DİNAMİK KOMŞULUKLU EŞZAMANSIZ DAĞINIK PARÇACIK SÜRÜ ENİYİLEME YÖNTEMİ VE ÇOK ROBOTLU ARAMA GÖREVİNDE UYGULANMASI

SALİH BURAK AKAT

YÜKSEK LİSANS TEZİ

ELEKTRİK VE ELEKTRONİK MÜHENDİSLİĞİ

TOBB EKONOMİ VE TEKNOLOJİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Şubat 2009

ANKARA

Fen Bilimleri Enstitü onayı

_____

Prof. Dr. Yücel Ercan

Müdür

Bu tezin Yüksek Lisans derecesinin tüm gereksinimlerini sağladığını onaylarım.

_____

Doç. Dr. M. Önder Efe

Anabilim Dalı Başkanı

SALİH BURAK AKAT tarafından hazırlanan DİNAMİK KOMŞULUKLU EŞZAMANSIZ DAĞINIK PARÇACIK SÜRÜ ENİYİLEME YÖNTEMİ VE ÇOK ROBOTLU ARAMA GÖREVİNDE UYGULANMASI adlı bu tezin Yüksek Lisans tezi olarak uygun olduğunu onaylarım.

_____

Doç. Dr. Veysel Gazi

Tez Danışmanı

Tez Jüri Üyeleri

Başkan : Doç. Dr. M.Önder Efe _____

Üye : Doç. Dr. Veysel Gazi _____

Üye : Yrd. Doç. Dr. Murat Özbayoğlu _____

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, ayrıca tez yazım kurallarına uygun olarak hazırlanan bu çalışmada orijinal olmayan her türlü kaynağa eksiksiz atıf yapıldığını bildiririm.

SALİH BURAK AKAT

Üniversitesi : **TOBB Ekonomi ve Teknoloji Üniversitesi**
Enstitüsü : **Fen Bilimleri Enstitüsü**
Anabilim Dalı : **Elektrik ve Elektronik Mühendisliği**
Tez Danışmanı : **Doç. Dr. Veysel Gazi**
Tez Türü ve Tarihi : **Yüksek Lisans - Şubat 2009**

SALİH BURAK AKAT

DİNAMİK KOMŞULUKLU EŞZAMANSIZ DAĞINIK PARÇACIK SÜRÜ ENİYİLEME YÖNTEMİ VE ÇOK ROBOTLU ARAMA GÖREVİNDE UYGULANMASI

ÖZET

Bu tez çalışmasında dinamik komşuluklu, eşzamansız ve dağınık parçacık sürü eniyileme yöntemi ve bu yöntemin çok robotlu arama görevinde uygulanması çalışılmıştır. Tezde çalışılan dinamik komşuluklu parçacık sürü eniyileme yöntemi parçacık komşuluklarının zamanla dinamik olarak değişmesine izin verilmektedir. Bu bakış açısı yöntemin paralel ve dağınık uygulamalarında çeşitli avantajlar sağlamaktadır. Yöntemin dinamik komşuluklu biçimi çeşitli denektaşı fonksiyonların farklı komşuluk dinamikleri altında eniyilenmesi ile sınanmıştır. Öte yandan yöntemin eşzamansız ve dağıtık biçimi parçacık sürü eniyileme yönteminin paralel ve dağınık olarak çalışmasına olanak sağlamaktadır. Yöntemin belirtilen biçiminde parçacıkların bağımsız zaman anlarında bilgi paylaşımında bulunmalarına, tahminlerini güncellemelerine ve parçacık komşuluklarının zamanla dinamik olarak değişmesine izin verilmiştir. Tek bir işlemci ve bir bilgisayar ağında bulunan birçok işlemci ile benzetimler gerçekleştirilmiştir. Yöntemin dinamik komşuluklu, eşzamansız ve dağınık biçimi sınırlı haberleşme/algılama yeteneğine sahip erkinlerden oluşan çok erkinli bir sistemin bilinmeyen bir ortamda arama görevinde kullanılmıştır. Erkinler eşzamansız olarak bilgi paylaşımı ve konum güncellemeleri gerçekleştirmekte ve haberleşen erkinlerin komşuluk yapısı dinamik olarak zamanla değişmektedir. Gerçekçi benzetim yazılımı ile benzetimler ve gerçek robotlar ile uygulamalar geliştirilen yöntemin verimliliğinin göstermek için gerçekleştirilmiştir.

Anahtar Kelimeler: Parçacık Sürü Eniyileme Yöntemi, Dinamik Parçacık Komşuluk Yapısı, Eşzamansızlık, Dağınık Sistemler, Çok Erkinli Sistemler ile Arama Görevi

University : **TOBB University of Economics and Technology**
Institute : **Institute of Natural and Applied Sciences**
Science Programme : **Electrical and Electronics Engineering**
Supervisor : **Associate Professor Veysel Gazi**
Degree Awarded and Date : **M.S. - February 2009**

SALİH BURAK AKAT

DECENTRALIZED ASYNCHRONOUS PARTICLE SWARM OPTIMIZATION WITH DYNAMIC NEIGHBORHOOD TOPOLOGY AND ITS IMPLEMENTATION ON MULTI ROBOT SEARCH TASK

ABSTRACT

In this thesis decentralized asynchronous particle swarm optimization (PSO) with dynamic neighborhood topology and its implementation to a search task of a multi-agent system were studied. Particle swarm optimization with dynamic neighborhood topology studied in this thesis allows the neighbors of particles or basically the neighborhood topology to change dynamically with time. Such a view of the algorithm is advantageous for its parallel and distributed implementations. The algorithm with dynamic neighborhood topology was tested on various benchmark functions under different neighborhood dynamics. Decentralized asynchronous realization of particle swarm optimization algorithm is suitable for parallel and distributed implementations. Such a version of the algorithm allows particles to exchange information and update their estimates at totally independent time instants and dynamically change neighborhood topology of particles with respect to time. Simulations were performed using a single processor and multiple processors in a computer network. The proposed algorithm is used for a search task of a multi-agent system in an unknown environment which consist of small robots with limited sensing capability. The method adopts asynchronous mechanism for information exchange and position updates of the agents and dynamic neighborhood topology of communicating agents. Simulations with a realistic simulator and implementation with real robots were performed to show the effectiveness of the proposed algorithm.

Keywords: Particle Swarm Optimization, Dynamic Particle Neighborhood Topology, Asynchronism, Distributed Systems, Search Task Using Multi-agent Systems

TEŞEKKÜR

Çalışmalarım boyunca yardım, katkı ve eleştirileri ile beni yönlendiren değerli hocam Doç. Dr. Veysel Gazi'ye ve yine kıymetli tecrübelerinden faydalandığım TOBB Ekonomi ve Teknoloji Üniversitesi Elektrik ve Elektronik Mühendisliği Bölümü öğretim üyelerine,

Her türlü zorluğa beraber göğüs gerdiğim Sürü Sistemler Laboratuvarındaki çalışma arkadaşlarım olan Ömer Çayırpunar, Yunus Ataş, Mirbek Turduev, Engin Karataş, Esma Gül, Sabahat Duran, İlter Köksal ve Andaç Töre Şamiloğlu'na

Çalışmalarımda yaptığı katkılardan dolayı Özgür Pekçağlıyan'a

Beni her zaman destekleyen ve bugünlere getiren aileme teşekkürlerimi sunarım.

Bu çalışma TÜBİTAK (Türkiye Bilimsel ve Teknolojik Araştırma Kurumu) tarafından 106E122 sayılı proje kapsamında ve Avrupa Komisyonu tarafından 6. Çerçeve Programı 045269 sözleşme numaralı özel amaçlı araştırma projesi kapsamında desteklenmiştir.

İÇİNDEKİLER

	Sayfa
ÖZET	iv
ABSTRACT	v
TEŞEKKÜR	vi
İÇİNDEKİLER	vii
1. GİRİŞ	2
2. PARÇACIK SÜRÜ ENİYİLEME YÖNTEMİ	5
2.1. Temel Parçacık Sürü Eniyileme Yöntemi	5
2.2. Komşuluk Yapıları	10
2.3. Melez Parçacık Sürü Eniyileme Algoritmaları	11
2.4. Parçacık Sürü Eniyileme Yönteminin Yakınsaklık Analizi	13
2.5. Parçacık Sürü Eniyileme Yönteminin Uygulama Alanları	14
2.5.1. Dinamik İzleme	14
2.5.2. Çeşitli Mühendislik Problemlerinde Uygulamaları	15
2.6. Tezin Amacı ve Konusu	15
2.7. Parçacık Sürü Eniyileme Yönteminin Dinamik Komşuluklu ve Paralel Biçimleri ile İlgili Çalışmalar	16
2.8. Denektaşı Fonksiyonlar	18
2.9. Parçacık Sürü Eniyileme Yönteminin Çok Erkinli Arama Görevinde Uygulandığı Çalışmalar	20
3. DİNAMİK KOMŞULUKLU PARÇACIK SÜRÜ ENİYİLEME YÖNTEMİ	22
3.1. Parçacık Komşuluklarının Dinamik Olarak Belirlenmesi	22
3.1.1. Arama Uzayında En Yakın Komşular Yöntemi	23

3.1.2. Fonksiyon Uzayında En Yakın Komşular Yöntemi 23

3.1.3. Rasgele Belirleme Yöntemi 25

3.1.4. Yönlü Çizgeler ile Komşuluk Gösterimi 26

3.2. Dinamik Komşuluk 28

3.3. Benzetim Sonuçları 34

4. EŞZAMANSIZ VE DAĞINIK PARÇACIK SÜRÜ ENİYİLEME YÖNTEMİ 50

4.1. Eşzamanlı Parçacık Sürü Eniyileme Algoritması 51

4.2. Eşzamansız ve Dağıtık Parçacık Sürü Eniyileme Algoritmasının Matematiksel Modeli 53

4.3. Eşzamansız ve Dağıtık Parçacık Sürü Eniyileme Algoritması 56

4.4. Benzetim Sonuçları 59

4.4.1. Tek bir İşlemci Üzerinde Gerçekleştirilen Benzetimler 60

4.4.2. Bir Bilgisayar Ağında Bulunan İşlemciler ile Gerçekleştirilen Benzetimler 66

5. EŞZAMANSIZ VE DİNAMİK KOMŞULUKLU PARÇACIK SÜRÜ ENİYİLEME YÖNTEMİNİN ÇOK ROBOTLU ARAMA GÖREVİNDE UYGULANMASI 74

5.1. Problem Tanımı 75

5.2. Eşzamansız Parçacık Sürü Eniyileme Yöntemi Tabanlı Arama Algoritması 79

5.3. Benzetim ve Uygulama 81

5.3.1. Benzetim Sonuçları 83

5.3.2. Uygulama Sonuçları 86

6. SONUÇ 91

6.1. Yorumlar 91

6.2. Gelecek Çalışmalar 92

KAYNAKLAR 94

ÖZGEÇMİŞ 102

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

Çizelge 2.1. Temel parçacık sürü eniyileme algoritmasının sahte kodu 6
Çizelge 2.2. Denektaşı fonksiyonlar 19
Çizelge 3.1. Arama uzayında en yakın komşular kuralı için benzetim sonuçları 47
Çizelge 3.2. Fonksiyon uzayında en yakın komşular kuralı için benzetim sonuçları 48
Çizelge 3.3. Rasgele komşuluk belirleme kuralı için benzetim sonuçları 49
Çizelge 4.1. Bir parçacık için eşzamansız ve dağıtık parçacık sürü eniyileme algoritmasının sahte kodu 57
Çizelge 4.2. Tek bir işlemci ile gerçekleştirilen benzetimlerde kullanılan algoritmanın sahte kodu 61
Çizelge 4.3. Türdeş ağda gerçekleştirilen benzetimlerde kullanılan işlemciler 68
Çizelge 4.4. Türdeş olmayan ağda gerçekleştirilen benzetimlerde kullanılan işlemciler 71
Çizelge 5.1. Eşzamansız parçacık sürü eniyileme yöntemi tabanlı arama algoritmasının sahte kodu 80

