18	F19	F20
PSO-RDL	Ort.	2.20E+02	2.22E+02	1.02E+03	9.85E+02	9.59E+02
	std.	1.74E+02	1.00E+02	1.19E+02	1.02E+02	1.06E+02
DMS-PSO	Ort.	9.48E+01	1.10E+02	7.61E+02	7.14E+02	8.22E+02
	std.	1.01E+01	4.35E+00	1.85E+02	2.01E+02	4.59E+01
SPC-PNX	Ort.	1.10E+02	1.19E+02	4.40E+02	3.80E+02	4.40E+02
	std.	9.87E+00	1.07E+01	2.25E+02	1.87E+02	2.29E+02
DE	Ort.	1.13E+02	1.15E+02	4.00E+02	4.20E+02	4.60E+02
	std.	1.80E+01	2.01E+01	2.04E+02	2.18E+02	2.38E+02
SaDE	Ort.	1.01E+02	1.14E+02	7.19E+02	7.05E+02	7.13E+02
	std.	6.17E+00	9.97E+00	2.09E+02	1.90E+02	2.01E+02
restartCMA-ES	Ort.	9.13E+01	1.23E+02	3.32E+02	2.26E+02	3.00E+02
	std.	3.49E+00	2.09E+01	1.12E+02	9.93E+01	0.00E+00
ABC (SF=1)	Ort.	1.75E+02	1.96E+02	4.46E+02	4.51E+02	4.38E+02
	std.	2.11E+01	2.25E+01	4.83E+01	4.09E+01	3.30E+01
ABC (SF=0.7)	Ort.	1.77E+02	2.04E+02	3.86E+02	4.31E+02	4.09E+02
	std.	1.64E+01	2.47E+01	9.60E+01	3.16E+01	2.66E+01
ABC (SF=0.5)	Ort.	2.12E+02	2.33E+02	4.16E+02	4.31E+02	4.39E+02
	std.	2.49E+01	2.87E+01	4.40E+01	4.66E+01	2.97E+01
ABC (SF=0.3)	Ort.	2.93E+02	3.06E+02	4.75E+02	4.78E+02	4.61E+02
	std.	4.40E+01	5.80E+01	1.32E+02	4.73E+01	4.80E+01
ABC (MR=1)	Ort.	1.98E+02	2.02E+02	5.50E+02	5.16E+02	5.04E+02
	std.	1.20E+01	1.75E+01	1.13E+02	1.19E+02	1.07E+02
ABC (MR=0.8)	Ort.	2.25E+02	2.41E+02	4.62E+02	4.83E+02	4.62E+02
	std.	1.17E+01	2.31E+01	2.15E+01	2.74E+01	3.12E+01
ABC (MR=0.6)	Ort.	2.02E+02	2.18E+02	5.03E+02	5.03E+02	5.19E+02
	std.	1.79E+01	1.55E+01	4.60E+01	6.14E+01	5.19E+01
ABC (MR=0.4)	Ort.	1.75E+02	1.90E+02	4.61E+02	5.20E+02	5.18E+02
	std.	1.80E+01	1.44E+01	4.14E+01	2.76E+01	3.23E+01
ABC (MR=0.2)	Ort.	1.59E+02	1.86E+02	4.33E+02	4.34E+02	4.24E+02
	std.	1.71E+01	1.66E+01	4.21E+01	5.99E+01	3.22E+01
ABC(ASF-MR:0.9)	Ort.	1.85E+02	1.75E+02	3.71E+02	3.92E+02	4.30E+02
	std.	3.62E+01	3.27E+01	5.78E+01	6.43E+01	7.00E+01

Tablo 4.44. 21-25 numaralı karma problemler için tüm algoritmaların sonuçları, D:10, Koloni Büyüklüğü: 20, Çevrim Sayısı: 5000, PSO-RDL: [8], DMS-PSO: [9], SPC-PNX [10], DE: [11], SADE [12], restartCMA-ES: [13]

		F21	F22	F23	F24	F25
PSO-RDL	Ort.	9.94E+02	8.87E+02	1.08E+03	7.20E+02	1.76E+03
	std.	3.27E+02	7.12E+01	2.87E+02	3.96E+02	1.54E+01
DMS-PSO	Ort.	5.36E+02	6.92E+02	7.30E+02	2.24E+02	3.66E+02
	std.	2.18E+02	1.56E+02	1.66E+02	8.31E+01	1.51E+02
SPC-PNX	Ort.	6.80E+02	7.49E+02	5.76E+02	2.00E+02	4.06E+02
	std.	2.69E+02	9.37E+01	8.22E+01	0.00E+00	2.38E-01
DE	Ort.	4.92E+02	7.18E+02	5.72E+02	2.00E+02	9.23E+02
	std.	4.00E+01	1.58E+02	4.48E+01	0.00E+00	3.40E-01
SaDE	Ort.	4.64E+02	7.32E+02	6.64E+02	2.00E+02	3.76E+02
	std.	1.58E+02	9.15E+01	1.53E+02	0.00E+00	3.15E+00
restartCMA-ES	Ort.	5.00E+02	7.29E+02	5.59E+02	2.00E+02	3.74E+02
	std.	0.00E+00	3.18E-13	6.86E+00	3.24E-11	3.22E+00
ABC (SF=1)	Ort.	4.07E+02	8.59E+02	4.98E+02	2.02E+02	2.00E+02
	std.	5.89E+01	7.29E+01	4.44E+01	5.76E-03	4.20E-03
ABC (SF=0.7)	Ort.	3.82E+02	7.79E+02	5.05E+02	2.02E+02	2.00E+02
	std.	6.89E+01	1.73E+02	3.78E+01	8.26E-03	2.52E-03
ABC (SF=0.5)	Ort.	3.80E+02	8.26E+02	5.08E+02	2.02E+02	2.00E+02
	std.	6.82E+01	1.64E+02	3.49E+01	7.62E-03	3.20E-03
ABC (SF=0.3)	Ort.	3.84E+02	9.20E+02	5.01E+02	2.02E+02	2.00E+02
	std.	6.43E+01	1.31E+02	3.87E+01	8.32E-03	2.51E-03
ABC (MR=1)	Ort.	7.76E+02	8.38E+02	8.00E+02	2.02E+02	2.00E+02
	std.	8.00E+01	3.40E+00	1.39E+02	4.26E-03	4.86E-04
ABC (MR=0.8)	Ort.	7.58E+02	8.40E+02	8.20E+02	2.02E+02	2.00E+02
	std.	7.62E+01	4.41E+00	9.93E+01	5.38E-03	7.01E-04
ABC (MR=0.6)	Ort.	7.08E+02	8.41E+02	7.49E+02	2.02E+02	2.00E+02
	std.	1.14E+02	3.85E+00	1.31E+02	5.42E-03	9.16E-04
ABC (MR=0.4)	Ort.	5.56E+02	8.43E+02	5.77E+02	2.02E+02	2.00E+02
	std.	9.68E+01	5.97E+00	6.66E+01	6.28E-03	1.94E-03
ABC (MR=0.2)	Ort.	4.39E+02	7.89E+02	5.09E+02	2.02E+02	2.00E+02
	std.	1.13E+02	1.32E+02	3.86E+01	5.79E-03	2.81E-03
ABC(ASF-MR:0.9)	Ort.	5.81E+02	8.20E+02	5.66E+02	2.02E+02	2.00E+02
	std.	1.46E+02	6.11E+00	4.07E+01	8.45E-03	4.13E-03

5. BÖLÜM

SINIRLAMALI OPTİMİZASYON PROBLEMLERİ ÜZERİNDE YAPAY ARI KOLONİSİ ALGORİTMASININ PERFORMANS ANALİZİ

5.1. Giriş

Yapısal optimizasyon, mühendislik tasarımı, VLSI tasarımı, ekonomi, atama ve yerleşim problemleri sınırlamalı optimizasyon problemlerine birkaç örnektir [215]. Sınırlamalı optimizasyon, (5.3) ile tanımlı eşitsizlik ve (5.4) ile tanımlı eşitlik sınırlamalarına tabi olarak (5.1) ile tanımlı $f(\vec{x})$ amaç fonksiyonunu minimize eden $\vec{x}$ değerini bulma işlemidir:

$$\text{minimize } f(\vec{x}), \qquad \vec{x} = (x_1, \ldots, x_n) \in R^n \tag{5.1}$$

$$l_i \leq x_i \leq u_i, \quad i = 1, \ldots, n \tag{5.2}$$

$$g_j(\vec{x}) \leq 0, \ for\ j = 1, \ldots, q \tag{5.3}$$

$$h_j(\vec{x}) = 0, \ for\ j = q+1, \ldots, m \tag{5.4}$$

Amaç fonksiyonu f, n-boyutlu R^n'de bulunan S araştırma uzayında ($S \subseteq R^n$) tanımlıdır. Değişken domeni değişkenlerin alt ve üst sınırları (Eşitlik 5.2) ile belirlenmektedir . Kabul edilebilir bölge $F \subseteq S$, m tane ($m \geq 0$) daha sınırlama fonksiyonuna tabidir (5.3 ve 5.4) ve $\vec{x}$, ($\vec{x} \in F \in S$) kabul edilebilir bölgede tanımlıdır. $\vec{x} \in F$ ise $g_k(\vec{x}) = 0$ şartını sağlayan g_k sınırlamaları, $\vec{x}$'de aktif olarak adlandırılır. Eşitlik sınırlamaları h_j fonksiyonları ise S'in her bölgesinde aktiftirler [216].

Sınırlamalı optimizasyon problemlerini çözmek zordur ve bu zorlukta hangi parametrenin (doğrusal, doğrusal olmayan, aktif sınırlamaların sayısı, $\rho = |F| / |S|$, fonksiyonun türü, değişken sayısı) etkili olduğu gösterilememiştir [217].

Bu bölümde ABC algoritmasının sınırlamalı problemlerin çözümünde kullanılılabilecek şekilde uyarlanması anlatılacak, literatürde mevcut test problemlerinin çözümünde kullanılarak sonuçlar verilecek ve bilinen diğer tekniklerle kıyaslanacaktır. ABC algoritması uyarlanırken kabul edilebilir bölgede çözümlerin tercih edilmesi amacıyla temel ABC algoritmasının seleksiyon biriminde kullanılan aç gözlü seleksiyon operatörü Deb'in seleksiyon mekanizması [218] ile değiştirilmiştir. Ayrıca kabul edilebilir bölgede olmayan çözümlerinde farklılık sağlamak amacıyla popülasyonda bulunmalarına izin verildiği için gözcü arıların bölge seçerken kullandıkları olasılık değeri sadece uygunluk değerine bağlı olarak değil kabul edilebilir bölgede olmayan çözümlere de bir olasılık değeri atanabilmesi gayesiyle sınırlamaların çiğnenme miktarları da olasılık hesabında kullanılmaktadır. ABC algoritması, başlatılması esnasında popülasyondaki tüm bireylerin kabul edilebilir bölgede üretilmesini gerektirmez. Bu özellik ABC'yi zaman kaybettirici bir süreçten kurtarmaktadır. Yani bu özellik ABC için bir avantajdır. Ayrıca ABC, amaç fonksiyonuna sınırlama değerlerinin eklenmesi şeklinde çalışan ceza (penalty) yaklaşımını da kullanmamaktadır. Ceza yaklaşımı da her bir sınırlamanın amaç fonksiyonuna katkısı yansıtılırken ağırlık katsayılarının optimum olarak belirlenmesi gerekmektedir. Bu işlemin kendisi de ayrı bir optimizasyon problemi olarak ele alınabilir. Bu ceza yönteminin dezavantajıdır.

Bu bölümde literatürde oldukça popüler olan ve [219]'deki çalışmada kullanılan 13 sınırlamalı test problemi üzerinde simülasyon çalışmalar yapılmış ve sonuçlar [14, 219–222] çalışmaları ile kıyaslanmıştır.

5.2. Literatürde Mevcut İlgili Çalışmalar

Bir çok optimizasyon algoritması öncelikle sınırlaması olmayan (sınırlamasız) optimizasyon problemlerini çözmek için geliştirilmişlerdir. Daha sonra bu algoritmalar sınırlamalı problemlere uygulanabilir hale çevrilmişlerdir. Sınırlama

ele alış tekniği kullanmanın amacı araştırmayı kabul edilebilir bölgeye doğru kaydırmaktır. Sınırlamaları ele alış metotları Koziel ve Michalewicz tarafından dört kategoride incelenmiştir [220]:

(i) Çözümleri kabul edilebilir bölgede tutan metotlar [223, 224];
(ii) Ceza terimine dayalı metotlar [225–228];
(iii) Kabul edilebilir bölge ve dışındaki bölgeler arasında ayrım yapan metotlar [218, 229–232];
(iv) Diğer karma metotlar [233–236]

(i) Çözümleri kabul edilebilir bölgede tutan metotlar kategorisinde iki grup bulunmaktadır. Bunlardan birinci grup kabul edilebilir olmayan çözümleri kabul edilebilir çözümlere dönüştürmek için bazı operatörler kullanmaktadır [223]. Bu metot, sınırlamaların doğrusal olduğunu ve başlangıç çözümlerinin kabul edilebilir bölgede bulunduğunu kabul etmektedir. Bu metodun zayıf tarafı konveks olmayan araştırma uzaylarında çalışma yeteneğiyle ilgilidir. Bu kategorideki ikinci grup ise kabul edilebilir bölgenin sınırlarını araştıran yaklaşımları içermektedir. Kabul edilebilir bölge ile diğer bölge arasında kritik bir seviye aracılığıyla bu sınır belirlenmektedir [224].

(ii) Ceza terimine dayalı metotlar sınırlama değerlerini aşan miktarlarını amaç fonksiyonuna eklemek suretiyle sınırlamalı problemi sınırlamasız probleme dönüştürürler. Ceza terimlerinin değeri çok büyükse, optimizasyon algoritmaları bölgesel minimumlara takılmakta; küçük olduğunda ise kabul edilebilir bölgenin sezilmesi zorlaşmaktadır. Basitlik ve doğrudan uygulanabilirlik avantajlarının yanında her bir sınırlamanın ağırlığının ayarlanmasından kaynaklanan zorlukları bulunmaktadır [237]. Ceza teriminin statik olması [226], dinamik olması [227, 228], adaptif olması [225] gibi durumlar mevcuttur.

(iii) Kabul edilebilir bölgede çözümleri araştıran metotlar kategorisinde, kabul edilebilir bölge ile diğer bölge arasında ayrım yapmaya odaklanan bir kaç yaklaşım mevcuttur. Birinci yaklaşım, yeterince kabul edilebilir çözüm oluşuncaya kadar sırayla her bir sınırlamayı dikkate almaktadır [230]. İkinci yaklaşım ise kabul

edilebilir bölgedeki her bir çözümün diğer bölgedeki çözüme tercih edilmesi fikrine dayanmaktadır. [218, 229, 231]. Üçüncü yaklaşım, kabul edilebilir bölgede olmayan çözümlerin onarılması şeklinde çalışır [232].

(iv) Son kategori, evrimsel hesaplama teknikleri ile deterministik süreçleri birleştiren karma metotları içermektedir [233–236].

EA, ES, PSO ve DE gibi stokastik optimizasyon algoritmalarına sınırlamalarla başa çıkabilen tekniklerin eklenmesiyle sınırlamalı optimizasyonda başarılı algoritmalar geliştirilmiştir [238–245].

Sınırlamalı optimizasyon problemlerini PSO algoritmasını kullanarak çözmek için araştırmacılar farklı yaklaşımlar benimsemişlerdir. Parsopoulos ve ark. durağan olmayan çok-aşama içeren atamalı ceza fonksiyonuna dayalı bir yaklaşım kullanmışlar ve iyi bilinen, yaygın olarak kullanılan test problemleri üzerinde çalışmalar yapmışlardır [215]. Ayrıca UPSO (Unified Particle Swarm Optimization) adı verilen bir yaklaşım öne sürerek sınırlamalı problemlerin çözümünde kullanmışlardır [241]. UPSO metodunda araştırmayı kabul edilebilir bölgede tutmak amacıyla ceza terimi kullanarak sınırlamaları dikkate almışlardır. PSO'nun bölgesel ve küresel çeşitleri ile karşılaştırma yapmışlardır. Hu ve Eberhart çözümlerin kabul edilebilirliğini muhafaza eden bir strateji ile PSO'yu sınırlamalı problemlere uygulamışlardır [242]. Hu ve ark. sınırlamalı mühendislik optimizasyon problemlerine değiştirilmiş bir PSO'yu uygulamışlardır [243]. Hu ve ark.larının önerdikleri bu değiştirilmiş PSO bir grup kabul edilebilir çözümle çalışmaya başlamakta ve sonra bir kabul edilebilirlik fonksiyosunu ile yeni bulunan çözümlerin tüm sınırlamaları sağlayıp sağlamadığı kontrol edilmektedir. Kabul edilebilir çözümlerle çalışmaya başladığından eşitlik sınırlamaların sağlayan rastgele çözümler bulmak oldukça güç bir durumdur. Hatta tüm uzaya göre kabul edilebilir bölgenin çok dar olduğu bazı fonksiyonlar için neredeyse imkansızdır. Kabul edilebilir çözümlerle algoritmaların başlatılması oldukça fazla zaman alan bir süreçtir. Zavala ve ark., Particle Evolutionary Swarm Optimization (PESO) adında kabul edilebilirlik kuralına dayalı sınırlama ele alış makanizması, bölgesel

en iyiyi takip eden bir halka topolojisine dayalı bir seleksiyon mekanizması; ve farklılığı ve keşif yeteneğini korumaya yardımcı iki değişim operatörü kullanan bir PSO geliştirmişlerdir [244].

DE algoritması sınırlamalı problemlerin çözümü için çeşitli çalışmalarda değişik yaklaşımlarla bir arada kullanılmıştır. Storn tüm çözümleri kabul edilebilir yapmak için öncelikle sınırlamaları serbest bırakan bir adaptif mekanizma önermiştir. Buna ilave olarak Storn bir çözümün çok fazla iterasyonlar boyunca gereksiz yere popülasyonda kalmasının önüne geçmek amacıyla yaşa dayalı bir yaklaşım önermiştir [246]. Lampinen ve Zelinka, DE algoritmasını sınırlamalı mühendislik problemlerinin çözümünde kullanmışlardır [247]. Esnek-sınırlama adı verilen statik ceza terimi yaklaşımı ile sınırlamaları dikkate almışlardır. Lampinen, yine sınırlamalı problemlerinin çözümüne yönelik olarak DE algoritmasını geliştirmiştir [248]. Yaklaşım, orjinal DE seleksiyon şeması yerine ebeveyn ve çocugu arasında tercih yapan bir seleksiyon şeması kullanmaktadır. Lampinen'in önerdiği bu şema Deb'in kabul edilebilirlik kurallarına benzemektedir [218]. Zhang ve Xie, DE ile PSO'yu birleştiren DEPSO adında bir karma teknik önermişlerdir. DEPSO çan şeklinde populasyondaki farklılığa bağlı olarak çalışan bir mutasyon gerçekleştirmektedir [245]. Mezura-Montes ve ark. Deb'in kurallarını seleksiyon kriteri olarak kullanan DE'ya dayalı bir yaklaşım önermişlerdir [249]. Ayrıca Mezura-Montes ve ark. DE algoritmasını yeni üretilen bireylerin o anki popülasyona yeniden eklenebilmesine izin veren bir yol izlemektedir [250].

Koziel ve Michalewicz, çözücü (decoder) tabanlı bir sınırlama ele alış yaklaşımını incelemiştir. Bu yaklaşımda, n-boyutlu bir küp ile kabul edilebilir bölge arasında eş yapılı haritalama (homomorphous mapping, HM) ile evrimsel algoritma birarada kullanılmıştır [220]. Metotda eşi yapılı haritalama ile sınırlamalı problem sınırlamasız bir probleme dönüştürülmektedir. Evrimsel algoritmalarla ilgili bir diğer çalışma Hamida and Schoenauer tarafından önerilen popülasyon seviyesinde adaptif ceza terimi oluşturan Adaptif Ayrıcılı (Adaptive Segregational Constraint Handling Evolutionary Algorithm, ASCHEA) EA'dır [222]. ASCHEA, yeniden oluşum operatöründe sınırlamaya göre çalışan bir eş seçim operatörü ve belli sayıda kabul edilebilir çözümü öne çıkaran bir seleksiyon operatörü kullanır. Eşitlik

sınırlamaları içinde öncelikle geniş bir uzaydan başlayarak yavaş yavaş uzayı daraltan bir strateji izlenir [222].

Sınırlamalı problemlerin çözümü için önerilen ES'lerde, Runarsson ve Yao amaç fonksiyonu ve ceza terimleri arasında bir denge kurmaya çalışarak stochastic ranking (SR) yapan bir yaklaşım önermişlerdir [219]. Yaklaşım sınırlamalı bir problemi sınırlamalarında amaç fonksiyonu gibi düşünüldüğü çok amaçlı bir optimizasyon problemi şeklinde ele alır ve ceza terimi katsayısı belirlenmesine gerek duymaz. Başka bir çalışmada da Runarsson ve Yao farklı sınırlama ele alış metotları kullanılması durumunda ES'nin performansını analiz etmişlerdir [221]. Bu çalışmada kabul edilebilir çözümleri amaç fonksiyonu değerine göre ardından kabul edilebilir olmayanları da amaç fonksiyonu değerine göre sıralayan ceza üstü (over-penaly approach, OPA) yaklaşımı tanımlanmıştır ve araştırmanın koordinat eksenine kayması problemini gideren bir araştırma mekanizması kullanarak ES'nin performansının artırıldığı bir modifikasyon yapmışlardır (Improved Stochastic Ranking,ISR) [221].

Mezura-Montes ve ark. küresl doğrusal olmayan optimizasyon problemlerini çözmek amacıyla basit çok üyeli ES (Simple Multi-membered Evolution Strategy, ES) adını verdikleri bir gelişim stratejisi önermişlerdir [14]. SMES yaklaşımı ceza fonksiyonu yerine kabul edilebilir olmayan çözümlerinde popülasyonda kalmasına izin veren bir farklılık mekanizması kullanmaktadır. Kabul edilebilirlik tabanlı kıyas mekanizamasıda araştırmanın uzayın kabul edilebilir bölgelerine doğru gitmesi için kullanılmaktadır. Ayrıca, gelişim stratejilerinde kullanılan adım büyüklüğünün başlangıç değeri daha hassas araştırma yapabilmek için ilerleyen adımlarda azaltılmakta ve panmiktik yeniden oluşum operatörü (çoklu bireyden yeni birey oluşumu) ile de popülasyon bilgisinin kullanımının artırılması amaçlanmaktadır [14].

Bu bilgilerin ışığında ABC algoritması da sınırlamalı optimizasyon problemlerinin çözümünde kullanılabilir. ABC algoritmasının sınırlamasız problemler üzerinde GA, PS-ES, DE, PSO, ES algoritmalarına kıyasla gösterdiği üstün performansından dolayı [133–137, 251], bir sınırlama ele alış tekniği kullanılarak sınırlamalı problemlerin çözümü için ABC algoritması uyarlanmıştır. Kullanılan sınırlama

ele alış tekniğinin de performansı etkileyeceği bilinmektedir. ABC algoritmasında, kabul edilebilir çözümlerin olmayanlara göre her zaman daha iyi olduğu kabulune dayalı basit, hesaplama maliyeti az olan ve ince ayar gerektirmeyen Deb'in kuralları kullanılmıştır [218]. Deb'in kuralları ve ABC algoritmasında yapılan değişiklikler sonraki bölümde detaylıca verilmektedir.

5.3. Sınırlamalı Problemlere ABC Algoritmasının Uyarlanması

Sınırlamalı problemlerin doğası gereği, sınırlamaların çizdiği kabul edilebilir bölge içerisinde çözümlerin aranması veya bu sınırlar dışında arama yapılsa dahi en son bulunacak en iyi çözümün kabul edilebilir bölgede olması gerekmektedir. Sınırlamalı problemleri çözmek için geliştirilen ABC algoritmasının temel adımlarının (Algoritma 2) sınırlamasız problemler için kullanılan versiyonunun temel adımlarından farkı yoktur. Adımların iç işleyişlerinde bazı değişiklikler bulunmaktadır.

Algoritma 2 ABC algoritmasının temel adımları

1: Algoritmanın başlatılması
2: Değerlendirme
3: Çevrim=1
4: **repeat**
5: İşçi arı fazı
6: Gözcüler için olasılık değerlerinin hesaplanması
7: Gözcü arı fazı
8: Kaşif arı fazı
9: Elde edilen en iyi çözümü hafızada tut
10: Çevrim=Çevrim+1
11: **until** Çevrim=MCN

ABC algoritması ilk adımda $SN/2$ tane rastgele üretilen çözümden (yiyecek kaynağı) oluşan başlangıç popülasyonu üretir. Her bir yiyecek kaynağı x_i $(i = 1, 2, ..., SN/2)$, bir D-boyutlu vektördür. D ise optimize edilmeye çalışılan parametre sayısıdır. Başlangıç aşamasında popülasyondaki bireylerin hepsini kabul edilebilir bölgede üretmek zaaman alıcı ve hatta bazı problemler için kabul edilebilir bölgede rastgele bir tane bile çözüm üretmek oldukça zor olduğu için ABC algoritması başlangıçta üretilen çözümlerin kabul edilebilir bölgede olmasını önemsemez. Parametrelerin alt $x^j_{\min}$ ve üst $x^j_{\max}$ sınırları arasında uniform dağılımlı rastgele değerler üretir

ve her bir yiyeceğin gelişememe sayacı olan $failure_i$ vektörünü sıfırlar. $failure_i$ sayacı ileriki adımlarda çözümlerin terk edilip edilmeyeceğini belirlemek amacıyla kullanılmaktadır. Başlatılma aşaması Algoritma 3'de verilmektedir:

Algoritma 3 ABC algoritmasının başlatılma aşaması

1: **for** $i = 1$ to $SN/2$ **do**
2: **for** $j = 1$ to D **do**
3: x_i çözümü üret

$$\mathrm{x_i^j} = x^j_{\min} + rand(0,1)(x^j_{\max} - x^j_{\min}) \tag{5.5}$$

4: **end for**
5: $failure_i = 0$
6: **end for**

Başlangıç aşamasından sonra popülasyonun tüm bireylerinin uygunluğu yani kalite değeri belirlenir ve görevli arı, gözcü arı ve kaşif arıdan oluşan araştırma süreci başlar. İşçi arı fazının adımları Algoritma 4'de verilmektedir:

Algoritma 4 ABC algoritmasının görevli arı fazı

1: **for** $i = 1$ to $SN/2$ **do**
2: **for** $j = 1$ to D **do**
3: x_i çözümünün görevli arısı için (5.6) eşitliği ile yeni bir çözüm (yiyecek kaynağı) υ_i üret:

$$\upsilon_{ij} = \begin{cases} x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) & , \quad if \quad R_j < MR \\ x_{ij} & , \quad \text{otherwise} \end{cases} \tag{5.6}$$

Burada $k \in \{1, 2, \ldots SN\}$ i'den farklı rastgele bir çözüm indisi, ϕ_{ij} [-1,1] aralığında uniform dağılımlı rastgele bir sayı, R_j [0,1] aralığında uniform dağılımlı rastgele bir sayı, MR ise ABC algoritmasının değişecek parametre sayısında etkili olan [0,1] aralığında değer alabilen kontrol parametresidir.
4: **end for**
5: En az bir parametrenin değişimi gerçekleşmemişse x_i çözümünün rastgele belirlenen bir parametresini ([1,D] aralığında rastgele belirlenen j. parametreyi) eşitlik (5.7) ile değiştir

$$\upsilon_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) \tag{5.7}$$

6: υ_i çözümünün kalitesini değerlendir
7: υ_i ve x_i arasında Deb'in metodunu kullanarak seleksiyon işlemi gerçekleştir
8: x_i çözümü komşuluğundaki çözümle gelişmemişse başarısızlık sayacını güncelle, $failure_i = failure_i + 1$, aksi takdirde sayacı sıfırla $failure_i = 0$
9: **end for**

Bir görevli arı hafızasındaki yiyecek kaynağının pozisyonu üzerinde (5.6) eşitliği ile ifade edilen işlemle komşu bir yiyecek kaynağı (çözüm) belirler ve bu kaynağın nektar miktarını (uygunluk) hesaplar. Eşitlik (5.6)'dan görüldüğü üzere $x_{i,j}$ ve $x_{k,j}$ çözümleri arasındaki fark azaldıkça $x_{i,j}$ çözümünün değişim miktarı da azalmaktadır. Böylece, araştırma uzayında optimum çözüme yaklaştıkça adım büyüklüğü de adaptif olarak azalmaktadır. Temel ABC algoritmasından ilk farklılık burada ortaya çıkmaktadır. Temel ABC algoritmasında değişim operatöründe yalnızca bir parametrenin değişimi ile yeni çözüm oluşturulurken burada çözümün her parametresinin değişme şansı vardır. Her x_{ij} parametresi için üretilen uniform dağılımlı rastgele sayı $(0 \leq R_j \leq 1)$ MR parametresinden küçük ise x_{ij} parametresi değiştirilir. Eğer hiç bir parametre değişmemişse rastgele bir parametre seçilir ve değiştirilir. Böylece en azından bir parametrenin değişimi garanti edilmektedir.

Yeni bir komşu yiyecek kaynağı üretildikten sonra ABC algoritması seleksiyon işlemi uygulamaktadır. ABC algoritmasında ikinci değişiklik bu işlemde yapılmıştır. Temel ABC algoritmasında mevcut çözüm ile komşu çözüm arasında aç gözlü seleksiyon uygulanırken sınırlamalı problemler için önerilen ABC algoritmasında aç gözlü seleksiyon yerine Deb'in kurallarına dayalı bir seleksiyon uygulanmaktadır. Deb'in kuralları ile araştırma kabul edilebilir bölgeye doğru kaymaktadır. Deb'in kuralları iki çözümün aşağıdaki kriterlere göre yarıştırıldığı bir turnuva seleksiyon operatörü gibi çalışmaktadır:

- Kabul edilebilir bölgedeki her bir çözüm ($violation_i \leq 0$) kabul edilebilir olmayan bölgedeki çözüme ($violation_j > 0$) tercih edilir (çözüm i dominanttır)
- Her iki çözümde kabul edilebilir bölgede ise ($violation_i \leq 0$, $violation_j \leq 0$), o zaman daha küçük maliyet değerine sahip çözüm tercih edilir($f_i < f_j$, çözüm i dominanttır),
- Her iki çözümde kabul edilebilir olmayan bölgede ise ($violation_i > 0, violation_j > 0$), daha küçük sınırlama aşım miktarına sahip olan çözüm tercih edilir ($violation_i < violation_j$, çözüm i dominanttır).

Tüm görevli arılar araştırmalarını tamamladıktan sonra, yiyecek kaynakları ile ilgili

sahip oldukları bilgiyi dans alanında gözcü arılarla paylaşırlar. ABC algoritması bu işlemi çözümlerin uygunluk değerinin popülasyonun toplam uygunluk değerine olan oranından yola çıkarak gerçekleştirmektedir. Gözcü arı seçeceği kaynak bölgesini belirlerken yiyecek kaynağının kalitesi ile orantılı olan dans şiddeti bilgisinden faydalanmaktadır. Burada da uygunluğu yani kalitesi fazla olan çözümün seçilme olasılığının daha fazla olduğu rulet tekerleği benzeri bir seleksiyon işlemi yapılmaktadır. Bu hesaplama ile ilgili işlem Algoritma 5'de verilmektedir:

Algoritma 5 Gözcü arılar için kaynakların seçilme olasılıklarının hesaplanması

1: **for** $i = 1$ to $SN/2$ **do**

2: (5.8) eşitliği ile uygunluk ve/veya sınırlama aşım miktarlarının kullanarak kaynakların seçilme olasılıklarını (p_i) hesapla

$$\mathrm{p_i} = \begin{cases} 0.5 + \left(\dfrac{\mathrm{fitness_i}}{\sum_{\mathrm{j=1}}^{\mathrm{SN}} \mathrm{fitness_j}} \right) * 0.5 & \textit{kabul edilebilir ise} \\ \left(1 - \dfrac{violation_i}{\sum_{j=1}^{SN} \mathrm{violation_j}} \right) * 0.5 & \textit{kabul edilebilir degilse} \end{cases} \tag{5.8}$$

$violation_i$ x_i çözümünün cezası yani sınırlama aşım miktarı, $fitness_i$ de x_i çözümünün nektar miktarı ile orantılı uygunluk değeridir. Uygunluk miktarı (5.9) eşitliği ile hesaplanmaktadır:

$$fitness_i = \begin{cases} 1/(1+f_i) & if f_i \geq 0 \\ 1 + abs(f_i) & if f_i < 0 \end{cases} \tag{5.9}$$

f_i, x_i çözümünün maliyet değeridir.

3: **end for**

Yeni ABC algoritması kabul edilebilir olmayan çözümlerin de popülasyonda olmasına izin vermesi gerektiğinden temel ABC algoritmasında olasılık değerinin hesaplanması ile ilgili işlem de üçüncü bir değişikliğe gerek duyulmuştur. Kabul edilebilir bölgede olmayan çözümlere de olasılık değeri atanması gerekliliğinden dolayı bu tür çözümlere 0 ile 0.5 arasında sınırlama aşım değerleri ile ters orantılı; kabul edilebilir olanlarada 0.5 ile 1 arasında uygunluk değerleriyle orantılı olasılık değerleri atanmaktadır. Yani kabul edilebilir çözümler hemen başlangıçta 0.5'lik bir eşik değerine sahip olmuş durumdadırlar.

Bir gözcü arı, görevli arılardan nektar miktarı ile ilişkili aldığı bilgiyi kullanarak

bir kaynak seçer ve seçtiği kaynağın komşuluğunda (5.6) eşitliği ile bir çözüm üretir. Yeni üretilen çözümün parametre değerleri alt ve üst sınır değerlerini aşıyorsa bu sınır değerlerine veya kabul edilebilir aralığa ötelenir. Üretilen kaynak değerlendirilir. İşçi arılarda olduğu gibi seçtiği kaynak ile yeni ürettiği kaynak arasında Deb'in kurallarına göre seleksiyon işlemi uygulanır ve başarısızlık sayaçları güncellenir. Gözcü arı aşamasında yapılan işlemler Algoritma 6'da verilmektedir:

Algoritma 6 ABC algoritmasının gözcü arı fazı

1: $t = 0$, $i = 1$
2: **repeat**
3: **if** $random < p_i$ **then**
4: t=t+1
5: **for** $j = 1$ to D **do**
6: Gözcü arı için seçtiği x_i komşuluğunda v_i çözümü üret (Eşitlik 5.6)
7: **end for**
8: v_i ve x_i arasında Deb'in kurallarına dayalı seleksiyon uygula
9: x_i çözümü gelişemiyorsa $failure_i = failure_i + 1$, gelişmişse $failure_i = 0$
10: **end if**
11: $i = i + 1$
12: $i = i \ mod \ (np + 1)$
13: **until** $t = SN/2$

Tüm gözcü arılar yiyecek kaynaklarına dağıtıldıkdan ve kovana geri döndükten sonra artık nektarı tükenmiş, gidilmeye gerek duyulmayan kaynaklar belirlenir. Bu belirleme işlemi "limit" parametresi ve başarısızlık sayacına göre yapılmaktadır. Eğer başarısızlık sayacı $failure_i$ "limit" değerini aşarsa bu çözümün bırakılmasına karar verilir. Bırakılmasına karar verilen yiyecek kaynağı (5.5) eşitliğince rastgele üretilen bir kaynakla yer değiştirir.

Temel ABC algoritması ile sınırlamalı problemler için bu tezde önerilen geliştirilen ABC algoritması arasındaki bir diğer fark da temel ABC algoritmasında başarısızlık sayacının "limit" parametresini aşıp aşmadığına her adımda bakılırken, sınırlamalı problemler için geliştirilen ABC algoritmasında belli peryotlarda bakılmasıdır. Bu peryot kaşif arı üretme peryodudur (Scout Production Period, SPP). Kaşif arı süreci yeni ve muhtemelen kabul edilebilir bölgede olmayan çözümlerin popülasyona katılarak farklılık sağlanması amacıyla çalışmaktadır. Kaşif arı sürecinin adımları Algoritma 7'de verilmektedir:

Algoritma 7 ABC algoritmasının kaşif arı fazı

1: **if** $cevrim\ mod\ SPP = 0$ **then**
2: **if** $max(failure_i) > limit$ **then**
3: x_i çözümünü (5.5) eşitliği ile üretilen rastgele bir çözümle değiştir
4: **end if**
5: **end if**

Netice olarak sınırlamalı problemler için önerilmiş ABC algoritması yakınsama hızını artırmak gayesiyle MR ve SPP olmak üzere iki kontrol parametresi daha kullanmakta ve temel ABC'deki aç gözlü seleksiyon yerine Deb'in kurallarına dayalı bir seleksiyon mekanizmasının kullanılmaktadır. Ayrıca popülasyonda kabul edilebilir olmayan çözümlerin de bulunmasından dolayı bu çözümlere de sınırlama aşım miktarları ile ters orantılı olarak kalite değeri ve buna bağlı olarak da gözcü arılarca bir seçilebilme olasılığı atanması temel ABC'den farklılığıdır.

5.4. Literatürdeki Diğer Algoritmalarla ABC'nin Performansının Kıyaslanması

Sınırlamalı problemlere uygulanmak amacıyla üzerinde bazı değişiklikler yapılan ABC algoritmasının performansını değerlendirmek için, [219] çalışmasında tanımlanan 13 sınırlamalı test fonksiyonu kullanılmıştır. Bu test setinde doğrusal, doğrusal olmayan veya quadratic tipte amaç fonksiyonuna sahip çeşitli test problemleri bulunmaktadır. Bu problemlere ait amaç fonksiyonunun tipi, doğrusal eşitliklerin sayısı (LE), doğrusal olmayan eşitliklerin sayısı (NE), doğrusal eşitsizliklerin sayısı (LI), doğrusal olmayan eşitsizliklerin sayısı (NI) ve değişken sayıları Tablo 5.1'de verilmektedir. Tablo 5.1'de görülen ρ değeri kabul edilebilir bölgedeki çözüm sayısı F, rastgele üretilen tüm çözümlerin sayısı S olmak üzere $\rho = |F|/|S|$ formülü ile hesaplanan kabul edilebilir bölgenin tüm araştırma uzayına oranının bir kestirimidir [216]. Problemler EK-4 bölümünde verilmektedir.

Değiştirilen ABC algoritmasıyla elde edilen sonuçlar, literatürde mevcut bazı popüler algoritmalara (örneğin homomorphous mapping (HM) metodu [220], stochastic ranking (SR) metodu [219], improved stochastic ranking (ISR) metodu [221], over-penalty approach (OPA) [221], adaptive segregational constraint handling evolutionary algorithm (ASCHEA) [222], simple multi-membered evolution strategy (SMES) [14], genetic algorithm (GA) [14], differential evolution (DE), and particle

swarm optimization (PSO) [244]) ait sonuçlarla kıyaslanmıştır. Bu algoritmalara ait bilgilere EK-1 bölümünde mevcuttur.

Tablo 5.1. 13 test problemine ait ρ değerleri, D:problemin boyutu, LI: doğrusal eşitsizlik sayısı, NI: doğrusal olmayan eşitsizlik sayısı, LE: doğrusal eşitlik sayısı, NE: doğrusal olmayan eşitlik sayısı [14].