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil 2.1. Yöntemde kullanılan çeşitli komşuluk yapıları a) Çember komşuluk yapısı (Yerel Yapı) b) Tekerlek komşuluk yapısı (Yerel Yapı) c) Yıldız komşuluk yapısı (Bütünsel Yapı). 10
Şekil 2.2. Denektaşı fonksiyonların şekilleri: sol üst köşe küre, sağ üst köşe Griewank, sol alt köşe Rastrigin, sağ alt köşe Rosenbrock. 20
Şekil 3.1. Arama uzayında en yakın komşular yönteminin gösterimi. 24
Şekil 3.2. Fonksiyon uzayında en yakın komşular yönteminin gösterimi. 24
Şekil 3.3. Parçacık komşuluklarının yönlü çizgeler ile gösterimi. 27
Şekil 3.4. Güçlü şekilde bağlı çizge. 28
Şekil 3.5. Varsayım 1 yönlü çizgeler ile gösterimi. 31
Şekil 3.6. Komşuluğun en iyi değerlerinin ortalamalarının arama uzayında parçacıkların algılama alanları büyüklüğüne göre değişimi (Küre). 36
Şekil 3.7. Komşuluğun en iyi değerlerinin ortalamalarının arama uzayında parçacıkların algılama alanları büyüklüğüne göre değişimi (Griewank). 36
Şekil 3.8. Komşuluğun en iyi değerlerinin ortalamalarının arama uzayında parçacıkların algılama alanları büyüklüğüne göre değişimi (Rastrigin). 37
Şekil 3.9. Komşuluğun en iyi değerlerinin ortalamalarının arama uzayında parçacıkların algılama alanları büyüklüğüne göre değişimi (Rosenbrock). 37
Şekil 3.10 Komşuluğun en iyi değerlerinin ortalamalarının fonksiyon uzayında parçacıkların algılama alanları büyüklüğüne göre değişimi (Küre). 38
Şekil 3.11 Komşuluğun en iyi değerlerinin ortalamalarının fonksiyon uzayında parçacıkların algılama alanları büyüklüğüne göre değişimi (Griewank). 38
Şekil 3.12 Komşuluğun en iyi değerlerinin ortalamalarının fonksiyon uzayında parçacıkların algılama alanları büyüklüğüne göre değişimi (Rastrigin). 39
Şekil 3.13 Komşuluğun en iyi değerlerinin ortalamalarının fonksiyon uzayında parçacıkların algılama alanları büyüklüğüne göre değişimi (Rosenbrock). 39
Şekil 3.14 Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 0.0001]$) (Küre). 41
Şekil 3.15 Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 0.0001]$) (Griewank). 41
Şekil 3.16 Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 0.0001]$) (Rastrigin). 42
Şekil 3.17 Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 0.0001]$) (Rosenbrock). 42
Şekil 3.18 Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 0.1]$) (Küre). 43
Şekil 3.19 Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 0.1]$) (Griewank). 43

Şekil 3.20Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 0.1]$) (Rastrigin). 44
Şekil 3.21Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 0.1]$) (Rosenbrock). 44
Şekil 3.22Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 1]$) (Küre). 45
Şekil 3.23Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 1]$) (Griewank). 45
Şekil 3.24Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 1]$) (Rastrigin). 46
Şekil 3.25Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 1]$) (Rosenbrock). 46
Şekil 4.1. Komşuluğun en iyi uyumluluk değerinin ortalamasının, parçacıkların komşu sayısının ortalamasının ve algılanabilen komşu parçacıkların sayısının ortalamasının komşuluk büyüklüğüne göre değişimi ve parçacıkların güncelleme yapma olasılığının ortalamasının yineleme sayısına göre değişimi (Küre). 62
Şekil 4.2. Komşuluğun en iyi uyumluluk değerinin ortalamasının, parçacıkların komşu sayısının ortalamasının ve algılanabilen komşu parçacıkların sayısının ortalamasının komşuluk büyüklüğüne göre değişimi ve parçacıkların güncelleme yapma olasılığının ortalamasının yineleme sayısına göre değişimi (Griewank). 63
Şekil 4.3. Komşuluğun en iyi uyumluluk değerinin ortalamasının, parçacıkların komşu sayısının ortalamasının ve algılanabilen komşu parçacıkların sayısının ortalamasının komşuluk büyüklüğüne göre değişimi ve parçacıkların güncelleme yapma olasılığının ortalamasının yineleme sayısına göre değişimi (Rastrigin). 64
Şekil 4.4. Komşuluğun en iyi uyumluluk değerinin ortalamasının, parçacıkların komşu sayısının ortalamasının ve algılanabilen komşu parçacıkların sayısının ortalamasının komşuluk büyüklüğüne göre değişimi ve parçacıkların güncelleme yapma olasılığının ortalamasının yineleme sayısına göre değişimi (Rosenbrock). 65
Şekil 4.5. Türdeş ağda komşuluğun en iyi uyumluluk değerlerin ve komşuluğun en iyi konumlarının yinelemelere göre değişimi (Küre). 69
Şekil 4.6. Türdeş ağda komşuluğun en iyi uyumluluk değerlerin ve komşuluğun en iyi konumlarının yinelemelere göre değişimi (Griewank). 69
Şekil 4.7. Türdeş ağda komşuluğun en iyi uyumluluk değerlerin ve komşuluğun en iyi konumlarının yinelemelere göre değişimi (Rastrigin). 70
Şekil 4.8. Türdeş ağda komşuluğun en iyi uyumluluk değerlerin ve komşuluğun en iyi konumlarının yinelemelere göre değişimi (Rosenbrock). 70
Şekil 4.9. Türdeş olmayan ağda komşuluğun en iyi uyumluluk değerlerin ve komşuluğun en iyi konumlarının yinelemelere göre değişimi (Küre). 71
Şekil 4.10Türdeş olmayan ağda komşuluğun en iyi uyumluluk değerlerin ve komşuluğun en iyi konumlarının yinelemelere göre değişimi (Griewank). 72

Şekil 4.11 Türdeş olmayan ağda komşuluğun en iyi uyumluluk değerlerin ve komşuluğun en iyi konumlarının yinelemelere göre değişimi (Rastrigin). 72
Şekil 4.12 Türdeş olmayan ağda komşuluğun en iyi uyumluluk değerlerin ve komşuluğun en iyi konumlarının yinelemelere göre değişimi (Rosenbrock). 73
Şekil 5.1. Hız-kısıtlı robot yapısı. 76
Şekil 5.2. Deneysel düzenek. 82
Şekil 5.3. Kaynak fonksiyonu. 83
Şekil 5.4. Player/Stage yazılımının blok gösterimi. 84
Şekil 5.5. İtim potansiyel fonksiyonunda kullanılan ses ötesi algılayıcılar ve açıları. 85
Şekil 5.6. Robotların ara noktaları ve gezingeleri. 87
Şekil 5.7. Hedefe olan ortalama uzaklık. 88
Şekil 5.8. Uygulamada kullanılan alan. 88
Şekil 5.9. Khepera III robotların gezingeleri, ara noktaları ve hedefe olan ortalama uzaklık. 89
Şekil 5.10 KheperaIII robotları ile gerçekleştirilen uygulamadan anlık kareler. 90
Şekil 5.11 İtim potansiyel fonksiyonunda kullanılan kızıl ötesi algılayıcılar ve açıları. 90

BÖLÜM 1

1. GİRİŞ

"Sürü zekası" disiplini merkezi olmayan denetim mekanizmaları ve kendini örgütleme[1] ile eşgüdümlenen birçok basit özerk erkinden[2] oluşan doğal ve yapay sistemler ile ilgilenmektedir. Bu disiplin sistemdeki erkinlerin birbirleri ve çevreleri ile etkileşimleri sonucu ortaya çıkan toplu[3] davranışlarına odaklanmaktadır. Bu durumda sürü zekası basit özerk erkin topluluklarında kendiliğinden ortaya çıkan[4] toplu zeka olarak tanımlanabilir [2]. "Sürü zekası" terimi ilk defa 1989 yılında Gerardo Beni ve Jing Wang tarafından hücresel robotik sistemlerin ele alındığı [1] çalışmasında kullanılmıştır. Bu çalışmada birçok basit erkin çeşitli örüntüler üretmek ve en yakın komşularını göz önünde bulundurarak kendiliğinden örgütlenmek amacıyla bir veya iki boyutlu ortamlara yerleşmiştir. Öte yandan çalışmada yapılan tanımlama sadece hücresel robotik sistemler için geçerli olmuştur ve sonraki çalışmalarda doğadaki sürüler ve onların davranışları modellenirken sürü zekasının tanımı ve kapsamı geliştirilmiştir.

Sürü zekası disiplininde sonraki çalışmalarda doğadaki böcek ve sürülerin modellenmesi ve sürülerin davranışlarının sürü zekası ile ilişkilendirilerek sürü zekasının kapsamı genişletilmeye çalışılmıştır. Doğadaki sosyal sürüler incelendiğinde sürünün eşgüdümlülüğünü sağlayan mekanizmaların ve toplu davranışların sürünün kendini örgütleme özelliği sayesinde ortaya çıktığı gözlemlenmiştir. Sosyal sürülerde karmaşık toplu davranışlar ve dolayısıyla toplu zeka, basit kurallara uyan erkinlerin birbirleri ile doğrudan veya çevre aracılığıyla etkileşimleri (stigmerji[5]) ve çevre ile olan etkileşimleri nedeniyle ortaya çıkmaktadır. Bu durumda doğadaki sosyal sürülerin karmaşık davranışlarını incelemek için, sürüdeki erkinlerin davranışlarını ortaya çıkaran karmaşık nedenlere bakmak yerine erkinler arasındaki basit etkileşimlere bakmak yeterlidir. Dahası bu durum kendini örgütleme özelliğine dayalı modellerde karmaşık işlemleri basit etkileşimler cinsinden betimlenmesine ve doğadaki sürülerin karmaşık davranışlarının mühendislik alanında akıllı sistemler tasarımında kullanılmasına olanak sağlamaktadır [2]. Doğadaki sosyal sürülerin kendi problemleri (yiyecek bulma veya yuva yapma) birçok mühendislik problemine

[1] ing:self-organized
[2] ing:agent
[3] ing:collective
[4] ing:emergent
[5] ing:stigmergy

benzemesi ve sosyal sürülerin bu problemleri etkin bir şekilde çözmesi sürü davranışlarının mühendislik problemlerinde kullanılmasını cazip bir hale getirmiştir.