	D	**Prob. Tipi**	ρ	**LI**	**NI**	**LE**	**NE**
g01	13	quadratik	0.0003%	9	0	0	0
g02	20	nonlineer	99.9973%	1	1	0	0
g03	10	nonlineer	0.0026%	0	0	0	1
g04	5	quadratik	27.0079%	0	6	0	0
g05	4	nonlineer	0.0000%	2	0	0	3
g06	2	nonlineer	0.0057%	0	2	0	0
g07	10	quadratik	0.0000%	3	5	0	0
g08	2	nonlineer	0.8581%	0	2	0	0
g09	7	nonlineer	0.5199%	0	4	0	0
g10	8	linear	0.0020%	3	3	0	0
g11	2	quadratik	0.0973%	0	0	0	1
g12	3	quadratik	4.7697%	0	9^3	0	0
g13	5	nonlineer	0.0000%	0	0	1	2

HM metodu decoder tabanlı bir sınırlama ele alış yaklaşımı kullanmakta ve gösterim için gray kodlama kullanmaktadır. Her bir değişken 25 bit ile temsil edilmekte ve algoritma olasılık tabanlı seleksiyon (elit yok), fonksiyon ölçekleme ve standart operatörleri kullanmaktadır. Popülasyon büyüklüğü 70, çaprazlama oranı 0.9, kuşak boşluğu (generation gap) %100 ve maksimum jenerasyon sayısı 20000 olarak alınmış ve toplamda da 1400000 fonksiyon değerlendirmesi yapılmıştır [220].

SR metodu, gelişim stratejisi içerisinde stokastik sıralama şemasına dayalı bir sınırlama ele alış yaklaşımı kullanmaktadır. (30,200)-ES yapısı 1750 jenerasyon koşulmakta ve böylece 200x1750=350000 değerlendirme yapılmaktadır. Bu yaklaşımda eşitlik sınırlamaları ϵ=0.0001 değeriyle $|h_j| \leq \epsilon$ şeklinde eşitsizlik sınırlamasına dönüştürülmektedir [219].

ISR metodu, (60,400) yapısında gelişim stratejisini 875 jenerasyon boyunca koşmakta ve böylece toplamda 400x875=350000 fonksiyon değerlendirmesi yapmaktadır. Algoritmanın geliştirilmişliği kullanılan sınırlama ele alış metodu ile ilgilidir. Metot, adım büyüklüğünü ölçeklemek amacıyla γ parametresini kullanmakta ve adım büyüklüğünü azaltmak için de 0.85 civarı bir değer

kullanmaktadır [221].

OPA metodu, (60,400) yapısında gelişim stratejisi ile ceza terimine dayalı bir sınırlama ele alış metodu kullanmaktadır. 87 jenerasyon boyunca koşulan g12 fonksiyonu dışında, diğer tüm fonksiyonlar için 875 jenerasyon koşulmuş ve toplam 400x875=350000 değerlendirme yapılmıştır [221].

ASCHEA metodu, ceza terimine dayalı sınırlama ele alış metodu kullanmakta ve tolerans değeri ϵ jenerasyona bağlı olarak değiştirilmektedir. Yaklaşım, (100+300)-ES yapısındadır ve ayrışmalı seleksiyon stratejisi kullanmaktadır. Kullanılan Gaussian mutasyonunun başlangıç değeri 0.03, öğrenme oranı 0.1, bölgesel öğrenme oranı 1'dir. Yeniden oluşum operatörü aritmetik çaprazlama kullanmaktadır. Çaprazlama ve mutasyon oranları 0.9 olarak seçilmiş ve 1500000 değerlendirme yapılmaktadır [222].

GA sınırlama ele alış metodu olarak Deb'in kurallarını kullanmakta ve eşitlik sınırlamaları dinamik olan ϵ değerleriyle $|h_j| \leq \epsilon$ şeklinde eşitsizlik sınırlamasına dönüştürülmektedir. Popülasyon büyüklüğü 200, maksimum jenerasyon sayısı 1200, çaprazlama oranı 0.8, mutasyon orano 0.6 ve maksimum değerlendirme sayısı 240000 olarak seçilmiştir [14].

SMES, sınırlama ele alış metodu olarak Deb'in kurallarını kullanmakta ve (100,300)-ES yapısındaki algoritma 800 jenerasyon boyunca koşulmaktadır. Böylece maksimum değerlendirme sayısı 240000 olmaktadır [14].

PSO algoritması da sınırlamaları ele almak için Deb'in kurallarını kullanmaktadır. Sürü 50 parçacıktan oluşmakta ve jenerasyon sayısı 7000'dir. Dolayısıyla toplam 350000 değerlendirme yapılmaktadır. Bilişsel ve sosyal bileşen parametre değerleri 1 olarak seçilmiş ve inertia parametresi [0.5,1] aralığında uniform olarak rastgele belirlenmiştir. Tüm eşitlik sınırlamaları ϵ=0.001 değeriyle $|h_j| \leq \epsilon$ şeklinde eşitsizlik sınırlamasına dönüştürülmüştür [244].

DE yaklaşımında vektörlerin farkını ölçekleyen parametre değeri F 0.5 olarak seçilmiş ve çaprazlama oranının değeri [203]'de önerildiği üzere 0.9 olarak alınmıştır. Deb'in kurallarını kullanan DE yaklaşımında popülasyon büyüklüğü 40

ve maksimum jenerasyon sayısı 6000 olarak alınmıştır.

ABC algoritmasında değişim oranı MR'nin değeri 0.8, koloni büyüklüğü (SN) 40, maksimum çevrim sayısı (MCN) 6000 alınarak maksimum değerlendirme sayısı SMES, GA ve DE'de olduğu gibi 240 000'e eşitlenmiştir. D problemin parametre sayısı olmak üzere "limit" parametresinin değeri $0.5xSNxD$ ve SPP parametresinin değeri de $0.5xSNxD$ şeklinde koloni büyüklüğüne ve parametre sayısına bağlanmıştır. Her başlatmada farklı rastgele üretilen başlatma popülasyonu kullanılarak her fonksiyon için testler 30 kez tekrarlanmış ve sonuçlar elde edilmiştir.

ABC algoritmasına ait en iyi, ortalama ve en kötü değerler Tablo 5.2'de verilmektedir. Algoritmaların yine en iyi, en kötü ve ortalama değerlere göre kıyaslamalı sonuçları da sırasıyla Tablo 5.3, Tablo 5.4 ve Tablo 5.5'de sunulmaktadır. Daha önce de belirtildiği gibi sınırlamalı optimizasyon problemlerinin zorluğunda sadece tek bir parametre etkili değildir (değişken sayısı, doğrusal olan ve olmayan eşitlik ve eşitsizliklerin sayısı, ρ) [217].

Tablo 5.2. ABC algoritmasının 240000 değerlendirme ile 13 fonksiyon üzerinde elde edilen 30 koşmaya ait istatistiksel sonuçlar.

Problem	**Optimal**	**En iyi**	**Ortalama**	**En kötü**	**Standart Sapma**
g01	-15.000	-15.000	-15.000	-15.000	0.000
g02	0.803619	0.803598	0.792412	0.749797	0.012
g03	1.000	1.000	1.000	1.000	0.000
g04	-30665.539	-30665.539	-30665.539	-30665.539	0.000
g05	5126.498	5126.484	5185.714	5438.387	75.358
g06	-6961.814	-6961.814	-6961.813	-6961.805	0.002
g07	24.306	24.330	24.473	25.190	0.186
g08	0.095825	0.095825	0.095825	0.095825	0.000
g09	680.63	680.634	680.640	680.653	0.004
g10	7049.25	7053.904	7224.407	7604.132	133.870
g11	0.75	0.750	0.750	0.750	0.000
g12	1.000	1.000	1.000	1.000	0.000
g13	0.053950	0.760	0.968	1.000	0.055

Tablo 5.3'deki en iyi sonuçlara bakıldığında, ABC algoritmasının 240 000 değerlendirme sonunda 13 fonksiyondan 7'sinde (g01, g03, g04, g06, g08, g11, g12) global minimumu bulabildiği, 5 fonksiyonda (g02, g04, g05, g07, g10) global optimuma oldukça yakın sonuçlar ürettiği ve bir problemde (g13) de ABC algoritmasının belirtilen değerlendirme sayısında optimum çözümü bulamadığı

görülmektedir.

Tablo 5.3'den ABC'nin HM'ye kıyasla dokuz problem üzerinde (g01, g02, g03, g04, g05, g06, g07, g09, g10) daha iyi sonuç ürettiği, üç problem (g08, g11, g12) üzerinde eşit performansa sahip oldukları görülmektedir. g13 için HM'nin sonuçları bulunmamaktadır.

ABC yaklaşımı SR'ye göre üç problem (g02, g05, g10) üzerinde daha iyi iken SR de ABC'ye göre üç problem (g07, g09, g13) üzerinde daha iyi sonuç vermiştir. Diğer yedi problem (g01, g03, g04, g06, g08, g11, g12) üzerinde iki algoritma benzer performans göstermiştir.

ISR ile ABC kıyaslandığında, ISR ABC'den altı problem (g02, g03, g07, g09, g10, g13) üzerinde daha iyi iken ABC bir problem (g05) üzerinde daha iyidir. Kalan altı problem (g01, g04, g06, g08, g11, g12) üzerinde ise benzer performans sergilemişlerdir.

OPA metodu beş problemde (g02,g07,g09,g10,g13), ABC iki problemde (g03,g05) daha iyi sonuç üretmiştir. Diğer altı problemde (g01, g04,g06,g08,g11,g12) eşit başarım göstermişlerdir.

ASCHEA'nın mevcut olan sonuçlarına göre altı problemde (g02, g04, g05, g06, g07, g10) ABC, bir problemde (g09) ASCHEA daha iyi sonuç üretmiştir. Diğer problemlerde (g01, g03, g08, g11) ise benzer performans sergilemişlerdir.

GA ile ABC kıyaslandığında g08, g11 ve g12 problemlerinde benzer sonuçlar üretmişlerdir. Sekiz problemde (g01, g02, g03, g04, g06, g07, g09, g10) ABC'nin performansı GA'nınkine göre daha iyidir. Bir problemde (g13) ise GA daha iyi sonuç üretmiştir.

ABC, SMES'ten beş problem (g02, g07, g09, g10, g13) üzerinde daha iyi iken SMES ABC'den bir problem (g05) üzerinde daha iyidir. Yedi problem (g01, g03, g04, g06, g08, g11, g12) üzerinde aralarında performans bakımından en iyi sonuca göre fark bulunmamaktadır.

PSO algoritması ABC'den üç problemde (g09,g10, g13) daha iyi, ABC ise beş

problemde (g02, g03, g07, g09, g11) daha iyidir.

En iyi sonuçlara göre DE'ye kıyasla ABC üç problemde (g02, g06, g11), DE ise dört problemde (g07, g09, g10, g13) daha iyi sonuç üretmiştir.

Benzer olarak Tablo 5.5'deki kararlılıkla ilgili fikir verebilecek ortalama sonuçlar incelendiğinde ABC HM'den dokuz problemde (g01, g02, g03, g04, g06, g07, g08, g09, g10), SR'den dört problemde (g02, g06, g09, g10), ISR'den bir problemde (g02), OPA'dan beş problemde (g02, g03, g05, g11, g12), ASCHEA'dan beş problemde (g01, g02,g04, g06, g07), GA'dan dokuz problemde (g01, g02, g03, g04, g06, g07, g08, g09, g10), SMES'ten beş problemde (g02, g06, g07, g08, g09, g10), PSO'dan altı problemde (g01, g02, g03, g07, g11, g12) ve DE'den beş problemde (g01, g02, g05, g06, g11) daha iyi performans göstermiştir. Ortalama sonuçlara göre HM ve GA hiç bir problemde ABC'den daha iyi değer üretememiştir. SR üç problemde(g05, g07, g13), ISR yedi problemde (g03, g05, g06, g07, g09, g13), OPA beş problemde (g06, g07, g09, g10, g13), ASCHEA iki problemde (g05, g10), SMES iki problemde (g05, g13), PSO beş problemde (g05, g06, g09, g10, g13) ve DE dört problemde (g07, g09, g10, g13) ABC'den daha iyi performans göstermiştir. Tablolardaki bu sonuçların daha iyi anlaşılabilmesi amacıyla sonuçlar Şekil 5.1(a)-5.2(g)'de verildiği gibi görselleştirilmiştir.

ABC algoritmasının toplam 240000 değerlendirme boyunca koşulmasının ardından diğer algoritmaların başarı oranları ile ikili olarak karşılaştırılmış ve Tablo 5.6'da sunulmuştur. Tablo 5.5'de sunulan ortalama değerlerden ve Tablo 5.6'daki başarı oranlarından, ABC algoritmasının HM, SR, ASCHEA, GA, SMES, DE ve PSO'dan daha iyi performans gösterdiği; OPA'ile benzer performans sergilediği ancak ISR'den biraz daha kötü sonuçlar ürettiği söylenebilir.

OPA, SR, ISR ve PSO'ya ait sonuçlar ilgili çalışmalarda 350000 değerlendirme sonucu elde edildiğinden ABC algoritması da aynı şekilde 350000 değerlendirme boyunca koşulmuş ve sonuçlar Tablo 5.7'de sunulmuştur. Tablo 5.5 ve 5.7'deki sonuçlara göre algoritmaların ABC ile ikili olarak başarı oranlarının karşıştırılmaları sonucu Tablo 5.8'de verilmektedir. Tablo 5.7'den görüldüğü gibi ABC algoritması daha uzun çevrim sayısı boyunca koşulduğunda en iyi, ortalama, en kötü ve standart

sapma değerlerinin geliştiği görülmektedir.

Ancak şu belirtilmelidir ki SMES metodu yüksek bir ϵ değeri ile başlamakta ve araştırma ilerledikçe dereceli olarak bu değer azaltılmaktadır. Ayrıca, ϵ değeri de her probleme göre özel seçilmiştir. Bu ise problem hakkında ön bilgi gerektirmektedir. Yani bu bir dezavantajdır. ABC algoritmasında ise bu şekilde bir yaklaşım söz konusu değildir. Tüm problemler için aynı ϵ değeri kullanılmıştır. Yani, ABC algoritması SMES'e göre daha basit bir yapıda ve problem hakkında ön bilgiye sahip olunmasını gerektirmemektedir.

Değiştirilmiş ABC algoritması, g05, g10 ve g13 problemleri için her koşulmasında global optimumu bulamamaktadır. g05 ve g13 doğrusal olmayan problemler olup bu problemler için ρ değeri %0.000'dir Ayrıca bu problemler doğrusal olmayan eşitlik sınırlamalarına sahiptirler. g10 ise doğrusaldır ve problemin ρ değeri %0.0020'dir. Herhangi bir doğrusal veya doğrusal olmayan eşitlik sınırlaması bulunmamaktadır. Bu nedenle ABC algoritmasının daha iyi olduğu veya olmadığı problemlerin karakteristiği hakkında bir genelleme yapılamamaktadır. Ancak herhangi bir karmaşık mekanizma kullanmadan basitliğine rağmen literatürdeki mevcut diğer yaklaşımlarla kıyaslanabilir derecede başarılı sonuçlar üretmektedir.

Herhangi bir gelişime dayalı algoritmanın sınırlamalı optimizasyon problemleri üzerindeki performansı, kullanılan sınırlama ele alış metoduna oldukça bağlı olduğu bilinen bir gerçektir. Amaç, ABC algoritmasının sınırlamalı problemlerdeki performansını değerlendirmek olduğundan algoritma ile ilgili değişimler minimum seviyede tutulmuştur. Değişim büyük oranda seleksiyon birimi ile ilgili yapılmıştır. Kabul edilebilir bölgede olan çözümler daha uygun çözümler olduğu için bu bölgedeki çözümlere öncelik vermek, bu bölgedeki çözümler arasından da uygunluğu daha yüksek olanın tercih edilebilmesi için aç gözlü seleksiyon yerine Deb'in kurallarına dayalı bir seleksiyon mekanizması kullanılmıştır. Başka bir sınırlama ele alış metodunun kullanılması algoritmanın performansında etkili olduğundan aslında ABC algoritmasını Deb'in kurallarını kullanan yaklaşımlarla ve benzer olarak yine kabul edilebilir bölgeye öncelik atayan OPA ile kıyaslamak daha adildir. Bu bağlamda değerlendirilirse sınırlamaları ele almak amacıyla Deb'in kurallarını

kullanan GA, DE, PSO, SMES metotları ve OPA metodu ile ABC kıyaslandığında Tablo 5.8'den ABC'nin bu yaklaşımların hepsinin üzerinde performans sergilediği açıktır.

Deb'in kuralları tek başına kullanıldığında sürekli kabul edilebilir bölgedeki çözümler seçildiği için popülasyonda farklılığın azalmasına sebep olmaktadır. Ancak farklılığın belli seviyede korunması problemin optimal çözümüne giden yolda oldukça önemli bir husustur [252]. Değiştirilmiş ABC algoritmasında bu özellik temel ABC algoritmasında da bulunan kaşif arı birimi ve gözcü arı birimi sayesinde sağlanmaktadır. Kaşif arı birimi ile kabul edilebilir bölgede olmayan çözümlerin de popülasyona katılması sağlanmakta; gözcü arı birimi ile de kabul edilebilir bölgede olmayan çözümlerinde sınırlama aşım değerleriyle ters orantılı olarak olasılık değeri atanmasıyla bu tür çözümlere seçilme olasılığı tanınmaktadır.

5.5. Kontrol Parametrelerinin Etkisi

Kontrol parametreleri için seçilen değerler araştırmanın performansını etkilediğinden incelenmesi gereken bir konudur. En uygun parametre değer setini bulmak için farklı parametre değerleri ile ilgili sonuçlar alınarak aralarında anlamlı fark olup olmadığını görmek amacıyla ortalama varyanslarını kullanan ANOVA testi kullanılmıştır. ANOVA testi için nul hipotez, farklı parametre değerlerinden gelen sonuçların ortalamalarının aynı popülasyondan geldiğidir, red hipotezi de en az bir grubun ortalamasının aynı popülasyondan gelmediğidir. Verilerden hesaplanan F değeri ile güven değeri ve serbestlik değeriyle ilişkili olarak F dağılımından hesaplanan F değeri kıyaslanır. Bu değer 0.05'ten küçükse sonuçların istatistiksel olarak anlamlı farklılık arz ettiği ve nul hipotezin reddedilerek tüm ortalamaların aynı popülasyondan gelmediği sonucuna varılır. ANOVA testinin yanı sıra en iyi parametre setinin belirlenebilmesi içi MATLAB platformu kullanılarak ANOM testi uygulanmıştır.

MR kontrol parametresi için 0.1 ile 0.9 arasında 0.1 adımla artan 9 değer, limit parametresi için $\{0.1xSNxD,\ 0.5xSNxD,\ SNxD,\ 2xSNxD,\ 4xSNxD\}$ değerlerinden oluşan 5 değer, ve SPP için $\{0.1xSNxD,\ 0.5xSNxD,\ SNxD,\ 2xSNxD,\ 4xSNxD\}$ değerlerinden oluşan 5 değer incelenmiştir. Her bir kontrol

parametresinin değerleri kendi içerisinde değerlendirilmiştir. Her bir değer için 30 koşma yapılmış; yani toplam 30x(9+5+5) koşma gerçekleştirilmiştir.

Tablo 5.9-5.11'de *MR*,*limit* ve *SPP* kontrol parametrelerinin değerlerinin hata üzerinde etkisi olup olmadığını analiz etmek amacıyla gerçekleştirilen ANOVA testinin sonuçları verilmektedir. Tablolardaki SS her bir kaynağın kareler toplamını, DF her bir kaynağın serbestlik derecesini ve MS=SS/df'den hesaplanan ortalama kareleri göstermektedir. F değeri de ortalama karelerin oranıdır. Tablo 5.9'daki sonuçlardan, g01, g06, g08 ve g12 üzerinde *MR* parametresinin hata değerini etkilemediği, Tablo 5.10'dan *limit* parametresinin g01, g08 ve g12 üzerinde etkili olmadığı, Tablo 5.11'den de g01, g04, g08 ve g12 üzerinde *SPP* parametresinin etkili olmadığı görülmüştür.

Kontrol parametrelerinin hangi değerlerinin en iyi sonucu ürettiği ile ilgili bir sonuca varmak için ANOM tabloları çıkarılmıştır. *MR*, *limit* ve *SPP* için elde edilen sırasıyla Tablo 5.12, 5.13, 5.14'de bu sonuçlar sunulmakta ve ANOM grafikleri de yine sırasıyla Şekil 5.3(a)-5.4(g), 5.5(a)-5.6(g), 5.7(a)-5.8(g)'de gösterilmektedir. ANOM, grup ortalamalarının mutlak sapmalarını tüm ortalama ile değerlendirerek diğer gruplardan farklı sonuç üreten grubu belirlemektedir. Şekiller ve tablolardan grup ortalamasından daha iyi ve benzer sonuçları ayıklayarak kontrol parametreleri için uygun değerler belirlenmiştir. Denenen tüm fonksiyonlar ve tüm kontrol parametre değerleri dikkate alındığında, *MR* için [0.3 - 0.8] aralığı, *limit* için [0.5xSNxD-SNxD] aralığı ve *SPP* için de [0.1xSNxD-2xSNxD] aralığının tavsiye edilebileceği sonucu çıkarılabilir.

Bu bölümde sınırlamalı optimizasyon problemlerinin çözümü için ABC algoritmasının kullanılması gerçeklenmiştir. Sınırlamalı problemlerin doğası gereği sınırlamalardan dolayı araştırma uzayı kısıtlanmaktadır. Uygun çözümlerin bu kısıtlı alanda aranması ve bu bölgede bir çözüm bulunması için araştırmanın bu yönde ilerlemesini sağlayacak bazı mekanizmaların algoritmalara dahil edilmesine ihtiyaç duyulmaktadır. Bu mekanizmalara sınırlama ele alış (constraint handling) metotları denmektedir. Sınırlamasız nümerik problemler için geliştirilen ABC algoritması da sınırlamalı problemleri çözmek için böyle bir düzenlemeye ihtiyaç

duymaktadır. Bu nedenle temel ABC algoritmasında yeni üretilen çözümün mevcut çözümle kıyaslandığı ve aç gözlü seleksiyonun uygulandığı birim, kabul edilebilir bölgede olmayan çözümlerin de değerlendirmeye alınması için Deb'in kurallarını kullanacak şekilde değiştirilmiştir. Ayrıca, gözcü arıların kullandığı olasılık tabanlı seçme işleminin gerçekleştiği yerde, kabul edilebilir bölgede olmayan çözümlerin de popülasyonda bulunmasından dolayı bu çözümlere de seçilme hakkı tanıyacak şekilde olasılık hesaplanmasında değişiklik yapılmıştır. Kaşif arıların her çevrimde değilde bazı çevrimlerde çıkması sağlanarak çok fazla rastgele çözümün popülasyona girmesinin önüne geçilmiştir. Değiştirilmiş ABC ile literatürdeki diğer algoritmalar sınırlamalı test problemleri üzerinde kıyaslanarak sonuçlar sunulmuştur. Gerçekleştirilen istatistiksel testlerle kontrol parametreleri için tavsiye edilebilecek değerler veya aralıkları belirlenmiştir.

Tablo 5.3. HM, SR, ISR, OPA, ASCHEA, SMES, GA, DE, PSO ve ABC algoritmalarının 13 fonksiyon üzerinde 30 koşmalarının en iyi sonuçları, – Herhangi bir kabul edilebilir çözümün bulunamadığını, Na = Sonucun olmadığını göstermektedir.

P	**Optimal**	[1]**HM [220]**	[2]**SR [219]**	[2]**ISR [221]**	[3]**OPA [221]**	[4]**ASCHEA [222]**	[5]**GA [14]**	[5]**SMES [14]**	[5]**PSO [244]**	[5]**DE**	[5]**ABC**
g01	-15.000	-14.7864	-15.000	-15.000	-15.000	-15.0	14.440	-15.000	15.000	-15.000	-15.000
g02	0.803619	0.79953	0.803515	0.803619	0.803619	0.785	0.796231	0.803601	0.669158	0.472	0.803598
g03	1.000	0.9997	1.000	1.001	0.747	1.0	0.990	1.000	0.993930	1.000	1.000
g04	-30665.539	-30664.5	-30665.539	-30665.539	-30665.539	-30665.5	-30626.053	-30665.539	-30665.539	-30665.539	-30665.539
g05	5126.498	–	5126.497	5126.497	5126.497	5126.5	–	5126.599	5126.484	5126.484	5126.484
g06	-6961.814	-6952.1	-6961.814	-6961.814	-6961.814	-6961.81	-6952.472	-6961.814	-6161.814	-6954.434	-6961.814
g07	24.306	24.620	24.307	24.306	24.306	24.3323	31.097	24.327	24.370153	24.306	24.330
g08	0.095825	0.0958250	0.095825	0.095825	0.095825	0.095825	0.095825	0.095825	0.095825	0.095825	0.095825
g09	680.63	680.91	680.630	680.630	680.630	680.630	685.994	680.632	680.630	680.630	680.634
g10	7049.25	7147.9	7054.316	7049.248	7049.248	7061.13	9079.770	7051.903	7049.381	7049.248	7053.904
g11	0.75	0.75	0.750	0.750	0.750	0.75	0.75	0.75	0.749	0.752	0.750
g12	1.000	NA	1.000	1.000	1.000	NA	1.000	1.000	1.000	1.00	1.000
g13	0.053950	NA	0.053957	0.053942	0.447118	NA	0.134057	0.053986	0.085655	0.385	0.760

Decoder tabanlı[1]
Stokastik sıralamaya dayalı[2]
Ceza terimi üzeri (over-penalized) yaklaşımı[3]
Ceza terimi tabanlı[4]
Deb'in kurallarına dayalı[5]

Tablo 5.4. HM, SR, ISR, OPA, ASCHEA, SMES, GA, DE, PSO ve ABC algoritmalarının 13 fonksiyon üzerinde 30 koşmalarının en kötü sonuçları, – Herhangi bir kabul edilebilir çözümün bulunamadığını, Na = Sonucun olmadığını göstermektedir.

P	**Optimal**	[1]**HM [220]**	[2]**SR [219]**	[2]**ISR [221]**	[3]**OPA [221]**	[4]**ASCHEA [222]**	[5]**GA [14]**	[5]**SMES [14]**	[5]**PSO [244]**	[5]**DE**	[5]**ABC**
g01	-15.000	-14.6154	-15.000	-15.000	-15.000	NA	-14.015	-15.000	-13.000	-11.828	-15.000
g02	0.803619	0.79119	0.726288	0.723591	0.712818	NA	0.779140	0.751322	0.299426	0.472	0.749797
g03	1.000	0.9978	1.000	1.001	0.031	NA	0.956	1.000	0.464	1.000	1.000
g04	-30665.539	-30645.9	-30665.539	-30665.539	-30665.539	NA	-30567.105	-30665.539	-30665.539	-30665.539	-30665.539
g05	5126.498	–	5142.472	5126.497	5826.807	NA	–	5304.167	5249.825	5534.610	5438.387
g06	-6961.814	-5473.9	-6350.262	-6961.814	-6961.814	NA	-6784.255	-6952.482	-6961.814	-6954.434	-6961.805
g07	24.306	25.069	24.642	24.306	24.311	NA	38.686	24.843	56.055	24.330	25.190
g08	0.095825	0.0291438	0.095825	0.095825	0.095825	NA	0.095723	0.095825	0.095825	0.095825	0.095825
g09	680.63	683.18	680.763	680.630	680.630	NA	698.297	680.719	680.631	680.631	680.653
g10	7049.25	9659.3	8835.655	7049.270	7049.252	NA	11003.533	7638.366	7894.812	9264.886	7604.132
g11	0.75	0.75	0.750	0.750	0.774	NA	0.752	0.75	0.749	1	0.750
g12	1.000	NA	1.000	1.000	0.999385	NA	0.999	1.000	0.994	1.000	1.000
g13	0.053950	NA	0.216915	0.438803	0.999225	NA	–	0.468294	1.793361	0.990	1.000

[1] Decoder tabanlı

[2] Stokastik sıralamaya dayalı

[3] Ceza terimi üzeri (over-penalized) yaklaşımı

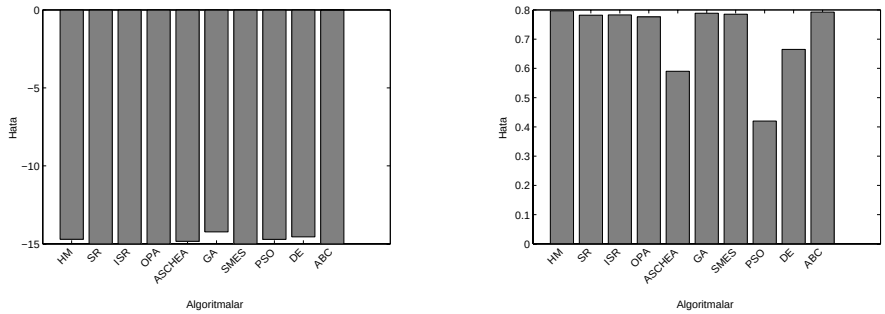
[4] Ceza terimi tabanlı

[5] Deb'in kurallarına dayalı

Tablo 5.5. HM, SR, ISR, OPA, ASCHEA, SMES, GA, DE, PSO ve ABC algoritmalarının 13 fonksiyon üzerinde 30 koşmalarının ortalama sonuçları – Herhangi bir kabul edilebilir çözümün bulunamadığını, Na = Sonucun olmadığını göstermektedir. Koyu renkle yazılan sonuçlar en iyi sonuçtur.

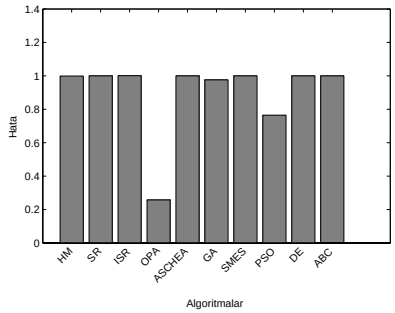
P	Optimal	[1]HM [220]	[2]SR [219]	[2]ISR [221]	[3]OPA [221]	[4]ASCHEA [222]	[5]GA [14]	[5]SMES [14]	[5]PSO [244]	[5]DE	[5]ABC
g01	-15.000	-14.7082	**-15.000**	**-15.000**	**-15.000**	-14.84	-14.236	**-15.000**	-14.710	-14.555	**-15.000**
g02	0.803619	**0.79671**	0.781975	0.782725	0.776283	0.59	0.788588	0.785238	0.419960	0.665	0.792412
g03	1.000	0.9989	1.000	**1.001**	0.257	0.99989	0.976	1.000	0.764813	1.000	1.000
g04	-30665.539	-30655.3	**-30665.539**	**-30665.539**	**-30665.539**	-30665.5	-30590.455	**-30665.539**	**-30665.539**	**-30665.539**	**-30665.539**
g05	5126.498	-	5128.881	**5126.497**	5268.610	5141.65	–	5174.492	5135.973	5264.270	5185.714
g06	-6961.814	-6342.6	-6875.940	**-6961.814**	**-6961.814**	-6961.81	-6872.204	-6961.284	**-6961.814**	–	-6961.813
g07	24.306	24.826	24.374	**24.306**	24.307	24.6636	34.980	24.475	32.407	24.310	24.473
g08	0.095825	0.0891568	**0.095825**	**0.095825**	**0.095825**	**0.095825**	0.095799	**0.095825**	**0.095825**	**0.095825**	**0.095825**
g09	680.63	681.16	680.656	**680.630**	**680.630**	680.641	692.064	680.643	680.630	**680.630**	680.640
g10	7049.25	8163.6	7559.192	7049.250	**7049.248**	7497.434	10003.225	7253.047	7205.5	7147.334	7224.407
g11	0.75	**0.75**	**0.750**	**0.750**	0.756	**0.75**	**0.75**	**0.75**	0.749	0.901	**0.750**
g12	1.000	NA	**1.000**	**1.000**	0.999889	NA	**1.000**	**1.000**	0.998875	**1.000**	**1.000**
g13	0.053950	NA	**0.057006**	0.066770	0.964323	NA	–	0.166385	0.569358	0.872	0.968

Decoder tabanlı[1]
Stokastik sıralamaya dayalı[2]
Ceza terimi üzeri (over-penalized) yaklaşımı[3]
Ceza terimi tabanlı[4]
Deb'in kurallarına dayalı[5]

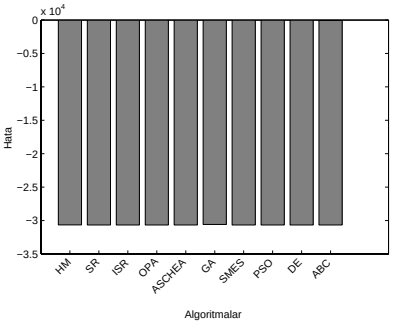


(a) G01Mean

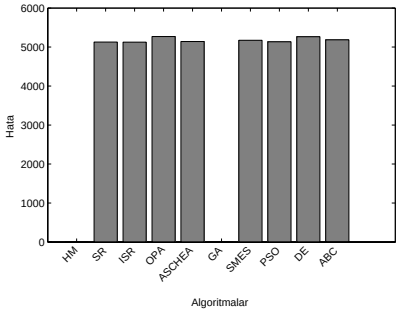
(b) G02Mean



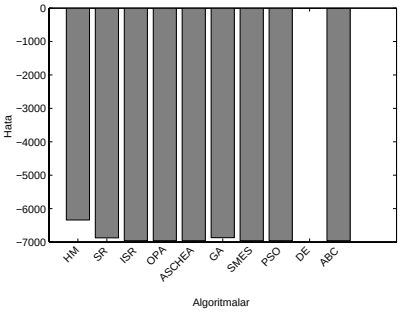
(c) G03Mean



(d) G04Mean

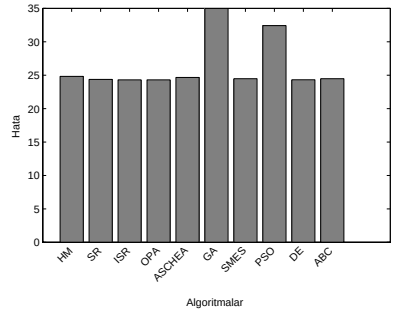


(e) G05Mean

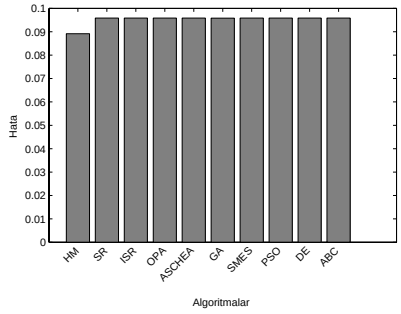


(f) G06Mean

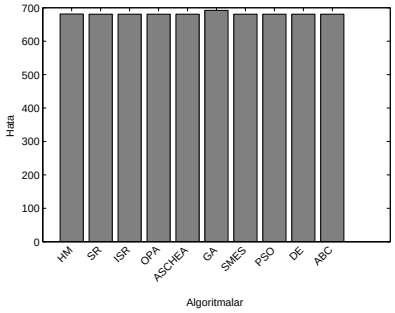
Şekil 5.1. g01-g06 problemlerinin ortalama sonuçlarına ait çubuk grafikler



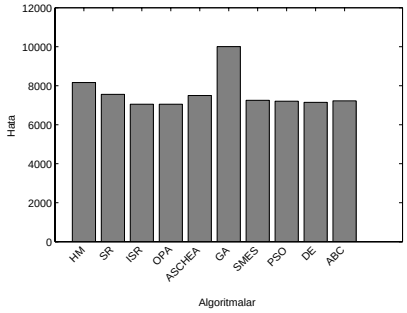
(a) G07Mean



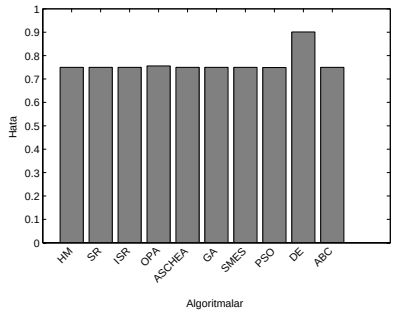
(b) G08Mean



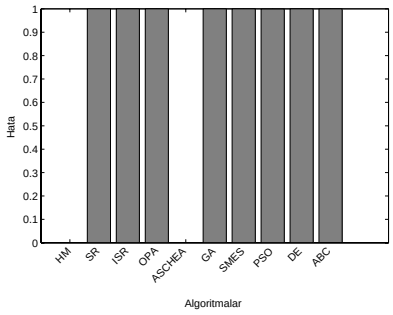
(c) G09Mean



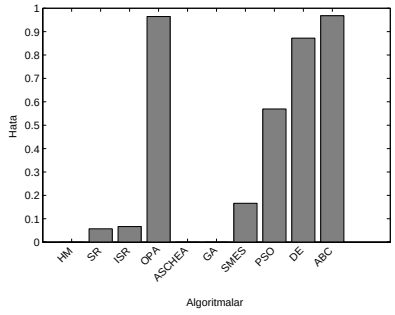
(d) G10Mean



(e) G11Mean



(f) G12Mean



(g) G13Mean

Şekil 5.2. g07-g13 problemlerinin ortalama sonuçlarına ait çubuk grafikler

Tablo 5.6. ABC algoritması ile diğer algoritmaların kıyaslanması ile elde edilen başarı oranları. + algoritmanın daha iyi olduğunu - daha kötü olduğunu göstermektedir.

Prb	**ABC-HM**		**ABC-SR**		**ABC-ISR**		**ABC-OPA**		**ABC-ASCHEA**		**ABC-GA**		**ABC-SMES**		**ABC-PSO**		**ABC-DE**	
	ABC	**HM**	**ABC**	**SR**	**ABC**	**ISR**	**ABC**	**OPA**	**ABC**	**ASCHEA**	**ABC**	**GA**	**ABC**	**SMES**	**ABC**	**PSO**	**ABC**	**DE**
g01	+	-	+	+	+	+	+	+	+	-	+	-	+	+	+	-	+	-
g02	-	+	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-
g03	+	+	+	+	-	+	+	-	+	+	+	-	+	+	+	-	+	+
g04	+	-	+	+	+	+	+	+	+	-	+	-	+	+	+	+	+	+
g05	+	-	-	+	-	+	+	-	-	+	-	-	-	+	-	+	+	-
g06	+	-	+	-	-	+	-	+	+	-	+	-	+	-	-	+	+	-
g07	+	-	-	+	-	+	-	+	+	-	+	-	+	-	+	-	-	+
g08	+	-	+	+	+	+	+	+	+	+	+	-	+	+	+	+	+	+
g09	+	-	+	-	-	+	-	+	+	-	+	-	+	-	-	+	-	+
g10	+	-	+	-	-	+	-	+	-	+	+	-	+	-	-	+	-	+
g11	+	-	+	+	+	+	+	-	+	+	+	+	+	+	+	-	+	-
g12	+	+	+	+	+	+	+	-	+	-	+	+	+	+	+	-	+	+
g13	+	-	-	+	-	+	-	+	+	-	-	-	-	+	-	+	-	+
Toplam	12	3	10	9	6	12	8	8	11	5	11	2	11	8	8	7	9	8

Tablo 5.7. ABC algoritmasının 350000 değerlendirme ile 13 fonksiyon üzerinde elde edilen 30 koşmaya ait istatistiksel sonuçlar.