Mühendislik alanında, doğadaki sosyal sürülerin davranışı çeşitli eniyileme yöntemleri aracılığıyla kullanılmaktadır. Doğadaki sürülerin davranışlarından bir başka deyişle kendi problemlerini çözerken en iyi çözümleri seçmesi ve çeşitli problemleri çözerken gerçekleştirdikleri davranışlar göz önüne alınarak çeşitli eniyileme yöntemleri geliştirilmiştir. Karınca sürülerinin yaşadıkları koloniden besin bulunan bölgeye gitmeleri için en kısa yolu bulmasından/kullanmasından esinlenilerek ortaya konulan karınca kolonisi eniyileme yöntemi [4], bal arılarının beslenme davranışlarından esinlenen arı kolonisi eniyileme yöntemi [5] ve kuş sürülerinin besin bulma amacıyla gösterdikleri davranışlardan esinlenen parçacık sürü eniyile yöntemi [3] bu yöntemlerin başında gelmektedirler.

Sürü robot sistemleri doğadaki sosyal sürülerin davranışının mühendislik alanında başka bir uygulamasıdır. Doğadaki sürülerde bulunan erkinlerin sınırlı yeteneklerine karşın birbirleri ve çevreleri ile olan etkileşimleri sonucu karmaşık görevleri yerine getirmesi, robot sistemlerinin bu özellikler göz önünde alınarak tasarlanması cazip bir hale gelmiştir. Bu tasarımlar sonucu son yıllarda ortaya çıkan ve büyük bir hızla gelişen sürü robot sistemleri disiplini ortaya çıkmıştır. Sürü sistemlerin yüksek seviyede gürbüzlüğe ve erkin bazında düşük karmaşıklığa sahip olması, maliyetinin az olması ve istenen görevleri daha etkin bir şekilde gerçekleştirmesi özellikleri ile diğer sistemlere göre daha avantajlı görülmektedir [6]. Bu nedenlerden ötürü karmaşık görevler için işlem yeteneği yüksek tek bir robot yerine işlem kapasiteleri sınırlı ve maliyetleri düşük birçok robottan oluşan sürü robot sistemleri tercih edilebilir. Sürü robot sistemlerin esneklik özelliği, farklı ortamlara veya görevlere göre kendini farklı biçimde örgütleme özelliği, bu tip sistemler ile farklı görevlerin gerçekleştirilmesine olanak sağlamaktadır. Ayrıca tek bir robotun yerine getiremeyeceği görevleri sürü robotlar aralarında işbirliği yaparak istenen görevi etkin bir biçimde gerçekleştirebilirler. Sürü robot sistemlerinin diğer robotik sistemlerden farkını ortaya koyabilmek için bu sistemlerin çeşitli özellikleri [7] çalışmasında ön plana çıkartılmıştır. Bu çalışmada sürü robot sistemlerin başlıca özellikleri özerk robotlardan oluşması, birçok sayıda robotun bulunması, az sayıda türdeş robot gruplarının olması, robotların ele alınan göreve göre yetersiz kapasiteye sahip olması ve robotların yerel algılama ve haberleşme yeteneklerine sahip olması olarak belirlenmiştir.

Sürü robotik sistemleri belirtilen avantajları nedeniyle birçok mühendislik projesinde

kullanılmaktadır. “Swarm-bots” projesinden “s-bots” adlı küçük ve maliyeti düşük erkinlerin kendini örgütlemesi ile “Swarm-bot” adlı tek bir sistemi meydana getirmesine odaklanılmıştır [8]. Küçük erkinlerden oluşan tek büyük erkinin küçük erkinlerin tek başına gerçekleştiremeyecekleri ağır bir yükü taşıma ve engebeli bir arazide gezinme gibi görevlerde kullanılması amaçlanmıştır. Öte yandan çevrenin durumuna göre küçük erkinler büyük erkini farklı geometrik şekillerde oluşturması da amaçlanmıştır. Bir başka proje olan “Swarmanoid” projesinde üç farklı özerk erkin grubu kullanılarak 3 boyutta hareket kabiliyetine sahip türdeş olmayan bir sistem tasarlanması amaçlanmaktadır [9]. “Eye-bots” adı verilen küçük erkin grubuna algılama ve bulunan ortamı analiz etme görevi, “Hand-bots” adı verilen erkin grubuna ortamda bulunan dik engellerden tırmanma ve çeşitli nesneleri kavrama görevleri, “Foot-bots” adı verilen grubuna ise engebeli alanda ilerleme ve nesne veya robotların taşıma görevleri atanmıştır. Belirtilen üç erkin grubundan oluşan ve dinamik olarak gruplardaki erkinlerin durumlarının değiştiği türdeş olmayan bir robot sisteminin denetimi ve eşgüdümlülüğü projenin asıl amacıdır. Sürü robotların tek bir robot sistemi oluşturması istenen projelerin dışında bu tip sistemlerin arama kurtarma işlemlerinde kullanılmaya amaçlandığı projelerde mevcuttur. “Guardians” projesi özerk erkinlerden oluşan bir sürünün bilinmeyen endüstriyel bir binadaki yangın ortamında (ortam çeşitli zararlı kimyasalların olduğu) arama ve kurtarma görevlerinde kullanılmasına odaklanılmıştır [10]. Projenin asıl amacı özerk bir sürünün ortamda bulunan bir itfaiyeciye arama sırasında yardım etmesi ve onu olası tehlikelerden uzak tutmasıdır. Sürü sistemlerin arama ve kurtarma görevlerinde kullanılmasının amaçlandığı bir başka proje olan “View-Finder” projesinde yarı özerk robot sürüsünün yangın ortamının haritasını çıkarması ve veri toplaması amaçlanmıştır [11]. Böylece itfaiyeciler arama kurtarma görevlerinde ortam hakkında gerekli bilgilere önceden sahip olacaktır.

BÖLÜM 2

2. PARÇACIK SÜRÜ ENİYİLEME YÖNTEMİ

2.1. Temel Parçacık Sürü Eniyileme Yöntemi

Parçacık sürü eniyileme yöntemi (PSO) Kennedy ve Eberhart tarafından 1995 yılında [12, 13] çalışmalarında geliştirilmiş evrimsel bir hesaplama ve arama yöntemidir. Parçacık sürü eniyileme yöntemi topluluk tabanlı bir yöntem olup parçacık adı verilen rasgele çözüm kümeleri ile ilklendirilir. Yöntem doğrudan bir arama yöntemidir[1], arama sırasındaki eğim bilgisi doğrudan kullanılmamaktadır ve yinelemeler boyunca sürüdeki parçacıklar paralel olarak arama yapmaktadırlar. Parçacıkların arama uzayında hareket etmeleri için belirli hızları vardır ve parçacıkların hızları arama sırasındaki davranışlarına göre dinamik olarak değişmektedir. Böylece parçacıklar arama yapılan uzayda istenilen en iyi noktalara ulaşmaktadır.

Parçacık sürü eniyileme yönteminin temel biçimi basitleştirilmiş toplumsal model benzetimlerinde ortaya çıkmıştır. Yöntem, kuş ve balık sürülerinin toplumsal davranışlarından ve sürü teorisinden[2] esinlenerek geliştirilmiştir. Yöntem ilk olarak belirli bir alanda besin bulmaya çalışan kuş sürüsünün davranışının benzetimi için kullanılmıştır. Tek bir kuş çevresindeki diğer kuşlar (komşuları) ile toplumsal işbirliği yaparak alandaki besini bulabilmektedir. Sonraki çalışmalarda yöntem çok boyutlu uzaylarda eniyileme probleminde kullanılmış ve parçacıkların bir komşuluk yapısında yer aldığı düşünülmüştür. Çok boyutlu bir uzayda $f(x), x = [x_1, x_2, \ldots, x_n]$ gibi bir uyumluluk fonksiyonu ele alınmıştır ve parçacıkların uzayda arama yaparak belirlenen fonksiyonun bütünsel minimum veya bütünsel maksimum noktasının bulunması amaçlanmıştır. Bu çalışmalar sonucunda temel parçacık sürü eniyileme algoritması her $i = 1, \ldots, N$ parçacığı için

$$\begin{aligned} v_i(t+1) &= v_i(t) + \varphi_1^i(t)\Big(b_i(t) - p_i(t)\Big) + \varphi_2^i(t)\Big(g_i(t) - p_i(t)\Big) \\ p_i(t+1) &= p_i(t) + v_i(t+1). \end{aligned} \tag{2.1}$$

şeklindeki denklem takımı olarak belirlenmiştir [12, 13]. Burada $p_i(t) \in \mathbb{R}^n$ i'inci parçacığın t zamanında uzaydaki konumunu (parçacığın eniyilenmekte olan

[1] ing: direct search method
[2] ing: swarm theory

fonksiyonun en küçük/en büyük noktası hakkında t zamanındaki tahmini), $b_i(t) \in \mathbb{R}^n$ i'inci parçacığın t zamanına kadar elde ettiği en iyi konumunu bir başka deyişle t zamanına kadar elde ettiği en iyi fonksiyon değerine karşılık gelen konumu, $g_i(t) \in \mathbb{R}^n$ i'inci parçacığın komşularının t zamanına kadar elde ettiği en iyi konumu bir başka deyişle parçacığın komşularının t zamanına kadar elde ettiği en iyi fonksiyon değerine karşılık gelen konumu belirtmektedir (bakınız Çizelge 2.1.). Parçacık arama sırasında hızını o andaki hızını, o andaki konumunu, o ana kadar elde ettiği en iyi konumunu ve komşularının o ana kadar elde ettiği en iyi konumu kullanarak dinamik olarak belirlemektedir. Öğrenme katsayıları olan $\varphi_1^i(t) \in [0, \bar{\varphi}_1]^n$ ve $\varphi_2^i(t_k) \in [0, \bar{\varphi}_2]^n$ düzgün dağılımlı n boyutlu rasgele vektörlerdir. Bu rasgele vektörler döngüdeki bilişsel[3] ve toplumsal/sosyal bileşenlerin göreli önemlerini/ağırlıklarını belirlemektedir. (Çizelge 2.1.'de parçacıkların bütünsel bir komşuluk yapısında bulunduğu kabul edilmiştir.)