Problem	**Optimal**	**En iyi**	**Ortalama**	**En kötü**	**Standart Sapma**
g01	-15.000	-15.000	-15.000	-15.000	0.000
g02	0.803619	0.803611	0.795430	0.770319	0.009466
g03	1.000	1.000	1.000	1.000	0.000
g04	-30665.539	-30665.539	-30665.539	-30665.539	0.000
g05	5126.498	5126.487	5182.868	5374.430	68.584
g06	-6961.814	-6961.814	-6961.814	-6961.813	0.0004
g07	24.306	24.324	24.447	24.835	0.113
g08	0.095825	0.095825	0.095825	0.095825	0.000
g09	680.63	680.631	680.636	680.641	0.0026
g10	7049.25	7058.823	7220.106	7493.943	122.589
g11	0.75	0.750	0.750	0.750	0.000
g12	1.000	1.000	1.000	1.000	0.000
g13	0.053950	0.757	0.975	1.000	0.051

Tablo 5.8. 350000 değerlendirme yapan ABC algoritması ile diğer algoritmaların kıyaslanması ile elde edilen başarı oranları. + algoritmanın daha iyi olduğunu - daha kötü olduğunu göstermektedir.

Prb	**ABC-HM**		**ABC-SR**		**ABC-ISR**		**ABC-OPA**		**ABC-ASCHEA**		**ABC-GA**		**ABC-SMES**		**ABC-PSO**		**ABC-DE**	
	ABC	**HM**	**ABC**	**SR**	**ABC**	**ISR**	**ABC**	**OPA**	**ABC**	**ASCHEA**	**ABC**	**GA**	**ABC**	**SMES**	**ABC**	**PSO**	**ABC**	**DE**
g01	+	-	+	+	+	+	+	+	+	-	+	-	+	+	+	-	+	-
g02	-	+	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-
g03	+	+	+	+	-	+	+	-	+	+	+	-	+	+	+	-	+	+
g04	+	-	+	+	+	+	+	+	+	-	+	-	+	+	+	+	+	+
g05	+	-	-	+	-	+	+	-	-	+	-	-	-	+	-	+	+	-
g06	+	-	+	-	+	+	+	+	+	-	+	-	+	-	+	+	+	-
g07	+	-	-	+	-	+	-	+	+	-	+	-	+	-	+	-	-	+
g08	+	-	+	+	+	+	+	+	+	+	+	-	+	+	+	+	+	+
g09	+	-	+	-	-	+	-	+	+	-	+	-	+	-	-	+	-	+
g10	+	-	+	-	-	+	-	+	-	+	+	-	+	-	-	+	-	+
g11	+	-	+	+	+	+	+	-	+	+	+	+	+	+	+	-	+	-
g12	+	+	+	+	+	+	+	-	+	-	+	+	+	+	+	-	+	+
g13	+	-	-	+	-	+	-	+	+	-	-	-	-	+	-	+	-	+
Toplam	12	3	10	9	7	12	9	8	11	5	11	2	11	8	9	7	9	8

Tablo 5.9. 9 farklı MR değeri ile elde edilen hata değerleri için ANOVA Tablosu

Fonksiyon	SS	DF	MS	F	$Prob > F$
g01	0	8	0	0	1
g02	0.00442	8	0.00055	8.24	0
g03	0.00042	8	5.28099e-005	199.49	0
g04	61136.1	8	7642.02	78.47	0
g05	758893.3	8	94861.7	2.82	0.0053
g06	0	8	0	-0	1
g07	29.8198	8	3.72748	57.08	0
g08	2.31112e-032	8	2.88889e-033	-3.77302e-016	1
g09	25.0808	8	8 3.1351	304.58	0
g10	1258569.7	8	157321.2	8.2	0
g11	6.20552e-007	8	7.7569e-008	4.53	0
g12	0	8	0	NaN	NaN
g13	3.62352	8	0.45294	28.47	0

Tablo 5.10. 5 farklı *limit* değeri ile elde edilen hata değerleri için ANOVA Tablosu

Fonksiyon	SS	DF	MS	F	$Prob > F$
g01	4.73317e-028	4	1.18329e-028	3.77302e-016	1
g02	0.00018	4	0.00004	0.38	0.8253
g03	0.00039	4	9.78255e-005	206.33	0
g04	55877.1	4	13969.3	79.93	0
g05	219937.6	4	54984.4	2.89	0.0252
g06	617376.5	4	154344.1	140.85	0
g07	28.3687	4	7.09218	82.25	0
g08	0	4	0	-0	1
g09	22.5267	4	5.63167	377.56	0
g10	997158.2	4	249289.6	13.25	0
g11	4.78378e-007	4	4 1.19594e-007	6.5	0.0001
g12	0	4	0	NaN	NaN
g13	0.80775	4	0.20194	15.42	0

Tablo 5.11. 5 farklı SPP değeri ile elde edilen hata değerleri için ANOVA Tablosu

Fonksiyon	SS	DF	MS	F	$Prob > F$
g01	4.73317e-028	4	1.18329e-028	3.77302e-016	1
g02	0.00091	4	0.00023	2.89	0.0253
g03	5.41396e-007	4	1.35349e-007	187.09	0
g04	2.49742e-019	4	6.24356e-020	-7.91077e-014	1
g05	321282.5	4	80320.6	5.12	0.0008
g06	155.567	4	38.8917	46.84	0
g07	0.09214	4	0.02304	1.35	0.2575
g08	0	4	0	-0	1
g09	0.00066	4	0.00017	15.04	0
g10	182792.5	4	45698.1	3.69	0.0073
g11	1.68171e-008	4	4.20427e-009	4.56	0.0019
g12	0	4	0	NaN	NaN
g13	0.06132	4	0.01533	2.52	0.0449

Tablo 5.12. 9 farklı MR değeri (0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9) ile elde edilen hata değerleri için ANOM Tablosu

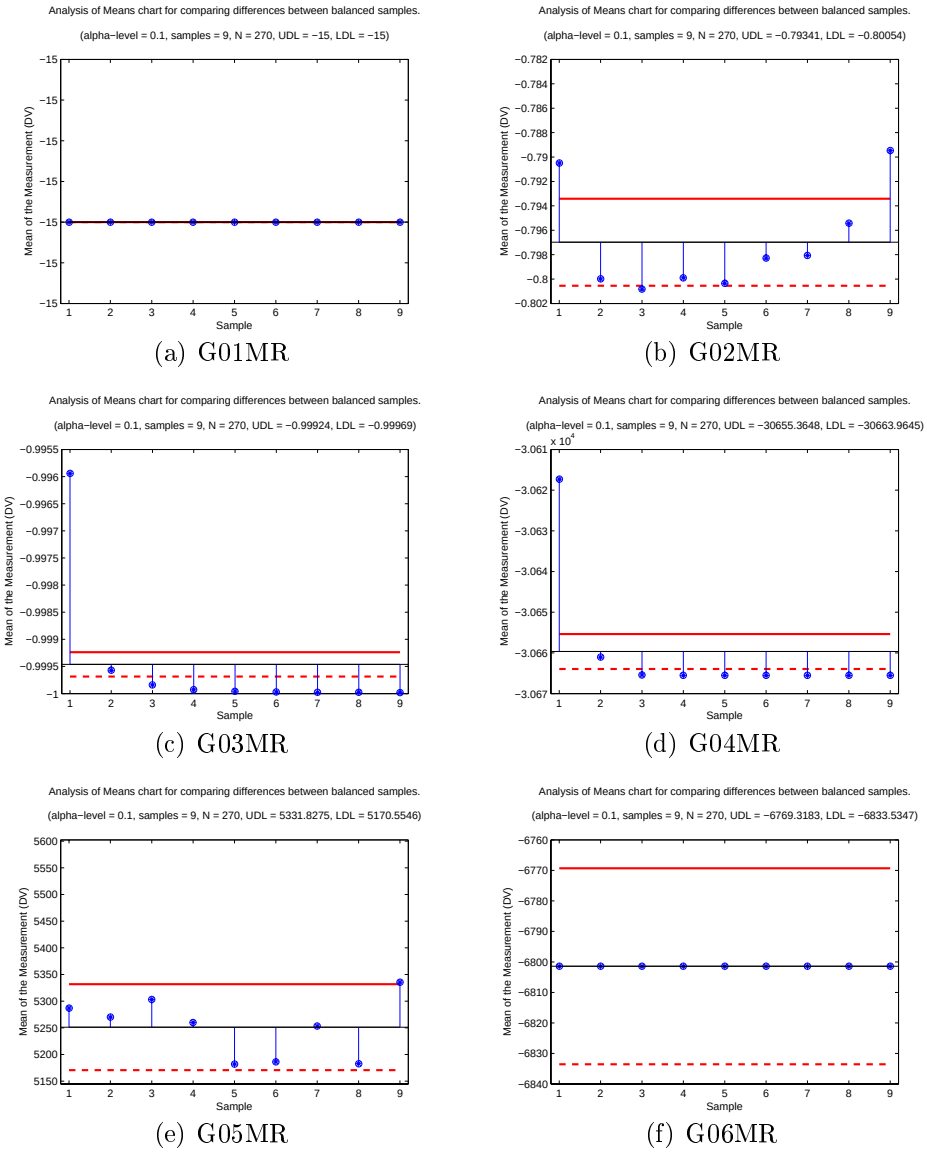
Fonksiyon	Limit üzeri	Limit altı	Farklılık yok
g01	-	-	0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9
g02	0.1, 0.9	0.3	0.2,0.4,0.5,0.6,0.7,0.8,
g03	0.1	0.3, 0. 4, 0.5, 0.6, 0.7, 0.8, 0.9	0.2
g04	0.1	0.3, 0. 4, 0.5, 0.6, 0.7, 0.8, 0.9	0.2
g05	0.9	-	0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,
g06	-	-	0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9
g07	0.1, 0.2	0.5, 0.6, 0.7, 0.8, 0.9	0.3, 0.4
g08	-	-	0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9
g09	0.1, 0.2	0.4,0.5,0.6,0.7,0.8,0.9	0.3
g10	0.1	-	0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9
g11	0.1	-	0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9
g12	-	-	0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9
g13	0.5, 0.6, 0.7, 0.8, 0.9	0.1, 0.2, 0.3	0.4

Tablo 5.13. 5 farklı *limit* değeri (0.1xSNxD, 0.5xSNxD, SNxD, 2xSNxD, 4xSNxD) ile elde edilen hata değerleri için ANOM Tablosu

Fonksiyon	Limit üzeri	Limit altı	Farklılık yok
g01	-	-	0.1, 0.5, 1, 2, 4
g02	-	-	0.1, 0.5, 1, 2, 4
g03	0.1	0.5, 1, 2, 4	-
g04	0.1	0.5, 1, 2, 4	-
g05	0.1	0.5, 1, 2, 4	-
g06	0.1	0.5, 1, 2, 4	-
g07	0.1	0.5, 1, 2, 4	-
g08	-	-	0.1, 0.5, 1, 2, 4
g09	0.1	0.5	1, 2, 4
g10	0.1	-	0.5, 1, 2, 4
g11	-	-	0.1, 0.5, 1, 2, 4
g12	-	-	0.1, 0.5, 1, 2, 4
g13	1	0.1	0.5, 2, 4

Tablo 5.14. 5 farklı SPP değeri (0.1xSNxD, 0.5xSNxD, SNxD, 2xSNxD, 4xSNxD) ile elde edilen hata değerleri için ANOM Tablosu

Fonksiyon	Limit üzeri	Limit altı	Farklılık yok
g01	-	-	0.1, 0.5, 1, 2, 4
g02	2	0.1	0.5, 1, 4
g03	0.1	0.5, 1, 2, 4	-
g04	0.1	-	0.5, 1, 2, 4
g05	4	-	0.1, 0.5, 1, 2
g06	0.1	0.5, 1, 2, 4	-
g07	-	-	0.1, 0.5, 1, 2, 4
g08	-	-	0.1, 0.5, 1, 2, 4
g09	0.1	1	0.5, 2, 4
g10	4	0.1	0.5, 1, 2
g11	0.1	-	0.5, 1, 2, 4
g12	-	-	0.1, 0.5, 1, 2, 4
g13	-	4	0.1, 0.5, 1, 2

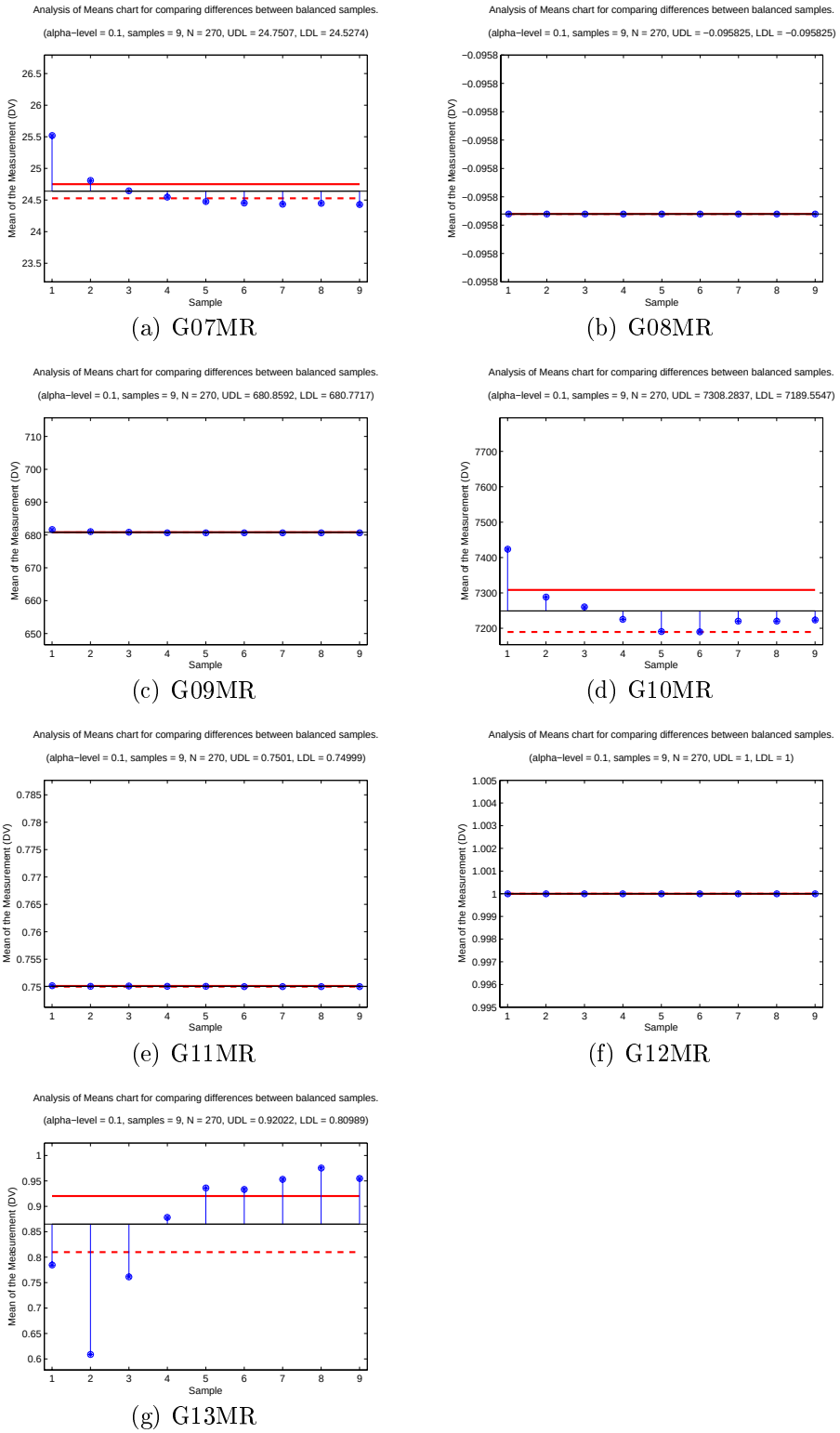


(a) G01MR (b) G02MR

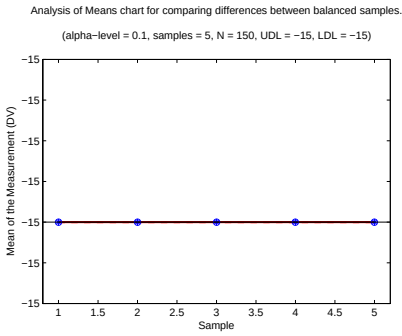
(c) G03MR (d) G04MR

(e) G05MR (f) G06MR

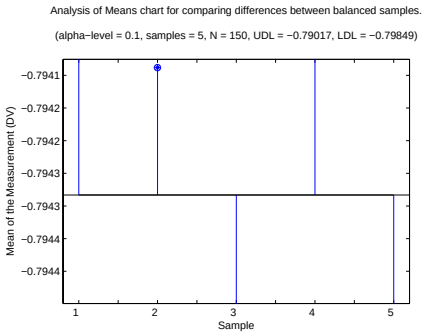
Şekil 5.3. 9 farklı MR değeri için g01-g06 problemlerinin ANOM grafikleri



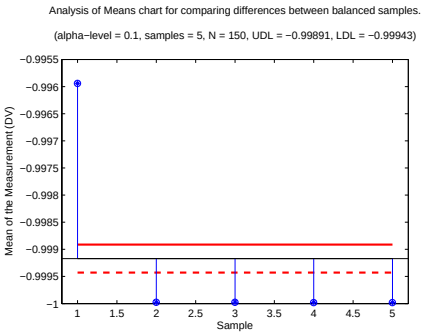
Şekil 5.4. 9 farklı MR değeri için g07-g13 problemlerinin ANOM grafikleri



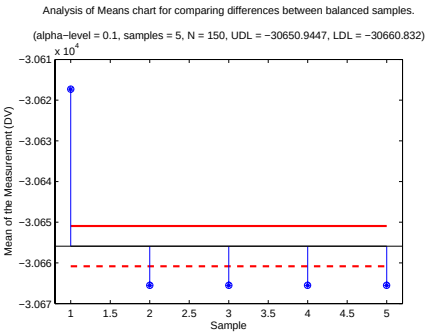
(a) G01L



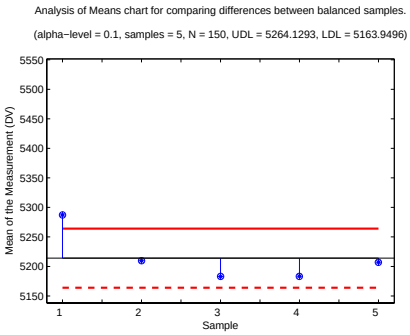
(b) G02L



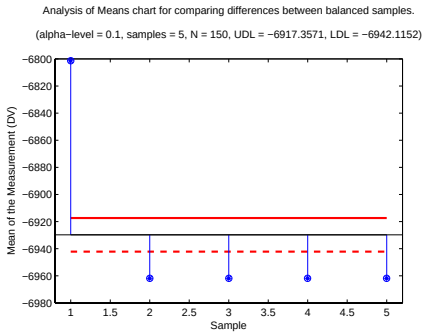
(c) G03L



(d) G04L

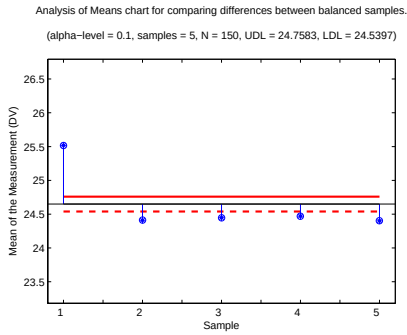


(e) G05L

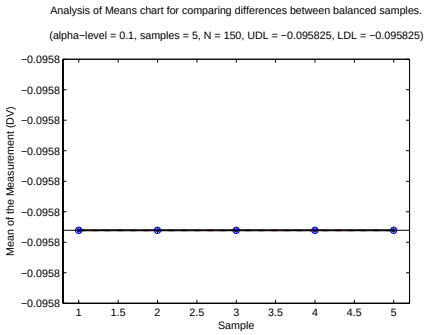


(f) G06L

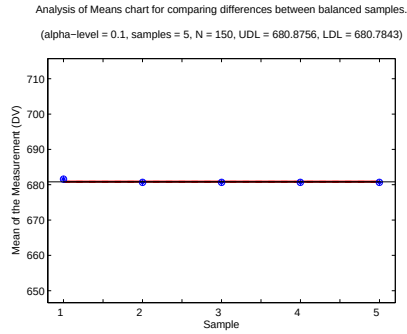
Şekil 5.5. 5 farklı *limit* değeri için g01-g06 problemlerinin ANOM grafikleri



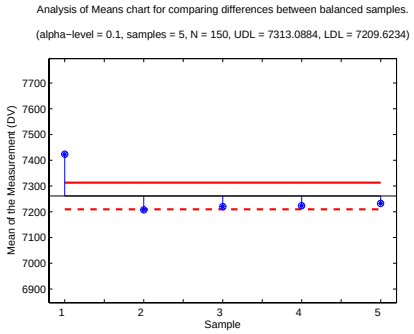
(a) G07L



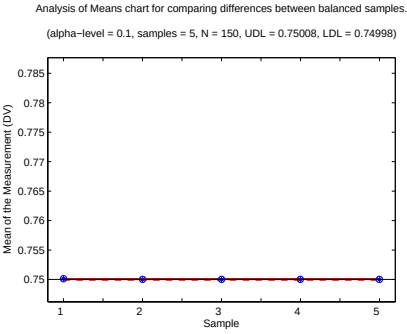
(b) G08L



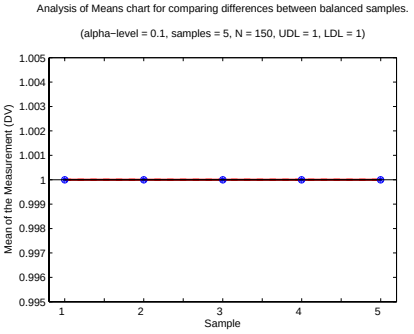
(c) G09L



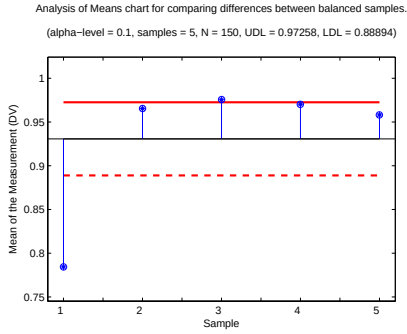
(d) G10L



(e) G11L

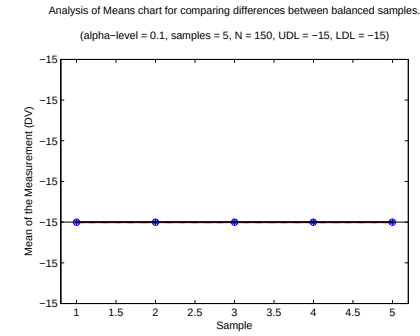


(f) G12L

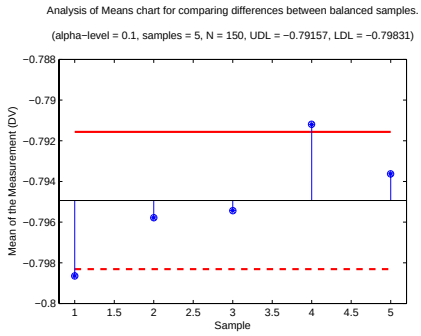


(g) G13L

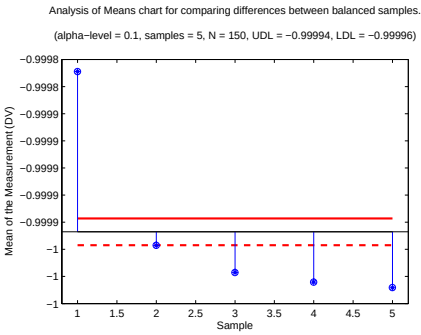
Şekil 5.6. 5 farklı *limit* değeri için g07-g13 problemlerinin ANOM grafikleri



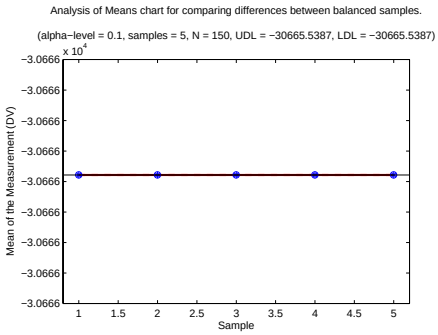
(a) G01S



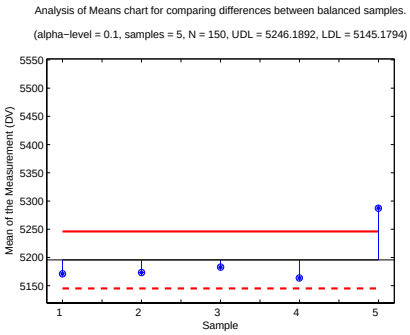
(b) G02S



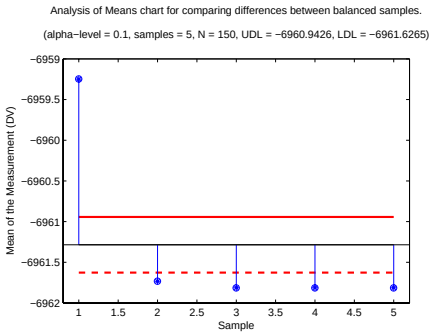
(c) G03S



(d) G04S

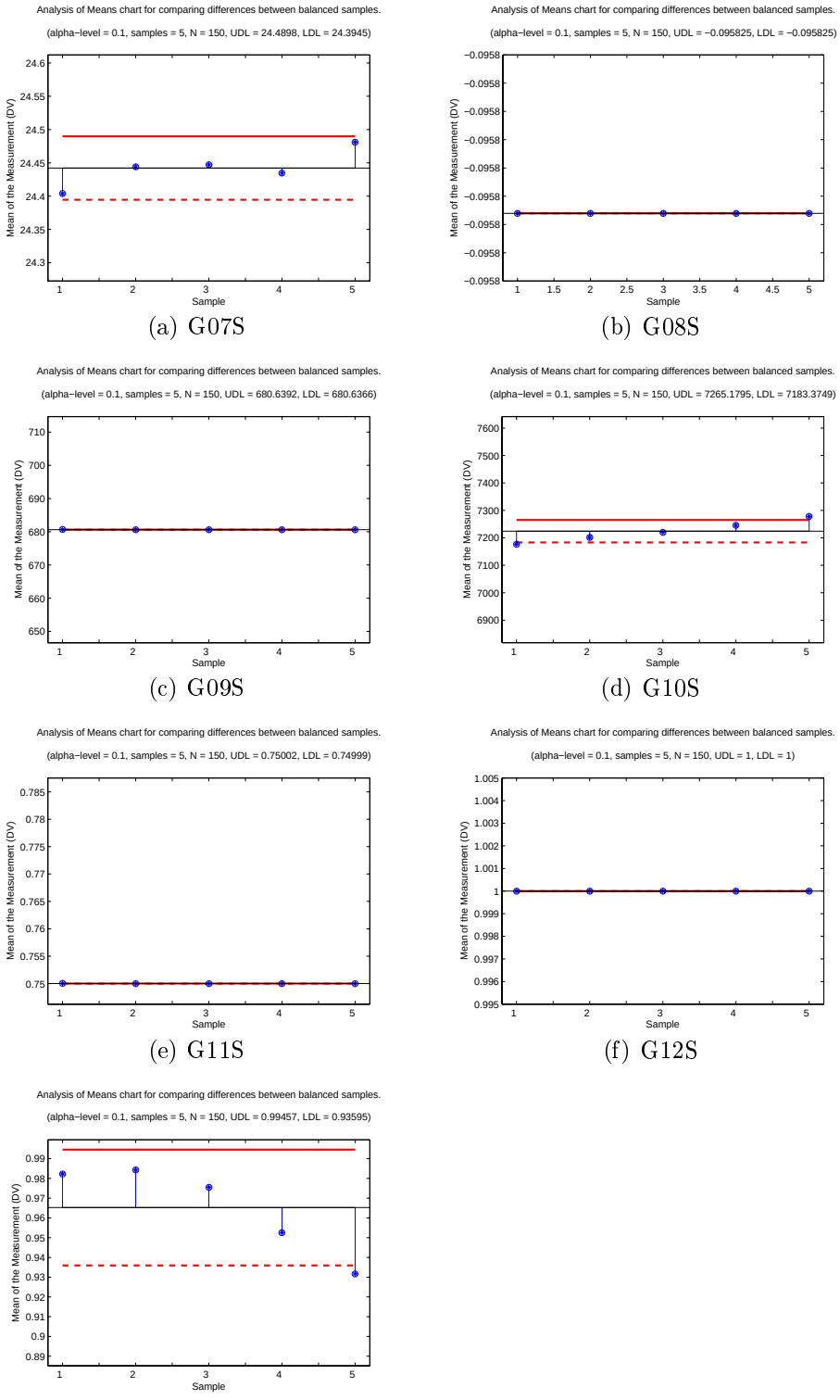


(e) G05S



(f) G06S

Şekil 5.7. 5 farklı SPP değeri için g01-g06 problemlerinin ANOM grafikleri



Şekil 5.8. 5 farklı SPP değeri için g07-g13 problemlerinin ANOM grafikleri

6. BÖLÜM

GERÇEK DÜNYA PROBLEMLERİNİN ÇÖZÜMÜNDE ABC ALGORİTMASININ KULLANILMASI

Şu ana kadar 4 ve 5. Bölümlerde ABC algoritmasının performans analizinde kullanılan problemler, test problemleri olarak adlandırılan ve optimizasyon literatüründe bilinen belli fonksiyonlardı. Bu bölümde yapay optimizasyon problemleri yerine kontrol mühendisliğinde ve makina mühendisliğinde karşılaşılan bazı sınırlamasız ve sınırlamalı gerçek dünya problemlerine ABC algoritması uygulanarak performansı analiz edilecektir. İlk olarak sınırlamasız bir optimizasyon problemi olan PID kontrolör tasarımı ele alınacak ve tasarıma ait parametreler ABC algoritması ile bulunarak, literatürdeki tekniklerle performansı kıyaslanacaktır. İkinci olarak ise makina mühendisliğinde karşılaşılan ve oldukça popüler olan altı adet sınırlamalı tasarım problemi için sonuçlar alınacak ve yorumlanacaktır.

6.1. PID Kontrolör Tasarımı

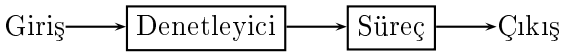
Kontrol sistemi belirli bir amaca yönelik olarak kendisini veya diğer bir sistemi ayarlayabilen fiziksel alt bileşenlerin tümüdür [253]. Kontrol sistemleri üretilen ürünlerin kalite kontrollerinde, otomatik montaj hatlarında, makine ve aletlerin kontrolünde, uzay teknolojilerinde, silah sistemlerinde, bilgisayarlı kontrol sistemlerinde, ulaşım ve güç sistemlerinde, robotik ve benzeri endüstri sistemlerinde oldukça yaygındır [254].

Endüstriyel anlamda bir kontrol sistemi; (i) kontrol edilen sistem ve (ii) kontrol elemanları (iii) sonuç yada çıkışlar olmak üzere üç temel ögeden oluşur [254]. Kontrol edilen sistem; özel bir niceliğin denetlendiği tesisat, süreç veya makinedir. Kontrol edilen sisteme uygulanacak uygun bir denetim sinyali sağlayan elemanda kontrol

elemanıdır.

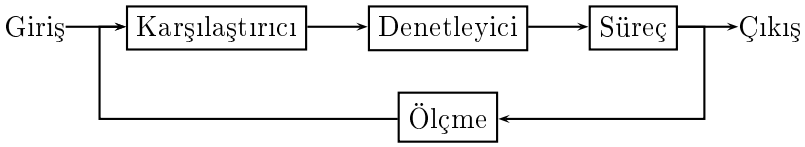
Kontrol sistemleri açık çevrimli (geribeslemesiz) ve kapalı çevrimli (geribeslemeli) kontrol sistemleri olmak üzere iki gruba ayrılır.

Açık çevrimli denetim sistemlerinde, denetim sistem çıkışından bağımsızdır. Açık çevrimli denetim sistemleri giriş-çıkış bağıntıları önceden belli olan ve iç veya dış bozuculara maruz kalmayan sistemlerde kullanılır [255]. Açık çevrimli sistem blok şeması Şekil 6.1 gösterilmiştir.



Şekil 6.1. Açık çevrimli sisteme ait blok şema

Kapalı çevrimli sistemler istenilen çıkış ile ölçülen çıkış arasındaki farkın değerlendirilerek, hatayı gidermek üzere sistemin değiştirilmesine dayanmaktadır. Kapalı çevrimli sistem blok şeması Şekil 6.2'de gösterilmiştir.



Şekil 6.2. Kapalı çevrimli sisteme ait blok şema

Geri beslemeli kontrol sistemi genelde çıkış ile daha önceden belirlenmiş olan sistemin referans girişi arasında hataları azaltmak amacıyla kullanılmaktadır. Ayrıca, kararlılık, bant genişliği, toplam kazanç, bozucu ve duyarlılık gibi sistem davranış karakteristiklerini de etkilemektedir [254].

Otomatik kontrol sistemi davranışı "geçici-durum" ve "kalıcı-durum" olmak üzere iki çeşittir. Geçici-durum davranışı, sistemin belli bir dış uyarı karşısında belli bir başlangıç değerinden bir nihai duruma kadar zaman değişimine bağlı olarak gösterdiği davranıştır. Kalıcı-durum davranışı ise sistemin geçici-durum davranışı tamamlandıktan sonra zaman sonsuza giderken koruduğu davranıştır. Bir sistemin

kalıcı durum haline en küçük hata veya sıfır hata ile ulaşması istenir. Kontrol sistemlerinde geri besleme kullanılmak suretiyle kalıcı durum hatası en küçük değere veya sıfıra düşürülebilir [255]. Kontrol sistemlerinin kararlı çalışması, hızlı cevap verebilmesi ve hatasının olmaması beklenir. Sistem çok hızlı ve duyarlı iken kararsız çalışmaya eğilimlidir veya kalıcı-durum hatası çok düşük iken sistemin cevap hızı veya duyarlılığı düşüktür. Kontrol sistemlerinin tasarımında bu kriterler arasında uygun bir uzlaşma sağlamaya çalışılır.

6.1.1. Geçici-Durum Cevabı

Doğrusal kontrol sistemlerinde geçici-durum yanıtının değerlendirilmesi genellikle birim basamak cevabından yararlanılarak yapılır. Sistemin basamak giriş cevabının bilinmesi halinde matematiksel olarak sistemin diğer herhangi bir giriş cevabı kolaylıkla hesaplanabilir. Birim basamak yanıtı ile ilişkili olarak kontrol sistemlerinin zaman domeni cevabı parametreleri Şekil 6.3'deki gibi verilebilir.

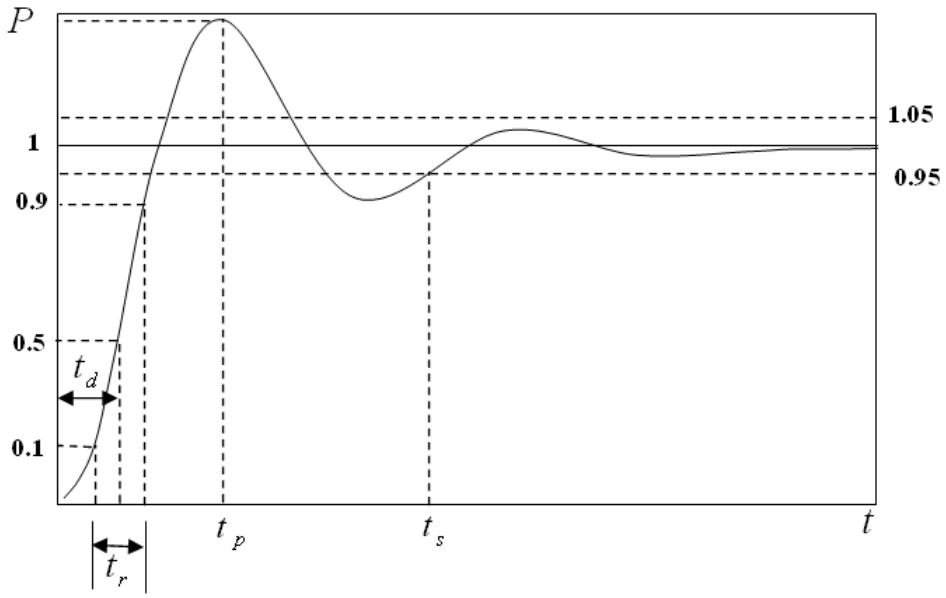
Gecikme zamanı, t_d: Basamak cevabının son değerinin yarısına ilk defa ulaşması için geçen zamandır.

Yükselme zamanı, t_r: Basamak cevabının son değerinin %10'undan %90'nına; %5'inden %95'ine veya %0'ından %100'üne kadar geçen zaman olarak tanımlanır. Genellikle aşırı sönümlü birinci dereceden sistemler için %0-100 yükselme zamanı olarak kullanılırken titreşimli sönümlü sistemlerde %10-90 olarak alınmaktadır.

Tepe zamanı, t_p: Basamak cevabının son değerini ilk defa aşarak ilk tepe (maksimum) noktaya erişmesi için geçen zamandır.

Yerleşme zamanı, t_s: Basamak cevabında titreşim genliklerinin izin verilen tolerans bandına düşmesi için geçen zamandır. İzin verilen tolerans değerleri ise genellikle son değerin %5 veya %2'lik aşma değerleri olarak tanımlanır.

Maksimum aşma (P): Basamak cevabın erişmesi gereken değerinden maksimum sapma gösterdiği değerdir. Maksimum aşma miktarı doğrudan doğruya sistemin bağıl kararlılığını belirler.



Şekil 6.3. Geçici-durum cevabı parametreleri

6.1.2. PID Kontrolör

Kontrol birimleri ikili denetim, orantı denetim (P), integral denetim (I) ve türev denetim (D) etkisi olmak üzere belli başlı dört temel denetim etkisine sahiptir.

Bu temel denetim etkilerinin bir veya birkaçının bir arada uygun şekilde kullanılmasıyla değişik kontrol etkilerinde çalışan PI, PD ve PID gibi denetim birimleri oluşturulur. PID denetim üç temel denetim etkisinin (P,I,D) birleşiminden meydana gelmiştir. PID kontrolör ve türevlerinin (P, PI, PD), yapılarının basit ve çalışma şartlarının geniş bir aralığında oldukça kararlı olmaları nedeniyle endüstriyel süreç kontrolünde yaygın olarak kullanılmaktadır [2].

Bir PID kontrol edicinin transfer fonksiyonu eşitlik 6.1 formundadır.

$$G_c(s) = \frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_d s \tag{6.1}$$

Burada K_p, K_i ve K_d sırasıyla ile oransal, integral ve türev kazançları, U(s) kontrol edicinin çıkışı, E(s) ise hata sinyali ve kontrol edicinin girişidir.

Tablo 6.1'de orantı, integral ve türev kazançlarının zaman domeni cevabı üzerindeki

etkisini vermektedir.

Tablo 6.1. Orantı, İntegral ve Türev karakteristikleri

Denetleyici	t_r	Sistem ani tepkisi	t_s	E_{ss}
K_p	Azalır	Artar	Küçük değişim	Azalır
K_i	Azalır	Artar	Artar	Yok eder
K_d	Küçük değişim	Azalır	Azalır	Küçük değişim

Burada E_{ss} kalıcı-durum hatasıdır.

İntegral etki sistemde çıkabilecek kalıcı-durum hatasını sıfırlarken türev etkide yalnızca PI denetim etkisi kullanılması haline göre sistemin aynı bağıl karalılığı için cevap hızını arttırmaktadır. Buna göre PID denetim birimi sistemde sıfır kalıcı-durum hatası olan hızlı bir cevap sağlamaktadır. PID denetimin sağlayacağı bu avantajdan yaralanmak için sırasıyla orantı (K_p), integral (K_i) ve türev (K_d) kazançlarının uygun şekilde ayarlanması gerekmektedir.

PID kontrolör parametreleri bulunurken sistem performansının değerlendirilmesinde farklı hata fonksiyonları (performans indisleri) kullanılabilmektedir. Kontrolör tasarımında kullanılan hata fonksiyonlarına göre ölçülen geçici-durum ve kalıcı-durum parametreleri farklılık gösterebilmektedir. Aşağıda yaygın olarak kullanılan hata fonksiyonları hakkında bilgi verilmiştir.

6.1.2.1. Mutlak Hatanın İntegrali (Integral of the Absolute Error, IAE)

IAE hata değeri, sistemin zaman domeni cevabının geçici-durumda çalışması halinde meydana gelen hata değerlerinin toplamı ile ifade edilir. Birim basamak girişi için IAE hata değeri eşitlik 6.2 ile hesaplanmaktadır.

$$\begin{aligned} IAE &= \int_0^\infty |e(t)|\, dt \\ e(t) &= 1 - y(i) \\ IAE &= \sum_i abs(1 - y(i)) \end{aligned} \tag{6.2}$$

6.1.2.2. Karesel Hatanın İntegrali (Integral of the Square Error, ISE)

ISE hata değeri, sistemin zaman domeni cevabının geçici-durumda çalışması halinde meydana gelen hata değerlerinin kareleri toplamı ile ifade edilir. ISE hata kriterinde hatanın karesi alındığından dolayı büyük hatanın etkisi, küçük hatanın etkisine göre çok daha fazla olmaktadır. Sistem zaman domeni cevabında büyük taşmalar olduğunda toplam hata çok yüksek çıkarken sürekli küçük salınım gösteren sistemlerde toplam hata daha az olmaktadır. Birim basamak girişi için ISE hata değeri eşitlik 6.3 ile hesaplanmaktadır.

$$\begin{aligned} ISE &= \int_0^\infty [e(t)]^2 dt \\ e(t) &= 1 - y(i) \\ ISE &= \sum_i (1 - y(i))^2 \end{aligned} \tag{6.3}$$

6.1.2.3. Zaman Ağırlıklı Mutlak Hatanın İntegrali (Integral of the Time-weighted Absolute Error, ITAE)

ITAE hata değeri, sistemin geçici-durum çalışması halinde meydana gelen hata değerlerinin hatanın meydana geldiği zamanla çarpılıp toplanması ile belirlenir. ITAE hata kriterinde hatanın zamanla çarpılmasından dolayı başlangıçta meydana gelen hatanın toplam hataya etkisi az olurken ilerleyen zamanlarda meydana gelen hataların toplam hataya etkisi daha fazla olur. E_{ss}'nin küçük olması istenen durumlarda tercih edilir. Sürekli salınım gösteren sistemlerde ise toplam hata çok yüksek değerler alır. Birim basamak girişi için ITAE hata değeri eşitlik 6.4 kullanılarak hesaplanmaktadır.

$$\begin{aligned} ITAE &= \int_0^\infty t\,|e(t)|\,dt \\ e(t) &= 1 - y(i) \\ ITAE &= \sum_i (abs(1 - y(i))) * t(i) \end{aligned} \tag{6.4}$$

6.1.2.4. Çıkış Farkı ve Yükselme Zamanına Bağlı Mutlak Hata (PWAE)

PWAE hata değeri, IAE hata değerine yükselme zamanını azaltmak amacıyla çıkışın önceki değeri ile arasındaki mutlak farkın ve ağırlıklandırılmış yükselme zamanının eklenmesi ile hesaplanmaktadır. PWAE değeri eşitlik 6.5 kullanılarak belirlenmektedir.

$$\begin{aligned} &if\ ey(t) < 0 \\ &F(t) = \int_0^\infty \left(\omega_1 \left|e(t)\right| + \omega_4 \left|ey(t)\right|\right) dt + \omega_3 t_r \\ &else \\ &F(t) = \int_0^\infty \left(\omega_1 \left|e(t)\right|\right) dt + \omega_3 t_r \end{aligned} \tag{6.5}$$

Burada $ey(t)$ çıkışın bir önceki değeri ile arasındaki farka karşılık gelmektedir.

6.1.2.5. Çıkış Farkı ve Yükselme Zamanına Bağlı Zaman ile Ağırlıklandırılmış Mutlak Hata (PWTAE)

PWAE hata değerinin ürettiği sonuçların oturma zamanını azaltmak için hata değeri ITAE hata değerinde olduğu gibi hatanın meydana geldiği zamanla çarpılıp toplanması ile belirlenir (6.6).

$$\begin{aligned} &if\ ey(t) < 0 \\ &F(t) = \int_0^\infty \left(\omega_1 \left|e(t) * t\right| + \omega_4 \left|ey(t)\right|\right) dt + \omega_3 t_r \\ &else \\ &F(t) = \int_0^\infty \left(\omega_1 \left|e(t) * t\right|\right) dt + \omega_3 t_r \end{aligned} \tag{6.6}$$

6.1.2.6. Çıkış Farkı, Yükselme Zamanı ve Maksimum Taşmaya Bağlı Zaman ile Ağırlıklandırılmış Mutlak Hata (PWOTAE)

PWTAE performans indisini kullanarak tasarlanan sistemin kararlılığını artırmak üzere yüzde maksimum aşma da amaç fonksiyonuna eklenmesiyle ifade edilebilir (6.7).

$$
\begin{aligned}
&if\ ey(t) < 0 \\
&F(t) = \int_0^\infty \left(\omega_1 \left|e(t) * t\right| + \omega_4 \left|ey(t)\right|\right) dt + \omega_3 t_r + PO \\
&else \\
&F(t) = \int_0^\infty \left(\omega_1 \left|e(t) * t\right|\right) dt + \omega_3 t_r + PO
\end{aligned}
\tag{6.7}
$$

Burada PO yüzde maksimum aşmaya karşılık gelmektedir.

6.1.3. ABC Algoritması ile PID Kontrolör Parametrelerinin En İyilenmesi

PID kontrolörlerin tasarımında, kontrolör parametrelerinin uygun şekilde belirlenmesi son derece önemlidir. Kontrol edilecek sistemin yapısı (nonlineerliği, derecesi v.b.) nedeniyle bu parametrelerin uygun şekilde belirlenmesi oldukça zor olabilmektedir. Diğer bir problem ise adaptif kontrolör tasarımında kontrolör parametrelerinin belirlenmesi için harcanacak sürenin sınırlı olmasıdır. Bu parametrelerin belirlenmesi amacıyla geliştirilmiş birçok klasik teknik bulunmaktadır [256, 257]. Bu klasik teknikler çoğunlukla karmaşık kontrol sistem tasarımlarında etkin sonuçlar vermemekle birlikte yoğun matematiksel işlemlere ihtiyaç duymaktadırlar. Ayrıca hedeflenen tasarım makul bir sürenin ötesinde zaman gerektirebilmektedir. Bu gibi nedenlerle kontrol alanında çalışan araştırmacıların sezgisel algoritmalar ve yapay sinir ağları kullanımına olan ilgisi gittikçe artmaktadır. Karmaşık kontrol sistemlerini tasarlamak için genetik algoritma, tabu araştırma, ısıl işlem ve yapay sinir ağları üzerine dayalı yaklaşımların kullanılmasına ilişkin literatürde çeşitli çalışmalar bulunmaktadır [255].

Bu bölümde, ABC algoritması kullanılarak PID denetleyici tasarımı gerçeklenecektir. ABC algoritmasının performansı iki adet sistem üzerinde test edilmiştir. ABC algoritmasından elde edilen sonuçlar klasik metotlardan Ziegler-Nichols (ZN) [256], CHR [257], Kitamori [258] metotları ile modern yaklaşımlardan tabu araştırma algoritması ile karşılaştırılmıştır. Her iki sistem içinde PID tasarım parametrelerinin alt ve üst sınırları [0,10] olarak alınmıştır. ABC algoritmasına ait koloni büyüklüğü 20, maksimum çevrim sayısı 100 ve limit değeri 10 olarak alınmıştır. Sistem 1 için zaman aralığı 0 ile 5 arası alınmış ve 0.01 aralıklarla ölçeklenmiş; sistem 2 için ise 0 ile 12 arası alınmış ve 0.02 aralıklarla

ölçeklenmiştir.

6.1.3.1. Sistem 1 [1]

Kontrolör tasarımı gerçekleştirilecek ilk sistem eşitlik 6.8 ile tanımlı transfer fonksiyonuna sahip üçüncü dereceden bir sistemdir.

$$G_p(s) = \frac{64}{s^3 + 14s^2 + 56s + 64} \tag{6.8}$$

Sistem 1 ile verilen sisteme ait PID kontrolör parametrelerinin tasarımı için ABC algoritması kullanılmış ve performansı geleneksel kontrolör tasarım metotlarından ZN [256] ve CHR [257] yöntemleri ile kıyaslanmıştır. Ayrıca ABC içerisinde amaç fonksiyonu olarak daha önce tanımlanan farklı performans indisleri kullanılarak performans indislerinin sistem cevabı üzerindeki etkisi incelenmiştir.

ZN, CHR ve ABC algoritmalarına ait K_d, K_p, K_i, PO (yüzde maksimum aşma), t_r, t_s ve IAE, ISE, ITAE, PWAE ve PWTAE hata değerleri Tablo 6.2'da verilmektedir. Hata değerlerinin bazıları karesel, bazıları mutlak olduğu için doğrudan farklı performans indislerinin ürettiği hata değerlerini kıyaslamak mantıklı olmamaktadır. CHR ve ZN'ya ait IAE değerleri hesaplanarak tabloda verilmiştir. ABC algoritmasının, IAE'ye dayalı amaç fonksiyonu kullanılarak koşulmasıyla elde edilen sonuçlar CHR ve ZN ile kıyaslandığında ABC algoritmasının IAE hata değerinin daha düşük olduğu görülmektedir. Ayrıca ABC algoritması daha düşük t_r ve t_s değerlerine sahiptir. Farklı performans indisi kullanan ABC algoritmaları kendi aralarında kıyaslanacak olursak PWAE ve PWTAE kullanıldığı durumlarda daha iyi sonuç elde edildiği görülmektedir. IAE performans indisi ISE ve ITAE'ye göre daha düşük maksimum aşma değeri üretirken ISE ve ITAE'nin yükselme zamanı daha düşüktür. Performans indisinde zaman çarpan olarak kullanıldığında sistem hızlı cevap vermekte ancak maksimum aşma arttığı için kararlılığı azalmaktadır. PWAE ve PWTAE, zamana bağlı çıkışlar arasındaki farkın negatif olduğu durumlardaki farkları da amaç fonksiyonunda kullandığı için yerleşme zamanı düşük, mutlak farka dayandığı için maksimum taşması düşük, ve yükselme zamanını da kullandığı için yükselme zamanı düşük bir sistem üretmektedir. Yani, hem hızlı cevap verebilen,

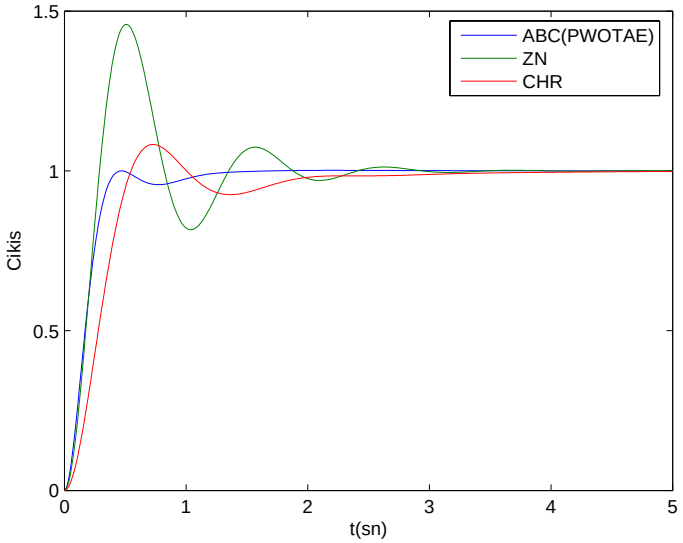
hem duyarlı hem de kararlı bir sistem gerçeklenmiş olmaktadır. ABC algoritmasının basit yapısı ve etkili sonuç üretimi bu amaç fonksiyonunun etkisi ile biraraya geldiğinde kararlı ve hızlı cevap verebilen bir sistem gürbüz, basit ancak etkili bir algoritma ile tasarlanmış olmaktadır. Literatürde yer alan çeşitli çalışmalarda da performans fonksiyonu seçiminin tasarlanacak sistemin geçici-durum ve kalıcı-durum parametreleri üzerinde etkili olduğu gösterilmiştir [2, 256].

ZN, CHR metotlarının ve ABC(PWAE) algoritmasının tasarladıkları sistemlerin birim basamak cevaplarına ait grafikler Şekil 6.4'de verilmektedir. ABC algoritmasının farklı performans indislerine göre koşulmasıyla gerçekleştirilen sistemlerin birim basamak cevaplarına ait grafikler de Şekil 6.5'de verilmektedir.

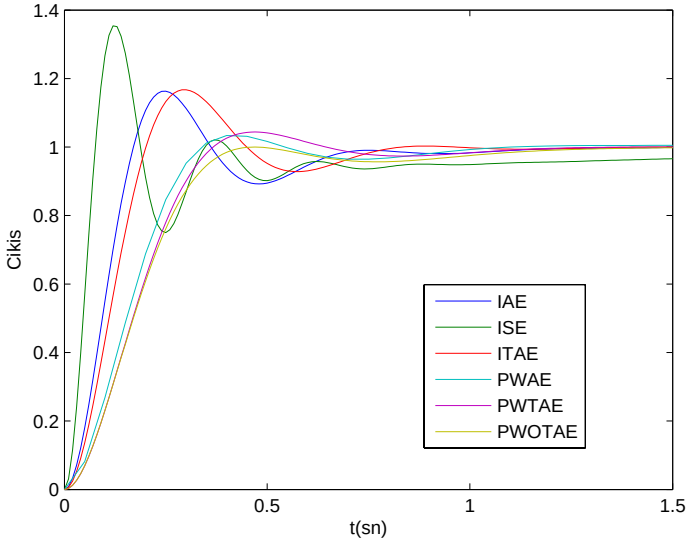
Sistem 1 için ABC (PWAE) algoritmasının 30 koşmasının en iyi PWAE değerlerinin ortalamalarına ait gelişim grafiği Şekil 6.6'da verilmektedir. Bu koşmalarda zaman aralığı 0 ile 5 aralığı olarak alınmış ve 0.05 aralıklarla ölçeklenmiştir.

Tablo 6.2. Sistem 1 için ZN, CHR ve ABC ile gerçekleştirilen tasarımların K_d, K_p, K_i, PO (yüzde maksimum aşma), t_r, t_s ve en iyilenme yapıldığı hata değerleri

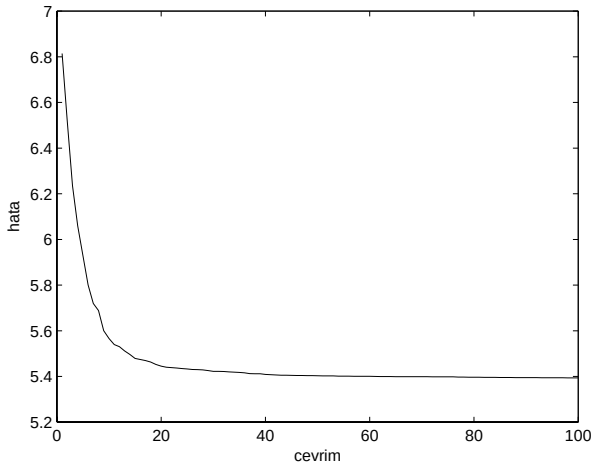
	K_d	K_p	K_i	PO	t_r	t_s	Hata
CHR [257](IAE)	0.331	3.31	2.9982	8.2632	0.4300	1.66	38.4584
ZN [256](IAE)	0.7087	6.75	16.0714	45.8456	0.2300	1.7	42.9923
ABC(IAE)	2.7796	8.8797	10.0000	16.3241	0.1300	0.61	15.0453
ABC(ITAE)	2.0455	8.6107	10.0000	16.7120	0.1600	0.67	2.79967
ABC(ISE)	10.0000	10.0000	10.0000	35.4011	0.0600	1.02	5.44144
ABC (PWAE)	1.2194	4.8889	6.3129	3.5111	0.2600	0.64	19.9057
ABC(PWTAE)	1.0175	4.6852	5.7031	4.3916	0.2800	0.33	4.62102
ABC(PWOTAE)	1.0376	4.2486	5.0349	0.1646	0.3100	0.36	5.8057



Şekil 6.4. Sistem 1 için ZN, CHR ve ABC ile gerçekleştirilen tasarımların birim basamak cevabı



Şekil 6.5. Sistem 1 için ABC algoritmasının farklı performans indislerine göre koşulmasıyla gerçekleştirilen tasarımların birim basamak cevabı



Şekil 6.6. Sistem 1 için PWAE performans indisi kullanan ABC algoritmasının 30 koşmasının en iyi değerlerinin ortalamalarına ait gelişim grafiği

6.1.3.2. Sistem 2 [2]

Eşitlik 6.9 ile transfer fonksiyonu tanımlanan ikinci sistem zaman gecikmeli ve ikinci dereceden bir sistemdir. Zaman gecikmesi içeren sistemler giriş işaretine belirli bir zaman geçmeden yanıt veremez. Zaman gecikmeli sistemlerin transfer fonksiyonları rasyonel olmadığı için sistemlerin işlenmesi daha zordur. Pade açılımı gibi yaklaşımlar kullanılarak bu tür sistemlerin transfer fonksiyonunun rasyonel olarak yazılabilmesi sağlanabilir.

$$G_p(s) = \frac{e^{-0.5s}}{(s+1)^2} \tag{6.9}$$

Sistem 2'ye ait kontrolör parametrelerinin tasarımı yine ABC algoritması kullanılarak yapılmış ve elde edilen sonuçlar geleneksel kontrolör tasarım metotlarından ZN [256], Kitamori [258] yöntemlerinin ve modern sezgisellerden Tabu Araştırma (TS) [2] algoritmasının sonuçları ile kıyaslanmıştır. ZN, Kitamori, TS ve ABC algoritmalarına ait K_d, K_p, K_i, PO (yüzde maksimum aşma), t_r, t_s ve hata değerleri Tablo 6.3'da verilmektedir. TS algoritması ITAE kullanılarak araştırma yapmıştır. Yine farklı performans indisleri kullanarak ABC algoritması ile sonuçlar alınmış ve bu tasarlanan sistema ait K_d, K_p, K_i, PO (yüzde maksimum

aşma), t_r, t_s ve hata değerleri de Tablo 6.3'da verilmiştir. ABC(ITAE) ve TS aynı performans metriklerini amaç fonksiyonu olarak kullandıkları için bu iki algoritma birbiri ile kıyaslandığında ABC algoritmasının daha düşük bir hata değeri ürettiği görülmektedir. ABC algoritmasının kullanılan performans indisinden bağımsız olarak araştırma kabiliyeti buradan görülmektedir. Performans indisleri arasında bir kıyaslama yapılacak olursa zaman ağırlıklı metriklerin oturma zamanlarının daha iyi olduğu, hata terimine maksimum taşma eklenmesiyle maksimum taşmanın azaltılabildiği görülmektedir.

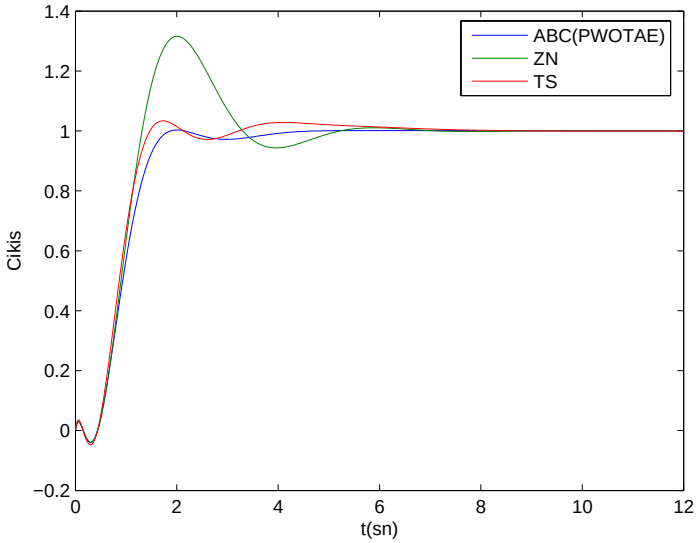
ZN, Kitamori [258], TS [2] ve ABC (PWAE) algoritmalarının birim basamak cevaplarına ait grafikler Şekil 6.7'de verilmektedir. ABC algoritmasının farklı performans indislerine göre koşulmasıyla gerçekleştirilen sistemlerin birim basamak cevaplarına ait grafikler de Şekil 6.8'de verilmektedir.

Sistem 2 için ABC algoritmasının 30 koşmasının en iyi değerlerinin ortalamalarına ait gelişim grafiği Şekil 6.9'da verilmektedir.

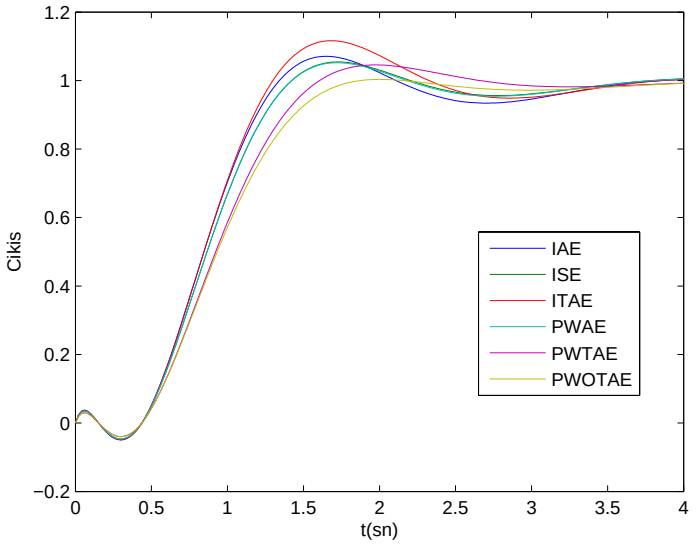
Ayrıca Sistem 1 ve Sistem 2 için ABC algoritmasının 30 koşmasına ait ortalama ve standard sapma değerleri Tablo 6.4'de verilmiştir. Tablodaki veriler incelendiğinde ABC algoritmasının oldukça kararlı olduğu sonucuna varılabilir.

Tablo 6.3. Sistem 2 için ZN, Kitamori, Tabu Araştırma ve ABC ile gerçekleştirilen tasarımların K_d, K_p, K_i, PO (yüzde maksimum aşma), t_r, t_s ve hata değerleri

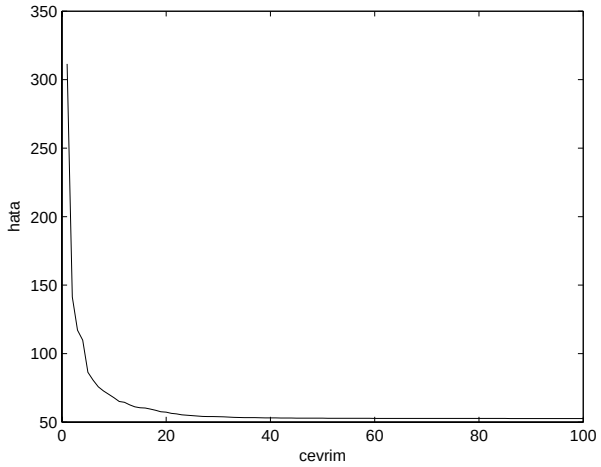
	K_d	K_p	K_i	PO	t_r	t_s	Hata
ZN [256](IAE)	1.151	2.808	1.7122	31.5959	0.76	4.26	68.0811
Kitamori [258](IAE)	1.148	2.212	1.0858	6.2992	0.98	2.36	51.9036
TS [2]	1.465	2.194	1.209	3.3541	0.86	1.34	43.7397
ABC(IAE)	1.5481	2.4441	1.1453	7.7303	0.76	3.06	49.0744
ABC(ITAE)	1.4904	2.5687	1.2141	11.6360	0.74	2.1	34.8458
ABC(ISE)	1.5225	2.4198	1.1325	7.0807	0.76	3.06	49.0630
ABC(PWAE)	1.4385	2.3220	1.1276	5.2063	0.8	1.8	52.6142
ABC (PWTAE)	1.2071	2.2119	1.0630	4.5640	0.96	1.46	34.8892
ABC(PWOTAE)	1.2022	2.0809	0.9919	0.3336	1.06	1.58	37.5402



Şekil 6.7. Sistem 2 için ZN, Kitamori, Tabu Araştırma ve ABC ile gerçekleştirilen tasarımların birim basamak cevabı



Şekil 6.8. Sistem 2 için ABC algoritmasının farklı performans indislerine göre koşulmasıyla gerçekleştirilen tasarımların birim basamak cevabı



Şekil 6.9. Sistem 2 için PWAE performans indisi kullanan ABC algoritmasının 30 koşmasının en iyi değerlerinin ortalamalarına ait gelişim grafiği

Tablo 6.4. Sistem 1 ve Sistem 2 için ABC algoritmasının 30 koşmasına ait ortalama PWAE hata ve standard sapma değerleri

	Ortalama	Standard Sapma
Sistem 1	5.3939	0.0244
Sistem 2	52.6165	0.0140

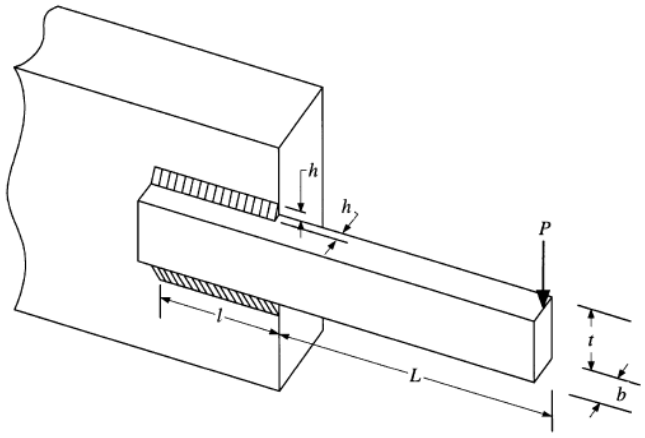
ABC algoritmasıyla tasarlanan PID kontrolör, kalıcı-durum hatası düşük aynı zamanda hızlı ve duyarlı bir sistem oluşturulmasını sağlarken algoritmanın kontrol parametre sayısının az oluşu da tasarıma esneklik kazandırmaktadır.

6.2. Sınırlamalı Mühendislik Tasarım Problemleri

6.2.1. Kaynak Yapılmış Kiriş (Welded Beam) Tasarım Problemi

Şekil 6.10'de gösterilen kaynak yapılmış kiriş tasarımı problemi kesme gerilimi τ, kirişteki bükülme gerilimi σ, çubuktaki burkulma gerilimi P_c, kirişin sehimi δ ve yön sınırlamaları altında kirişin maliyetinin minimize edildiği Şekil 6.10'deki h, l, t and b değişkenlerine karşılık gelen x_1, x_2, x_3 and x_4 değişkeni yani dört tasarım parametreli bir optimizasyon problemidir.

Minimize edilecek kiriş maliyet fonksiyonu ve sınırlamalar aşağıda verilmektedir:



Şekil 6.10. Kaynak yapılmış kiriş tasarım problemi

$$\min_{X} f(X) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2)$$
$$g_1(X) : \tau(x) - \tau_{\max} \leq 0$$
$$g_2(X) : \sigma(x) - \sigma_{\max} \leq 0$$
$$g_3(X) : x_1 - x_4 \leq 0$$
$$g_4(X) : 0.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5.0 \leq 0$$
$$g_5(X) : 0.125 - x_1 \leq 0$$
$$g_6(X) : \delta(x) - \delta_{\max} \leq 0$$
$$g_7(X) : P - P_c(x) \leq 0$$

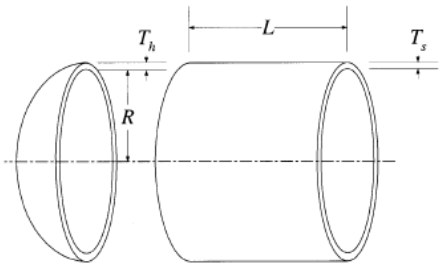
Burada $\tau(X) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x2}{2R} + (\tau'')^2}$, $\tau' = \frac{P}{\sqrt{2}x_1x_2}$, $\tau'' = \frac{MR}{J}$,
$M = P(L + \frac{x_2}{2})$, $R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1+x_3}{2}\right)^2}$, $J = 2\left\{\frac{x_1x_2}{\sqrt{2}}\left[\frac{x_2^2}{12} + \left(\frac{x_1+x_3}{2}\right)^2\right]\right\}$,
$\sigma(X) = \frac{6PL}{x_4x_3^2}$, $\delta(X) = \frac{4PL^3}{Ex_3^3x_4}$, $P_c = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)$ 'dir

Kullanılan sabitler ve değerleri ise P=6000 lb., L=14 in, $\delta_{\max} = 0.25$in, E = 30x10^{6}psi, G = 12x10^{6}psi, $\tau_{\max}$= 13600psi, $\sigma_{\max}$ = 30000psi'dir.

Değişken aralıkları ise $X = (x_1, x_2, x_3, x_4)^{\mathrm{T}}$, $0.1 \leq x_1, x_4 \leq 2.0$, $0.1 \leq x_2, x_3 \leq 10$ şeklinde tanımlıdır.

6.2.2. Basınç Tankı (Pressure Vessel) Tasarım Problemi

İkinci tasarım problemi Şekil 6.11'de gösterilen silindir basınç tankının malzeme ve kaynak maliyetlerinin minimize edilmesidir.



Şekil 6.11. Basınç tankı tasarım problemi

Problem, dört tane tasarım parametresine sahip ve bunlar x_1 (dış kalınlık T_S), x_2 (baş kısmın kalınlığı T_H), x_3 (iç çap R) and x_4 (baş kısım olmaksızın silindir kısmın uzunluğu L) dir. x_1 ve x_2 mevcut çelik plakaların kalınlığı olan 0.0625'in tam sayı katları olmak zorundadır. Çap değişkeni x_3 ve uzunluk değişkeni x_4 sürekli değişkenlerdir.

Basınç tankı tasarım problemine ait sınırlamalar ve amaç fonksiyonu aşağıda verilmektedir:

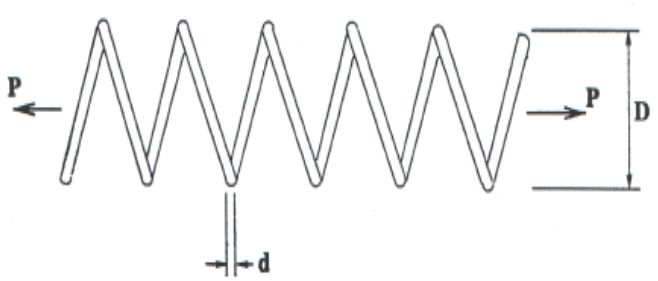
$$\begin{aligned}
&\min_X f(X) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \\
&\quad g_1(X) : -x_1 + 0.0193x_3 \leq 0 \\
&\quad g_2(X) : -x_2 + 0.00954 \leq 0 \\
&\quad g_3(X) : -\pi x_3^2 x_4 - \tfrac{4}{3}\pi x_3^3 + 1296000 \leq 0 \\
&\quad g_4(X) : x_4 - 240 \leq 0
\end{aligned}$$

$X = (x_1, x_2, x_3, x_4)^{\mathrm{T}}$. Tasarım değişkenlerinin aralıkları $1 \leq x_1, x_2 \leq 99$, $10 \leq x_3, x_4 \leq 200$ şeklindedir.

6.2.3. Yay (Tension/Compression Spring) Tasarım Problemi

Yay tasarım problemi Şekil 6.12'de gösterilen yayın ağırlığının minimizasyonu ile ilgilenir. Buna ilave olarak minimum burkulma, kesme gerilimleri, gerilme frekansı, çap boyutu gibi sınırlamaları sağlamak zorundadır.

Tasarım değişkenleri malzemenin çapı, d, ana sarım çapı, D, ve aktif sarımların sayısı N'dir. Amaç fonksiyonu ve sınırlamalar aşağıda formülize edilmiştir:

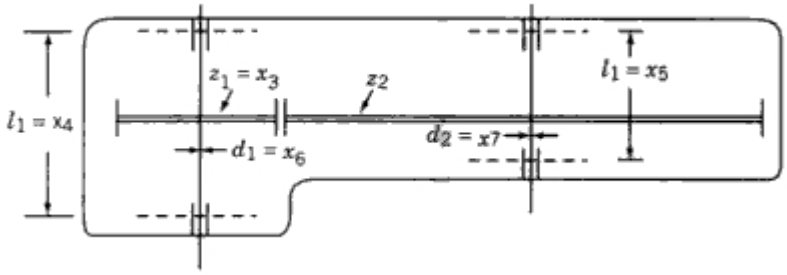


Şekil 6.12. Yay tasarım problemi

$$\min_X f(X) = (N+2)Dd^2$$
$$g_1(X) : 1 - \frac{D^3 N}{71785d^4} \leq 0$$
$$g_2(X) : \frac{4D^2 - dD}{12566(Dd^3 - d^4)} + \frac{1}{5108d^2} - 1 \leq 0$$
$$g_3(X) : 1 - \frac{140.45d}{D^2 N} \leq 0$$
$$g_4(X) : \frac{D+d}{1.5} - 1 \leq 0$$
$$X = (d, D, N)^{\mathrm{T}},\ 0.05 \leq d \leq 2.0,\ 0.25 \leq D \leq 1.3,\ 2.0 \leq N \leq 15.0$$

6.2.4. Hız İndirgeyici (Speed Reducer) Tasarım Problemi

Şekil 6.13'de gösterilen hız indirgeci tasarımının amacı, çark dişlilerinin bükme gerilimi, yüzey gerilimi, şaftların çapraz sehimleri ve şaft gerilimleri sınırlamalarını sağlayacak şekilde indirgeyici ağırlığının minimize edilmesidir.



Şekil 6.13. Hız indirgeyici tasarım problemi

Tasarım parametreleri yüzey genişliği (b), dişli modülü (m), pinyonlardaki dişli sayısı (z), birinci şaftın rulmanları arasındaki uzunluk (l_1), ikinci şaftın rulmanları arasındaki uzunluk (l_2), birinci şaftın çapı (d_1) ve ikinic şaftın çapı (d_2) dır ve bunlar sırasıyla $x_1, x_2, ..., x_7$ ile gösterilmektedir. Dişli sayısı olan üçüncü değişken z tam sayı değerler alırken diğer değişkenler süreklidir. Bu tasarım problemi karma tamsayı

problemi olduğundan bazı algoritmaların kabul edilebilir bölgede çözüm bulmaları oldukça zordur [259].

Tasarım parametreleriyle ilgili amaç fonksiyonu ve sınırlamalar aşağıda tanımlanmaktadır:

$$f(x) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) -$$

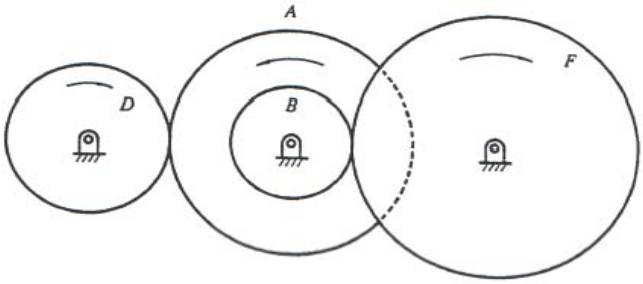
$$1.508x_1(x_6^2 + x_7^2) + 7.4777(x_6^3 + x_7^3)$$

$$\begin{aligned}
g_1(x) &= \frac{27}{x_1x_2^2x_3} - 1 \leq 0 \\
g_2(x) &= \frac{397.5}{x_1x_2^2x_3^2} - 1 \leq 0 \\
g_3(x) &= \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \leq 0 \\
g_4(x) &= \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \leq 0 \\
g_5(x) &= \frac{\left(\left(\frac{745x_4}{x_2x_3}\right)^2 + 16.9\text{x}10^6\right)^{1/2}}{110.0x_6^3} - 1 \leq 0 \\
g_6(x) &= \frac{\left(\left(\frac{745x_4}{x_2x_3}\right)^2 + 157.5\text{x}10^6\right)^{1/2}}{85.0x_7^3} - 1 \leq 0 \\
g_7(x) &= \frac{x_2x_3}{40} - 1 \leq 0 \\
g_8(x) &= \frac{5x_2}{x_1} - 1 \leq 0 \\
g_9(x) &= \frac{x_1}{12x_2} - 1 \leq 0 \\
g_{10}(x) &= \frac{1.5x_6 + 1.9}{x_4} - 1 \leq 0 \\
g_{11}(x) &= \frac{1.1x_7 + 1.9}{x_5} - 1 \leq 0
\end{aligned}$$

Değişkenlerin alt ve üst sınırları ise $2.6 \leq x_1 \leq 3.6, 0.7 \leq x_2 \leq 0.8, 17 \leq x_3 \leq 28, 7.3 \leq x_4 \leq 8.3, 7.8 \leq x_5 \leq 8.3, 2.9 \leq x_6 \leq 3.9, 5.0 \leq x_7 \leq 5.5$ şeklindedir.

6.2.5. Dişli Takımı (Gear Train) Tasarım Problemi

Şekil 6.14'de gösterilen dişli takımı tasarım probleminin amacı dişli takımının oranını minimize etmektedir.