Çizelge 2.1. Temel parçacık sürü eniyileme algoritmasının sahte kodu

Her i parçacığı için konum, $p_i(t)$, ve hız, $v_i(t)$, vektörlerinin ilklendirilmesi
Komşuluğun en iyi uyumluluk değerinin, $evrensel_{eniyi}$, ve her parçacık için en iyi uyumluluk değerinin, $kisisel_{eniyi}$, ilklendirilmesi
Komşuluğun en iyi konumunun, $g_i(t)$, ve her parçacığın için en iyi konumunun, $b_i(t)$, ilklendirilmesi
while Durma koşulu sağlanmıyor ise **do**
  Her parçacığın uyumluluk değerinin, $f(p_i(t))$, hesaplanması
  for i = 1:Parçacık sayısı **do**
    if $f(p_i(t)) < kisisel_{eniyi}$ **then**
      $kisisel_{eniyi} = f(p_i(t))$
      $b_i(t) = p_i(t)$
    end if
  end for
  En küçük uyumluluk değeri, $min_{kisisel_{eniyi}}$ ve bu değere sahip olan parçacık, min_i seçilir
  if $min_{kisisel_{eniyi}} < evrensel_{eniyi}$ **then**
    $evrensel_{eniyi} = min_{kisisel_{eniyi}}$
    $g_i(t) = p_{min_i}(t)$
  end if
  for i=1:Parçacık sayısı **do**
    $v_i(t+1) = v_i(t) + \varphi_1^i(t)\Big(b_i(t) - p_i(t)\Big) + \varphi_2^i(t)\Big(g_i(t) - p_i(t)\Big)$
    $p_i(t+1) = p_i(t) + v_i(t+1)$
  end for
end while

Denklem 2.1 sürüdeki parçacıkların çok boyutlu bir uzaydaki hareketlerini

[3]ing: cognitive

betimlemektedir. Denklem 2.1'de bulunan ilk denklem parçacık hızların dinamik olarak nasıl değiştiğini, ikinci denklem ise sürüdeki parçacıkların arama uzayındaki konumlarını nasıl güncellendiklerini göstermektedir. Denklem 2.1'de bulunan ilk denklem üç kısma ayrılabilir. Birinci kısım devinirlik (momentum) bileşenidir. Bu kısım güncellenen hızın o zaman anındaki hız değerini göz önüne alarak güncellenmesini sağlar, böylece güncellenen hızın ani olarak değişimi önlenir. İkinci kısım olan bilişsel bileşen bir başka deyişle parçacıkların hafızası olduğunu ve geçmiş tecrübelerinden yararlanabilme yetisini gösterir. Üçüncü kısım olan toplumsal bileşen ise toplumsal işbirliği, parçacıkların komşularının tecrübelerin yararlanarak karar verebilme yetisini göstermektedir.

Parçacık sürü eniyileme yöntemi diğer evrimsel programlama yöntemleri gibi rasgele ilklendirilen ve diğer topluluk üyeleriyle etkileşimin olduğu topluluk tabanlı bir arama algoritmasıdır. Bazı evrimsel programlama yöntemlerinin aksine parçacık sürü eniyileme yönteminde herhangi seçme işleci[4] bulunmamaktadır. Parçacıklar arama uzayında hareket edebilir ve nesilden nesile parçacıklar elde ettikleri en iyi konum bilgisini kullanma yetisine sahiplerdir [26, 27]. Öte yandan bazı evrimsel programlama yöntemleri ise sadece bir nesildeki üyelerden en iyi çözüme sahip olanları bir sonraki nesle aktarılır. Ayrıca PSO yönteminde çaprazlama işleci[5] bulunmamakta ve bu nedenden dolayı parçacıklar hem kendi elde ettikleri bilgileri hem de bütün komşularında elde ettikleri bilgileri kullanarak arama uzayındaki hareketlerini belirlemektedirler [28]. Parçacık sürü eniyileme yönteminde birçok evrimsel programlama yöntemlerinde kullanılan mutasyon işleci kullanılmamaktadır, ancak parçacığın uzaydaki hareketini belirlerken kendi bilgisini ve komşularının bilgilerini kullanması parçacığın uygun çözümler bölgesine doğru yönelmesini sağlamaktadır. Bu durumda parçacık sınırlı sayıda arama yönünde uygun çözümlerin olduğu bölgede arama yapmakta ve bu durum bir mutasyon işleci olarak kabul edilmektedir [15]. Parçacık sürü eniyileme yöntemi ve evrimsel algoritmaların başarımının karşılaştırıldığı çalışmalarda yöntemin uygun çözümlerin olduğu bölgelere evrimsel algoritmalardan daha hızlı yakınsadığı ancak bu bölgelerdeki detaylı arama yeteneğinin evrimsel algoritmalardan daha düşük olduğu gösterilmiştir [26, 29].

Denklem 2.1'de bulunan ilk denklemde belirtilen üç kısım arasındaki denge yöntemin evrensel ve yerel arama yeteneklerini belirlemekte, dolayısıyla yöntemin başarımını etkilemektedir. Dikkat edilirse denklemde bulunan düzgün dağılımlı n

[4] ing: selection operator
[5] ing: crossover operator

boyutlu vektörler olan bilişsel φ_1^i ve toplumsal φ_2^i öğrenme katsayılarının rasgele olmaları nedeniyle yöntemin evrensel ve yerel arama kabiliyetlerini önemli ölçüde belirlemektedirler. Bir parçacık için bilişsel öğrenme katsayısının, φ_1^i, değerinin artırılması yerel arama kabiliyetini artırırken, toplumsal öğrenme katsayısının, φ_2^i, değerinin artırılması evrensel arama kabiliyetini artırmaktadır. Öğrenme katsayılarını rasgele olarak seçilmesi yöntemin arama kabiliyetini geliştirmesine karşın; bu katsayılara göre güncellenen parçacık hızlarının istenmeyen değerler alması ve arama uzayındaki parçacıkların yüksek hızlarla hareket ederek uzayda saçılması olası bir durumdur. Bu durum yönteminin "patlama" davranışı olarak adlandırılmıştır ve parçacıkların uzayda bir noktaya/bölgeye yakınsayamaması olarak belirlenmiştir. Belirtilen sorunun çözülmesi veya en aza indirgenmesi için birçok çalışma gerekçeleştirilmiş ve yöntemin farklı biçimleri ortaya konulmuştur. "Patlama" davranışının ilk çözümü parçacıkların her boyutta bulunan hız değerlerinin istenen bir aralığa sabit $\pm V_{maks}$ sınır değerleri kullanılarak sınırlandırılması olarak belirlenmiştir. Yöntemin verimliliği açısından V_{maks} hız sınırının dinamik olarak değiştiği bir durum daha uygun olabileceği [14] çalışmasında gösterilmiştir.

Öte yandan parçacık hızlarının her boyutta sınırlandırılması yöntemin arama kabiliyetini düşürebilmektedir. Parçacık hızlarının sınırlandırılmadan patlama davranışının engellenmesi için yöntemin dinamik denklemlerine çeşitli katsayılar eklenmiştir. Dinamik denklemlere eklenen ilk katsayı eylemsizlik ağırlık katsayısıdır[6]. Bu katsayı ile yöntemin dinamik denklemleri

$$\begin{aligned} v_i(t+1) &= w v_i(t) + \varphi_1^i(t)\Big(b_i(t) - p_i(t)\Big) + \varphi_2^i(t)\Big(g_i(t) - p_i(t)\Big) \\ p_i(t+1) &= p_i(t) + v_i(t+1). \end{aligned} \tag{2.2}$$

biçimini almıştır [15, 16]. Eylemsizlik ağırlık katsayısı, yöntemin yerel ve evrensel arama kabiliyetlerini dengelemek amacıyla ortaya konulmuştur. Eylemsizlik ağırlık katsayısının büyük değerler alması yöntemin evrensel arama kabiliyetini artırırken, küçük değerler alması yöntemin yerel arama kabiliyetini artırır. Eylemsizlik ağırlık katsayısının doğrusal olarak artırılarak [17] veya azaltılarak [15] yöntemin başarımını inceleyen çalışmalar sonucunda yöntem için en iyi başarımı sağlayan katsayının değeri belirlenememiştir. Bulanık mantık ile eylemsizlik ağırlık katsayısının ayarlandığı [18] çalışmasında ve katsayının zaman ile parçacıkların ivmesine göre değiştiği [19] çalışmasında yöntemin başarımının daha iyi olduğu gözlemlenmiştir. [25] çalışmasında Denklem 2.2 benimsenmiş ve parçacıkların davranışı ayrık zamanlı

[6]ing: inertia weight parameter

sistem kuramından yararlanılarak incelenmiş ve yöntem için parametre seçiminde önerilerde bulunulmuştur.

Parçacık sürü eniyileme yönteminin patlama davranışının engellenmesi için bir başka katsayı olan kısıtlama katsayısı[7] χ çeşitli çalışmalarda ortaya konulmuştur [20, 21]. Yeni parametre için yöntemin dinamik denklemleri

$$\begin{aligned} v_i(t+1) &= \chi\Big[v_i(t) + \varphi_1^i(t)\Big(b_i(t) - p_i(t)\Big) + \varphi_2^i(t)\Big(g_i(t) - p_i(t)\Big)\Big] \\ p_i(t+1) &= p_i(t) + v_i(t+1). \end{aligned} \tag{2.3}$$

biçiminde tanımlanmıştır. Kısıtlama katsayısı bilişsel ve sosyal öğrenme katsayıları olan φ_1 ve φ_2 fonksiyonu

$$\chi = \begin{cases} \frac{2\kappa}{\varphi - 2 + \sqrt{\varphi^2 - 4\varphi}} & \text{eğer } \varphi > 4, \\ \sqrt{\kappa} & \text{aksi taktirde}, \end{cases} \tag{2.4}$$

biçiminde tanımlanmıştır [20]. Bu tez çalışmasında yöntemin denklem 2.3'te öne sürülen kısıtlama katsayılı biçimi kullanılmaktadır. Burada $\varphi_1 + \varphi_2 = \varphi$ ve $\kappa \in [0, 1]$.

Denklem 2.2'de bulunan eylemsizlik ağırlık katsayısı kısıtlama kaysayısına eşitlenir ve öğrenme katsayıları φ_1 ve φ_2 $\varphi_1 + \varphi_2 = \varphi$, $\varphi > 4$ koşulunu sağlayacak şekilde seçilirse, eylemsizlik ağırlık katsayılı yöntem ile kısıtlama katsayılı yöntem biribirine eşdeğer olurlar. Bu nedenden dolayı kısıtlama katsayılı yöntem, eylemsizlik ağırlık katsayılı yöntemin özel bir durumu olarak görülebilir. Yöntemin eylemsizlik katsayılı biçimi ve kısıtlama katsayılı biçimi [22] çalışmasında karşılaştırılmış ve arama uzayının boyutlarının iyi ayarlandığı taktirde kısıtlama katsayılı yöntemin daha hızlı yakınsadığı gözlemlenmiştir. Yöntemin Denklem 2.3'teki biçimi benimsenerek yöntem için en iyi sonucu verecek parametre kümesini (parçacık sayısı, parçacık komşuluğunun büyüklüğü, kısıtlama ve ağırlık eylemsizlik katsayılarının değerleri, öğrenme katsayılarının üst değerleri) bulunması amacı ile çeşitli çalışmalar yapılmıştır ancak genel bir sonuca varılamamıştır [23,24]. [20] çalışmasında öne sürülen kısıtlama katsayılı denklemde öğrenme katsayısının (φ) üst sınrının 4.1 olarak alınması ve Denklem 2.4'de bulunan κ değerinin 1 alınması ile parçacıkların yakınsamalarının yavaş olacağı, böylece parçacıkların arama uzayını daha iyi bir biçimde arayacağı belirtilmiştir.

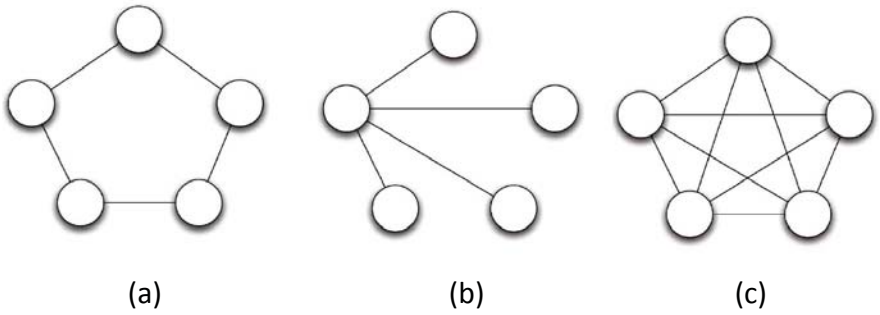
Parçacık sürü eniyileme yönteminin incelenmesi ve parametrelerinin ayarlanması

[7] ing: constriction factor

içeren çalışmalar dışında, yöntemde parçacıkların komşuluk yapılarının[8] incelendiği, yöntemin başarımının artırılması için evrimsel veya evrimsel olmayan yöntemler kullanılarak melez algoritmaların geliştirildiği, yöntemin yakınsaklık analizinin gerçekleştirildiği ve yöntemin çeşitli uygulamalarda kullanıldığı çalışmalar yöntem ile ilgili yapılan çalışmaların başında gelmektedir.

2.2. Komşuluk Yapıları

Parçacık sürü eniyileme yönteminin temel biçiminde yapısal komşuluk sürüdeki her parçacığın her diğer parçacık ile komşu olduğu bütünsel bir komşuluk olarak tanımlanır (bakınız Şekil 2.1.). Her parçacık her diğer parçacığın en iyi konum bilgisini alabilmekte ve bu bilgileri kullanılarak parçacıklar komşuluğun en iyi konumunu belirleyebilmektedir. Böylece parçacığın hızı o zamana kadar elde ettiği en iyi konuma ve diğer bütün parçacıkların elde ettiği en iyi konumlara göre dinamik olarak ayarlanır. Bir diğer komşuluk yapısı olarak yerel yapılı komşuluklar benimsenmiştir (bakınız Şekil 2.1.). Bu tip komşuluk yapılarında parçacıklar sadece sürüdeki bazı parçacıklardan bilgi alabilmekte ve parçacık hızını o zamana kadar elde ettiği en iyi konuma ve komşuluğundaki parçacıklarından elde ettiği en iyi konumlara göre belirlemektedir. Parçacık sürü eniyileme yöntemindeki parçacık komşulukları gibi sosyal yapılarda yapıdaki üyeler arası bağlanırlık, üyelerin öbekleşme miktarı ve üyeler arası mesafe gibi çeşitli özellikler yapıdaki bilgi akışını etkilemektedir [33]. Bu nedenden dolayı parçacıkların komşuluk yapıları yöntemin başarımını da etkilemektedir.



Şekil 2.1.: Yöntemde kullanılan çeşitli komşuluk yapıları a) Çember komşuluk yapısı (Yerel Yapı) b) Tekerlek komşuluk yapısı (Yerel Yapı) c) Yıldız komşuluk yapısı (Bütünsel Yapı).

Parçacık komşuluklarının yerel komşuluklar yapıları olan çember, tekerlek, bütünsel

[8]ing: neighborhood topologies

bir komşuluk yapısı olan yıldız ve rasgele belirlenen komşuluk yapıları kullanılarak yöntemin başarımı incelenmiştir [30]. Komşuluk yapılarının yöntemin başarımını büyük ölçüde etkilediği tespit edilmiştir. Öte yandan çok doruklu fonksiyonların eniyilemesinde yerel komşuluk yapılarının bütünsel komşuluk yapılarına göre yerel minimum noktalara yakınsama riskinin daha az olduğu, tek doruklu fonksiyonlarda ise bütünsel komşuluk yapılarının yerel komşuluk yapılarından daha hızlı bütünsel minimum noktaya yakınsadığı belirlenmiştir. Yerel ve bütünsel komşuluk yapılarının yönteme olan etkisini inceleyen daha genel bir çalışma [31] çalışmasında sunulmuştur. Bütünsel parçacık komşuluklarında bilgi akışının yerel komşuluk yapılarından daha hızlı olduğu böylece sürünün komşuluğun en iyi konumuna hızlı bir şekilde yakınsadığı gösterilmiş, bu durumun bütünsel komşuluk yapılarının çok doruklu fonksiyonların eniyilemesindeki yerel minimum noktalara yakınsamasının nedeni olabileceği ve parçacıkların başlangıç koşullarının yöntemin başarımını etkileyen önemli bir unsur olduğu belirlenmiştir. [32] çalışmasında belirli bir komşulukta bulunan parçacıklarından en iyi uyumluluk değerine sahip olan parçacığın arama yönünün diğer parçacıkların arama yönünden daha uygun varsayılmasının hatalı olabileceğine dikkat çekilmiştir. Çalışmada yerel ve bütünsel komşuluk yapılarını ele alınarak parçacıkların hızlarının güncellenirken komşuluktaki bütün parçacıkların etkisi olduğu öne sürülmüş, bu nedenden dolayı her parçacığa bir uyumluluk değeri ve sürüdeki parçacık sayısına göre bir ağırlık parametresi atanmıştır. Yöntemin ulaşması istenen koşula göre (evrensel minimum noktaları bulma başarısı veya yakınsama hızı) yerel veya bütünsel komşuluk yapılarının seçilmesi tavsiye edilmiştir.

Yöntemin parçacık komşuluk yapıları ile ilgili yapılan çalışmalarında en iyi başarımı sağlayacak tek bir yapı bulunamamıştır. Bu çalışmalar sonucunda eniyilenecek fonksiyonun biçimine göre komşuluk yapısının belirlenmesinin yöntemin verimliliğini büyük ölçüde arttıracağı belirlenmiştir.

2.3. Melez Parçacık Sürü Eniyileme Algoritmaları

Parçacık sürü eniyileme yönteminin evrensel arama kabiliyetinin (uzayda uygun çözümlerin olduğu bölgelere yakınsama hızının) diğer evrimsel algoritmalardan daha iyi olduğu ancak yerel arama kabiliyetinin (uygun çözümlerin bulunduğu bölgelerde ayrıntılı arama) daha düşük olduğu önceden belirtilmiştir. Bu nedenden dolayı yöntemin yerel minimum noktalara yakınsama riski bulunmaktadır. Yöntemin bu sorununu aşmak ve arama kabiliyetini geliştirmek için evrimsel işleçler olan seçme, çaprazlama, mutasyon işleçler ve evrimsel olmayan diğer yöntemler PSO yöntemi ile

birlikte kullanılarak bir çok çalışma gerçekleştirilmiştir.

Yöntemde seçme işleci kullanılarak sadece en iyi uyumluluk değerine sahip parçacıklar bir sonraki nesle aktarılarak yöntemin bulunan en iyi çözümler yönünde arama yapması sağlanmıştır [26]. Sürüdeki parçacıkların alt sürülere bölünmesi ve bu alt sürüler arasında çaprazlama işlecinin kullanılması parçacıkların uzayda çeşitlemesini[9] artırırken yerel minimum noktalara yakınsamasını engellemekte ve parçacıkların uzayda yeni bölgelerde arama yapmasını sağlamaktadır [34]. Mutasyon işleci yöntem ile beraber kullanılan en sık evrimsel işleçlerden biridir. Mutasyon işleci bir toplulukta farklı bireyler ortaya çıkararak topluluktaki çeşitlemeyi artırmaktadır [35]. Ayrıca mutasyon işlecinin diğer evrimsel işleçlerden daha kolay uygulanması nedeniyle yöntemde daha sık kullanılmaktadır. Yöntemdeki kısıtlama katsayısının ve eylemsizlik ağırlık katsayılarının mutasyon işleci ile değiştirildiği ve parçacıkların çeşitlemesinin artırıldığı [36] çalışmasında ve parçacıkların yerel ve bütünsel arama kabiliyetini belirleyen φ_1^i ve φ_2^i öğrenme katsayılarının mutasyon işleci ile değiştirildiği [37] çalışmasında yöntemin başarımının artığı gözlemlenmiştir. Doğrusal olmayan mutasyon işleci kullanılarak parçacıkların konum bilgisinin değiştirildiği [38], Gauss ve Cauchy mutasyon işleçleri kullanılarak parçacıkların konum ve hız bilgisinin değiştirildiği [39] ve [40] çalışmalarında yöntemin özellikle çok doruklu fonksiyonların eniyilenmesinde başarımının arttığı gözlemlenmiştir.

Evrimsel olmayan yöntemler de kullanılarak melez parçacık sürü eniyileme algoritmaları geliştirilmiştir. Bu çalışmaların bir kısmında çeşitli koşullar göz önünde bulundurularak sürünün arama sırasında konumunu yeniden ilklendirmesi veya bulunduğu konumun yakınlarında aramaya devam etmesi benimsenerek sürünün daha uygun değerler bulunması amaçlanmıştır [21]. Bazı çalışmaların bakış açısı ise uzayda parçacıkların çeşitlemesinin artırılarak yerel minimum noktalara yakınsamadan arama yapmalarını sağlamak olmuştur [41–44]. [45] ve [46] çalışmalarında parçacıkların uzayda aranan bir bölgeyi bir daha aramaması anlayışı benimsenmiş böylece parçacıkların diğer evrensel minimum noktaları araması ve ilk anda bulunan evrensel minimum noktalara erken yakınsamaması sağlanmıştır. [47] çalışmasında bireyler arasındaki işbirliği felsefesi kabul edilmiş ve çözüm vektörünün (diğer sürülerdeki en iyi konum ve değer bilgisinin bulunduğu vektör) sürü sayısı kadar bileşene bölünmesi ve her sürünün sadece bir bileşeni eniyilemesi benimsenmiştir. Böylece arama uzayının bölündüğü ve başarımın artığı gözlemlenmiştir. [48] çalışmalarında parçacık hızlarının Kalman filtresi kullanılarak belirlenmesine odaklanılmış böylece

[9] ing: diversity

parçacıkların belirli bir bölgede detaylı arama yapabilecekleri ve yöntemin uygun çözümlerin olduğu bölgeleri hızlı bir şekilde yakınsama özelliğinin de korunacağı belirtilmiştir.

Yöntemin verimliliğinin artırılması amacıyla diğer evrimsel hesaplama yöntemleri ile beraber kullanıldığı çeşitli çalışmalar da literatürde bulunmaktadır. [49] çalışmasında sürü alt sürülere bölünmüş ve parçacık sürü eniyileme, genetik algoritma veya tırmanış arama algoritmalarının kullanıldığı öngörüsel[10] bir arama yöntemi sunulmuştur. Belirli kurallara göre uzaydaki alt sürüler belirtilen algoritmalar arasında geçiş yaparak arama işlemini gerçekleştirmektedirler. [50] çalışmasında yöntem karınca eniyileme yöntemi ile, [51] çalışmasında ise yöntem diferansiyel evrim algoritması ile beraber kullanılarak yöntemin melez biçimleri ortaya konulmuştur.

Yerel minimum noktalara yakınsama sorununun parçacıkların çeşitlemesinin arttırılmasıyla aşılacağı doğrultusundaki düşünceler yöntemin başarımını arttırmayı amaçlayan birçok çalışmada beyan edilmiştir. Öte yandan çeşitlenmenin çok arttırılması sonucunda ise arama işlemin çok uzun sürebileceği ve herhangi olumlu bir sonuç alınamayabileceği de belirlemiştir.

2.4. Parçacık Sürü Eniyileme Yönteminin Yakınsaklık Analizi

Yöntemin çeşitli denektaşı fonksiyonlarda evrensel minimum/maksimum noktalara yakın konumlara yakınsaması ve çeşitli uygulama alanlarında istenen sonuçları verdiği gözlemlenmesine karşın yöntemin yakınsaklık analizinin gerçekleştirilmesi, yöntemin dinamiklerinin daha iyi anlaşılması ve yöntemin başarımının arttırılması için yöntemde yapılacak değişikliklerin belirlenmesi için önem arz etmektedir.