Şekil 6.14. Dişli takımı tasarım problemi

Dişli takımı oranı aşağıdaki gibi tanımlanmaktadır: $gear\ ratio = \frac{n_B n_D}{n_F n_A}$

Tasarım parametreleri n_A, n_B, n_D, n_F; sırasıyla x_1, x_2, x_3, x_4 değişkenleri ile gösterilmekte ve tüm değişkenler [12,60] aralığında tam sayı olmak zorundadır. Amaç fonksiyonu ve sınır değerleri aşağıda verilmektedir:

$$\min_X f(X) = \left(\frac{1}{6.931} - \frac{x_3 x_2}{x_1 x_4}\right)^2$$
$$12 \leq x_i \leq 60, i = 1, \ldots, 4$$

6.3. Çalışmalar ve Testler

Yukarıda sunulan gerçek tasarım problemleri üzerinde SCA [260], PSO versiyonları (PSO, UPSOm) [241,261], ES ($\mu + \lambda - ES$) [259] ve ABC algoritmalarını kıyaslayan bir çalışma yapılmıştır. Algoritmalara özgü kontrol parametrelerinin değerleri Tablo 6.5'de verilmiştir. Tam sayı değer alması gereken değişkenlerin değerleri reel değişkenlerin değerlerinin en yakın tam sayıya kesilmesiyle elde edilmiştir. Her bir tasarım problemi için 30 farklı koşma gerçekleştirilmiş ve bu koşmaların en iyi, ortalama, standard sapma değerleri ve algoritmaların durdurulduğu maksimum değerlendirme sayıları Tablo 6.6'de sunulmuştur. En iyi sonuç, algoritmanın en iyi sonucu bulabilme yeteneği ile ilgili bilgi verirken, ortalama ve standard sapma değerleri de algoritmanın kararlılığı ve gürbüzlüğü konusunda bilgi vermektedir. Maksimum değerlendirme sayısı da yakınsama hızı ile ilgili bir metrik sayılabilir.

Tablo 6.6'deki sonuçlara bakılacak olursa, çözümün kabul edilebilir bölgenin sınırları üzerinde olduğu kaynatılmış kiriş probleminde, en iyi sonuç ($\mu + \lambda$) ES ve ABC algoritmaları tarafından elde edilmiştir. Problem çözümünün kabul edilebilir bölge sınırları üzerinde olması ve dolayısıyla mevcut çözüm komşularının bazılarının kabul edilebilir olması bazılarının kabul edilebilir olmaması problemi zorlaştırmaktadır. Bulunan en iyi sonuçlar açısından ($\mu + \lambda$) ES ve ABC algoritmalarının performansı aynı iken, ortalama sonuçlara bakıldığında ABC algoritmasının daha üstün olduğu görülmektedir. Kaynak yapılmış kiriş probleminde ABC algoritmasının 30 kez koşulmasına ait gelişimlerin ortalaması Şekil 6.15(a)'da verilmektedir. Şekilden görüldüğü gibi ABC algoritması ortalama 800-1000 çevrim sonunda arzu edilen sonuca ulaşmaktadır.

Basınç tankı tasarım problemi için algoritmalar en iyi çözüm açısından değerlendirilirse $(\mu+\lambda)$ ES en başarılı algoritma olarak görülmektedir. ABC ve [261] numaralı referansta tanıtılan PSO ise benzer sonuçlar üretmişlerdir. Ancak ortalama çözüm açısından değerlendirildiğinde ABC'nin en düşük değerleri ürettiği açıktır. Bu problem üzerinde ABC'nin 30 koşmasına ait ilk 500 çevrimi için ortalama gelişim grafiği Şekil 6.15(b)'de verilmektedir. ABC'nin ortalama 100-200 çevrimde en düşük değeri ürettiği görülmektedir.

Yay tasarım probleminde en iyi, ortalama, standard sapma ve değerlendirme sayısı bakımından ABC algoritması en iyi sonuçları üretmiştir. İlk 500 çevrim için ABC'nin gelişim grafiği Şekil 6.15(c)'de verilmektedir. Şekilden de görüldüğü gibi ABC algoritması maksimum çevrim sayısına ulaşmadan optimum çözümü bulabilmektedir.

Hız indirgeyici tasarım problemi için en iyi sonuç SCA algoritması ile bulunmasına rağmen ortalama en iyi sonuç $(\mu + \lambda)$ ES yaklaşımı tarafından elde edilmiştir. [261]'de tanıtılan PSO'nun ve UPSOm'nun [241] sonuçları bu problem için bulunmamaktadır. ABC algoritmasının ilk 500 çevrim için ortalama hata gelişim grafiği Şekil 6.15(d)'de verilmektedir. Ortalama 50-100 çevrim içerisinde ABC algoritması en iyi sonucunu üretmektedir.

Dişli takımı tasarım problemi için UPSOm ve ABC dışındaki algoritmaların sonuçları bulunmamaktadır. UPSOm ve ABC algoritmalarının ürettikleri sonuçlar hemen hemen aynı olmasına karşın UPSOm algoritmasının değerlendirme sayısı ABC algoritmasına göre oldukça fazladır.

Problemler için elde edilen en iyi sonuçlara ait parametre ve sınırlama değerleri Tablo 6.7-6.11'de verilmektedir.

[261]'de sunulan PSO algoritması araştırma uzayında yalnızca kabul edilebilir bölge içerisinde hareket edecek şekilde geliştirilmiştir. Bu nedenle başlangıç popülasyonunun kabul edilebilir bölgede olmasına ihtiyaç duymaktadır. SCA optimizasyon metodu ise kümeleme rutinlerine ihtiyaç duyduğundan ekstra hesaplama maliyetine sahiptir [259]. [259] numaralı referansta sunulan $(\mu + \lambda)$ ES

yaklaşımında popülasyonda farklılığı sağlayacak kabul edebilir bölgede olmayan daha fazla çözüme ihtiyaç duyulduğu belirtilmiştir. ABC algoritması başlangıç çözümlerinin kabul edebilir bölgede olmasına ihtiyaç duymamaktadır. Yine ABC, ekstra karmaşıklık getirecek, hesaplama maliyetini artıran rutinlere sahip değildir. Kaşif arı ve gözcü arı birimiyle kabul edilebilir bölgede olmayan çözümlerinde popülasyona katılmasına izin verir yani keşif yeteneği, yeni çözüm bulma yeteneği gelişmiştir.

Netice olarak, bu bölümde genellikle doğrusal olmayan ve sınırlamalı karakteristikteki mühendislik tasarım problemlerinin çözümü için ABC algoritması kullanılmış ve literatürde bulunan diğer yaklaşımlara ait sonuçlarla benzer veya daha başarılı sonuçlar elde edilmiştir.

Tablo 6.5. Algoritmaların kontrol parametrelerinin değerleri. D: Problem boyutu, CS: Medeniyet sayısı, TS: Zaman aralığı, σ: Normal dağılım varyansı, SS: Sürü boyutu, MGN: Maksimum jenerasyon sayısı, ω: inertia, c_1: Bilişsel bileşen, c_2: Sosyal bileşen, S_r: Seleksiyon oranı, LR: Öğrenme oranı, MSS: Mutasyon adım büyüklüğü, χ: Konstriksiyon faktörü, NR: Komşuluk çapı, UF: Unification Faktörü, MtR: Mutation Oranı, CSabc: Koloni büyüklüğü, MCN: Maksimum çevrim sayısı, SPP: Kaşif arı üretme peryodu, MR: Değişim oranı

SCA		PSO		$(\mu+\lambda)$-ES		UPSOm		ABC	
CS	10*D	SS	30	μ	15	χ	0.729	CSabc	30
TS	1000	MGN	1000	λ	100	c_1	2.05	MCN	1000
σ	0.25$\sim$1	ω	0.8	S_r	0.97	c_2	2.05	MR	0.9
		c_1	0.5	MGN	300	NR	1	SPP	400
		c_2	0.5	LR	$\tau=\left(\sqrt{2\sqrt{\mathrm{n}}}\right)^{-1}$ $\tau'=\left(\sqrt{2n}\right)^{-1}$	UF	0.1	Limit	CSabc*D*5
				MSS	$\sigma_{\mathrm{i}}(0)=0.4\left({\Delta \mathrm{xi}}/{\sqrt{\mathrm{n}}}\right)$	MGN	5000		
						SS	20		
						MtR	0.01		
						w1	100		
						w2	100		

Tablo 6.6. Algoritmaların mühendislik problemleri için ürettikleri istatistiki sonuçlar. Koyu yazılı değerler en iyi sonucu göstermektedir.

Problem	İstatistik	SCA	PSO	$(\mu+\lambda)$-ES	UPSOm	ABC
Kaynatılmış Kiriş	En iyi	2.385435	2.380957	**1.724852**	1.92199	**1.724852**
	Ort.	3.255137	2.381932	1.777692	2.83721	**1.741913**
	Std.Sapma	9.6E-1	5.2E-3	8.8E-2	0.682980	**3.1E-02**
	Değ. Sayısı	33000	30000	30000	100000	30000
Basınç Tankı	En iyi	6171.00	6059.7143	**6059.701610**	6544.27	6059.714736
	Ort.	6335.05	6289.92881	6379.938037	9032.55	**6245.308144**
	Std.Sapma	NA	3.1E+2	2.1E+2	995.573	**2.05.E+02**
	Değ. Sayısı	20000	30000	30000	100000	30000
Yay	En iyi	0.012669	0.012665	0.012689	0.0131200	**0.009872**
	Ort.	0.012923	0.012702	0.013165	0.0229478	**0.009872**
	Std.Sapma	5.9E-4	4.1E-5	3.9E-4	0.00720571	**2.68E-11**
	Değ. Sayısı	25167	15000	30000	100000	4000
Hız İndirgeyici	En iyi	**2994.744241**	NA	2996.348094	NA	2997.058412
	Ort.	3001.758264	NA	**2996.348094**	NA	2997.058412
	Std.Sapma	4.0E+0	NA	**0**	NA	0
	Değ. Sayısı	54456	NA	30000	NA	30000
Dişli takımı	En iyi	NA	NA	NA	**2.70085E-12**	**2.700857E-12**
	Ort.	NA	NA	NA	**3.80562 E-8**	**3.641339E-10**
	Std.Sapma	NA	NA	NA	**1.09631 E-7**	**5.525811E-10**
	Değ. Sayısı	NA	NA	NA	100000	60

Tablo 6.7. Kaynak yapılmış kiriş problemi için en iyi sonuca ait parametre ve sınırlama değerleri

	SCA	PSO	$(\mu + \lambda)$ ES	ABC
x1	0.244438	0.244369	0.205730	0.205730
x2	6.237967	6.217520	3.470489	3.470489
x3	8.288576	8.291471	9.036624	9.036624
x4	0.244566	0.244369	0.205729	0.205730
g1	-5760.110471	-5741.176933	0.000000	0.000000
g2	-3.245428	0.000001	0.000002	-0.000002
g3	-0.000128	0.000000	0.000000	0.000000
g4	-3.020055	-3.022955	-3.432984	-3.432984
g5	-0.119438	-0.119369	-0.080730	-0.080730
g6	-0.234237	-0.234241	-0.235540	-0.235540
g7	-13.079305	-0.000309	-0.000001	0.000000
f(x)	2.38119	2.380956	1.724852	1.724852

Tablo 6.8. Basınç tankı problemi için en iyi sonuca ait parametre ve sınırlama değerleri

	SCA	PSO	$(\mu + \lambda)$ ES	ABC
x1	0.8125	0.8125	0.8125	0.8125
x2	0.4375	0.4375	0.4375	0.4375
x3	41.9768	42.098446	42.098446	42.098446
x4	182.2845	176.636052	176.636596	176.636596
g1	-0.0023	0.000000	0.000000	0.000000
g2	-0.0370	-0.035881	0.035880	-0.035881
g3	-23420.5966	0.000000	0.000000	-0.000226
g4	-57.7155	-63.363948	-63.363404	-63.363404
f(x)	6171.0	6059.701610	6059.7143	6059.714339

Tablo 6.9. Yay problemi için en iyi sonuca ait parametre ve sınırlama değerleri

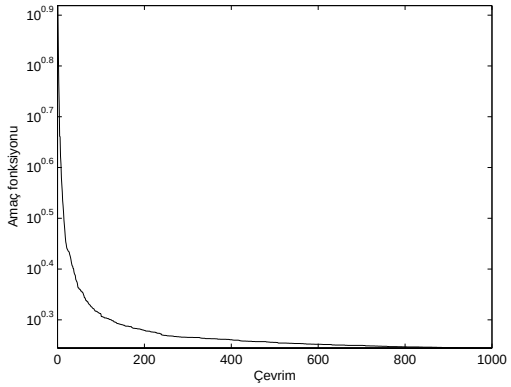
	SCA	PSO	$(\mu + \lambda)$ ES	ABC
x1	0.0521602	0.051690	0.052836	0.05
x2	0.368159	0.356750	0.384942	0.3744317
x3	10.648442	11.287126	9.807729	8.546720
g1	0.000000	0.000000	-0.000001	0.000000
g2	0.000000	0.000000	0.000000	0.000000
g3	-4.075805	-4.053827	-4.106146	-4.860699
g4	-0.719787	-0.727706	-0.708148	-0.717046
f(x)	0.012669	0.012665	0.012689	0.009872

Tablo 6.10. Hız indirgeyici problemi için en iyi sonuca ait parametre ve sınırlama değerleri

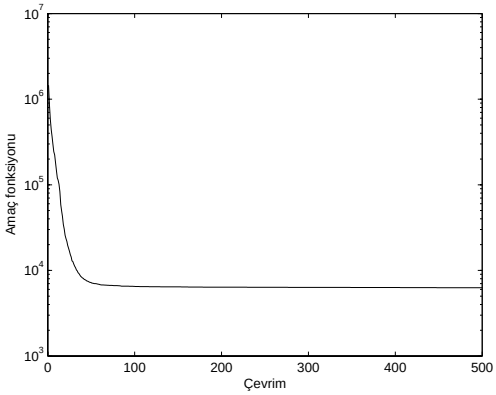
	SCA	$(\mu + \lambda)$ ES	ABC
x1	3.500000	3.499999	3.499999
x2	0.700000	0.699999	0.7
x3	17	17	17
x4	7.327602	7.300000	7.3
x5	7.715321	7.800000	7.8
x6	3.350267	3.350215	3.350215
x7	5.286655	5.286683	5.287800
g1	-0.073915	-0.073915	-0.073915
g2	-0.197999	-0.197998	-0.197999
g3	-0.493501	-0.499172	-0.499172
g4	-0.904644	-0.901472	-0.901555
g5	0.000000	0.000000	0.000000
g6	0.000633	0.000000	0.000000
g7	-0.7025	-0.702500	-0.7025
g8	0.000000	0.000000	0.000000
g9	-0.583333	-0.583333	-0.583333
g10	-0.054889	-0.051325	-0.051326
g11	0.000000	-0.010852	-0.010695
f(x)	2994.744241	2996.348094	2997.058412

Tablo 6.11. Dişli takımı problemi için en iyi sonuca ait parametre ve sınırlama değerleri

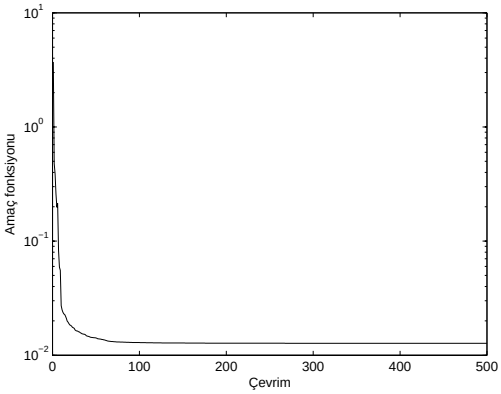
x1	x2	x3	x4	f(x)
49	16	19	43	0



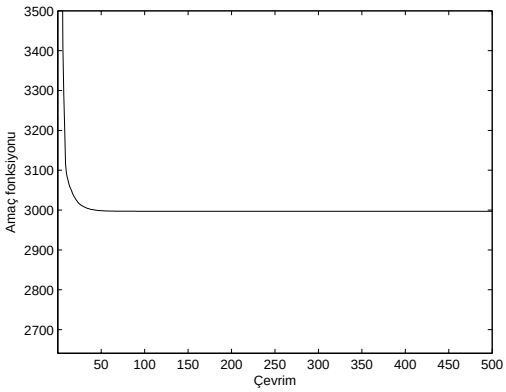
(a) Kaynak yapılmış kiriş problemi



(b) Basınç Tankı problemi



(c) Yay problemi



(d) Hız indirgeyici problemi

Şekil 6.15. ABC algoritmasının 30 koşmasında en iyi sonuçların ortalamalarının gelişimi

7. BÖLÜM

SONUÇLAR

Optimizasyon, bir problemin olası çözümleri arasından en iyisini bulma işlemidir. Bazı problemler çözülürken eldeki alternatif çözümler arasından deneme yanılma suretiyle en iyisini bulmak mümkün iken gerçek hayat problemlerinde bazen zaman, maliyet ve gerçeklenebilirlik kısıtlarından dolayı deneme yanılma sürecini kullanmak imkansızdır. Bu nedenle bu tür problemlerin çözümünde bilgisayar simülasyonu kullanmak gerekir. Simülasyon esnasında ise çeşitli çözüm teknikleri kullanılabilir. En iyi çözümün bulunmasına yönelik geliştirilen bilgisayar programlarının performansları, klasik teknikler veya sezgisel yaklaşımlarla çözüm üretilmesine göre farklılıklar gösterebilir. Kullanılabilecek alternatif tekniklerin probleme özgü performans sergilemesinden dolayı hangi klasik tekniğin veya hangi sezgiselin kullanılması gerektiği ayrı bir problemdir. Bu nedenle belirli problem türlerinde çok iyi performans sergileyen teknikler yerine, genel olarak çoğu problem türünde iyi performans sergileyen algoritmaların geliştirilmesi oldukça önemlidir. Algoritmaların bu karakteristiğe sahip olup olmadığının ortaya konması içinde performanslarının analiz edilmesi ve literatürdeki bilinen algoritmalarla kıyaslanarak davranışının incelenmesi gerekmektedir. Algoritmaların performans analizi yapılırken çok çeşitli test problem türleri üzerinde testler yapılır ve elde edilen sonuçlar çeşitli metriklerle istatistiksel analizlere tabi tutulurlar. Algoritmaların sınırlamasız uzayda davranışları, sınırlamalı uzaydaki davranışları, probleme özgü koşullara karşı dayanıklılığı ve algoritmaya özgü kontrol parametrelerinin performans üzerindeki etkisi incelenmesi gereken hususlardan bazılarıdır.

Literatürdeki klasik tekniklerin deterministik, hızlı ve bölgesel tam sonuç verme gibi avantajları olmasına rağmen, optimum çözümü bulması bazı problemlerde

oldukça zaman almakta veya gerçek dünya problemlerine uygulanmaları mümkün olmamaktadır. Bu nedenle genel amaçlı, kabul edilebilir süre içerisinde makul çözümler üreten, esnek, küresel araştırma yapabilen sezgisel algoritmalara ihtiyaç duyulmuştur. Bu amaç doğrultusunda geliştirilen komşuluk tabanlı, popülasyon tabanlı ve sürü tabanlı sezgisel algoritmalar literatürde mevcuttur. Sürü tabanlı algoritmalarda, bireyler global düzeyde herhangi bir ilişkileri olmaksızın bölgesel kurallar kullanarak kendi kendilerine organize olabilmekte ve toplu olarak, yani sürü olarak, bir işi gerçekleştirebilmektedirler. Karıncaların, kuş ve balık sürülerinin davranışlarını temel alan sürü zekası temelli algoritmalar oldukça popülerdir. Literatürde arıların farklı davranışlarını temel alan çeşitli sürü zekası tabanlı yaklaşımlar da bulunmaktadır. Arılardaki sürü zekasını modelleyen algoritmalar çok farklı alanlardaki çeşitli problemlerin çözümünde kullanılmıştır. Modellerin çoğu kombinasyonel problemler için geliştirilmiş ve uygulanmıştır. Nümerik optimizasyon problemlerinin çözümü amacıyla geliştirilen arıların davranışına dayalı algoritmalar Yapay Arı Algoritması (VBA), Arı Algoritması (BA) ve Yapay Arı Kolonisi (ABC) algoritmasıdır.

Yapay arı kolonisi algoritması, Karaboga [22] tarafından 2005 yılında literatüre kazandırılmış, arıların yiyecek arama davranışını modelleyen oldukça yeni ve popüler sürü tabanlı bir algoritmadır. Bu tez çalışmasında, literatüre yeni sunulan ABC algoritmasının detaylı performans analizi yapılmıştır.

Bölüm 1'de optimizasyon konusu ile ilgili genel tanımlamalar yapılarak gelişime dayalı algoritmaların bileşenleri tanıtılmıştır. Ayrıca bir sezgisel algoritmanın performans analizi yapılırken izlenecek metotlar ve dikkat edilecek hususlar belirtilmiştir. Bölüm 2'de sürü zekası kavramı tanıtılarak, arılardaki sürü zekasına dayalı çalışmalar verilmiştir. Bölüm 3'de arıların yiyecek arama davranışını temel alan ABC algoritması detaylı olarak anlatılmış ve özellikleri verilmiştir.

Bölüm 4'de ABC algoritmasının sınırlamasız problemler üzerinde test edilmesi esnasında kontrol parametrelerine olan duyarlılığı incelenmiştir. ABC algoritması nispeten düşük popülasyon büyüklüklerinde de iyi sonuçlar üretmekte ve popülasyon büyüklüğüne olan duyarlılığı PSO ve DE algoritmalarına göre daha azdır.

Belirli bir değerden sonra popülasyonun artması sadece hesaplama maliyetini artırmakta, performans üzerinde olumlu bir etki yapmamaktadır. “Limit" kontrol parametresinin düşük değerleri ihtiyaçdan fazla kaşif arı üretilmesine sebep olurken, büyük limit değerleri ise gereğinden az kaşif arı çıkmasına ve algoritmanın global araştırma yeteneğinin zayıflamasına sebep olmaktadır. ABC algoritmasının performansının başlangıç aralığına olan bağlılığı ile ilgili yapılan çalışmadan da, çok dar, kısıtlı bir bölgede başaltılmış olsa dahi algoritmanın keşif yeteneğinin etkili olmasından dolayı tüm uzayı araştırabildiği görülmüştür. Sınırlamasız problemler üzerinde, literatürde oldukça popüler olan GA, DE, PSO, CES, FES, CMA-ES, ESLAT, SOM-ES ve NG-ES algoritmaları ile ABC'yi karşılaştıran çalışmalar yapılmıştır. Sonuçlardan ABC algoritmasının basit sınırlamasız fonksiyonlar üzerindeki performansının çalışmada kıyaslanan diğer algoritmalara ya benzer yada daha iyi olduğu gözlemlenmiştir. Daha az kontrol parametresine sahip basit ABC algoritmasının çok modlu ve çok parametreli optimizasyon problemlerinin çözümünde oldukça iyi sonuçlar ürettiği söylenebilir.

Bu bölümde yine doğrusal programlamanın bir türü olan ve değişkenlerin tam sayı değerler alabildiği tam sayı programlama problemlerinin çözümü için ABC algoritmasının yeni bir versiyosu önerilmiş ve performansı BB tekniği ve PSO türevleri ile kıyaslanmıştır. ABC algoritması, bu çalışmada dikkate alınan tam sayı programlama problemeri üzerinde PSO türevleri ve BB tekniği ile kıyaslanabilir veya daha iyi performans sergilemiştir. ABC algoritması kesin teknikler kadar başarılı sonuçlar üretirken hesaplama maliyeti de oldukça düşüktür.

ABC algoritmasının performans analizi, farklı basit fonksiyonların ötelenmesi, döndürülmesi ve toplanamsı ile elde edilen, oldukça zor problemler olan karma problemler üzerinde de yapılmıştır. Bu tez çalışmasında önerilen değişim oranı ve ölçekleme faktörü parametrelerinin algoritmanın performansı üzerindeki etkisi karma problemler açısından incelenmiştir. Algoritmanın zaman göre karmaşıklık analizi yapılmıştır. Yine, ABC'nin basit yapısına rağmen karmaşık yapıdaki algoritmalar kadar bu problemler üzerinde başarılı performans sergilediği görülmüştür.

Bölüm 5'de ABC algoritmasının, araştırma uzayının bazı eşitlik ve eşitsizliklerle sınırlandığı ve optimumun kabul edilebilir bölge içinde olması gerekliliği taşıyan sınırlamalı test problemlerini çözmek amacıyla yeni bir versiyonu önerilmiştir. Bu tür problemlere bir algoritmanın uygulanabilmesi için sınırlamaların dikkate alınması gerektiğinden sınırlamasız problemleri çözmek amacıyla geliştirilmiş algoritmalara sınırlama ele alış teknikleri entegre edilir. ABC algoritmasının görevli ve gözcü arı fazlarında kullanılan aç gözlü seleksiyon yerine, kullanım kolaylığı ve ek hesaplama maliyeti olmayan Deb'in kurallarının eklenmesiyle ABC ile sınırlamalı problemler çözülebilir hale getirilmiştir. Yeni algoritma ile elde edilen sonuçlar literatürde mevcut başka algoritmalarla bulunmuş sonuçlarla karşılaştırılmış ve sınırlamalı problemler üzerinde kontrol parametreleri için tavsiye edilebilecek değerler belirlenmiştir.

Bölüm 6'da bir gerçek dünya problemi olan, endüstriyel süreçlerin denetiminde yaygın olarak kullanılan PID denetleyicilerin tasarımı için ABC algoritması kullanılmış ve sınırlamasız türde olan bu mühendislik tasarım problemi için ürettiği sonuçlar literatürde yer alan klasik ve sezgisel algoritmaların sonuçları ile karşılaştırılmıştır. ABC algoritmasıyla tasarlanan PID kontrolör, kalıcı-durum hatası düşük aynı zamanda hızlı ve duyarlı bir sistem oluşturulmasını sağlamıştır. Bu bölümde, ayrıca sınırlamalı problemler türünde olan ve genellikle doğrusal olmayan karakteristiğe sahip literatürde popüler olan bazı makine mühendisliği tasarım problemlerinin çözümü için ABC algoritması kullanılmış ve oldukça başarılı sonuçlar elde edilmiştir.

7.1. Gelecekte Yapılacak Çalışmalar

ABC algoritmasının literatürdeki zor test problemleri üzerinde göstermiş olduğu başarım, dünyadaki araştırmacıların dikkatini çekmeye başlamış ve farklı alanlarda çok sayıda değişik probleme uygulanmıştır. Bazı uygulamaların doğası gereği çözülmeye çalışılan problemler kombinasyonel tipte yada çok amaçlı formda olabilmektedir. ABC algoritmasının genel yapısında değişiklik yapmadan mekanizmaların kombinasyonel tipteki problemlere uygulanabilecek forma getirilmesi amaçlanmaktadır. Ayrıca, çok amaçlı problemlerin çözümüne

uyarlanıp performansının analiz edilmesi düşünülmektedir. Yine, gözcü ve kaşif arı fazlarında yapılacak değişimlerle ABC'nin performansının geliştirilmesi hedeflenmektedir.

KAYNAKLAR

1. Stefani, R., et al., Design of Feedback Control System, Saunders College Publishing, 1994.

2. Karaboga, D., Kalinli, A., Tuning pid controller parameters using tabu search algorithm, IEEE International Conference on Systems, Man and Cybernetics, 134–136, 1996.

3. Hedar, A., Fukushima, M., Evolution strategies learned with automatic termination criteria, Proceedings of SCIS–ISIS 2006, Tokyo, Japan, 2006.

4. Yao, X., Liu, Y., Fast evolution strategies, Control and Cybernetics, 26(3), 467–496, 1997.

5. Milano, M., et al., Self–organizing nets for optimization, IEEE Transactions on Neural Networks,, 15(3), 758–765, 2004.

6. Liu, J., et al., Neural Information Processing, LNCS, volume 4233/2006, chapter Quantum-Behaved Particle Swarm Optimization for Integer Programming, 1042–1050, Springer Berlin / Heidelberg, 2006.

7. Laskari, E.C., et al., Particle swarm optimization for integer programming, CEC '02: Proceedings of the Evolutionary Computation on 2002. CEC '02. Proceedings of the 2002 Congress, 1582–1587, IEEE Computer Society, Washington, DC, USA, 2002, ISBN 0-7803-7282-4.

8. Jian, M.C., Introducing recombination with dynamic linkage discovery to particle swarm optimization, Technical Report NCL-TR-2006006, Natural Computing Laboratory (NCLab), Department of Computer Science, National Chiao Tung University, 2006.

9. Liang, J., Suganthan, P., Dynamic multi-swarm particle swarm optimizer with local search, Evolutionary Computation, 2005. The 2005 IEEE Congress on, volume 1, 522–528, Sept. 2005.

10. Ballester, P., et al., Real-parameter optimization performance study on the cec-2005 benchmark with spc-pnx, volume 2, 498–505, Sept. 2005.

11. Ronkkonen, J., et al., Real-parameter optimization with differential evolution, Evolutionary Computation, 2005. The 2005 IEEE Congress on, volume 1, 506–513, Sept. 2005.

12. Qin, A., Suganthan, P., Self-adaptive differential evolution algorithm for numerical optimization, volume 2, 1785–1791, Sept. 2005.

13. Auger, A., Hansen, N., Performance evaluation of an advanced local search evolutionary algorithm, Evolutionary Computation, 2005. The 2005 IEEE Congress on, volume 2, 1777–1784, Sept. 2005.

14. Mezura-Montes, E., Coello Coello, C., A Simple Multimembered Evolution Strategy to Solve Constrained Optimization Problems, Technical Report EVOCINV-04-2003, Evolutionary Computation Group at CINVESTAV, Sección de Computación, Departamento de Ingeniería Eléctrica, CINVESTAV-IPN, México D.F., México, 2003, available in the Constraint Handling Techniques in Evolutionary Algorithms Repository at http://www.cs.cinvestav.mx/~constraint/.

15. Vesterstrom, J., Riget, J., Particle swarms extensions for improved local, multimodal and dynamic search in numerical optimization, Master's thesis, Dept. Computer Science, Univ Aarhus, Aarhus C, Denmark, May 2002.

16. Kirkpatrick, S., et al., Optimization by simulated annealing, Science, 220(4598), 671–680, 1983.

17. Metropolis, N., et al., Equations of state calculations by fast computing machines, Journal of Chemical Physics, 21(6), 1087–1092, 1953.

18. Glover, F., Tabu search - part i, ORSA Journal on Computing, 1(3), 190–206, 1989.

19. Glover, F., Tabu search - part ii, ORSA Journal on Computing, 2(1), 4–32, 1990.

20. Holland, J.H., Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, MI, 1975.

21. Dorigo, M., et al., Positive feedback as a search strategy, Technical Report 91-016, Politecnico di Milano, Italy, 1991.

22. Karaboga, D., An idea based on honey bee swarm for numerical optimization, Technical Report TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

23. Arora, J., S., Introduction to Optimum Design, McGraw Hill, 1989.

24. Arora, J., S., Guide to structural optimization, ASCE Publications, 1997.

25. Dumitrescu, I., Stützle, T., Combinations of local search and exact algorithms, Applications of Evolutionary Computing, LNCS, volume 2611/2003, 57–68, 2003.

26. Eiben, A., Smith, J., Introduction to Evolutionary Computing, Springer, 2003.

27. Zhang, Q.L., Metrics for meta-heuristic algorithm evaluation, Machine Learning and Cybernetics, 2003 International Conference on, volume 2, 1241–1244 Vol.2, Nov. 2003.

28. Barr, R.S., et al., Designing and reporting on computational experiments with heuristic methods, Journal of Heuristics, 1(1), 9–32, 1995.

29. Alba, E., Luque, G., Parallel Metaheuristics, chapter Measuring the Performance of Parallel Metaheuristics, 43–62, John Wiley Sons, Inc., 2005.

30. Bonabeau, E., et al., Swarm intelligence: from natural to artificial systems, Oxford University Press, Inc., New York, NY, USA, 1999, ISBN 0195131592, URL `http://portal.acm.org/citation.cfm?id=328320`.

31. Calabi, P., Advances in Myrmecology, chapter Behavioral flexibility in Hymenoptera: a re-examination of the concept of caste, 237–258, Leiden:Brill Press, 1988.

32. Robinson, G.E., Regulation of division of labor in insect societies, Annual Review of Entomology, 37, 637–665, 1992.

33. Bonabeau, E., et al., Adaptive task allocation inspired by a model of division of labor in social insects, Biocomputing and emergent computation: Proceedings of BCEC97, 36–45, World Scientific Press, 1997, ISBN 981-02-3262-4.

34. Waibel, M., et al., Division of labour and colony efficiency in social insects: effects of interactions between genetic architecture, colony kin structure and rate of perturbations, Proceedings of the Royal Society B, 273, 1815–23, 2006.

35. Millonas, M., Artificial Life III, chapter Swarms, phase transitions, and collective intelligence., Reading, MA:Addison-Wesley, 1994.

36. Grosan, C., Abraham, A., Stigmergic Optimization, Studies in Computational Intelligence, volume 31, chapter Stigmergic Optimization: Inspiration, Technologies and Perspectives, 1–24, Springer-Verlag Berlin Heidelberg, 2006.

37. Kennedy, J., Eberhart, R., Particle swarm optimization, IEEE Int. Conf. on Neural Networks, 1942–1948, Piscataway, NJ, 1995.

38. Eberhart, R.C., et al., Swarm Intelligence, The Morgan Kaufmann Series in Artificial Intelligence, Morgan Kaufmann, San Francisco, CA, USA, 2001, ISBN 978-1-55860-595-4, URL http://www.elsevierdirect.com/product.jsp?isbn=9781558605954.

39. Sierra, M. R., C.A.C., Multi-objective particle swarm optimizers: A survey of the state-of-the-art, International Journal of Computational Intelligence Research, 2(3), 287–308, 2006.

40. Blum, C., Ant colony optimization: Introduction and recent trends, Physics of Life, 2, 353–373, 2005.

41. Dorigo, M., Blum, C., Ant colony optimization theory: A survey, Theoretical Computer Science, 344, 243–278, 2005.

42. Beekman, M., et al., What makes a honeybee scout?, Behavioral Ecology and Sociobiology, 61, 985–995, 2007, URL http://dx.doi.org/10.1007/s00265-006-0331-9.

43. Dornhaus, A., et al., Task selection in honeybees - experiments using multi-agent simulation, In Proc of GWAL'98, September 18-19 1998.

44. Von Frisch, K., The Dancing Bees: An Account of the Life and Senses of Honey Bee, Harcourt, Brace, 1953.

45. Von Frisch, K., Lindauer, M., The "language" and orientation of the honey bee, Annu. Rev. Entomol, 1, 45– 58, 1956.

46. Seeley, T., Honeybee ecology: A study of adaptation in social life, Princeton University Press, Princeton, 1985.

47. Seeley, T., Visscer, P., Group decision making in nest-site selection by honey bees, Apidologie, 35, 101–116, 2006.

48. Menzel, R., et al., Spatial memory, navigation and dance behaviour in apis mellifera, J Comp Physiol A, 192, 889–903, 2006.

49. Jung, S.H., Queen-bee evolution for genetic algorithms, Electronics Letters, 39(6), 575– 576, 2003.

50. Qin, L., et al., A queen-bee evolution based on genetic algorithm for economic power dispatch, 39th International Universities Power Engineering Conference, 2004. UPEC 2004., volume 1, 453– 456, 2004.

51. Azeem, M., Saad, A., Modified queen bee evolution based genetic algorithm for tuning of scaling factors of fuzzy knowledge base controller, Proceedings of the IEEE INDICON 2004, First India Annual Conference, 299– 303, 2004.

52. Azeem, M., A novel parent selection operator in ga for tuning of scaling factors of fkbc, IEEE International Conference on Fuzzy Systems, 1742– 1747, 2006.

53. Karci, A., Imitation of bee reproduction as a crossover operator in genetic algorithms, PRICAI 2004: Trends in Artificial Intelligence, LNCS, volume 3157/2004, 1015–1016, September 2004.

54. Xu, C., et al., A bee swarm genetic algorithm for the optimization of dna encoding, Innovative Computing Information and Control, 2008. ICICIC '08. 3rd International Conference on, 35–35, June 2008.

55. Lu, X., Zhou, Y., A genetic algorithm based on multi-bee population evolutionary for numerical optimization, Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on, 1294–1298, June 2008.

56. Xiongm, Y., et al., Telecommunications Modeling, Policy, and Technology, Operations Research/Computer Science Interfaces, volume 44, chapter The Label-Constrained Minimum Spanning Tree Problem, 39–58, Springer, 2008.

57. Sato, T., Hagiwara, M., Bee system: finding solution by a concentrated search, Systems, Man, and Cybernetics, IEEE International Conference on Computational Cybernetics and Simulation, volume 4, 3954– 3959, Orlando, FL, USA, 1997.

58. Walker, R., Emulating the honeybee information sharing model, International Conference on Integration of Knowledge Intensive Multi-Agent Systems, 497– 504, 2003.

59. Gordon, N., et al., Discrete bee dance algorithm for pattern formation on a grid, IEEE/WIC International Conference on Intelligent Agent Technology, IAT 2003, 545– 549, 2003.

60. Wedde, H.F., et al., Beehive: An efficient fault-tolerant routing algorithm inspired by honey bee behavior, Ant Colony, Optimization and Swarm Intelligence: 4th International Workshop, ANTS 2004, Brussels, Belgium, September 5-8, 2004. Proceedings, LNCS, volume 3172 / 2004, 83– 94, 2004.

61. Wedde, H., Farooq, M., Beehive: Routing algorithms inspired by honey bee behavior, Kunstliche Intelligenz. Schwerpunkt: Swarm Intelligence, 18–24, 2005.

62. Wedde, H., Farooq, M., Computer and Information Science, chapter BeeHive: New Ideas for Developing Routing Algorithms Inspired by Honey Bee Behavior, 321– 339, Chapman & Hall-CRC, 2005.

63. Wedde, H., Farooq, M., A comprehensive review of nature inspired routing algorithms for fixed telecommunication networks, Journal of Systems Architecture, 52, 461– 484, 2006.

64. Wedde, H., et al., Beehiveguard: A step towards secure nature inspired routing algorithms, Applications of Evolutionary Computing, LNCS, volume 3907/2006, 243– 254, 2006.

65. Wedde, H., et al., Beehiveais: A simple, efficient, scalable and secure routing framework inspired by ais, Parallel Problem Solving from Nature - PPSN IX, LNCS, volume 4193/2006, 623–632, 2006.

66. Wang, X., et al., A beehive algorithm based qos unicast routing scheme with abc supported, Advanced Parallel Processing Technologies, LNCS, volume 4847, 450–459, 2007.

67. Wedde, H., et al., A novel class of multi-agent algorithms for highly dynamic transport planning inspired by honey bee behavior, IEEE Conference on Emerging Technologies & Factory Automation, 2007. ETFA., 1157 – 1164, 2007.

68. Wedde, H., et al., Highly dynamic and adaptive traffic congestion avoidance in real-time inspired by honey bee behavior, Mobilität und Echtzeit, Informatik aktuell, 21– 31, 2008.

69. Navrat, P., Bee hive metaphor for web search, International Conference on Computer Systems and Technologies - CompSysTech' 06, 2006.

70. Navrat, P., Kovacik, M., Web search engine as a bee hive, WI '06: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence, 694–701, IEEE Computer Society, Washington, DC, USA, 2006, ISBN 0-7695-2747-7.

71. Navrat, P., et al., Exploring social behaviour of honey bees searching on the web, WI-IATW '07: Proceedings of the 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Workshops, 21–25, IEEE Computer Society, Washington, DC, USA, 2007, ISBN 0-7695-3028-1.

72. Olague, G., Puente, C., The honeybee search algorithm for three-dimensional reconstruction, Applications of Evolutionary Computing, LNCS, volume 3907/2006, 427– 437, 2006.