Yöntemde parçacıkların dinamiklerini inceleyen ilk çalışmalardan biri [53] çalışmasıdır. Bu çalışmada bütünsel komşuluğa sahip parçacık sürü eniyileme yönteminin dinamik denklemleri basitleştirilerek öğrenme katsayılarının (φ_1, φ_2 ve $\varphi = \varphi_1 + \varphi_2$) tek bir parçacığın tek boyutlu bir arama uzayındaki gezingesindeki etkisi üzerinde çalışılmıştır. $\varphi > 4$ olduğu durumda parçacığın ıraksıyan bir zarf içinde salınım yaptığı ve parçacığın arama uzayında arama yaptığı, $0 < \varphi < 4$ olduğu durumda ise parçacık gezingesinin rasgele genlik ve frekansa sahip bir sinüs dalgası olduğu ve parçacığın arama uzayında bir noktaya yakınsadığı gözlemlenmiştir. Daha sonraki çalışmalarda parçacığın çok boyutlu bir uzaydaki hareketi ele alınmıştır ve parçacık gezingelerinin belirli bölgelerdeki (farklı φ değerine

[10] ing: heuristic

sahip bölgelerdeki) davranışı incelenmiştir [54]. [15, 16] çalışmalarında öne sürülen eylemsizlik ağırlık katsayısının parçacığın davranışını etkilediği ve bu nedenden dolayı parçacığın gezingesini belirlediği vurgulanmıştır. [20] çalışmasında parçacık gezingesi ayrık zamanda incelenmiş ve elde edilen sonuçlar ile parçacık gezingesi sürekli zamanda da incelenmiştir. Parçacıkların arama uzayındaki bütün bölgeleri araması ve bir noktaya yakınsamaları için kısıtlama katsayısı olan χ yöntemin dinamik denklemine eklenmiş ve kısıtlama katsayısının farklı değerleri için parçacıkların faz uzayındaki gezingeleri incelenmiştir. [55] çalışmasında parçacık dinamiklerini doğrusal olmayan geri beslemeli bir sistem olarak betimlemiş, parçacık dinamiklerinin yakınsaklık analizinin doğrusal olmayan geri beslemeli bir sistemin mutlak yakınsaklık problemi Lure yakınsaklık problemi olarak düşünülmüştür. Pasiflik kuramı ve Lyapunov kararlılık metodu kullanılarak denge noktasına yakınsama için gerekli koşullar sunulmuştur. Yapılan çalışmada sadece mutlak yakınsaklık konusunu sunulduğu ancak asıl amacın eniyileme işlemi sırasında yakınsaklığın korunabilmesi olduğuna dikkat çekmişlerdir. [25] çalışmasında yöntemin dinamik davranışını ile yakınsaklık analizini dinamik sistem kuramının sonuçlarını kullanarak incelemiştir. Yöntemin belirlenimci[11] biçimi ele alınarak parçacıkların gezingelerinin ve yakınsama biçimlerinin belirlenimci denklemlerindeki özdeğerlerin değişimine göre belirlendiği gözlemlenmiştir. [56] çalışmasında yöntemin eylemsizlik ağırlık katsayılı biçimi ele alınmıştır ve çok boyutlu uzayda rasgele bileşenlerin ($\varphi_1^i(t)$ ve $\varphi_2^i(t)$) parçacıkların uzaydaki gezingelerine olan etkisi de düşünülmüştür. Parçacıkların uzayda bir noktaya yakınsamaları için yöntemin katsayılarının ayarlanması ile ilgili çeşitli öngörüler de belirtilmiştir.

Yöntemin yakınsaklık analizi ile ilgili yapılan çalışmalarda katsayıların uygun seçildiği durumda uzayda bir noktaya yakınsayacağı belirtilmiştir, ancak bu noktanın bütünsel minimum nokta olamayabileceğinin üstünde durulmuştur.

2.5. Parçacık Sürü Eniyileme Yönteminin Uygulama Alanları

2.5.1. Dinamik İzleme

Dinamik izleme problemi tüm evrimsel hesaplama algoritmaları için çözülmesi zor bir problemdir. Bunun nedeni ise eniyilenecek fonksiyonun zamana göre biçiminin değişmesidir. Bu nedenden dolayı bir zaman anı için bulunan iyi bir sonuç sonraki bir zaman anı için iyi bir çözüm olmayabilir.

[11] ing: deterministic

Dinamik izleme probleminin çözüm yollarından biri bu tip problemlere parçacık sürü eniyileme algoritmasının uygulanmasıdır. Bu düşüncenin benimsendiği [57] çalışmasında parçacık sürü eniyileme yöntemi uyumluluk fonksiyonunun zamana göre biçiminin değişmesi için bir döndürme matrisi ile döndürülmesi ve uyumluluk fonksiyonuna Gauss dağılımlı rasgele bir terimin eklenmesi durumlarında uygulanmıştır. Sonuçlar yöntemin gürültüye karşı gürbüzlüğünü ortaya koymuştur ancak diğer dinamik ve gerçek zamanlı ortamlar için de denenmesinin gerekliliği belirtilmiştir. Başka bir çözüm yöntemi olarak fonksiyonun biçimi değiştiği zaman o zaman anına ait en iyi değerin sıfırlanması ve yeniden hesaplanması olarak ortaya konulmuştur [58]. Ancak bu çözüm sadece zamanla yavaş değişen fonksiyonlar için uygulanabilir olduğu belirtilmiştir. Yukarıdaki çözüm önerisi doğrultusunda eniyilenecek fonksiyonun biçimindeki değişimi gözlemlenmesi ve değişim olduğunda parçacıkların tekrar rasgele olarak yerleştirilmesi olası bir çözüm olarak sunulmuştur [59].

2.5.2. Çeşitli Mühendislik Problemlerinde Uygulamaları

Parçacık sürü eniyileme yöntemi yukarıda bahsedilen uygulama alanları dışında birçok mühendislik probleminin çözümünde özellikle pratik mühendislik problemlerin çözümünde başarıyla uygulanmıştır. Yöntem yapay sinir ağlarındaki bağlantılar arasındaki ağırlık ve nöronlardaki bias katsayılarının ayarlanması için kullanılmaktadır [60–63]. Diğer yöntemlerin aksine yöntemin katsayılarının dikkatli ayarlanması sinir ağının eğitimi için yeterlidir. Bunun dışında yöntem görüntü işleme problemlerinin [64, 65], yapay zeka ile oyun oynamanın ve öğrenmenin [66, 67] ve güç elektriği sistemleri tasarımının ve denetiminin [37, 68] ele alındığı çeşitli çalışmalarda uygulanmıştır. Genel olarak parçacık sürü eniyileme yöntemi eniyileme problemlerinde ve eniyileme problemi olarak betimlenebilen diğer mühendislik problemlerinde uygulanabilmektedir.

2.6. Tezin Amacı ve Konusu

Bu tez çalışmasında parçacık sürü eniyileme yönteminin dinamik komşuluklu, parçacık komşulukların zaman ile değiştiği, dağıtık, eşzamansız ve paralel biçimi ele alınmıştır. Yöntemin dinamik komşuluk biçiminde parçacık komşuluklarının belirlenmesi için olasılıklı ve mesafeye bağlı komşuluk kuralları belirlenmiş, sürüdeki zaman ile değişen komşuluk yapısı zaman ile değişen bir çizge yapısı olarak betimlenmiştir. Geliştirilen biçiminin yöntemin paralel ve dağıtık uygulamalarında daha uygun olacağı belirlenmiş ve çeşitli denektaşı fonksiyonlar kullanılarak

benzetimler yapılmıştır. Yöntemin dinamik komşuluklu, eşzamansız, dağıtık ve paralel biçiminde parçacıkların tamamen bağımsız zaman anlarında bilgi değişimi yapmalarına ve konumlarını güncellemelerine izin verilmiştir. Ayrıca parçacıklar arası bilgi değişimi gecikmeleri de düşünülerek güncel olmayan bilgiler de kullanılmış ve parçacıkların komşularının zaman ile değiştiği düşünülmüştür. Geliştirilen yöntemin matematiksel modeli paralel ve dağıtık hesaplama literatüründeki sonuçlar göz önünde bulundurularak ortaya konulmuş, yöntemin verimliliğinin gösterilmesi için tek bir bilgisayar ve yerel ağda bulunan bilgisayarlar kullanılarak çeşitli denektaşı fonksiyonların eniyilemesi gerçekleştirilmiştir. Yöntemin geliştirilen biçimleri çok erkinli bir sistemin bilinmeyen bir ortamda arama görevinde kullanılmış, erkinler eşzamansız bilgi paylaşımı ve konum güncellemeleri yaparak arama görevini gerçekleştirmişlerdir. Ayrıca haberleşen erkinler zaman ile değişen bir komşuluk yapısı olarak benimsenmiştir. Geliştirilen arama yönteminin verimliliğinin gösterilmesi için benzetimler ve gerçek robotlar ile uygulama yapılmıştır.

2.7. Parçacık Sürü Eniyileme Yönteminin Dinamik Komşuluklu ve Paralel Biçimleri ile İlgili Çalışmalar

Parçacık sürü eniyileme yöntemi doğadaki kuş ve balık sürülerinin toplumsal davranışından esinlenmiştir. Doğadaki sürüler beslenme amacıyla bulundukları ortamı ve konumları dinamik olarak değişmekte ve sürü üyeleri arasındaki etkileşimler zaman içinde değişmektedir. Ayrıca sürüdeki üyeler arasındaki etkileşimlerin iki yönlü olamayabileceği gerçeği etkileşimlerin dinamik olarak değişebileceğini göstermektedir. Bu nedenlerden dolayı yöntemin dinamik komşuluklu biçiminin daha uygun olabileceği öngörülmüş ve yöntemin dinamik komşuluk biçimi ile ilgili çeşitli çalışmalar gerçekleştirilmiştir. Yöntemin dinamik komşuluklu biçimi ile ilgili ilk çalışma [69] çalışmasında sunulmuştur. Sunulan yöntemde ilk olarak yerel komşuluk yapısı ile uzayda arama yapılmakta, daha sonra parçacık komşulukları genişletilerek bütünsel bir komşuluk yapısı ile arama yapılmaktadır. Parçacık komşulukları belirli bir parçacığın arama uzayında üstünde veya altında yer alan parçacıkların veya parçacığa arama uzayında belirli bir mesafeden yakın olan parçacıkların komşu olması ile belirlenmiştir. [70] çalışmasında sürünün birçok alt sürüye bölünerek uzayda arama yapması benimsenmiştir. Alt sürüler uzayda belirli bir zaman arama yaptıktan sonra tekrar büyük sürüyü oluşturmak için birleşmekte ve oluşan büyük sürü rasgele olarak alt sürülere bölünerek komşuluk yapısı dinamik olarak değişmektedir. Önerilen yöntemin çok doruklu fonksiyonlarda daha iyi başarım gösterdiği belirlenmiştir. [71] çalışmasında yöntemin dinamik komşuluklu biçimi çok

amaçlı[12] eniyileme probleminde uygulanmıştır. Eniyilenen ilk fonksiyon parçacıkların fonksiyon uzayındaki mesafelerine göre komşulukları belirlerken, ikinci fonksiyon ise parçacıkların uyumluluk değerlerini belirlemektedir. Daha sonraki çalışmada parçacıkların bütünsel en iyi Pareto çözümlerini de kullanılarak eniyileme işlemini gerçekleştirmişlerdir [72]. [73] çalışmasında komşuluk yapısının yönlü çizgeler olarak belirtilmiş ve tek yönlü komşuluk ilişkileri düşünülmüştür. Komşuluğun dinamik olarak tanımlanması için komşu sayısı birden fazla olan düğümlerden bir kenar koparılarak başka bir düğüme bağlanması ve belirli bir zaman sonra komşuluk çizgesinin yeniden tamamen ilklendirilmesi olmak üzere iki yöntem belirlenmiştir. [74] çalışmasında arama işleminin başında parçacık komşuluk yapısının yerel bir komşuluk yapısı olan çember komşuluk yapısı olarak belirlenerek arama sırasında parçacıklar arasındaki komşuluklar artırılarak bütünsel komşuluk yapısına ulaşılmış, böylece parçacık komşulukları dinamik olarak değişerek arama gerçekleştirilmiştir.