73. Nakrani, S., Tovey, C., On honey bees and dynamic server allocation in internet hosting centers, Adaptive Behavior - Animals, Animats, Software Agents, Robots, Adaptive Systems, 12(3-4), 223–240, 2004, ISSN 1059-7123.

74. Nakrani, S., Tovey, C., Honey bee waggle dance protocol and autonomic server orchestration in internet hosting centers, Nature Inspired Approaches to Network and Telecommunication in 8th International Conference on Parallel Problem Solving from Nature, 2004.

75. Nakrani, S., Tovey, C., From honeybees to internet servers: biomimicry for distributed management of internet hosting centers, Bioinspiration and Biomimetics, 2, 182– 197, 2007.

76. Gupta, A., Koul, N., Swan: A swarm intelligence based framework for network management of ip networks, Conference on Computational Intelligence and Multimedia Applications, 2007. International Conference on, volume 1, 114–118, Dec. 2007.

77. Sadik, S., et al., Using honey bee teamwork strategy in software agents, Computer Supported Cooperative Work in Design, 2006. CSCWD '06. 10th International Conference on, 1–6, May 2006.

78. Sadik, S., et al., Honey bee teamwork architecture in multi-agent systems, Computer Supported Cooperative Work in Design III, Lecture Notes in Computer Science, volume 4402/2007, 428–437, Springer Berlin / Heidelberg, 2007.

79. Yonezawa, Y., Kikuchi, T., Ecological algorithm for optimal ordering used by collective honey bee behavior, Micro Machine and Human Science, 1996., Proceedings of the Seventh International Symposium, 249–256, Oct 1996.

80. Passino, K., Systems biology of group decision making, Control and Automation, 2006. MED '06. 14th Mediterranean Conference on, 1–1, June 2006.

81. Gutierrez, R.L.Z., Huhns, M., Robust Intelligent Systems, chapter Multiagent-Based Fault Tolerance Management for Robustness, 23–41, Springer London, 2008.

82. Abbass, H.A., Marriage in honey bees optimisation: A haplometrosis polygynous swarming approach, The Congress on Evolutionary Computation, CEC2001, volume 1, 207 – 214, Seoul, Korea, May 2001.

83. Abbass, H.A., A single queen single worker honey bees approach to 3-sat, The Genetic and Evolutionary Computation Conference, GECCO2001, San Francisco, USA, July 7-11 2001.

84. Abbass, H.A., A monogenous mbo approach to satisfiability, International Conference on Computational Intelligence for Modelling, Control and Automation CIMCA'2001, July 9-11 2001.

85. Teo, J., Abbass, H.A., An annealing approach to the mating-flight trajectories in the marriage in honey bees optimization algorithm, Technical report, 2001.

86. Abbass, H.A., Teo, J., A true annealing approach to the marriage in honey-bees optimization algorithm, International Journal of Computational Intelligence and Applications, 3, 199–211, 2003.

87. Benatchba, K., et al., Using bees to solve a data-mining problem expressed as a max-sat one, Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach, LNCS, volume 3562/2005, 212–220, June 2005.

88. Koudil, M., et al., Using artificial bees to solve partitioning and scheduling problems in codesign, Applied Mathematics and Computation, 186(2), 1710–1722, March 2007.

89. Curkovic, P., Jerbic, B., Honey-bees optımization algorithm applied to path planning problem, International Journal of Simulation Modelling, 6(3), 154–165, 2007.

90. Chang, H.S., Converging marriage in honey-bees optimization and application to stochastic dynamic programming, Journal of Global Optimization, 35(3), 423– 441, July 2006.

91. Haddad, O. B., A.A., Mariño, M., Honey-bees mating optimization (hbmo) algorithm: A new heuristic approach for water resources optimization, Water Resources Management, 20(5), 661–680, October 2006.

92. Afshar, A., et al., Honey-bee mating optimization (hbmo) algorithm for optimal reservoir operation, Journal of the Franklin Institut, 344(5), 452– 462, August 2007.

93. Haddad, O.B., et al., Honey-bee mating optimization (hbmo) algorithm in deriving optimal operation rules for reservoirs, Journal of Hydroinformatics, 10(3), 257–264, 2008.

94. Haddad, O.B., et al., Optimum rehabilitation strategy of water distribution systems using the hbmo algorithm, Journal of Water Supply: Research and Technology-AQUA, 57(5), 337–350, 2008.

95. Amiri, B., Fathian, M., Integration of self organizing feature maps and honey bee mating optimization algorithm for market segmentation, Journal of Theoretical and Applied Information Technology, 3(3), 70–86, 2007.

96. Fathian, M., et al., Application of honey-bee mating optimization algorithm on clustering, Applied Mathematics and Computation, 190(2), 1502–1513, july 2007.

97. Fathian, M., Amiri, B., A honeybee-mating approach for cluster analysis, The International Journal of Advanced Manufacturing Technology, 38(7–8), 809–821, 2008.

98. Yang, C., et al., Algorithm of fast marriage in honey bees optimization and convergence analysis, IEEE International Conference on Automation and Logistics, 1794 – 1799, Jinan, August 2007.

99. Yang, C., et al., Algorithm of marriage in honey bees optimization based on the nelder-mead method, International Conference on Intelligent Systems and Knowledge Engineering (ISKE 2007), Advances in Intelligent Systems Research, October 2007.

100. Marinakis, Y., et al., Honey bees mating optimization algorithm for the vehicle routing problem, Nature Inspired Cooperative Strategies for Optimization (NICSO 2007), Studies in Computational Intelligence, volume 129/2008, 139–148, June 2008.

101. Marinakis, Y., et al., A hybrid clustering algorithm based on honey bees mating optimization and greedy randomized adaptive search procedure, Learning and Intelligent Optimization, Lecture Notes in Computer Science, volume 5313/2008, 138–152, 2008.

102. Niknam, T., et al., A hybrid algorithm based on hbmo and fuzzy set for multi-objective distribution feeder reconfiguration, World Applied Sciences Journal, 4(2), 308– 315, 2008.

103. Yang, C., et al., Algorithm of marriage in honey bees optimization based on the wolf pack search, The 2007 International Conference on Intelligent Pervasive Computing, 2007. IPC., 462–467, Oct. 2007.

104. Niknam, T., Application of honey-bee mating optimization on state estimation of a power distribution system including distributed generators, Journal Journal of Zhejiang University - Science A, 9(12), 1753–1764, Dec. 2008.

105. Yang, C.R., et al., Optimization of ground anti-aircraft weapon system networks based on direction probability and algorithm of improved marriage in honey bee optimization, Ordnance Acta Armamentarii, 29(2), 2008.

106. Sumpter, D.J.T., Broomhead, D.S., Formalising the link between worker and society in honey bee colonies, Multi-Agent Systems and Agent-Based Simulation, Lecture Notes in Computer Science, volume 1534/1998, 95–110, 1998.

107. Lucic, P., Teodorovic, D., Bee system: Modeling combinatorial optimization transportation engineering problems by swarm intelligence, Preprints of the TRISTAN IV Triennial Symposium on Transportation Analysis, 441–445, Sao Miguel, Azores Islands, Portugal, June 2001.

108. Lucic, P., Modeling transportation problems using concepts of swarm intelligence and soft computing, Ph.D. thesis, Virginia Polytechnic Institute and State University, 2002, chair-Dusan Teodorovic.

109. Lucic, P., Teodorovic, D., Transportation modeling: an artificial life approach, 14th IEEE International Conference on Tools with Artificial Intelligence, 2002. (ICTAI 2002)., 216 – 223, november 2002.

110. Teodorovic, D., Transport modeling by multi-agent systems: a swarm intelligence approach., Transportation Planning and Technology, 26(4), 289– 312, August 2003.

111. Lucic, P., Teodorovic, D., Computing with bees: Attacking complex transportation engineering problems, International Journal on Artificial Intelligence Tools, 12(3), 375– 394, 2003.

112. Lucic, P., Teodorovic, D., Fuzzy Sets Based Heuristics for Optimization, chapter Vehicle Routing Problem With Uncertain Demand at Nodes: The Bee System and Fuzzy Logic Approach, 67– 82, Springer - Verlag, Berlin Heidelberg, 2003.

113. Teodorovic, D., Dell, M.O., Bee colony optimization - a cooperative learning approach to complex transportation problems, Advanced OR and AI Methods in Transportation, 51– 60, 2005.

114. Markovic, G., et al., Routing and wavelength assignment in all-optical networks based on the bee colony optimization, AI Communications, European Journal of Artificial Intelligence, 20, 273–285, 2007.

115. Vassiliadis, V., Dounias, G., Nature inspired intelligence for the constrained portfolio optimization problem, Artificial Intelligence: Theories, Models and Applications, Lecture Notes in Computer Science, volume 5138/2008, 431–436, 2008.

116. Banarjee, S., et al., Metaheuristics for Scheduling in Industrial and Manufacturing Applications, Studies in Computational Intelligence, volume 128, chapter Modelling Process and Supply Chain Scheduling using Hybrid Meta-heuristics, 277–300, Springer, 2008.

117. Wong, L., et al., Bee colony optimization algorithm for traveling salesman problem, Second Asia International Conference on Modeling and Simulation, 2008. AICMS 08., 818– 823, May 2008.

118. Teodorovic, D., L.P., et al., Bee colony optimization: Principles and applications, 8th Seminar on Neural Network Applications in Electrical Engineering, 2006. NEUREL 2006., 151–156, september 2006.

119. Teodorovic, D., Dell'orco, M., Mitigating traffic congestion: Solving the ride-matching problem by bee colony optimization, Transportation Planning and Technology,, 31(2), 135–152, April 2008.

120. Teodorovic, D., Swarm intelligence systems for transportation engineering: Principles and applications, Transportation Research Part C: Emerging Technologies, 16(6), 651 – 667, 2008, ISSN 0968-090X, URL http://www.sciencedirect.com/science/article/B6VGJ-4SV0YTN-1/2/020a5d7efd4298ce3e18862af26424fa.

121. V., T., Reaction-diffusion model of a honeybee colony's foraging behaviour, PPSN VI: Proceedings of the 6th International Conference on Parallel Problem Solving from Nature, 807–816, Springer-Verlag, London, UK, 2000, ISBN 3-540-41056-2.

122. Tereshko, V., Lee, T., How information-mapping patterns determine foraging behaviour of a honey bee colony, Open Systems & Information Dynamics, 9(2), 181–193, 2002, ISSN 1230-1612.

123. Tereshko, V., Loengarov, A., Collective decision making in honey-bee foraging dynamics, Computing and Informaton Systems, 9(3), 1– 7, 2005.

124. Loengarov, A., Tereshko, V., Phase transitions and bistability in honeybee foraging dynamics, Artif. Life, 14(1), 111–120, 2008, ISSN 1064-5462.

125. Ghosh, S., Marshall, I., Simple model of Collective Decision Making during Nectar Source selection by Honey Bees, CD Rom of Workshop on Memory and Learning Mechanisms in Autonomous Robots (ECAL 2005), 10, September 2005, URL http://www.cs.kent.ac.uk/pubs/2005/2251.

126. Walker, R., Honeybee search strategies: adaptive exploration of an information ecosystem, Evolutionary Computation, 2004. CEC2004., volume 1, 1209– 1216, 2004.

127. Wedde, H., Farooq, M., The wisdom of the hive applied to mobile ad-hoc networks, Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE, 341–348, June 2005.

128. Wedde, H.F., et al., Beeadhoc: an energy efficient routing algorithm for mobile ad hoc networks inspired by bee behavior, GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation, 153–160, ACM, New York, NY, USA, 2005, ISBN 1-59593-010-8.

129. Mazhar, N., Farooq, M., Vulnerability analysis and security framework (beesec) for nature inspired manet routing protocols, GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation, 102–109, ACM, New York, NY, USA, 2007, ISBN 978-1-59593-697-4.

130. Saleem, M., Farooq, M., Beesensor: A bee-inspired power aware routing protocol for wireless sensor networks, Applications of Evolutionary Computing, LNCS, volume 4448/2007, 81– 90, 2007.

131. Saleem, M., et al., Formal modeling of beeadhoc: a bio-inspired mobile ad hoc network routing protocol, ANTS Conference, 315–322, 2008.

132. Mazhar, N., Farooq, M., A sense of danger: dendritic cells inspired artificial immune system for manet security, GECCO '08: Proceedings of the 10th annual conference on Genetic and evolutionary computation, 63–70, ACM, New York, NY, USA, 2008, ISBN 978-1-60558-130-9.

133. Basturk, B., Karaboga, D., An artificial bee colony (abc) algorithm for numeric function optimization, IEEE Swarm Intelligence Symposium 2006, Indianapolis, Indiana, USA, May 2006.

134. Karaboga, D., Basturk, B., A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (abc) algorithm, Journal of Global Optimization, 39(3), 459–471, 2007.

135. Karaboga, D., Basturk, B., On the performance of artificial bee colony (abc) algorithm, Applied Soft Computing, 8(1), 687–697, 2008.

136. Karaboga, D., Akay, B., Solving large scale numerical problems using artificial bee colony algorithm, 6th International Symposium on Intelligent and Manufacturin Systems Features, Strategies and Innovation, Sakarya, Turkiye, October 14-17 2008.

137. Karaboga, D., Akay, B., Effect of region scaling on the initialization of particle swarm optimization differential evolution and artificial bee colony algorithms on multimodal high dimensional problems, International Conference on Multivariate Statistical Modelling and High Dimensional Data Mining, Kayseri, TURKEY, June 19-23 2008.

138. Karaboga, D., Basturk, B., Advances in Soft Computing: Foundations of Fuzzy Logic and Soft Computing, LNCS, volume 4529/2007, chapter Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization Problems, 789–798, Springer-Verlag, 2007.

139. Karaboga, D., Akay, B., An artificial bee colony (abc) algorithm on training artificial neural networks, 15th IEEE Signal Processing and Communications Applications, SIU 2007, 1–4, Eskisehir, Turkiye, June 2007.

140. Karaboga, D., et al., Modeling Decisions for Artificial Intelligence, LNCS, volume 4617/2007, chapter Artificial Bee Colony (ABC) Optimization Algorithm for Training Feed-Forward Neural Networks, 318–329, Springer-Verlag, 2007.

141. Karaboga, D., et al., Training neural networks with abc optimization algorithm on medical pattern classification, International Conference on Multivariate Statistical Modelling and High Dimensional Data Mining, Kayseri, TURKEY, June 19-23 2008.

142. Ozturk, C., Karaboga, D., Classification by neural networks and clustering with artificial bee colony (abc) algorithm, 6th International Symposium on Intelligent and Manufacturin Systems Features, Strategies and Innovation, Sakarya, Turkiye, October 14-17 2008.

143. Fenglei, L., et al., The parameter improvement of bee colony algorithm in tsp problem, Nov. 2007, science Paper Online.

144. Singh, A., An artificial bee colony algorithm for the leaf-constrained minimum spanning tree problem, Applied Soft Computing, In Press, 2008.

145. Rao, R.S., et al., Optimization of distribution network configuration for loss reduction using artificial bee colony algorithm, International Journal of Electrical Power and Energy Systems Engineering, 1(2), 116–122, 2008.

146. Bendes, E., Ozkan, C., Direk lineer trasformasyon yönteminde yapay zeka tekniklerinin uygulanması, UZALCBS08, Kayseri, Turkiye, September 13-15 2008.

147. Karaboga, N., A new design method based on artificial bee colony algorithm for digital iir filters, Journal of The Franklin Institute, In Press, 2008.

148. Qingxian, F., Haijun, D., Bee colony algorithm for the function optimization, Aug. 2008, science Paper Online.

149. Hemamalini, S., Simon, S.P., Economic load dispatch with valve-point effect using artificial bee colony algorithm, XXXII National Systems Conference, India, Dec. 17-19 2008.

150. Quan, H., Shi, X., On the analysis of performance of the improved artificial-bee-colony algorithm, Fourth IEEE International Conference on Natural Computation, ICNC 2008, Jinan, China, August 25-27 2008.

151. Pawar, P., et al., Optimization of process parameters of abrasive flow machining process using artificial bee colony algorithm, Advances in Mechanical Engineering (AME-2008), Surat, India, December 15-17 2008.

152. Pawar, P., et al., Optimization of process parameters of milling process using particle swarm optimization and artificial bee colony algorithm, Advances in Mechanical Engineering (AME-2008), Surat, India, December 15-17 2008.

153. Pawar, P., et al., Multi-objective optimization of electro-chemical machining process parameters using artificial bee colony (abc) algorithm, Advances in Mechanical Engineering (AME-2008), Surat, India, December 15-17 2008.

154. Tsai, P-W., P.J.S.L.B.Y., Chu, S.C., Interactive artificial bee colony (iabc) optimization, ISI2008, Taiwan, December 12 2008.

155. Baykasoglu, A., et al., Swarm Intelligence Focus on Ant and Particle Swarm Optimization, chapter Artificial Bee Colony Algorithm and Its Application to Generalized Assignment Problem, 113– 144, I-Tech Education and Publishing, Vienna, Austria, December 2007, ISBN 978-3-902613-09-7.

156. Yang, X.S., Engineering optimizations via nature-inspired virtual bee algorithms, Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach, LNCS, volume 3562/2005, 317– 323, June 2005.

157. Pham, D.T., et al., The bees algorithm, Technical report, Manufacturing Engineering Centre, Cardiff University, UK, 2005.

158. Pham, D.T., et al., The bees algorithm - a novel tool for complex optimisation problems, Proceedings of IPROMS 2006 Conference, 454– 461, Cardiff, UK., 2006.

159. Pham, D.T., et al., Optimising neural networks for identification of wood defects using the bees algorithm, Proc 2006 IEEE International Conference on Industrial Informatics, 1346– 1351, Singapore, 2006.

160. Pham, D.T., et al., Optimisation of the weights of multi-layered perceptrons using the bees algorithm, Proc 5th International Symposium on Intelligent Manufacturing Systems, 2006.

161. Pham, D.T., et al., Application of the bees algorithm to the training of radial basis function networks for control chart pattern recognition, Proc 5th CIRP International Seminar on Intelligent Computation in Manufacturing Engineering (CIRP ICME '06), Ischia, Italy, 2006.

162. Pham, D.T., et al., Application of the bees algorithm to the training of learning vector quantisation networks for control chart pattern recognition, Proc Information and Communication Technologies (ICTTA'06), 1624– 1629, 2006.

163. Pham, D.T., Ghanbarzadeh, A., Multi-objective optimisation using the bees algorithm, Proceedings of IPROMS 2007 Conference, Cardiff, UK., 2007.

164. Pham, D.T., et al., Manufacturing cell formation using the bees algorithm, IPROMS 2007 Innovative Production Machines and Systems Virtual Conference, Cardiff, UK., Cardiff, UK., 2007.

165. Pham, D.T., et al., Using the bees algorithm to schedule jobs for a machine, Proc Eighth International Conference on Laser Metrology, CMM and Machine Tool Performance, 430– 439, 2007.

166. Pham, D.T., et al., Using the bees algorithm to tune a fuzzy logic controller for a robot gymnast, Proceedings of IPROMS 2007 Conference, Cardiff, UK., 2007.

167. Pham, D.T., et al., Data clustering using the bees algorithm, Proc 40th CIRP Int. Manufacturing Systems Seminar, 2007.

168. Pham, D.T., et al., Using the bees algorithm to optimise a support vector machine for wood defect classification, IPROMS 2007 Innovative Production Machines and Systems Virtual Conference, Cardiff, UK., 2007.

169. Pham, D.T., et al., Preliminary design using the bees algorithm, Proc Eighth International Conference on Laser Metrology, CMM and Machine Tool Performance, LAMDAMAP, 420– 429, Euspen, UK, Cardiff, 2007.

170. Pham, D.T., et al., Some applications of the bees algorithm in engineering design and manufacture, Proc Int. Conference on Manufacturing Automation (ICMA 2007), Singapore, 2007.

171. Bahamish, H., et al., Protein conformational search using bees algorithm, Modeling and Simulation, 2008. AICMS 08. Second Asia International Conference on, 911–916, May 2008.

172. Guney, K., Onay, M., Bees algorithm for design of dual-beam linear antenna arrays with digital attenuators and digital phase shifters, Int. J. RF Microw. Comput.-Aided Eng., 18(4), 337–347, 2008, ISSN 1096-4290.

173. Lee, J.Y., H., D.A., Multi-objective environmental/economic dispatch using the bees algorithm with weighted sum, EKC2008 Proceedings of the EU-Korea Conference on Science and Technology, Springer Proceedings in Physics, volume 124, 267–274, 2008.

174. Drias, H., S.S., Yahi, S., Cooperative bees swarm for solving the maximum weighted satisfiability problem, Computational Intelligence and Bioinspired Systems, LNCS, volume 3512/2005, 318– 325, June 2005.

175. Sadeg, S., Drias, H., A selective approach to parallelise bees swarm optimisation metaheuristic and application to max-w-sat, Int. J. Innov. Comput. Appl., 1(2), 146–158, 2007, ISSN 1751-648X.

176. Chong, C.S., et al., A bee colony optimization algorithm to job shop scheduling, WSC '06: Proceedings of the 38th conference on Winter simulation, 1954–1961, Winter Simulation Conference, 2006, ISBN 1-4244-0501-7.

177. Chong, C.S., et al., Using a bee colony algorithm for neighborhood search in job shop scheduling problems, 21st European Conference on Modeling and Simulation (ECMS 2007), 2007.

178. Quijano, N., Passino, K., Honey bee social foraging algorithms for resource allocation, part i: Algorithm and theory, American Control Conference, 2007. ACC '07, 3383–3388, July 2007, ISSN 0743-1619.

179. Quijano, N., Passino, K., Honey bee social foraging algorithms for resource allocation, part ii: Application, American Control Conference, 2007. ACC '07, 3389–3394, July 2007, ISSN 0743-1619.

180. Baig, A., Rashid, M., Foraging for fitness: A honey bee behavior based algorithm for function optimization, Technical report, NUCES, Pakistan, November 2006.

181. Baig, A.R., Rashid, M., Honey bee foraging algorithm for multimodal & dynamic optimization problems, GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation, 169–169, ACM, New York, NY, USA, 2007, ISBN 978-1-59593-697-4.

182. Lu, X., Zhou, Y., A novel global convergence algorithm: Bee collecting pollen algorithm, ICIC '08: Proceedings of the 4th international conference on Intelligent Computing, 518–525, Springer-Verlag, Berlin, Heidelberg, 2008, ISBN 978-3-540-85983-3.

183. Ko, S.Y., et al., A new class of nature-inspired algorithms for self-adaptive peer-to-peer computing, ACM Trans. Auton. Adapt. Syst., 3(3), 1–34, 2008, ISSN 1556-4665.

184. Ashlock, D., Oftelie, J., Simulation of floral specialization in bees, Evolutionary Computation, 2004. CEC2004., volume 2, 1859– 1864, 2004.

185. Purnamadjaja, A.H., Russell, R.A., Pheromone communication in a robot swarm: necrophoric bee behaviour and its replication, Robotica, 23(6), 731–742, 2005, ISSN 0263-5747.

186. Purnamadjaja, A., Russell, R., Guiding robots' behaviors using pheromone communication, Auton. Robots, 23(2), 113–130, 2007, ISSN 0929-5593.

187. Bianco, G., Getting inspired from bees to perform large scale visual precise navigation, Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on, volume 1, 619–624 vol.1, 2004.

188. Lemmens, N., et al., A bee algorithm for multi-agent systems: Recruitment and navigation combined., Adaptive and Learning Agents (ALAg-07), 2007.

189. Lemmens, N., et al., Bee behaviour in multi-agent systems: A bee foraging algorithm, K. Tuyls, A. Nowe, Z. Guessoum, D. Kudenko, eds., Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning, Lecture Notes in Artificial Intelligence, volume 4865/2008, 145–156, 2008.

190. Lemmens, N., et al., Bee system with inhibition pheromones, European Conference on Complex Systems, 2007.

191. Walker, A., et al., Bee-havior in a mobile robot: the construction of a self-organized cognitive map and its use in robot navigation within a complex, natural environment, 1451–1456 vol.3, 1993.

192. Hamdan, K., How do bees make honey, 2008, bee Research Unit,National Center for Agriculture Research and Technology Transfer, bee. (NCARTT), http://www.jordanbru.info/how do Bees make hony.htm.

193. Mackean, D.G., The honey bee (apis mellifera), 2008, resources for Biology Education, http://www.biology-resources.com/bee-01.html.

194. Boyer, D.O., et al., Crossover operator for evolutionary algorithms based on population features, Journal of Artificial Intelligence Research, 24, 1–48, 2005.

195. Junior, A.D., et al., Performance and parameterization of the algorithm simplified generalized simulated annealing, Genetics and Molecular Biology, 27(4), 616–622, 2004.

196. Digalakis, J.G., Margaritis, K.G., An experimental stduy of benchmarking functions for genetic algorithms, Intern. J. Computer Math., 79(4), 403–416, 2002.

197. DeJong, K., Parameter setting in eas: a 30 year perspective, Parameter Setting in Evolutionary Algorithms, 1–18, 2007.

198. Eiben, A., et al., Parameter control in evolutionary algorithms, Evolutionary Computation, IEEE Transactions on, 3(2), 124–141, Jul 1999, ISSN 1089-778X.

199. Eiben, A.E., et al., Parameter control in evolutionary algorithms, Parameter Setting in Evolutionary Algorithms, 19–46, 2007.

200. Liang, J., et al., Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, Evolutionary Computation, IEEE Transactions on, 10(3), 281–295, June 2006, ISSN 1089-778X.

201. Bäck, T., Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms, Oxford University Press, Oxford, UK, 1996, ISBN 0-19-509971-0.

202. García, S., et al., A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the cec'2005 special session on real parameter optimization, Journal of Heuristics, 2008, URL `http://dx.doi.org/10.1007/s10732-008-9080-4`.

203. Corne, D., et al., New Ideas in Optimization, McGraw–Hill, 1999.

204. Vesterstrom, J., Thomsen, R., A comparative study of differential evolution particle swarm optimization and evolutionary algorithms on numerical benchmark problems, IEEE Congress on Evolutionary Computation (CEC'2004), volume 3, 1980–1987, Piscataway, New Jersey, June 2004.

205. Bratton, D., Kennedy, J., Defining a standard for particle swarm optimization, Swarm Intelligence Symposium, 2007. SIS 2007. IEEE, 120–127, Honolulu, HI, 2007.

206. Michalewicz, Z., Janikow, C., Genetic Algorithms + Data Structures = Evolution Programs, Springer–Verlag, Newyork, 3rd edition, 1996.

207. Nemhauser, G., Wolsey, L., Handbooks in Operations Research and Management Science, 1, chapter Integer Progamming, Elsevier Science and Technology, 1989.

208. Misra, K., Sharma, U., An efficient algorithm to solve integer-programming problems arising in system-reliability design, IEEE Transactions On Reliability, 40(1), 81–91, 1991.

209. Rouillon, S., et al., An extended branch-and-bound method for locomotive assignment, Transportation Research Part B: Methodological, 40(5), 404–423, 2006.

210. Rudolph, G., An evolutionary algorithm for integer programming, PPSN III: Proceedings of the International Conference on Evolutionary Computation. The Third Conference on Parallel Problem Solving from Nature, 139–148, Springer-Verlag, London, UK, 1994, ISBN 3-540-58484-6.

211. Glankwahmdee, A., et al., Unconstrained discrete nonlinear programming, Engineering Optimization, 4, 95–107, 1979.

212. Rao, S.S., Engineering Optimization- Theory and Practice, Wiley Eastern:New Delhi, 1996.

213. Suganthan, P.N., et al., Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization, Technical report, Nanyang Technological University, Singapore, http://www.ntu.edu.sg/home/EPNSugan, 2005.

214. Liang, J., et al., Novel composition test functions for numerical global optimization, Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE, 68–75, June 2005.

215. Parsopoulos, K.E., Vrahatis, M.N., Particle swarm optimization method for constrained optimization problems, In Proceedings of the Euro-International Symposium on Computational Intelligence 2002, 214–220, Press, 2002.

216. Michalewicz, Z., Schoenauer, M., Evolutionary algorithms for constrained parameter optimization problems, Evolutionary Computation, 4(1), 1– 32, 1995.

217. Michalewicz, Z., et al., Evolutionary Algorithms for Engineering Applications, K. Miettinen, P. Neittaanmäki, M.M. Mäkelä, J. Périaux, eds., Evolutionary

Algorithms in Engineering and Computer Science, 73–94, John Wiley and Sons, Chichester, England, 1999.

218. Deb, K., An efficient constraint handling method for genetic algorithms, Computer Methods in Applied Mechanics and Engineering, 186(2- 4), 311–338, 2000.

219. Runarsson, T.P., Yao, X., Stochastic ranking for constrained evolutionary optimization, Evolutionary Computation, IEEE Transactions on, 4(3), 284–294, Sep 2000, ISSN 1089-778X.

220. Koziel, S., Michalewicz, Z., Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization, Evol. Comput., 7(1), 19–44, 1999, ISSN 1063-6560.

221. Runarsson, T.P., Yao, X., Search biases in constrained evolutionary optimization, IEEE Transactions on Systems, Man, and Cybernetics Part C—Applications and Reviews, 35(2), 233–243, May 2005.

222. Hamida, S.B., Schoenauer, M., ASCHEA: New Results Using Adaptive Segregational Constraint Handling, Proceedings of the Congress on Evolutionary Computation 2002 (CEC'2002), volume 1, 884–889, IEEE Service Center, Piscataway, New Jersey, May 2002.

223. Michalewicz, Z., Janikow, C.Z., Handling Constraints in Genetic Algorithms, R.K. Belew, L.B. Booker, eds., Proceedings of the Fourth International Conference on Genetic Algorithms (ICGA-91), 151–157, University of California, San Diego, Morgan Kaufmann Publishers, San Mateo, California, 1991.

224. Michalewicz, Z., Schoenauer, M., Evolutionary Algorithms for Constrained Parameter Optimization Problems, Evolutionary Computation, 4(1), 1–32, 1996.

225. Bean, J., Hadj-Alouane, A.B., A Dual Genetic Algorithm for Bounded Integer Programs, Technical Report TR 92-53, Department of Industrial and Operations Engineering, The University of Michigan, 1992, to appear in R.A.I.R.O.-R.O. (invited submission to special issue on GAs and OR).

226. Homaifar, A., et al., Constrained Optimization via Genetic Algorithms, Simulation, 62(4), 242–254, 1994.

227. Joines, J., Houck, C., On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GAs, D. Fogel, ed., Proceedings of the first IEEE Conference on Evolutionary Computation, 579–584, IEEE Press, Orlando, Florida, 1994.

228. Michalewicz, Z., Attia, N.F., Evolutionary Optimization of Constrained Problems, Proceedings of the 3rd Annual Conference on Evolutionary Programming, 98–108, World Scientific, 1994.

229. Richardson, J.T., et al., Some Guidelines for Genetic Algorithms with Penalty Functions, J.D. Schaffer, ed., Proceedings of the Third International Conference on Genetic Algorithms (ICGA-89), 191–197, George Mason University, Morgan Kaufmann Publishers, San Mateo, California, June 1989.

230. Schoenauer, M., Xanthakis, S., Constrained GA Optimization, S. Forrest, ed., Proceedings of the Fifth International Conference on Genetic Algorithms (ICGA-93), 573–580, University of Illinois at Urbana-Champaign, Morgan Kauffman Publishers, San Mateo, California, July 1993.

231. Powell, D., Skolnick, M.M., Using genetic algorithms in engineering design optimization with non-linear constraints, S. Forrest, ed., Proceedings of the Fifth International Conference on Genetic Algorithms (ICGA-93), 424–431, University of Illinois at Urbana-Champaign, Morgan Kaufmann Publishers, San Mateo, California, July 1993.

232. Michalewicz, Z., Nazhiyath, G., Genocop III: A co-evolutionary algorithm for numerical optimization with nonlinear constraints, D.B. Fogel, ed., Proceedings of the Second IEEE International Conference on Evolutionary Computation, 647–651, IEEE Press, Piscataway, New Jersey, 1995.

233. Paredis, J., Co-evolutionary Constraint Satisfaction, Proceedings of the 3rd Conference on Parallel Problem Solving from Nature, 46–55, Springer Verlag, New York, 1994.

234. Parmee, I.C., Purchase, G., The development of a directed genetic search technique for heavily constrained design spaces, I.C. Parmee, ed., Adaptive Computing in Engineering Design and Control-'94, 97–102, University of Plymouth, Plymouth, UK, 1994.

235. Myung, H., et al., Preliminary investigation into a two-stage method of evolutionary optimization on constrained problems, J.R. McDonnell, R.G. Reynolds, D.B. Fogel, eds., Proceedings of the Fourth Annual Conference on Evolutionary Programming, 449–463, MIT Press, Cambridge, Massachusetts, 1995.

236. Reynolds, R.G., et al., Using cultural algorithms for constraint handling in GENOCOP, J.R. McDonnell, R.G. Reynolds, D.B. Fogel, eds., Proceedings of the Fourth Annual Conference on Evolutionary Programming, 298–305, MIT Press, Cambridge, Massachusetts, 1995.

237. Michalewicz, Z., Genetic Algorithms, Numerical Optimization, and Constraints, L.J. Eshelman, ed., Proceedings of the Sixth International Conference on Genetic Algorithms (ICGA-95), 151–158, University of Pittsburgh, Morgan Kaufmann Publishers, San Mateo, California, July 1995.

238. Michalewicz, Z., Janikow, C.Z., Handling Constraints in Genetic Algorithms, R.K. Belew, L.B. Booker, eds., Proceedings of the Fourth International Conference on Genetic Algorithms (ICGA-91), 151–157, University of California, San Diego, Morgan Kaufmann Publishers, San Mateo, California, 1991.

239. Michalewicz, Z., Naguib, F. Attia, N., Evolutionary Optimization of Constrained Problems, Proceedings of the 3rd Annual Conference on Evolutionary Programming, 98–108, World Scientific, 1994.

240. Joines, J.A., Houck, C.R., On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GAs, IEEE Int. Conf. Evol. Comp., 579–584, IEEE Press, 1994.

241. Parsopoulos, K., Vrahatis, M., Unified Particle Swarm Optimization for solving constrained engineering optimization problems, Advances in Natural Computation, Pt. 3, 582–591, 2005, lecture Notes in Computer Science Vol. 3612.

242. Hu, X., Eberhart, R., Solving constrained nonlinear optimization problems with particle swarm optimization, In Proceedings of the Sixth World Multiconference on Systemics, Cybernetics and Informatics 2002 (SCI 2002, 203–206, 2002.

243. Hu, X., et al., Engineering optimization with particle swarm, Swarm Intelligence Symposium, 2003. SIS '03. Proceedings of the 2003 IEEE, 53–57, April 2003.

244. Muñoz-Zavala, A., et al., Constrained Optimization via Particle Evolutionary Swarm Optimization Algorithm (PESO), H.G. Beyer, U.M. O'Reilly, D. Arnold, W. Banzhaf, C. Blum, E. Bonabeau, E. Cant Paz, D. Dasgupta, K. Deb, J. Foste r, E. de Jong, H. Lipson, X. Llora, S. Mancoridis, M. Pelikan, G. Raidl, T. Soule, A. Tyrrell, J.P. Watson, E. Zitzler, eds., Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2005), volume 1, 209–216, Washington DC, USA, ACM Press, New York, June 2005, iSBN 1-59593-010-8.

245. Zhang, W., Xie, X., Depso: hybrid particle swarm with differential evolution operator, volume 4, 3816– 3821, Oct. 2003, ISSN 1062-922X.

246. Storn, R., System Design by Constraint Adaptation and Differential Evolution, IEEE Transactions on Evolutionary Computation, 3(1), 22–34, April 1999.

247. Lampinen, J., Zelinka, I., New Ideas in Optimization, chapter Mechanical Engineering Design Optimization by Differential Evolution, 127–146, Mc Graw-Hill, UK, 1999.

248. Lampinen, J., A Constraint Handling Approach for the Diifferential Evolution Algorithm, Proceedings of the Congress on Evolutionary Computation 2002 (CEC'2002), volume 2, 1468–1473, IEEE Service Center, Piscataway, New Jersey, May 2002.

249. Mezura-Montes, E., et al., Simple Feasibility Rules and Differential Evolution for Constrained Optimization, R. Monroy, G. Arroyo-Figueroa, L.E. Sucar, H. Sossa, eds., Proceedings of the 3rd Mexican International Conference on Artificial Intelligence (MICAI'2004), 707–716, Springer Verlag, Heidelberg, Germany, April 2004, lecture Notes in Artificial Intelligence No. 2972.

250. Mezura-Montes, E., et al., Promising Infeasibility and Multiple Offspring Incorporated to Differential Evolution for Constrained Optimization, H.G. Beyer, U.M. O'Reilly, D. Arnold, W. Banzhaf, C. Blum, E. Bonabeau, E. Cant Paz, D. Dasgupta, K. Deb, J. Foste r, E. de Jong, H. Lipson, X. Llora, S. Mancoridis, M. Pelikan, G. Raidl, T. Soule, A. Tyrrell, J.P. Watson, E. Zitzler, eds., Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2005), volume 1, 225–232, Washington DC, USA, ACM Press, New York, June 2005, iSBN 1-59593-010-8.

251. Karaboga, D., Akay, B., A comparative study of artificial bee colony algorithm, Applied Mathematics and Computation, 214, 108–132, 2009.

252. Mezura-Montes, E., Coello Coello, C., Adding a diversity mechanism to a simple evolution strategy to solve constrained optimization problems, Evolutionary Computation, 2003. CEC '03. The 2003 Congress on, 1, 6–13 Vol.1, Dec. 2003.

253. Ogata, K., Modern control engineering, Prentice-Hall, Inc., USA, second edition, 1990.

254. Kuo, B., Otomatik Kontrol Sistemleri, Literatür Yayınları, 2002.

255. Akay, R., Zeki programlama teknikleri kullanarak kontrolör tasarımı, Master's thesis, Erciyes Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Müh. ABD., 2006.

256. Ziegler, J., Nichols, N., Optimum settmgs for automtic controllers, TW. ASME, 64, 759–768, 1942.

257. Chien, K., et al., On the automatic control of generalized passive systems, Trans. ASME, 74, 175–185, 1952.

258. Kitamori, T., A method for control system design based upon partial knowledge about controlled process, Trans. SICE Japan, 15, 549–555, 1979.

259. Mezura-Montes, E., Coello Coello, C., Useful infeasible solutions in engineering optimization with evolutionary algorithms, MICAI 2005: Advances in Artificial Intelligence, Lecture Notes in Computer Science, volume 3789/2005, 652–662, Springer Berlin / Heidelberg, 2005.

260. Ray, T., Liew, K., Society and civilization: An optimization algorithm based on the simulation of social behavior, IEEE Transactions on Evolutionary Computation, 7, 386–396, 2003.

261. He, S., et al., An improved particle swarm optimizer for mechanical design optimization problems, Engineering Optimization, 36, 585–605, 2004.

262. Rechenberg, I., Cybernetic solution path of an experimental problem, Library translation 1122, Royal Aircraft Establishment, Farnborough, Hants, U.K., 1965.

263. Schwefel, H.P., Kybernetische evolution als strategie der experimentellen forschung in der stromungstechnik, Master's thesis, Technical University of Berlin, Germany, 1965.