Paralel işleme büyük çaplı bir problemin küçük parçalara bölünmesi ve bu problemlerin aynı zamanda çözülmesi düşüncesine dayanan birçok işlemin aynı zamanda parelel olarak gerçekleştirildiği hesaplama biçimidir. Paralel işleme yüksek başarım hesaplama disiplininde uzun yıllar boyu kullanılan bir hesapalama biçimidir ancak fiziksel kısıtlar nedeniyle işlemci çalışma frekanslarının ölçeklendirilmesinin sınırlı olması nedeniyle bu hesaplama biçimine olan ilgili son yıllarda artmıştır. İşlemcilerin güç tüketiminin (dolayısıyla işlemciler tarafından yayılan ısının) büyük bir sorun teşkil etmesi paralel işleme mantığın bilgisayar mimarisindeki önemi daha da artmıştır. Paralel bilgisayarlar tek bir donanımın paralelliği desteklediği çok işlemcili/çekirdekli bilgisayarlar ve bir ağda bulunan bilgisayar kümeleri ve öbekleri olarak tanımlanabilir. Bu durumda paralel bilgisayar yaklaşımları çoklu işleme, bilgisayar kümelenmesi, paralel süper bilgisayarlar, dağıtık hesaplama ve ızgara[13] hesaplama olarak belirlenebilir.

Paralel işleme için gerekli paralel kodların geliştirilmesi bilinen ve geleneksel ardışık kodlara[14] göre daha zordur. Paralel kodların geliştirilmesinde birçok yazılım hatası, problemin küçük parçalarının aynı anda çözümü sırasında çözümler arasında yarış durumunun yaşanması ve farklı alt işlemlerini haberleşme problemleri en sık görülen problemler arasındadır. Bu etkenlerin dikkatlice ayarlanması paralel kodların ardaşık kodlara göre çok daha hızlı çalışmalarına ve paralel verimliliğin artmasına olanak sağlar.

[12] ing: multiobjective
[13] ing: grid
[14] ing: sequential codes

Karmaşık mühendislik eniyileme problemlerinin hesaplama yükünün yüksek olmasından dolayı paralel olarak çalışan eniyileme yöntemlerinin geliştirilmesi büyük önem kazanmıştır. Birçok karmaşık mühendislik eniyileme probleminde kullanılan paralel parçacık sürü eniyileme yöntemi literatürdeki çalışmalarda yer almaktadır. [75] çalışmasında yöntemin paralel biçimi parçacıkların farklı işlemcilerde uyumluluk değerlerinin hesaplaması olarak belirlenmiştir. İşlemciler arasında usta yamak[15] ilişkisi benimsenerek usta işlemci parçacıkların konum güncellemelerini gerçekleştirerek uyumluluk değerlerini hesaplayan yamak işlemcilere bu bilgiyi eşzamanlı olarak iletilmektedir. Daha sonraki çalışmada sistemdeki işlemcilere eşit olmayacak şekilde dağılan iş yükünün başarıma olan olumsuz etkisini en aza indirgenmesi ve paralel verimliliğinin artması için eşzamansız biçimi sunulmuştur [76]. Benzer bir yaklaşım [77] çalışmasında yamak işlemciler arasındaki iş yükünün dinamik olarak ayarlanmasında kullanılmıştır. [79] ve [78] çalışmalarında yöntemin paralel biçiminin modellenmesi ve haberleşme kurallarının belirlenmesi ile yöntemin başarımı ve paralel verimliliği artırılmıştır. Yukarıdaki çalışmalarda sunulan yöntemin paralel ve eşzamansız biçimleri işlemciler arasında usta yamak ilişkisi benimsendiğinden dolayı merkezi olmayan karar verebilme ve dağıtık özellikleri görülmemektedir. Öte yandan usta işlemcinin herhangi bir nedenden dolayı devre dışı kalması eniyileme işleminin başarısız olmasına neden olacaktır. Bu sorunun aşılması amacıyla bu tez çalışmasında yöntemin eşzamansız, paralel ve dağıtık çalışan ve parçacık komşuluklarının dinamik olarak değiştiği biçimi geliştirilmiştir.

2.8. Denektaşı Fonksiyonlar

Bu tezde parçacık sürü eniyileme yönteminin parçacık komşuluklarının zaman ile değiştiği ve eşzamansız, dağıtık biçimlerinin başarımının sınanması amacıyla çeşitli denektaşı fonksiyonlarının eniyilenmesi problemi ele alınmıştır. Benzetimlerde küre, Griewank, Rastrigin ve Rosenbrock fonksiyonları kullanılmıştır. Kullanılan bu dört fonksiyonun farklı özellikleri vardır ve yöntemin geliştirilen biçimlerinin farklı özelliklerdeki fonksiyonların evrensel en küçük noktalarına ne kadar yaklaştıkları gözlemlenerek sınanmıştır.

İlk denektaşı fonksiyonu olarak küre fonksiyonu ele alınmıştır. Küre fonksiyon tek doruklu[16] bir fonksiyondur ve evrensel en küçük noktası $x^* = [0, 0, \ldots, 0]$ dır ve bu noktaya karşılık gelen fonksiyon değeri $f(x^*) = 0$ olmaktadır (bakınız Şekil 2.2.). Fonksiyonun herhangi yerel minimum noktası bulunmadığı için birçok eğim

[15] ing: master-slave
[16] ing:unimodal

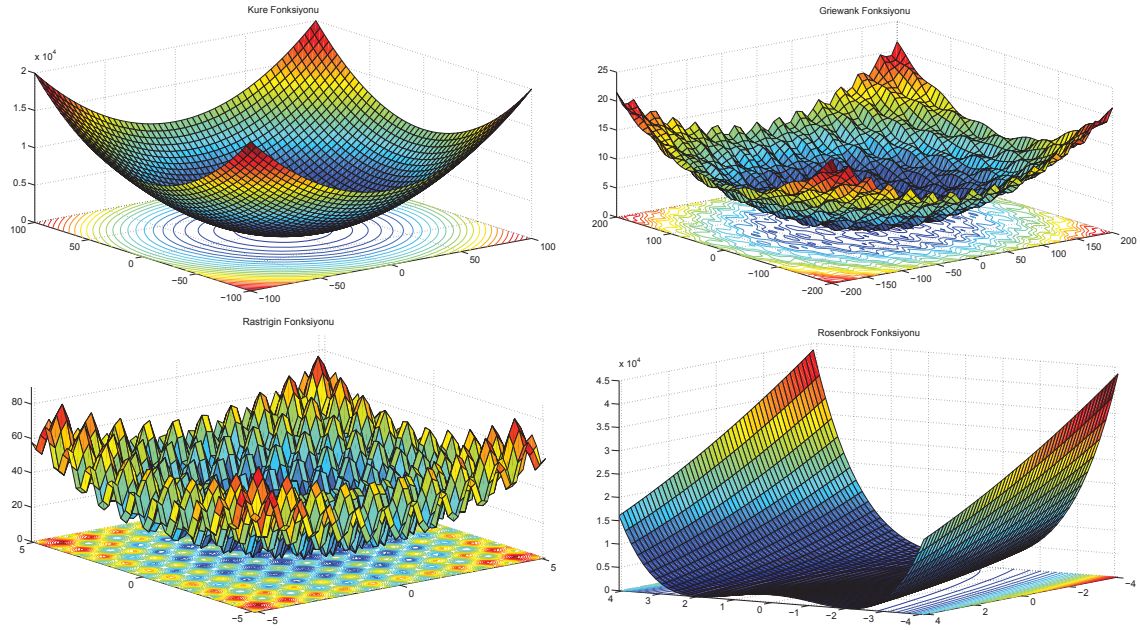
tabanlı olmayan doğrudan arama/eniyileme yöntemi hızlı ve kolay biçimde bütünsel en küçük noktayı bulabilmektedir. Ayrıca fonksiyonda tutarlı eğim değişimleri olduğu için birçok eğim tabanlı arama/eniyileme yöntemi de küre fonksiyonun bütünsel en küçük noktayı kolayca bulabilmektedir. Griewank fonksiyonu çok doruklu[17] bir fonksiyondur ve birçok yerel minimum noktaya sahiptir (bakınız Şekil 2.2.). Evrensel en küçük noktası $x^* = [0, 0, \ldots, 0]$ dır ve bu noktaya karşılık gelen fonksiyon değeri $f(x^*) = 0$ dır. Küre fonksiyonu ile benzerlikler taşımaktadır ancak küre fonksiyonunun aksine birçok yerel minimum noktası bulunmaktadır. Bu tür fonksiyonlarda eğim tabanlı arama yöntemlerin yerel minimum noktalara yakınsama riski çok yüksek olmasına karşın doğrudan arama yöntemleri eğim tabanlı yöntemlerden daha iyi sonuç vermektedir. Rastrigin fonksiyonu ise yine çok doruklu bir fonksiyondur ve evrensel minimum noktası $x^* = [0, 0, \ldots, 0]$ dır ve bu noktadaki değeri $f(x^*) = 0$ olmaktadır (bakınız Şekil 2.2.). Çok doruklu griewank fonksiyonunun aksine rastrigin fonksiyonunda yerel minimum noktaları arasındaki eğim değişimleri daha büyük olmakta ve bu durum eğim tabanlı yöntemlerinin bu fonksiyonda kullanılmasına elverişsiz kılmaktadır. Rastrigin fonksiyonu doğrudan olmayan arama/eniyileme yöntemleri için de eniyilenmesi zor bir fonksiyondur. Son olarak rosenbrock fonksiyonu çok doruklu bir fonksiyon olup, evrensel minimum noktası $x^* = [1, 1, \ldots, 1]$ de bulunmakta ve $f(x^*) = 0$ olmaktadır (bakınız Şekil 2.2.). Dikkat edilirse evrensek minimum nokta uzun, dar ve parabolik bir yarık içinde bulunmaktadır. Birçok arama yöntemi için bu yarığı bulmak kolaydır ancak bu yarık içindeki evrensel minimum noktayı bulmak zor bir işlemdir.