264. Bäck, T., Schwefel, H.P., An overview of evolutionary algorithms for parameter optimization, Evolutionary Computation, 1(1), 1–23, 1993.

265. Kohonen, T., Self–Organizing Maps, New York: Springer Verlag, 1995.

266. Martinetz, T., Schulten, S., A neural–gas network learns topologies, K.K. et al, ed., Artificial Neural Networks, 397–402, Elsevier, 1991.

267. Hansen, N., Ostermeier, A., Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation, IEEE Int. Conf. Evolution. Comput. (ICEC) Proc., 312–317, 1996.

268. Auger, A., Hansen, N., A restart cma evolution strategy with increasing population size, Evolutionary Computation, 2005. The 2005 IEEE Congress on, volume 2, 1769–1776, Sept. 2005.

269. Runarsson, T.P., Yao, X., Search biases in constrained evolutionary optimization, IEEE Transactions on Systems, Man, and Cybernetics Part C—Applications and Reviews, 35(2), 233–243, May 2005.

270. Pham, D.T., Karaboga, D., Optimum design of fuzzy logic controllers using genetic algorithms, Journal of Systems Engineering, 1, 114–118, 1991.

271. Ballester, P., Carter, J., An effective real-parameter genetic algorithm with parent centric normal crossover for multimodal optimization, Genetic and Evolutionary Computation, GECCO 2004, Lecture Notes in Computer Science, volume 3102/2004, 901–913, 2004.

272. Ballester, P., New computational methods to address nonlinear inverse problems, Ph.D. thesis, Department of Earth Science and Engineering, Imperial College, University of London, 2005.

273. Storn, R., Price, K., Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces, Technical report, International Computer Science Institute, Berkley, 1995.

274. Storn, R., Price, K., Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces., Journal of Global Optimization, 11, 341–359, 1997.

275. Koziel, S., Michalewicz, Z., Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization, Evol. Comput., 7(1), 19–44, 1999, ISSN 1063-6560.

276. Sun, J., et al., Particle swarm optimization with particles having quantum behavior, Evolutionary Computation, 2004. CEC2004. Congress on, volume 1, 325–331 Vol.1, June 2004.

EK-1.

Karşılaştırma Yapılan Algoritmalar

1.1. Gelişim Stratejileri

Gelişim stratejisi (Evolution Strategy, ES) [262, 263] nümerik optimizasyon problemleri için geliştirilmiş en eski algoritmalardan biridir. Ebeveyn vektör üzerinde mutasyon gerçekleştirerek n boyutlu reel değerli bir vektör oluşturur. Bu şekilde oluşan yeni birey değerlendirilir ve bir sonraki iterasyonda hangi bireyin hayatta kalacağını belirlemek için seleksiyon işlemi uygulanır. Bu basit seleksiyon mekanizmasına (1+1)-seleksiyon adı verilir. Çok sayıda üyenin dahil olduğu (multi-membered)(μ+1)-ES'de μ tane ebeveyn biraraya gelerek yeni bir birey (döl, offspring)oluşturur ve bu bireyin daha iyi olması durumunda popülasyondaki en kötü bireyle değiştirilir [264]. $\mu/\rho, \lambda$)-ES veya $(\mu/\rho + \lambda)$-ES olarak ifade edilen kanonik ES (CES)'de μ ebeveyn sayısı, $\rho \leq \mu$ de yeni birey oluşumunda görev alan ebeveyn sayısı ve λ da oluşan birey sayısıdır. Ebeveynler bir çocuk kümesinden deterministik olarak seçilirler. Buna virgül seçme (comma-selection) adı verilir ve $\mu < \lambda$ olmalıdır. Seleksiyon bireylerin uygunluk değerlerine göre yapılır. ES'nin temel adımları aşağıda verilmektedir:

1: Popülasyonun başlatılması
2: **repeat**
3: Yeniden oluşum
4: Mutasyon
5: Değerlendirme
6: Seçme

7: **until** Durma Kriteri

Mutasyon her bir vektöre normal dağılımlı rasgele bir terim eklenmesi ile gerçekleştirilir. Cauchy mutasyon operatorü kullanan ES lere hızlı ES'ler (Fast Evolution Strategy, FES) adı verilir [4]. Bir diğer mutasyon operatörü kendi kendine organize olabilen haritalar (SOM) kullanan mutasyon operatörüdür. Bu operatörde daha başarılı adaylara daha çok seçilme olasılığr verecek şekilde başarılı bireyler üzerinde değişim gerçekleştirilir [265]. Kohonen'in geliştirdiği ES (KSOM-ES), SOM'a dayalı mutasyon operatörünü kullanmaktadır [5]. Sinirsel gaz ağları gelişim stratejisi (neural gas networks evolution strategy, NG-ES) KSOM-ES'nin kötü ölçekleme özelliğini gidermek için geliştirilmiştir [266]. Kovaryans matrisi uyarlamalı ES'ler (covariance matrix adaptation-ES, CMA-ES) lokal bilginin en iyi şekilde kullanımını sağlamak için geliştirilmişlerdir ve adım büyüklüğü self-adaptasyon yada kovaryans matris adaptasyonuna göre belirlenebilir [267]. LR-CMA-ES (Local Restart-CMA-ES), (μ_W, λ) algoritmasının bir versiyonu olup kontrol parametresizmiş gibi çalışan (μ_W, λ)'nın lokal arama özelliklerini kullanan bir global optimizasyon algoritmasıdır. Yeniden başlama (restart) stratejisi ile bir durdurma kriteri sağlanan kadar çalışmaya devam eder ve sonrasında yeniden başlatılır [13]. Artan popülasyonlu yeniden başlatmalı CMA-ES (restart (μ_W, λ)-CMA-ES with increasing population,IPOP-CMA-ES) de ise bu durdurup başlatma işlemlerinin her birinde popülasyon büyüklüğü iki katına çıkarılır [268].

Otomatik durdurma kriterli ES'ler (Evolution Strategies Learned with Automatic Termination Criteria, ESLAT) bir gen matrisine göre belirlenen bir otomatik durdurma kriterine sahip başka bir ES çeşididir [3].

Basit çoğul üyeli gelişim stratejisi (Simple Multimembered Evolution Strategy, SMES) $(\mu+\lambda)$-ES şeklinde çalışan bir yaklaşımdır. Standart $(\mu+\lambda)$-ES'den farkları başlangıç adım büyüklüğünün azaltılması, panmiktik rekombinasyon operatörünün kullanılması, ES'nin deterministik yer değiştirme operatörü yerine fizibıllığa dayalı bir karşılaştırma operatörünün kullanılması, bir sonraki popülasyona en iyi fizibıl olmayan çözümün de dahil edilmesidir [14].

Over-penalized yaklaşımı (Over-Penalized Approach, OPA) sınırlamalı problemerde

fizibıl çözümleri amaç fonksiyonu değerlerine göre, fizibıl olmayan çözümleri ise penaltı fonksiyonu değerlerine göre derecelendiren bir gelişim stratejisidir [269].

ES'lerde kullanılan bir diğer sınırlama ele alış metodu amaç fonksiyonu ve penaltı değerlerinin dengelenmesi ihtiyacından doğan stokastik sıralamadır (stokastik ranking, SR) [219]. Sıralama baloncuk sıralama metodu ile sağlanmaktadır. r_i değeri iki bitişik bireyin amaç fonksiyonu değerlerinin farkının penaltı değerleri farkına oranı ve r_g adaptif olarak ayarlanan penaltı katsayısı olmak üzere r_g değeri alt sınırından küçük ise penaltı altıdır (underpenalization) ve kıyaslamalar uygunluk fonksiyonuna göre yapılır, üst sınırdan büyük ise penaltı üstüdür (overpenalization) ve kıyaslamalar penaltı değerine göre yapılır, alt ve üst sınır değerleri arasında ise kıyaslamalar amaç fonksiyonu ve penaltı fonksiyonunun kombinasyonuna göre yapılır. Optimal r_g değerlerinin bulunması zor olduğu için, stokastik sıralamada P_f olasılık değeri terimi getirilmiştir. Eğer kıyaslanan iki bitişik bireyin her ikiside fizibıl ise bu olasılık değeri 1 değerini alır (underpenelization), değilse önceden belirlenen ve adaptif olmayan P_f değeri kullanılır. P_f'nin 0 olması penaltı üstü (overpenalization)'a karşılık gelir. Çözümlerin her ikisi fizibıl ise yada P_f değeri uniform dağılımlı rasgele bir sayıdan büyük ise çözümler amaç fonksiyon değerine göre sıralanır, her ikiside fizibıl değil ve uniform dağılımlı rasgele sayı P_f'den büyük ise penaltı değeri küçük olana göre sıralama yapılır.

Bazı problemlerde penaltı değerlerinin bazılarında ise amaç fonksiyonu değerinin etkili olması sınırlamaların ele alınış biçiminin ve arama operatörünün optimizasyon sürecindeki etkisini göstermektedir. Geliştirilmiş stokastik ranking (improved stokastik ranking, ISR) ile koordinat eksenlerine dayalı araştırmanın kayması ile bu sorunun önüne geçilmesi amaçlanmıştır. Metodun kullandığı kontrol parametresi (γ) ile adım büyüklüğü ölçeklenmektedir. Daha küçük değerler daha iyi gelişim göstermiştir [269].

1.2. Gelişim Algoritmaları

1.2.1. Genetik Algoritma

Standart bir Genetik Algoritma (GA) [20] 5 temel bileşenden oluşur. Bunlar rasgele sayı jeneratörü, uygunluk hesaplama birimi, yeni birey üretimi için kullanılan genetik operatör (reproduction), çaprazlama ve mutasyon. Algoritmik formda yazacak olursak:

1: Popülasyonun başlatılması
2: **repeat**
3: Değerlendirme
4: Yeniden üretim
5: Çaprazlama
6: Mutasyon
7: **until** Durma kriteri

Algoritmanın başlangıcında oluşturulan başlangıç popülasyonu rasgele sayı üretecinin ürettiği sayı dizilerinden oluşmaktadır. Her bir dizi optimizasyon probleminin bulmaya çalıştığı parametreleri ifade eden bir çözümdür. İkili diziler genellikle kullanılmaktadır. Değerlendirme biriminde her bir diziye ait uygunluk değeri hesaplanmaktadır. Genetik operatörlerin amacı eldeki dizi kümesini daha yüksek uygunluk değerine sahip olan bir kümeye dönüştürmektir. Yeniden üretim (reproduction) birimi *seeded selection* adı verilen doğal seleksiyon işlemini gerçekleştirir. Bir dizi uygunluk değerine göre kümede kalır yada uygunluk değeri ile orantılı olarak kalma olasılığı vardır. Çaprazlama operatörü rasgele belirlediği iki diziden yeni bir çift çözüm üretir. En basit çaprazlama rasgele belirlenen bir noktadan iki dizinin yer değiştirilmedir. Çaprazlama işleminin sayısı çaprazlama oranına bağlıdır. Mutasyon operatörü rasgele belirlediği bir noktadaki dizi elemanını tersine çevirir. Mutasyon işleminin sayısıda mutasyon oranına bağlıdır. Algoritmanın her bir çevrimi değerlendirme, yeniden üretim, çaprazlama ve mutasyon işlemlerini içerir. Her bir döngüde yeni bir çözüm popülasyonu oluşur [270].

1.2.2. SPC-PNX

SPC-PNX metodu reel değerli vektörler üzerinde çalışan sürekli durumlu bir genetik algoritmadır [271]. Yer biliminde ortaya çıkan bazı lineer olmayan parametre kestirim problemlerine uygulanmıştır [272]. SPC-PNX'te o anki popülasyondan çaprazlama operatörü aracılığıyla bir çocuk birey oluşturmak için iki ebeveyn seçilir. Her bir çocuğa ait aöaç fonksiyonu değeri hesaplanır. Çocuk da popülasyona eklenir ve yer değiştirme operatörü ile popülasyonun boyutunun sabit kalması sağlanır. Seçme, çaprazlama, değerlendirme ve yer değiştirmeden oluşan bu dört adım GA'ya has adımlar olmasına rağmen operatörler SPC-PNX'e hastır. Seçme işleminde GA'daki uygunluk değerine dayalı seçme işleminden farklı olarak uniform rasgele seçme yapılmaktadır. Çaprazma adımında PNX çaprazlama operatörü kullanılmaktadır. Bu operatörde iki bireyin her bir parametresi için eğer rasgele atılan bir sayı 0.5 den büyük ise bir bireyden değilse diğer bireyden alınacak şekilde çaprazlama işlemi gerçekleştirilir. Yer değiştirme operatöründe ölçeklenmil olasılıksal yükleme (scaled probabilistic crowding) şeması kullanılmaktadır. Bu operatörde daha önceden seçilen iki bireyden en yakın olanı çocuk bireyle olasılıksal turnuva seleksiyon işlemine tabi tutulmaktadır.

1.2.3. Diferansiyel Gelişim Algoritması

Diferansiyel Gelişim (Differential Evolution, DE) algoritması [273] GA gibi popülasyon tabanlı, mutasyon, çaprazlama ve seleksiyon gibi benzer operatörleri olan bir algoritmadır. Daha iyi yeni çözüm üretmedeki temel farklılık GA'nın çaprazlama operatörü, DE'nin ise bunu mutasyon operatörü ile sağlamasıdır. DE algoritması rasgele seçilen çözümlerin farkına dayalı olarak bu mutasyon işlemini gerçekleştirmektedir. Araştırma mekanizması olarak mutasyon işlemi araştırmayı potansiyel bölgelere yönlendirmek içinde seleksiyon operatörü kullanılmaktadır. DE algoritması yeni çözümün bir kısmını ebeveynden diğer kısmını aday çözümden alacak şekilde uniform olmayan bir çaprazlama operatörü kullanmaktadır. DE algoritmasında da rasgele bir başlangıç popülasyonu üretilir ve bu popülasyon mutasyon, çaprazlama ve seleksiyon operatörleri ile daha iyi hale getirilir. DE algoritmasında her bir yeni çözüm bu çözümden üretilen yeni çözüm ile kıyaslanır ve daha iyi olan popülasyonda kalır. DE algoritması araştırmacılar

tarafından oldukça fazla ilgi görmüştür ve çeşitli gerçek hayat problemlerine uygulanmıştır [203, 273, 274]. Algoritmanın temel adımları aşağıda verilmektedir:

1: Popülasyonun başlatılması
2: Değerlendirme
3: **repeat**
4: Mutasyon
5: Çaprazlama
6: Değerlendirme
7: Seleksiyon
8: **until** Durma kriteri

Mutasyon: Çözüm vektörünün her bir elemanı mutasyona uğrar. Mutasyona uğramış x_i çözümü, $\hat{x}_i$, (EK-1.1) ile elde edilir:

$$\hat{x}_i = x_{r_1} + F(x_{r_3} - x_{r_2}) \tag{EK-1.1}$$

Burada F [0,1] arasında değer alan ölçekleme faktörüdür ve x_{r1}, x_{r2} ve x_{r3} çözümleri rasgele seçilen ve (EK-1.2)'yi sağlaması gereken çözüm vektörleridir.

$$x_{r_1}, x_{r_2}, x_{r_3} | r_1 \neq r_2 \neq r_3 \neq i \tag{EK-1.2}$$

i: o anki çözümün indisidir.

Çaprazlama: Ebeveyn çözüm mutasyona uğramış çözüm ile (EK-1.3) kuralınca karıştırılır:

$$y_i^j = \begin{cases} \hat{x}_i^j & R_j \leq CR \\ x_i^j & R_j > CR \end{cases} \tag{EK-1.3}$$

Burada CR çaprazlama oranı, R_j [0,1] arasında rasgele bir reel sayı ve j çözümün j. parametresidir.

Popülasyondaki tüm çözümler uygunluk değerlerine bakılmaksızın ebeveyn olarak seçilme hakkına sahiptirler. Mutasyon ve çaprazlama sonrasında oluşan çocuk birey

değerlendirilir ve çocuk bireyin performansı ile ebeveyninki karşılaştırılarak daha iyi olan popülasyonda kalmak üzere seçilir.

Diferansiyel gelişim algoritmasının performansının CR, F ve NP'ye olan bağımlılığını ortadan kaldırmak için öğrenme stratejilerini ve kontrol parametrelerini otomatik olarak uyarlayabilen kendi kendine adapte olabilen diferansiyel gelişim(Self Adaptive Differential Evolution, SADE) algoritması geliştirilmiştir [12].

1.2.4. Adaptif Ayrışmalı Sınırlamalı Gelişim Algoritması

Adaptif Ayrışmalı Sınırlamalı Gelişim Algoritması (Adaptive Segregational Constraint Handling Evolutionary Algorithm, ASCHEA) adaptif penaltı fonksiyonu kullanan sınırlamalı problemleri optimize etmeye çalışan bir algoritmadır. Penaltı katsayılarını ayarlarken popülasyon düzeyinde bir adaptif yapı kullanır. ASCHEA eş seçiminde adaptif bir seleksiyon yapısı ve fizibıllığa dayalı bir seleksiyon operatörü kullanır. Eşitlik sınırlamalarının sağlanabilmesi için her bir eşitlik için geniş bir fizibıl bölge ile başlayıp bu bölgenin daraltılması şeklinde bir strateji kullanmaktadır [222].

1.2.5. Biçimdeş Haritalamalı Gelişim Algoritması

Bu yaklaşımda gelişim algoritması sınırlamalı problemlere uygulanırken sınırlamaların ele alınış biçimi olarak biçimdeş haritalama (homomorphous mapping, HM) kullanılmıştır. Bu metotta fizibıl araştırma uzayı ile n boyutlu uzay arasında bir haritalama yapılmaktadır. Bu yaklaşım decoder tabalı yaklaşımlardan biridir [275].

1.3. Parçacık Sürüsü Optimizasyon Algoritması

Parçacık Sürüsü Optimizasyon (Particle Swarm Optimization, PSO) algoritmasında [37], bir parçacık popülasyonu araştırma uzayında o anki optimum parçacığı takip ederek ve pozisyonlarını değiştirerek hareket ederler. Bir parçacığın pozisyonu optimize edilecek problemin çözümüne karşılık gelmektedir. Fonksiyonda parçacığın pozisyonu yerine koyularak çözümün kalitesi belirlenir. Her iterasyonda, her bir

parçaçık o ana kadarki en iyi değerinin ($\vec{p}(t)$, particle best) ve tüm popülasyondaki parçacıkların en iyisinin ($\vec{g}(t)$, global best) takibinde değerini değiştirir. Bir parçacık topolojik komşular olarak popülasyonun bir kısmını kullanırsa bu lokal en iyidir. Parçacıklar tüm popülasyona bilginin yayılması ile araştırma uzayındaki daha iyi yerlere gitme eğilimindedirler. Bir parçacık her bir t anında güncellenen hız değeri ile bir sonraki pozisyonlarını hesaplarlar. Bu yeni pozisyon daha önceki pozisyon ve yeni hız değerinin toplanması ile bulunur (Eşitlik EK-1.4):

$$\vec{x}(t+1) = \vec{x}(t) + \vec{v}(t+1) \tag{EK-1.4}$$

Hız değerinin güncellenmesi ise Eşitlik EK-1.5 ile hesaplanır:

$$\begin{aligned} \vec{v}(t+1) = \quad & \omega\vec{v}(t) + \phi_1 \text{rand}(0,1)(\vec{p}(t) - \vec{x}(t)) \\ & + \phi_2 \text{rand}(0,1)(\vec{g}(t) - \vec{x}(t)) \end{aligned} \tag{EK-1.5}$$

ω atalet (inertia) parametresi olarak isimlendirilir ve yeni hız değerinin hesaplanmasında bir önceki hız değerinin $\vec{v}(t)$ ağırlığını ifade eder. ϕ_1 ve ϕ_2 sırasıyla $\vec{p}(t)$ ve $\vec{g}(t)$'nin önem derecelerini belirlemektedir. Ayrıca, υ_i'ın alabileceği maksimum değer υ_{max} ile sınırlanmaktadır. PSO algoritmasında sürü her bir parçacığa rasgele bir pozisyon atanmasıyla başlatılır ve hız vektörlerine de [υ_{min},υ_{max}] değerleri arasında başlangıç değerleri atanır. Algoritmanın temel adımları aşağıdaki gibidir:

1: Popülasyonun başlatılması
2: **repeat**
3: Parçacıkların uygunluk değerlerinin hesaplanması
4: Sürüdeki en iyi parçacıkların modifiye edilmesi
5: En iyi parçacığın belirlenmesi
6: Parçacıkların hızlarının hesaplanması
7: Parçacıkların pozisyonlarının güncellenmesi
8: **until** Durma kriteri

1.3.1. Kuantum Davranışlı PSO

Kuantum davranışlı PSO (Quantum Behaves Particle Swarm Optimization, QPSO) [276], süper pozisyon durumu ve belirsizlik prensiplerini irdeleyen Kuantum Teorisi PSO algoritması ile entegre eidlmiştir. p(pbest) merkezli x pozisyonundaki bir parçacığın olasılığı $Q(x) = |\Psi(x,t)|^2$'dır. Bu parçacığın dalga fonksiyonu Eşitlik (EK-1.6) ile hesaplanır:

$$\psi(x) = \frac{1}{\sqrt{L}} \exp(-\|p - x\| / L) \tag{EK-1.6}$$

Bir parçacığın araştırma çevresi enerji yoğunluğuna bağlı olarak L parametresi ile belirlenir (Eşitlik (EK-1.7)):

$$L(t+1) = 2 * \alpha * |p - x(t)| \tag{EK-1.7}$$

Popülasyon büyüklüğü az olduğunda erken yakınsamanın önlenmesi için tüm parçacıkların ağırlık merkezini ifade eden bir ortalama en iyi pozisyon (mbest) kavramı geliştirilmiştir. mbest (EK-1.8) eşitliği ile tanımlanır:

$$\begin{aligned} mbest &= \sum_{i=1}^{M} p_i / M \\ mbest &= \left(\sum_{i=1}^{M} p_{i1}/M, \sum_{i=1}^{M} p_{i2}/M, \ldots, \sum_{i=1}^{M} p_{id}/M \right) \end{aligned} \tag{EK-1.8}$$

Burada M popülasyon büyüklüğü ve p_i $i.$ parçacığın pbest değeridir. Böylece L değerinin tanımı şu şekilde elde edilir (EK-1.9):

$$L(t+1) = 2 * \beta * |mbest - x(t)| \tag{EK-1.9}$$

β parametresi her bir parçacığın yakınsama hızını ve algoritmanın performansını etkileyen yaratıcılık (creativity) katsayısıdır. Monte Carlo simülasyonun kullanılmasıyla her bir parçacığın pozisyonu Eşitlik (EK-1.10) ile hesaplanır:

$$x(t) = p \pm \frac{L}{2} \ln(1/u) \tag{EK-1.10}$$

L parametresini $x(t)$'de yerine koyarak bir sonraki pozisyon değeri şu şekilde hesaplanır (EK-1.11):

$$x(t+1) = p \pm \beta * |mbest - x(t)| * \ln(1/u) \qquad \text{(EK-1.11)}$$

1.3.2. Rekombinasyon ve Dinamik Bağlantılı PSO

PSO algoritmasında, parçacıklar reel sayı vektörleri olarak kodlanmaktadır. Bu gösterimden solayı boyutlar arasındaki ilişkiyi belirtmek için bağlantı (linkage) terimi kullanılmaktadır. Optimizasyon sürecinin değişik aşamalarında uygunluk yüzeyine (fitness landscape) ve ilgili popülasyonun dağılımına göre bağlantı yapıladırması farklı olabilir. Bu nedenle farklı boyutlar arasındaki ilişki popülasyon açısından optimizasyon süreci boyunca dinamik olarak değişmektedir hipotezinden yola çıkarak popülasyon dağılımındaki fonksiyon yapısı ile ilgili bilgiye göre bağlantı yapılandırması güncellenmelidir. Bağlantı uyarlaması için ekstra yapay kriter kullanmak yerine doğal seleksiyona dayandırılmaktadır. Optimizasyon sürecinde, rekombinasyon ve dinamik bağlantılı PSO (PSO with recombination and dynamic linkage discovery, PSO-RDL) algoritması rasgele bağlantı grupları oluşturarak amaç fonksiyonunun değerine göre bu bağlantılar güncellenmektedir. Popülasyonun ortalama kalite değeri belli bir eşik değerinin üzerinde olacak şekilde gelişiyorsa, o anki bağlantı yapısının uygun olduğu düşünülür ve değişmeden kalır. Aksi takdirde, bağlantı grupları yeniden düzenlenir. Dinamik bağlantı standard PSO'daki rasgele komşuluğa benzer gibi görünsede rasgele komşuluk parçacıklar üzerinde çalışırken dinamikbağlantı boyutlar üzerinde çalışır [8].

1.3.3. Dinamik Çok Sürülü PSO

Bazı lokal PSO türevleri her bir sürünün bir özel komşuluk olarak tanımlanabileceği çoklu sürülerden oluşan bir yapı kullanmaktadırlar. Bu sürüler önceden tanımlanabilecekleri gibi uzaklıpıa göre dinamik olarak da ayarlanabilirler. Bundan dolayı sürülerin özgürlükleri sınırlıdır. Dinamik çok sürülü PSO (Dynamic Multi-Swarm PSO, DMS-PSO)'da komşuluk topolojisi dinamiktir ve rasgele

atanmaktadır. DMS-PSO çok modlu problemler üzerinde iyi performans sergilerden lokal arama performansı düşüktür [9].

1.4. Dal ve Sınır Tekniği

Ayır ve bağla (branch and bound, BB) metodu tam sayı çözümlerin ağaç yapılarına dayalıdır. Ağacın olabildiğince kısa tutulması gerekmektedir. Potansiyel düğümlerin büyümesine izin verilir ve bunlar alt dallara ayrılır, alt ve üst sınırlarda alt dallar belirdiği zaman bunlarda bağlanır.

BB tekniğinin adımları aşağıda verilmektedir [7]:

Adım 1. S fasible bölge olmak üzere kümesi ile başla ve and M_0'ı M_i, $i = 1, 2, \ldots, m$ olacak şekilde alt kümlere böl.

Adım 2. f amaç fonksiyonu değeri olmak üzere her bir alt küme için $\beta(M_i) \leq inf\ f(M_i \cap S) \leq \alpha(M_i)$ şartını sağlayan alt $\beta(M_i)$ ve üst $\alpha(M_i)$ sınırları belirle.

Bu sınırlar $\beta \leq min\ f(S) \leq \alpha$ olacak şekilde $\beta = \min\limits_{i=1,2,\ldots,m} \beta(M_i)$ ve $\alpha = \min\limits_{i=1,2,\ldots,m} \alpha(M_i)$ olarak tanımlanır.

Adım 3. $\beta = \alpha$ ise dur, değil ise M_i alt kümelerinin bazılarını seçerek bunları parçala ve adımları tekrarla.

EK-2.

Fonksiyonlara Ait Veri Tabloları

Tablo EK-2.1. a and c parameters of Langerman Function

i	$a_{ij}, j = 1, ..., 10$										c_i
1	9.681	0.667	4.783	9.095	3.517	9.325	6.544	0.211	5.122	2.020	0.806
2	9.400	2.041	3.788	7.931	2.882	2.672	3.568	1.284	7.033	7.374	0.517
3	8.025	9.152	5.114	7.621	4.564	4.711	2.996	6.126	0.734	4.982	1.5
4	2.196	0.415	5.649	6.979	9.510	9.166	6.304	6.054	9.377	1.426	0.908
5	8.074	8.777	3.467	1.863	6.708	6.349	4.534	0.276	7.633	1.567	0.965
6	7.650	5.658	0.720	2.764	3.278	5.283	7.474	6.274	1.409	8.208	0.669
7	1.256	3.605	8.623	6.905	0.584	8.133	6.071	6.888	4.187	5.448	0.524
8	8.314	2.261	4.224	1.781	4.124	0.932	8.129	8.658	1.208	5.762	0.902
9	0.226	8.858	1.420	0.945	1.622	4.698	6.228	9.096	0.972	7.637	0.531
10	7.305	2.228	1.242	5.928	9.133	1.826	4.060	5.204	8.713	8.247	0.876
11	0.652	7.027	0.508	4.876	8.807	4.632	5.808	6.937	3.291	7.016	0.462
12	2.699	3.516	5.874	4.119	4.461	7.496	8.817	0.690	6.593	9.789	0.491
13	8.327	3.897	2.017	9.570	9.825	1.150	1.395	3.885	6.354	0.109	0.463
14	2.132	7.006	7.136	2.641	1.882	5.943	7.273	7.691	2.880	0.564	0.714
15	4.707	5.579	4.080	0.581	9.698	8.542	8.077	8.515	9.231	4.670	0.352
16	8.304	7.559	8.567	0.322	7.128	8.392	1.472	8.524	2.277	7.826	0.869
17	8.632	4.409	4.832	5.768	7.050	6.715	1.711	4.323	4.405	4.591	0.813
18	4.887	9.112	0.170	8.967	9.693	9.867	7.508	7.770	8.382	6.740	0.811
19	2.440	6.686	4.299	1.007	7.008	1.427	9.398	8.480	9.950	1.675	0.828
20	6.306	8.583	6.084	1.138	4.350	3.134	7.853	6.061	7.457	2.258	0.964
21	0.652	2.343	1.370	0.821	1.310	1.063	0.689	8.819	8.833	9.070	0.789
22	5.558	1.272	5.756	9.857	2.279	2.764	1.284	1.677	1.244	1.234	0.360
23	3.352	7.549	9.817	9.437	8.687	4.167	2.570	6.540	0.228	0.027	0.369
24	8.798	0.880	2.370	0.168	1.701	3.680	1.231	2.390	2.499	0.064	0.992
25	1.460	8.057	1.336	7.217	7.914	3.615	9.981	9.198	5.292	1.224	0.332
26	0.432	8.645	8.774	0.249	8.081	7.461	4.416	0.652	4.002	4.644	0.817
27	0.679	2.800	5.523	3.049	2.968	7.225	6.730	4.199	9.614	9.229	0.632
28	4.263	1.074	7.286	5.599	8.291	5.200	9.214	8.272	4.398	4.506	0.883
29	9.496	4.830	3.150	8.270	5.079	1.231	5.731	9.494	1.883	9.732	0.608
30	4.138	2.562	2.532	9.661	5.611	5.500	6.886	2.341	9.699	6.500	0.326

Tablo EK-2.2. A parameter of Fletcher-Powell Function

i	$A_{ij}, j = 1, ..., 20$																			
1	-79	56	-62	-9	92	48	-22	-34	-39	-40	-95	-69	-20	-66	-98	-66	-67	37	-83	-45
2	91	-9	-18	-59	99	-45	88	-14	-29	26	71	-65	19	45	88	18	-11	-81	-10	42
3	-38	8	-12	-73	40	26	-64	29	-82	-32	-89	-3	88	98	53	58	45	-39	34	-23
4	-78	-18	-49	65	66	-40	88	-95	-57	10	-98	-11	-16	-55	33	84	21	-43	45	100
5	-1	-43	93	-18	-76	-68	-42	22	46	-14	69	27	-12	-26	57	-13	0	1	56	17
6	34	-96	26	-56	-36	-85	-62	13	93	78	-43	96	77	65	-34	-52	82	18	-59	-55
7	52	-46	-69	99	-47	-72	-11	55	-55	91	-30	7	-35	23	-20	55	61	-39	-58	13
8	81	47	35	55	67	-13	33	14	83	-42	8	-45	-44	12	100	-9	-33	-11	21	14
9	5	-43	-45	46	56	-94	-62	52	66	55	-86	-29	-52	-71	-91	-46	27	-27	6	67
10	-50	66	-47	-75	89	-16	82	6	-85	-62	-30	31	-7	-75	-26	-24	46	-95	-71	-57
11	24	98	-50	68	-97	-64	-24	81	-59	-7	85	-92	2	61	52	-59	-91	74	-99	-95
12	-30	-63	-32	-90	-35	44	-64	57	27	87	-70	-39	-18	-89	99	40	14	-58	-5	-42
13	56	3	88	38	-14	-15	84	-9	65	-20	-75	-37	74	66	-44	72	74	90	-83	-40
14	84	1	73	43	84	-99	-35	24	-78	-58	47	-83	94	-86	-65	63	-22	65	50	-40
15	-21	-8	-48	68	-91	17	-52	-99	-23	43	-8	-5	-98	-17	-62	-79	60	-18	54	74
16	35	93	-98	-88	-8	64	15	69	-65	-86	58	-44	-9	-94	68	-27	-79	-67	-35	-56
17	-91	73	51	68	96	49	10	-13	-6	-23	50	-89	19	-67	36	-97	0	3	1	39
18	53	66	23	10	-33	62	-73	22	-65	37	-83	-65	59	-51	-56	98	-57	-11	-48	88
19	83	48	67	27	91	-33	-90	-34	39	-36	-68	17	-7	14	11	-10	96	98	-32	56
20	52	-52	-5	19	-25	15	-1	-11	8	-70	-4	-7	-4	-6	48	88	13	-56	85	-65

Tablo EK-2.3. B parameter of Fletcher-Powell Function

i	$B_{ij}, j = 1, ..., 21$																				
1	-65	-11	76	78	30	93	-86	-99	-37	52	-20	-10	-97	-71	16	9	-99	-84	90	-18	-94
2	59	67	49	-45	52	-33	-34	29	-39	-80	22	7	3	-19	-15	7	-83	-4	84	-60	-4
3	21	-23	-80	86	86	-30	39	-73	-91	5	83	-2	-45	-54	-81	-8	14	83	73	45	32
4	-91	-75	20	-64	-15	17	-89	36	-49	-2	56	-6	76	56	2	-68	-59	-70	48	2	24
5	-79	99	-31	-8	-67	-72	-43	-55	76	-57	1	-58	3	-59	30	32	57	29	66	50	-80
6	-89	-35	-55	75	15	-6	-53	-56	-96	87	-90	-93	52	-86	-38	-55	-53	94	98	4	-79
7	-76	45	74	12	-12	-69	2	71	75	-60	-50	23	0	6	44	-82	37	91	84	-15	-63
8	-50	-88	93	68	10	-13	84	-21	65	14	4	92	11	67	-18	-51	4	21	-38	75	-59
9	-23	-95	99	62	-37	96	27	69	-64	-92	-12	87	93	-19	-99	-92	-34	-77	17	-72	29
10	-5	-57	-30	-6	-96	75	25	-6	96	77	-35	-10	82	82	97	-39	-65	-8	34	72	65
11	85	-9	-14	27	-45	70	55	26	-87	-98	-25	-12	60	-45	-24	-42	-88	-46	-95	53	28
12	80	-47	38	-6	43	-59	91	-41	90	-63	11	-54	33	-61	74	96	21	-77	-58	-75	-9
13	-66	-98	-4	96	-11	88	-99	5	5	58	-53	52	-98	-97	50	49	97	-62	79	-10	-80
14	80	-95	82	5	-68	-54	64	-2	5	10	85	-33	-54	-30	-65	58	40	-21	-84	-66	-11
15	94	85	-31	37	-25	60	55	-13	48	-23	-50	84	-71	54	47	18	-67	-30	5	-46	53
16	-29	54	-10	-68	-54	-24	-16	21	32	33	-27	48	37	-61	97	45	-90	87	-95	85	67
17	76	-11	-48	38	-7	86	-55	51	26	8	-96	99	69	-84	41	78	-53	4	29	38	16
18	-8	48	95	47	39	-11	-72	-95	-17	33	65	96	-52	-17	-22	-15	-91	-41	-16	23	14
19	92	87	63	-63	-80	96	-62	71	-58	17	-89	-35	-96	-79	7	46	-74	88	93	-44	52
20	-21	35	16	-17	54	-22	-93	27	88	0	-67	94	-24	-30	-90	-5	-48	45	-90	32	-81
21	-86	31	-80	-79	-5	11	-20	9	52	-38	67	64	-49	23	-86	39	-97	76	10	81	20

Tablo EK-2.4. α parameter of Fletcher-Powell Function

$\alpha_j, j = 1, ..., 20$
-2.7910
2.5623
-1.0429
0.5097
-2.8096
1.1883
2.0771
-2.9926
0.0715
0.4142
-2.5010
1.7731
1.6473
0.4934
2.1038
-1.9930
0.3813
-2.2144
-2.5572
2.9449

Tablo EK-2.5. a parameter of FoxHoles Function

j	$a_{ij}, i = 1, 2$	
1	-32	-32
2	-16	-32
3	0	-32
4	16	-32
5	32	-32
6	-32	-16
7	-16	-16
8	0	-16
9	16	-16
10	32	-16
11	-32	0
12	-16	0
13	0	0
14	16	0
15	32	0
16	-32	16
17	-16	16
18	0	16
19	16	16
20	32	16
21	-32	32
22	-16	32
23	0	32
24	16	32
25	32	32

Tablo EK-2.6. a and b parameters of Kowalik Function

i	a_i	b_i^{-1}
1	0.1957	0.25
2	0.1947	0.5
3	0.1735	1
4	0.1600	2
5	0.0844	4
6	0.0627	6
7	0.0456	8
8	0.0342	10
9	0.0323	12
10	0.0235	14
11	0.0246	16

Tablo EK-2.7. a and c parameters of Shekel Functions

i	$a_{ij}, j = 1, ..., 6$				c_i
1	4	4	4	4	0.1
2	1	1	1	1	0.2
3	8	8	8	8	0.2
4	6	6	6	6	0.4
5	3	7	3	7	0.4
6	2	9	2	9	0.6
7	5	5	3	3	0.3
8	8	1	8	1	0.7
9	6	2	6	2	0.5
10	7	3.6	7	3.6	0.5

Tablo EK-2.8. a, c and p parameters of 3-parameter Hartman Function

i	$a_{ij}, j = 1, 2, 3$			c_i	$p_{ij}, j = 1, 2, 3$		
1	3	10	30	1	0.3689	0.1170	0.2673
2	0.1	10	35	1.2	0.4699	0.4387	0.7470
3	3	10	30	3	0.1091	0.8732	0.5547
4	0.1	10	35	3.2	0.03815	0.5743	0.8828

Tablo EK-2.9. a, c and p parameters of 6-parameter Hartman Function

i	$a_{ij}, j = 1, ..., 6$						c_i	$p_{ij}, j = 1, ..., 6$					
1	10	3	17	3.5	1.7	8	1	0.1312	0.1696	0.5569	.0124	0.8283	0.5886
2	0.05	10	17	0.1	8	14	1.2	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
3	3	3.5	1.7	10	17	8	3	0.2348	0.1415	0.3522	0.2883	0.3047	0.6650
4	17	8	0.05	10	0.1	14	3.2	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381

EK-3.

Karma Fonksiyonlar

3.1. F_1: Kaydırılmış Sphere Fonksiyonu

$$F_1(x) = \sum_{i=1}^{D} z_i^2 + f_{bias}$$
$$z = x - o$$
$$x = [x_1, x_2, \ldots, x_D]$$
$$o = [o_1, o_2, \ldots, o_D]$$
$$x \in [-100, 100]$$
$$x* = o, F_1(x*) = f_{bias} = -450$$

3.2. F_2: Kaydırılmış Schwefel's Problem 1.2

$$F_1(x) = \sum_{i=1}^{D} \sum_{j=1}^{i} z_j^2 + f_{bias}$$
$$z = x - o$$
$$x = [x_1, x_2, \ldots, x_D]$$
$$o = [o_1, o_2, \ldots, o_D]$$
$$x \in [-100, 100]$$
$$x* = o, F_2(x*) = f_{bias} = -450$$

3.3. F_3: Kaydırılmış Döndürülmüş Yüksek Dereceli Eliptik Fonksiyon

$$F_3(x) = \sum_{i=1}^{D} (10^6)^{(\frac{i-1}{D-1})} z_i^2 + f_{bias}$$
$$z = (x - o) * M$$
$$M : ortagonalmatris,$$
$$x = [x_1, x_2, \ldots, x_D]$$
$$o = [o_1, o_2, \ldots, o_D]$$
$$x \in [-100, 100]$$
$$x* = o, F_3(x*) = f_{bias} = -450$$

3.4. F_4:Shifted Schwefel's Problem 1.2 Gürültülü

$F_4(x) = (\sum_{i=1}^{D} \sum_{j=1}^{i} z_j^2) * (1 + 0.4|N(0,1)|) + f_{bias}$
$z = x - o$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D]$
$x \in [-100, 100]$
$x* = o, F_4(x*) = f_{bias} = -450$

3.5. F_5: Genişletilmiş Schwefel's Problem 2.6 Global Optimum Sınırlar Üzerinde

$f(x) = \max\{|x_1 + 2x_2 - 7|, |2x_1 + x_2 - 5|\}, i = 1, \ldots, n, x* = [1, 3], f(x*) = 0$
$F_5(x) = \max\{|Aix - Bi|\} + f_{bias}, i = 1, \ldots, D, x = [x_1, x_2, \ldots x_D]$
$A_i : DxD\ matris, aij$ random integer in the range$[-500, 500], \det(A) \neq 0$
$B_i = Ai * o, o_i \in [-100, 100]$
$o_i = 100 fori = 1, 2, \ldots, [D/4]$

3.6. F_6: Kaydırılmış Rosenbrock Fonksiyonu

$F_6(x) = \sum_{i=1}^{D-1} 100(z_i^2 - z_{i+1})^2 + (z_i - 1)^2 + f_{bias}$
$z = x - o + 1$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D]$
$x \in [-100, 100]$
$x* = o, F_6(x*) = f_{bias} = 390$

3.7. F_7: Kaydırılmış DöndürülmüşGriewank Fonksiyonu Parametre Sınırlamasız

$F_7(x) = \sum_{i=1}^{D} \frac{z_i^2}{4000} - \prod_{i=1}^{D} cos(\frac{z_i}{\sqrt{i}}) + 1 + f_{bias}$
$z = (x - o) * M$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D]$
$M :$ dorusaldnmmatrisi, conditionnumber $= 3$
$M : M'(1 + 0.3|N(0,1)|)$
Baslangicpoplasyonu $\in [0, 600]$
$x* = o, F_7(x*) = f_{bias} = -180$

3.8. F_8: Kaydırılmış Döndürülmüş Ackley Fonksiyonu Global Minimum Sınırlar Üzerinde

$$F_8(x) = -20exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2}) - exp(\frac{1}{D}\sum_{i=1}^{D} cos(2\pi z_i)) + 20 + e + f_{bias}$$
$z = (x - o) * M$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D], o_{2j-1} = -32, o_{2j} : rasgele$
$M : dorusaldnmmatrisi, conditionnumber = 100$
$x \in [-32, 32]$
$x* = o, F_8(x*) = f_{bias} = -140$

3.9. F_9: Kaydırılmış Rastrigin Fonksiyonu

$$F_9(x) = \sum_{i=1}^{D} z_i^2 - 10cos(2\pi z_i) + 10 + f_{bias}$$
$z = (x - o)$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D]$
$x \in [-5, 5]$
$x* = o, F_9(x*) = f_{bias} = -330$

3.10. F_{10}: Kaydırılmış Döndürülmüş Rastrigin Fonksiyonu

$$F_{10}(x) = \sum_{i=1}^{D} z_i^2 - 10cos(2\pi z_i) + 10 + f_{bias}$$
$z = (x - o) * M$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D], o_{2j-1} = -32, o_{2j} : rasgele$
$M : dorusaldnmmatrisi, conditionnumber = 2$
$x \in [-5, 5]$
$x* = o, F_{10}(x*) = f_{bias} = -330$

3.11. F_{11}: Kaydırılmış Döndürülmüş Weierstrass Fonksiyonu

$$F_{11}(x) = \sum_{i=1}^{D} \sum_{k=0} k_{max}([a^k cos(2\pi b^k (z_i + 0.5))]) - D \sum_{k=0}^{k_{max}} [a^k cos(\pi b^k)] + f_{bias}$$
$z = (x - o) * M$
$a = 0.5, b = 3, k_{max} = 20$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D]$
$M : dorusaldnmmatrisi, conditionnumber = 5$
$x \in [-0.5, 0.5]$
$x* = o, F_{11}(x*) = f_{bias} = 90$

3.12. F_{12}: Schwefel Problem 2.13

$F_{12}(x) = \sum_{i=1}^{D} (A_i - B_i(x))^2 + f_{bias}$
$A_i = \sum_{j=1}^{D} a_{ij} sin\alpha_j + b_{ij} cos\alpha_j$
$B_i = \sum_{j=1}^{D} a_{ij} sinx_j + b_{ij} cosx_j$
$\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_D], random[-\pi, \pi]$
a_{ij}, b_{ij} $[-100, 100]$ $araliginda\ rasgele\ tam\ sayi$
$x = [x_1, x_2, \ldots, x_D]$
$x \in [-\pi, \pi]$
$x* = \alpha, F_{12}(x*) = f_{bias} = -460$

3.13. F_{13}: Kaydırılmış Genişletilmiş Griewank + Rosenbrock Fonksiyonu

$F_{13}(x) = F_8(F_2(z_1, z_2)) + F_8(F_2(z_2, z_3)) + \ldots + F_8(F_2(z_{D-1}, z_D)) + F_8(F_2(z_D, z_1)) + f_{bias}$
$F_8 : Griewank = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} cos(\frac{x_i}{\sqrt{i}}) + 1$
$F_2 : Rosenbrock = \sum_{i=1}^{D-1} 100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2$
$z = x - o + 1$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D]$
$x \in [-3, 1]$
$x* = o, F_{13}(x*) = f_{bias} = -130$

3.14. F_{14}: Kaydırılmış Döndürülmüş Genişletilmiş Schaffer F6 Fonksiyonu

$F_{14}(x) = EF(z_1, z_2, \ldots, Z_D) = F(z_1, z_2) + F_2(z_2, z_3) + \ldots + F(z_{D-1}, z_D) + F(z_D, z_1) + f_{bias}$
$F(x, y) = 0.5 + \frac{(sin^2(\sqrt{x^2+y^2})) - 0.5}{(1+0.001(x^2+y^2))^2}$
$z = (x - o) * M$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D]$
$M : dorusaldnmmatrisi, conditionnumber = 3$
$x \in [-100, 100]$
$x* = o, F_{14}(x*) = f_{bias} = -300$

3.15. F_{15}: Karma Birleşim Fonksiyonu

$f_{1,2}(x) = Rastriginfonksiyonu$
$f_{3,4}(x) = Weierstrassfonksiyonu$
$f_{5,6}(x) = Griewankfonksiyonu$
$f_{7,8}(x) = Ackleyfonksiyonu$
$f_{9,10}(x) = Spherefonksiyonu$
$z = ((x - o_i)/\lambda_i) * M_i$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D]$
$M_i : Birimmatris$
$\lambda = [1, 1, 10, 10, 5/60, 5/60, 5/32, 5/32, 5/100, 5/100]$
$\sigma = [1, 1, \ldots, 1]$
$x \in [-5, 5]$
$x* = o_1, F_{15}(x*) = f_{bias} = -120$

3.16. F_{16}: Döndürülmüş Karma Birleşim Fonksiyonu

$f_{1,2}(x) = Rastriginfonksiyonu$
$f_{3,4}(x) = Weierstrassfonksiyonu$
$f_{5,6}(x) = Griewankfonksiyonu$
$f_{7,8}(x) = Ackleyfonksiyonu$
$f_{9,10}(x) = Spherefonksiyonu$
$z = ((x - o_i)/\lambda_i) * M_i$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D]$
$M_i : Dorusaldnmmatrisi, conditionnumber = 2$
$\lambda = [1, 1, 10, 10, 5/60, 5/60, 5/32, 5/32, 5/100, 5/100]$
$\sigma = [1, 1, \ldots, 1]$
$x \in [-5, 5]$
$x* = o_1, F_{16}(x*) = f_{bias} = 120$

3.17. F_{17}: Gürültülü Döndürülmüş Karma Birleşim Fonksiyonu

$f_{1,2}(x) = Rastriginfonksiyonu$
$f_{3,4}(x) = Weierstrassfonksiyonu$
$f_{5,6}(x) = Griewankfonksiyonu$
$f_{7,8}(x) = Ackleyfonksiyonu$
$f_{9,10}(x) = Spherefonksiyonu$
$z = ((x - o_i)/\lambda_i) * M_i$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D]$
$M_i : Dorusaldnmmatrisi, conditionnumber = 2$
$\lambda = [1, 1, 10, 10, 5/60, 5/60, 5/32, 5/32, 5/100, 5/100]$
$\sigma = [1, 1, \ldots, 1]$
$F_{17} = (F_16 - f_{bias}) * (1 + 0.2|N(0, 1)|) + f_{bias}$
$x \in [-5, 5]$
$x* = o_1, F_{17}(x*) = f_{bias} = 120$

3.18. F_{18}: Döndürülmüş Karma Birleşim Fonksiyonu

$f_{1,2}(x) = Ackleyfonksiyonu$
$f_{3,4}(x) = Rastriginfonksiyonu$
$f_{5,6}(x) = Spherefonksiyonu$
$f_{7,8}(x) = Weierstrassfonksiyonu$
$f_{9,10}(x) = Griewankfonksiyonu$
$z = ((x - o_i)/\lambda_i) * M_i$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D], o_{10} = [0, 0, \ldots, 0]$
$M_i : Dorusaldnmmatrisi, conditionnumber = [2323232030200300]$
$\lambda = [2 * 5/32, 5/32, 2 * 1, 1, 2 * 5/100, 5/100, 2 * 10, 10, 2 * 5/60, 5/60]$
$\sigma = [1, 2, 1.5, 1.5, 1, 1, 1.5, 1.5, 2, 2]$
$x \in [-5, 5]$
$x* = o_1, F_{18}(x*) = f_{bias} = 10$

3.19. F_{19}: Global Optimumu Dar Alanda Olan Döndürülmüş Karma Birleşim Fonksiyonu

$f_{1,2}(x) = Ackleyfonksiyonu$
$f_{3,4}(x) = Rastriginfonksiyonu$
$f_{5,6}(x) = Spherefonksiyonu$
$f_{7,8}(x) = Weierstrassfonksiyonu$
$f_{9,10}(x) = Griewankfonksiyonu$
$z = ((x - o_i)/\lambda_i) * M_i$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D], o_{10} = [0, 0, \ldots, 0]$
$M_i : Dorusaldnmmatrisi, conditionnumber = [2323232030200300]$
$\lambda = [0.1 * 5/32, 5/32, 2 * 1, 1, 2 * 5/100, 5/100, 2 * 10, 10, 2 * 5/60, 5/60]$
$\sigma = [0.1, 2, 1.5, 1.5, 1, 1, 1.5, 1.5, 2, 2]$
$x \in [-5, 5]$
$x* = o_1, F_{19}(x*) = f_{bias} = 10$

3.20. F_{20}: Global Optimumun Sınır Üzerinde Olduğu Döndürülmüş Karma Birleşim Fonksiyonu

$f_{1,2}(x) = Ackleyfonksiyonu$
$f_{3,4}(x) = Rastriginfonksiyonu$
$f_{5,6}(x) = Spherefonksiyonu$
$f_{7,8}(x) = Weierstrassfonksiyonu$
$f_{9,10}(x) = Griewankfonksiyonu$
$z = ((x - o_i)/\lambda_i) * M_i$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D], o_{10} = [0, 0, \ldots, 0], o_{1,2j} = 5, j = 1, 2, \ldots, D/2$
$M_i : Dorusaldnmmatrisi, conditionnumber = [2323232030200300]$
$\lambda = [0.1 * 5/32, 5/32, 2 * 1, 1, 2 * 5/100, 5/100, 2 * 10, 10, 2 * 5/60, 5/60]$
$\sigma = [0.1, 2, 1.5, 1.5, 1, 1, 1.5, 1.5, 2, 2]$
$x \in [-5, 5]$
$x* = o_1, F_{20}(x*) = f_{bias} = 10$

3.21. F_{21}: Döndürülmüş Karma Birleşim Fonksiyonu

$f_{1,2}(x) = DndrlmGeniletilmiSchafferFonksiyonu$
$f_{3,4}(x) = Rastriginfonksiyonu$
$f_{5,6}(x) = F8F2Fonksiyonu(Griewank + Rosenbrock)$
$f_{7,8}(x) = Weierstrassfonksiyonu$
$f_{9,10}(x) = Griewankfonksiyonu$
$z = ((x - o_i)/\lambda_i) * M_i$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D]$
$M_i : ortogonalmatrisler, conditionnumber = [2323232030200300]$
$\lambda = [5 * 5/100, 5/100, 5 * 1, 1, 5 * 1, 1, 5 * 10, 10, 5 * 5/200, 5/200]$
$\sigma = [1, 1, 1, 1, 1, 2, 2, 2, 2, 2]$
$x \in [-5, 5]$
$x* = o_1, F_{21}(x*) = f_{bias} = 360$

3.22. F_{22}: Yüksek Dereceli Döndürülmüş Karma Birleşim Fonksiyonu

$f_{1,2}(x) = DndrlmGeniletilmiSchafferFonksiyonu$
$f_{3,4}(x) = Rastriginfonksiyonu$
$f_{5,6}(x) = F8F2Fonksiyonu(Griewank + Rosenbrock)$
$f_{7,8}(x) = Weierstrassfonksiyonu$
$f_{9,10}(x) = Griewankfonksiyonu$
$z = ((x - o_i)/\lambda_i) * M_i$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D]$
$M_i : ortogonalmatrisler, conditionnumber = [10205010020010002000300040005000]$
$\lambda = [5 * 5/100, 5/100, 5 * 1, 1, 5 * 1, 1, 5 * 10, 10, 5 * 5/200, 5/200]$
$\sigma = [1, 1, 1, 1, 1, 2, 2, 2, 2, 2]$
$x \in [-5, 5]$
$x* = o_1, F_{22}(x*) = f_{bias} = 360$

3.23. F_{23}: Süreksiz Döndürülmüş Karma Birleşim Fonksiyonu

$f_{1,2}(x) = DndrlmGeniletilmiSchafferFonksiyonu$
$f_{3,4}(x) = Rastriginfonksiyonu$
$f_{5,6}(x) = F8F2Fonksiyonu(Griewank + Rosenbrock)$
$f_{7,8}(x) = Weierstrassfonksiyonu$
$f_{9,10}(x) = Griewankfonksiyonu$
$z = ((x - o_i)/\lambda_i) * M_i$
$x_j = \begin{cases} x_j & |xj - o1j| < \frac{1}{2} \\ round(2xj)/2 & |xj - o1j| \geq \frac{1}{2} \end{cases}$
$round(x) = \begin{cases} a-1 & if \quad (x \leq 0)\&(b \geq 0.5) \\ a & if \quad b < 0.5 \\ a+1 & if \quad (x > 0)\&(b \geq 0.5) \end{cases}$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D]$
$M_i : ortogonalmatrisler, conditionnumber = [2323232030200300]$
$\lambda = [5 * 5/100, 5/100, 5 * 1, 1, 5 * 1, 1, 5 * 10, 10, 5 * 5/200, 5/200]$
$\sigma = [1, 1, 1, 1, 1, 2, 2, 2, 2, 2]$
$x \in [-5, 5]$
$x* = o_1, F_{23}(x*) = f_{bias} = 360$

3.24. F_{24}: Döndürülmüş Karma Birleşim Fonksiyonu

$f_1(x) = WeierstrassFonksiyonu$
$f_2(x) = DndrlmGeniletilmiSchafferFonksiyonu$
$f_3(x) = F8F2Fonksiyonu(Griewank + Rosenbrock)$
$f_4(x) = AckleyFonksiyonu$
$f_5(x) = RastriginFonksiyonu$
$f_6(x) = GriewankFonksiyonu$
$f_7(x) = SreksizGeniletilmiSchafferFonksiyonu$
$f_8(x) = SreksizRastriginFonksiyonu)$
$f_9(x) = YksekdereceliEliptikFonksiyonu$
$f_{10}(x) = GrltlSphereFonksiyonu$
$z = ((x - o_i)/\lambda_i) * M_i$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D]$
$M_i : ortogonalmatrisler, conditionnumber = [100503010554322]$
$\lambda = [10, 5/20, 1, 5/32, 1, 5/100, 5/50, 1, 5/100, 5/100]$
$\sigma = [2, 2, \ldots, 2]$
$x \in [-5, 5]$
$x* = o_1, F_{24}(x*) = f_{bias} = 260$

3.25. F_{25}: Araştırma Bölgesi Sınırlamasız Döndürülmüş Karma Birleşim Fonksiyonu

$f_1(x) = Weierstrass Fonksiyonu$
$f_2(x) = Dndrlm Geniletilmi Schaffer Fonksiyonu$
$f_3(x) = F8F2 Fonksiyonu(Griewank + Rosenbrock)$
$f_4(x) = Ackley Fonksiyonu$
$f_5(x) = Rastrigin Fonksiyonu$
$f_6(x) = Griewank Fonksiyonu$
$f_7(x) = Sreksiz Geniletilmi Schaffer Fonksiyonu$
$f_8(x) = Sreksiz Rastrigin Fonksiyonu)$
$f_9(x) = Yksekdereceli Eliptik Fonksiyonu$
$f_{10}(x) = Grltl Sphere Fonksiyonu$
$z = ((x - o_i)/\lambda_i) * M_i$
$x = [x_1, x_2, \ldots, x_D]$
$o = [o_1, o_2, \ldots, o_D]$
$M_i : ortogonal matrisler, condition number = [100503010554322]$
$\lambda = [10, 5/20, 1, 5/32, 1, 5/100, 5/50, 1, 5/100, 5/100]$
$\sigma = [2, 2, \ldots, 2]$
$Baslangic x \in [2, 5]$
$x* = o_1, F_{25}(x*) = f_{bias} = 260$

EK-4.

Sınırlamalı Test Problemleri

4.1. g01:

$$Minimize\ f(\vec{x}) = 5\sum_{i=1}^{4} x_i - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i$$

subject to

$$g_1(\vec{x}) = 2x_1 + 2x_2 + x_{10} + x_{11} - 10 \leq 0$$
$$g_2(\vec{x}) = 2x_1 + 2x_3 + x_{10} + x_{12} - 10 \leq 0$$
$$g_3(\vec{x}) = 2x_2 + 2x_3 + x_{11} + x_{12} - 10 \leq 0$$
$$g_4(\vec{x}) = -8x_1 + x_{10} \leq 0$$
$$g_5(\vec{x}) = -8x_2 + x_{11} \leq 0$$
$$g_6(\vec{x}) = -8x_3 + x_{12} \leq 0$$
$$g_7(\vec{x}) = -2x_4 - x_5 + x_{10} \leq 0$$
$$g_8(\vec{x}) = -2x_6 - x_7 + x_{11} \leq 0$$
$$g_9(\vec{x}) = -2x_8 - x_9 + x_{12} \leq 0$$

$0 \leq x_i \leq 1\ (i = 1, \ldots, 9, 13)\ ,\ 0 \leq x_i \leq 100\ (i = 10, 11, 12)$. global optimum: $x^* = (1, 1, 1, 1, 1, 1, 1, 1, 1, , 3, 3, 3, 1)$, $f(x^*) = -15$.

$g_1,\ g_2,\ g_3,\ g_4,\ g_5$ ve g_6 are aktif.

4.2. g02:

$$Maximize\ f(\vec{x}) = \left| \frac{\sum_{i=1}^{n} \cos^4(x_i) - 2\prod_{i=1}^{n} \cos^2(x_i)}{\sqrt{\sum_{i=1}^{n} i x_i^2}} \right|$$

subject to

$$g_1(\vec{x}) = 0.75 - \prod_{i=1}^{n} x_i \leq 0$$
$$g_2(\vec{x}) = \sum_{i=1}^{n} x_i - 7.5n \leq 0$$

n=20 ve $0 \leq x_i \leq 10$ $(i = 1, \ldots, n)$. Bilinen global maksimum: $x_i^* = 1/\sqrt{n}$ $(i = 1, \ldots, n)$, $f(x^*) = 0.803619$. g_1 aktifliğe yakın $(g_1 = -10^{-8})$

4.3. g03:

$$Maximize\ f(\vec{x}) = (\sqrt{n})^n \prod_{i=1}^{n} x_i$$

subject to

$$h(\vec{x}) = \sum_{i=1}^{n} x_i^2 - 1 = 0$$

n=10 ve $0 \leq x_i \leq 1$ $(1 = 1, \ldots, n)$. Global maksimum: $x_i^* = 1/\sqrt{(}n)$ $(i = 1, \ldots, n)$, $f(x^*) = 1$

4.4. g04:

$$Minimize\ f(\vec{x}) = 5.3578547x_3^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141$$

subject to

$$\begin{aligned}
g_1(\vec{x}) &= 85.334407 + 0.0056858x_2x_5 \\
&\quad + 0.0006262x_1x_4 - 0.0022053x_3x_5 \\
&\quad - 92 \leq 0 \\
g_2(\vec{x}) &= -85.334407 - 0.0056858x_2x_5 \\
&\quad - 0.0006262x_1x_4 + 0.0022053x_3x_5 \leq 0 \\
g_3(\vec{x}) &= 80.51249 + 0.0071317x_2x_5 \\
&\quad + 0.0029955x_1x_2 - 0.0021813x_3^2 \\
&\quad - 110 \leq 0 \\
g_4(\vec{x}) &= -80.51249 - 0.0071317x2x5 \\
&\quad + 0.0029955x_1x_2 - 0.0021813x_3^2 \\
&\quad + 90 \leq 0 \\
g_5(\vec{x}) &= 9.300961 - 0.0047026x_3x_5 \\
&\quad - 0.0012547x_1x_3 - 0.0019085x_3x_4 \\
&\quad - 25 \leq 0 \\
g_6(\vec{x}) &= -9.300961 - 0.0047026x_3x_5 \\
&\quad - 0.0012547x_1x_3 - 0.0019085x_3x_4 \\
&\quad + 20 \leq 0
\end{aligned}$$

$78 \leq x_1 \leq 102,\ 33 \leq x_2 \leq 45,\ 27 \leq x_i\ \leq 45$

$(i = 3, 4, 5)$. Optimum çözüm:

$x^* = (78, 33, 29.995256025682, 45, 36.775812905788)$, $f(x^*) = -30665.539$. g_1 ve g_6 sınırlamaları aktif.

4.5. g05:

$$Minimize\ f(\vec{x}) = 3x_1 + 0.000001x_1^3 + 2x_2 + \left(\frac{0.000002}{3}\right) x_2^3$$

subject to

$$\begin{aligned}
g_1(\vec{x}) &= -x4 + x3 - 0.55 \leq 0 \\
g_2(\vec{x}) &= -x3 + x4 - 0.55 \leq 0 \\
h_1(\vec{x}) &= 1000\sin(-x_3 - 0.25) \\
&\quad + 1000\sin(-x_4 - 0.25) + 894.8 \\
&\quad - x_1 = 0 \\
h_2(\vec{x}) &= 1000\sin(x_3 - 0.25) \\
&\quad + 1000\sin(x_3 - x_4 - 0.25) + 894.8 \\
&\quad - x_2 = 0 \\
h_3(\vec{x}) &= 1000\sin(x_4 - 0.25) \\
&\quad + 1000\sin(x_4 - x_3 - 0.25) \\
&\quad + 1294.8 = 0
\end{aligned}$$

$0 \leq x_1 \leq 1200,\ 0 \leq x_2 \leq 1200, -0.55 \leq x_3 \leq 0.55$, ve $-0.55 \leq x_4 \leq 0.55$. Bilinen en iyi çözüm: $x^* = (679.9453, 1026.067, 0.1188764, -0.3962336)$, $f(x^*) = 5126.4981$.

4.6. g06:

$$Minimize\ f(\vec{x}) = (x_1 - 10)^3 + (x_2 - 20)^3$$

subject to

$$\begin{aligned}
g_1(\vec{x}) &= -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \leq 0 \\
g_2(\vec{x}) &= (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \leq 0
\end{aligned}$$

$13 \leq x_1 \leq 100$ ve $0 \leq x_2 \leq 100$. Optimum çözüm: $x^* = (14.095, 0.84296)$, $f(x^*) = -6961.81388$. Her iki sınırlama da aktif.

4.7. g07:

$$\begin{aligned} Minimize\ f(\vec{x}) &= x_1^2 + x_2^2 + x_1x_2 - 14x_1 \\ &\quad - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 \\ &\quad + (x_5 - 3)^2 + 2(x_6 - 1)^2 + 5x_7^2 \\ &\quad + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 \\ &\quad + (x_{10} - 7)^2 + 45 \end{aligned}$$

subject to

$$\begin{aligned} g_1(\vec{x}) &= -105 + 4x1 + 5x2 - 3x7 + 9x8 \leq 0 \\ g_2(\vec{x}) &= 10x_1 - 8x_2 - 17x_7 + 2x_8 \leq 0 \\ g_3(\vec{x}) &= -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \leq 0 \\ g_4(\vec{x}) &= 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x4 \\ &\quad - 120 \leq 0 \\ g_5(\vec{x}) &= 5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 \leq 0 \\ g_6(\vec{x}) &= x_1^2 + 2(x_2 - 2)^2 - 2x_1x_2 + 14x_5 \\ &\quad - 6x_6 \leq 0 \\ g_7(\vec{x}) &= 0.5(x1 - 8)2 + 2(x2 - 4)2 + 3x_5^2 - x6 \\ &\quad - 30 \leq 0 \\ g_8(\vec{x}) &= -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} \leq 0 \end{aligned}$$

$-10 \leq x_i \leq 10\ (i = 1, \ldots, 10)$. Global optimum:

$x^* = (2.171996, 2.363683, 8.773926, 5.095984, 0.9906548,$

$1.430574, 1.321644, 9.828726, 8.280092, 8.375927),$

$f(x^*) = 24.3062091$. $g_1,\ g_2,\ g_3,\ g_4,\ g_5$ ve g_6 sınırlamaları aktif.

4.8. g08:

$$Maximize\ f(\vec{x}) = \frac{\sin^3(2\pi x_1)\sin(2\pi x_2)}{x_1^3(x_1 + x_2)}$$

subject to

$$\begin{aligned} g_1(\vec{x}) &= x_1^2 - x_2 + 1 \leq 0 \\ g_2(\vec{x}) &= 1 - x_1 + (x_2 - 4)^2 \leq 0 \end{aligned}$$

$0 \leq x_i \leq 10\ (i = 1, 2)$. Optimum çözüm:$x^* = (1.2279713, 4.2453733)$,

$f(x^*) = 0.0095825$.

4.9. g09:

$$\begin{aligned} Minimize\ f(\vec{x}) &= (x_1-10)^2 + 5(x_2-12)^2 \\ &+ x_3^4 + 3(x_4-11)^2 + 10x_5^6 \\ &+ 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7 \end{aligned}$$

subject to

$$\begin{aligned} g_1(\vec{x}) &= -127 + 2x_1^2 + 3x_2^4 + x3 + 4x_4^2 \\ &+ 5x5 \leq 0 \\ g_2(\vec{x}) &= -282 + 7x_2 + 3x_2 + 10x_3^2 + x_4 \\ &- x_5 \leq 0 \\ g_3(\vec{x}) &= -196 + 23x_1 + x_2^2 + 6x_6^2 - 8x_7 \leq 0 \\ g_4(\vec{x}) &= 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \leq 0 \end{aligned}$$

$-10 \leq x_i \leq 10,\ (i = 1, \ldots, 7)$.

Global çözüm:

$x^* = (2.330499, 1.951372, -0.4775414,$

$4.365726, -0.6244870, 1.038131, 1.594227)$,

$f(x^*) = 680.6300573$. g_1 ve g_4 sınırlamaları aktif. are active.

4.10. g10:

$$Minimize\ f(\vec{x}) = x_1 + x_2 + x_3$$

subject to

$$\begin{aligned} g_1(\vec{x}) &= -1 + 0.0025(x_4 + x_6) \leq 0 \\ g_2(\vec{x}) &= -1 + 0.0025(x_5 + x_7 - x_4) \leq 0 \\ g_3(\vec{x}) &= -1 + 0.01(x_8 - x_5) \leq 0 \\ g_4(\vec{x}) &= -x_1x_6 + 833.33252x_4 + 100x_1 \\ &- 83.333333 \leq 0 \\ g_5(\vec{x}) &= -x_2x_7 + 1250x_5 + x_2x_4 \\ &- 1250x_4 \leq 0 \\ g_6(\vec{x}) &= -x_3x_8 + 1250000 + x_3x_5 - 2500x_5 \leq 0 \end{aligned}$$

$100 \leq x_1 \leq 10000,\ 1000 \leq x_i \leq 10000, (i = 2, 3),\ 10 \leq x_i \leq 1000, (i = 4, \ldots, 8)$.Global optimum:

$x^* = (579.19, 1360.13, 5109.92, 182.0174, 295.5985, 217.9799,$

$286.40, 395.5979)$, $f(x^*) = 7049.25$. $g_1,\ g_2$ ve g_3 aktif.

4.11. g11:

$Minimize\ f(\vec{x}) = x_1^2 + (x_2 - 1)^2$

subject to

$h(\vec{x}) = x_2 - x_1^2 = 0$

where $-1 \leq x_1 \leq 1$, $-1 \leq x_2 \leq 1$. Optimum çözüm:$x^* = (\pm 1/\sqrt{(}2), 1/2)$, $f(x^*) = 0.75$.

4.12. g12:

$Maximize\ f(\vec{x}) = \frac{100-(x_1-5)^2-(x_2-5)^2-(x_3-5)^2}{100}$

subject to

$g_1(\vec{x}) = (x_i - p)^2 + (x_2 - q)^2 + (x_3 - r)^2 - 0.0625 \leq 0$

$0 \leq x_i \leq 10$, $(i = 1, 2, 3)$ ve p, r, q=$1, \ldots, 9$. Global optimum:$x^* = (5, 5, 5)$, $f(x^*) = 1$.

4.13. g13:

$Minimize\ f(\vec{x}) = e^{x_1 x_2 x_3 x_4 x_5}$

subject to

$h_1(\vec{x}) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 = 0$
$h_2(\vec{x}) = x_2 x_3 - 5 x_4 x_5 = 0$
$h_3(\vec{x}) = x_1^3 + x_2^3 + 1 = 0$

$-2.3 \leq x_i \leq 2.3$ $(i = 1, 2)$, $-3.2 \leq x_i \leq 3.2$,$(i = 3, 4, 5)$. Global Optimum:
$x^* = (-1.717143, 1.5957091, -0.736413, -0.763645)$, $f(x^*) = 0.0539498$.

EK-5.

ABC Algoritmasının Kodları

```
clear all close all clc



% Set ABC Control Parameters
ABCOpts = struct( ...
    ... %Variables of the ABC
    'ColonySize',  20, ...
    'MaxCycles', 500,...
    'ErrGoal',     1e-20, ...
    'Dim',       10 , ...
    'Limit',   100, ...
    'lb', -30, ...
    'ub',  30, ...
    'ObjFun' , 'rosenbrock', ...
    'RunTime',1);

GlobalMins=zeros(ABCOpts.RunTime,ABCOpts.MaxCycles);

for r=1:ABCOpts.RunTime

% Initialise population
Range = repmat((ABCOpts.ub-ABCOpts.lb),[ABCOpts.ColonySize
```

```
ABCOpts.Dim]); Lower = repmat(ABCOpts.lb, [ABCOpts.ColonySize
ABCOpts.Dim]); Colony = rand(ABCOpts.ColonySize,ABCOpts.Dim) .*
Range + Lower;

Employers=Colony(1:(ABCOpts.ColonySize/2),:);



%evaluate and calculate fitness
ObjEmp=feval(ABCOpts.ObjFun,Employers);
FitEmp=calculateFitness(ObjEmp);

%set initial values of Bas
Bas=zeros(1,(ABCOpts.ColonySize/2));



GlobalMin=ObjEmp(find(ObjEmp==min(ObjEmp),end));
GlobalParams=Employers(find(ObjEmp==min(ObjEmp),end),:);

Cycle=1; while ((Cycle <= ABCOpts.MaxCycles)),

    %%%%% Employer
    %Employers2=produceOffSpring(ABCOpts,Employers);
    Employers2=Employers;
    for i=1:ABCOpts.ColonySize/2
        Param2Change=fix(rand*ABCOpts.Dim)+1;
        neighbour=fix(rand*(ABCOpts.ColonySize/2))+1;
            while(neighbour==i)
                neighbour=fix(rand*(ABCOpts.ColonySize/2))+1;
            end;
        Employers2(i,Param2Change)=Employers(i,Param2Change)+ ...
        (Employers(i,Param2Change)-Employers(neighbour,Param2Change))* ...
        (rand-0.5)*2;
```

```
          if (Employers2(i,Param2Change)<ABCOpts.lb)
              Employers2(i,Param2Change)=ABCOpts.lb;
          end;
          if (Employers2(i,Param2Change)>ABCOpts.ub)
              Employers2(i,Param2Change)=ABCOpts.ub;
          end;

end;

    ObjEmp2=feval(ABCOpts.ObjFun,Employers2);
    FitEmp2=calculateFitness(ObjEmp2);
    [Employers ObjEmp FitEmp Bas]=GreedySelection(Employers,Employers2, ...
    ObjEmp,ObjEmp2,FitEmp,FitEmp2,Bas,ABCOpts);

    %Normalize
    NormFit=FitEmp./sum(FitEmp);%

    %%% Onlooker
    %[Employers ObjEmp FitEmp Bas]=OnlookerProbSelection(ABCOpts,NormFit,...
    Employers,ObjEmp,FitEmp,Bas,xd);

Employers2=Employers; i=1; t=0; while(t<ABCOpts.ColonySize/2)
    if(rand<NormFit(i))
        t=t+1;
        Param2Change=fix(rand*ABCOpts.Dim)+1;
        neighbour=fix(rand*(ABCOpts.ColonySize/2))+1;
            while(neighbour==i)
                neighbour=fix(rand*(ABCOpts.ColonySize/2))+1;
            end;
         Employers2(i,:)=Employers(i,:);
         Employers2(i,Param2Change)=Employers(i,Param2Change)+ ...
         (Employers(i,Param2Change)-Employers(neighbour,Param2Change))*...
```

```
          (rand-0.5)*2;
          if (Employers2(i,Param2Change)<ABCOpts.lb)
              Employers2(i,Param2Change)=ABCOpts.lb;
          end;
         if (Employers2(i,Param2Change)>ABCOpts.ub)
             Employers2(i,Param2Change)=ABCOpts.ub;
          end;
    ObjEmp2=feval(ABCOpts.ObjFun,Employers2);
    FitEmp2=calculateFitness(ObjEmp2);
    [Employers ObjEmp FitEmp Bas]=GreedySelection(Employers,Employers2,...
    ObjEmp,ObjEmp2,FitEmp,FitEmp2,Bas,ABCOpts,i);

   end;

    i=i+1;
    if (i==(ABCOpts.ColonySize/2)+1)
        i=1;
    end;
end;


    %%%Memorize Best
 CycleBestIndex=find(FitEmp==max(FitEmp));
 CycleBestIndex=CycleBestIndex(end);
 CycleBestParams=Employers(CycleBestIndex,:);
 CycleMin=ObjEmp(CycleBestIndex);

 if CycleMin<GlobalMin
       GlobalMin=CycleMin;
       GlobalParams=CycleBestParams;
 end
```

```
 GlobalMins(r,Cycle)=GlobalMin;
    %% Scout

ind=find(Bas==max(Bas)); ind=ind(end); if (Bas(ind)>ABCOpts.Limit)
Bas(ind)=0;
Employers(ind,:)=(ABCOpts.ub-ABCOpts.lb)*(0.5-rand(1,ABCOpts.Dim))*2;
%message=strcat('burada',num2str(ind))
end; ObjEmp=feval(ABCOpts.ObjFun,Employers);
FitEmp=calculateFitness(ObjEmp);



    fprintf('Cycle=%d ObjVal=%g\n',Cycle,GlobalMin);

    Cycle=Cycle+1;

end % End of ABC

end; %end of runs

semilogy(GlobalMins(end,:));


function fFitness=calculateFitness(fObjV)
fFitness=zeros(size(fObjV)); ind=find(fObjV>=0);
fFitness(ind)=1./(fObjV(ind)+1); ind=find(fObjV<0);
fFitness(ind)=1+abs(fObjV(ind));



function [Colony Obj Fit oBas]= GreedySelection ...
(Colony1,Colony2,ObjEmp,ObjEmp2,FitEmp,FitEmp2,fbas,ABCOpts,i)%
```

```
oBas=fbas; Obj=ObjEmp; Fit=FitEmp; Colony=Colony1; if (nargin==8)
for ind=1:size(Colony1,1)
    if (FitEmp2(ind)>FitEmp(ind))
        oBas(ind)=0;
        Fit(ind)=FitEmp2(ind);
        Obj(ind)=ObjEmp2(ind);
        Colony(ind,:)=Colony2(ind,:);
    else
        oBas(ind)=fbas(ind)+1;
        Fit(ind)=FitEmp(ind);
        Obj(ind)=ObjEmp(ind);
        Colony(ind,:)=Colony1(ind,:);
    end;
end; %for
end; %if
if(nargin==9)
    ind=i;
    if (FitEmp2(ind)>FitEmp(ind))
        oBas(ind)=0;
        Fit(ind)=FitEmp2(ind);
        Obj(ind)=ObjEmp2(ind);
        Colony(ind,:)=Colony2(ind,:);
    else
        oBas(ind)=fbas(ind)+1;
        Fit(ind)=FitEmp(ind);
        Obj(ind)=ObjEmp(ind);
        Colony(ind,:)=Colony1(ind,:);
    end;
end;
```

ÖZGEÇMİŞ

Bahriye Akay 1981 yılında Kayseri'de doğdu. İlkokulu Ahmet Eskiyapan İlkokulu'nda, orta okulu Kocasinan Atatürk Lisesi'nde, liseyi Kayseri Sümer Lisesi'nde bitirdikten sonra 1999 yılında Erciyes Üniversitesi Bilgisayar Mühendisliği Bölümünü kazandı. 2003 yılında bu lisans programından mezun olduktan sonra Erciyes Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'ında yüksek lisans eğitimine başladı. 2004 yılında Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda araştırma görevlisi olarak atandı. 2005 yılında yüksek mühendis ünvanını aldı ve aynı yıl Erciyes Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'ında doktora eğitimine başladı. Doktora eğitimi sırasında tez konusu yapay arı kolonisi algoritması ile ilgili bilimsel yayınlar ve tebliğler hazırlayarak sunmuştur.

Adres	:	Barbaros Mahallesi Hakimiyet Caddesi No:1-9 38080 Kocasinan KAYSERİ
Telefon	:	+90 352 338 54 21
e-posta	:	bahriye@erciyes.edu.tr