Çizelge 2.2. belirtilen denektaşı fonksiyonların matematiksel ifadelerini ve benzetimler sırasında göz önünde bulundurulan arama alanlarını vermektedir. Denektaşı fonksiyonların evrensel minimum noktalarını arama alanında ortalamak ve elde edilen sonuçların karşılaştırılabilir olması için literatürde sıkça kullanılan arama alanları boyutları seçilmiştir (bakınız [23, 25]) .

Çizelge 2.2. Denektaşı fonksiyonlar

No	Fonksiyon	Matematiksel İfade	Arama Alanı
1	Küre	$\sum_{i=1}^{n} x_i^2$	$[-100 \;\; 100]^n$
2	Griewank	$\sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} cos(\frac{x_i}{\sqrt{i}}) + 1$	$[-600 \;\; 600]^n$
3	Rastrigin	$\sum_{i=1}^{n} x_i^2 - 10cos(2\pi x_i) + 10$	$[-5.12 \;\; 5.12]^n$
4	Rosenbrock	$\sum_{i=1}^{n-1} (1 - x_i)^2 + 100(x_{i+1} - x_i^2)^2$	$[-2.048 \;\; 2.048]^n$

[17] ing:multimodal



Şekil 2.2.: Denektaşı fonksiyonların şekilleri: sol üst köşe küre, sağ üst köşe Griewank, sol alt köşe Rastrigin, sağ alt köşe Rosenbrock.

2.9. Parçacık Sürü Eniyileme Yönteminin Çok Erkinli Arama Görevinde Uygulandığı Çalışmalar

Parçacık sürü eniyileme yönteminin doğrudan bir arama yöntemi, hesaplama yükünün az ve dağıtık/merkezi olmayan biçimde çalışma özelliğine sahip olmasından dolayı yöntem çok erkinli sistemlerin arama görevinde sıkça kullanılmaktadır. [80] çalışmasında çok robotlu bir sistemin bir veya birden fazla hedefin bulunduğu bir ortamda arama görevini gerçekleştirmesi incelemişlerdir. Aramanın verimli gerçekleşmesi için yöntemin parametreleri olan eylemsizlik ağırlık katsayısının (w) ve öğrenme katsayılarının üst sınırları olan $\bar{\varphi}_1$ ve $\bar{\varphi}_2$ katsayılarının en iyi değerlerinin bulunmasına odaklanılmıştır ve iki seviyeli bir PSO algoritması kullanılmıştır. [81–83] çalışmalarında sistemdeki her erkin parçacık sürüsündeki bir parçacık olarak düşünülmüş ve her erkinin parçacık sürü eniyileme algoritmasını kullanarak hareketini planlaması benimsenmiştir. Robotların sadece bütünsel komşuluğun en iyi konumunu paylaştıkları ve robotlar arasındaki haberleşmenin en aza indirgenmesine odaklanılarak yöntemin verimliliği benzetim ve uygulamalar ile gösterilmiştir. [84] çalışmasında yöntemin çok erkinli sistemlerin arama görevine uyarlanmasını incelemişlerdir. Erkinlerin arama görevi sırasında bütünsel konumlarını bilmeleri ve erkinlerin arama görevi sırasında sadece kendi bilgilerini kullanmaları olmak üzere iki durum incelenmiştir. Öte yandan robotların sınırlı haberleşme alanına

sahip olmalarından dolayı komşuluğun dinamik olarak değişmesi de benimsenmiştir. [86,87] çalışmalarında dinamik olarak değişen bir ortamda koku kaynağının bulunması görevi için parçacık sürü eniyileme yönteminin dinamik olarak değişen ortamlara uygun bir biçimi geliştirilmiştir. Değişen ortamda parçacıkların yerel minimum noktalara takılmamaları için sürüdeki parçacıklar yüklü veya yüksüz parçacıklar olarak tanımlanarak parçacık çeşitlemesi artırılmış ve gerçekçi ortam koşulları göz önünde bulundurularak benzetimler yapılmıştır. [85] çalışmasında arama alanındaki koku kaynaklarını belirlemek amacıyla parçacık sürü eniyileme algoritmasından esinlenerek bir arama yöntemi geliştirmişlerdir. Yerel ve evrensel arama olarak iki seviyeli arama yöntemi geliştirilmiş ve parçacık sürü eniyileme yöntemi yerel aramada kullanılarak koku takibi ve koku kaynağının yerinin bulunması gerçekleştirilmiştir. [88] çalışmasında yerel ve evrensel arama olmak üzere arama işlemi iki seviyede yapılmaktadır. Hedefin sinyali robotlar tarafından algılanmadığı taktirde robotlar belirli bir bölgede yerel arama yaparak hedefin sinyalini algılamaya çalışmaktadırlar. Robotlar hedeften herhangi bir sinyal algıladıkları anlarda evrensel arama yöntemi olan parçacık sürü eniyileme yöntemi kullanılmaktadır. Ayrıca çalışmada robotların belirli bir haberleşme alanı olduğu ve robot komşuluklarının dinamik olarak değiştiği belirtilmiştir.

Yapılan çalışmalarda çok erkinli sistemlerin özelliklerinden biri olan eşzamansız çalışma özelliği ele alınmamıştır. Bu tez çalışmasında geliştirilen arama yöntemi erkinlerin eşzamansız olarak çalışmasına olanak sağlayarak sürüdeki haberleşme gecikmelerine karşın sürünün arama işlemine devam etmesini sağlamaktadır. Ayrıca erkinlerin haberleşme problemi yaşamalarına veya devre dışı kalmalarına karşın sistemin arama görevine devam edebilmesi de öngörülmüştür. Öte yandan erkinlerin arası etkileşimlerin zamana bağlı olarak değiştiği gerçeğinden yola çıkılarak, haberleşen erkinlerin komşuluk yapısı dinamik olarak değişen bir yapı olarak benimsenmiştir.

BÖLÜM 3

3. DİNAMİK KOMŞULUKLU PARÇACIK SÜRÜ ENİYİLEME YÖNTEMİ

Literatürde parçacık sürü eniyileme yöntemi ile ilgili yapılan çalışmaların ve uygulamaların büyük bir bölümünde parçacık komşulukları yöntemin başında belirlenir ve yöntemin yinelemeleri boyunca sabit kalır. Bu yöntemlerden birisi sürüdeki her parçacığın iki komşusu olacak şekilde (parçacığın sanal sağında ve solunda olmak üzere) yerel bir komşuluk yapısı olan çember (örgü) komşuluk yapısının tanımlamaktır. Diğer bir yöntem ise sürüdeki her parçacığın her diğer parçacık ile komşu olduğu bütünsel bir komşuluk yapısının tanımlanmasıdır. Yukarıda belirtilen parçacık komşuluklarının ve diğer komşuluk yapılarının yöntemin başarımına olan etkisi [30–32] çalışmalarında incelenmiştir.

Bu tez çalışmasının ilk amacı parçacık sürü eniyileme yönteminin dinamik (zaman ile değişen) parçacık komşuluk yapısına sahip biçiminin matematiksel modelinin geliştirilmesi ve değişen parçacık komşuluklarına göre yöntemin başarımının incelenmesidir. Parçacık sürü eniyileme yönteminin paralel ve dağıtık (merkezi olmayan) uygulamalarında yöntemin temel biçiminde benimsenen sabit parçacık komşuluk yapılarının yerine zaman ile değişen komşuluk yapılarının kullanılmasının daha uygun olacağı öngörülmüştür. Bu amaç ile parçacık komşuluklarının dinamik olarak belirlenmesi için rasgele belirleme ve uzaklık tabanlı yöntemler olan arama uzayında ve fonksiyon uzayında parçacıklar arası mesafelere göre komşuluk belirleme yöntemleri benimsenmiş ve parçacık komşuluklarının zaman ile değişen yönlü çizgeler kullanılarak betimlenmiştir. Bu bölümde sunulan çalışma ilk olarak [89] çalışmasında ortaya atılan fikrin geliştirilmesi ile elde edilen sonuçlardır. Elde edilen sonuçların bir bölümü [91] çalışmasında da yayınlanmıştır.

3.1. Parçacık Komşuluklarının Dinamik Olarak Belirlenmesi

Yukarıda da belirtildiği gibi bu bölümde parçacıkların komşuluk yapısının dinamik olarak zaman ile değiştiği kabul edilmiştir. Bir başka deyişle her adımda sürüdeki her parçacık zamana bağlı olarak sürüdeki diğer parçacıkların sadece bir kısmı bir başka deyişle sürünün bir alt kümesi ile haberleşebilmekte ve bu alt küme zaman ile değişmektedir. Sürüdeki bu alt küme parçacığın belirli bir adımdaki komşuları olarak adlandırılmıştır. Yöntemin farklı uygulamaları veya bir başka deyişle ele

alınan farklı biçimlerdeki eniyileme problemleri için sözü edilen alt kümenin (parçacık komşuluklarının) belirlenmesi için çeşitli yöntemlerin kullanılması mümkündür. Bu bölümde parçacık komşuluklarının belirlenmesi için üç farklı yöntem anlatılmaktadır. Ayrıca belirtilen komşuluk belirleme yöntemlerinin yöntemin başarımına olan etkisi de incelenecektir.

3.1.1. Arama Uzayında En Yakın Komşular Yöntemi

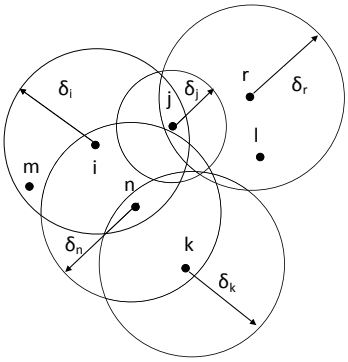
Parçacıkların komşularının belirlenmesi için olası bir yöntem arama uzayında parçacıklar arasındaki mesafelerin göz önünde bulundurularak en yakın komşular kuralından faydalanmaktır. Bu yöntemde her parçacığın bir algılama alanına sahip olduğu ve sadece bu algılama alanın içinde bulunan parçacıklar ile haberleşebilmektedir. (Şekil 3.1 bazı parçacıkların algılama/haberleşme alanlarını farklı yarıçaplardaki çemberler olarak göstermektedir). Parçacığın algılama alanları içinde bulunan parçacıklar parçacığın komşuları olarak düşünülmüştür. Sürüdeki i'inci parçacığın haberleşme/algılama alanı veya yarıçapı $\delta_i > 0$ olduğu düşünülürse, matematiksel olarak t zaman anında i'inci parçacığın komşuluğu

$$\mathcal{N}_i(t) = \{j, j \neq i \mid \|x_i(t) - x_j(t)\| \leq \delta_i\} \tag{3.1}$$

biçiminde betimlenir. Burada $\mathcal{N}_i(t)$ kümesi, $\delta_i > 0$ gibi bir algılama/haberleşme alanına sahip i'inci parçacığın t zaman anındaki komşularını göstermektedir. Öte yandan, genelde sürüdeki parçacıkların algılama/haberleşme alanları birbirlerinden farklı olabilir. Sürüdeki bütün parçacıkların algılama alanları eşit olduğu durumda bu haberleşme alanının boyutu olan herhangi bir $\delta > 0$ için $\delta_i = \delta$ olur. Eğer sürüdeki bütün parçacıklar aynı algılama/haberleşme alanına sahip ise komşuluk ilişkileri karşılıklıdır. Eğer t zaman anında i'inci parçacık j'inci parçacığın komşusu ise aynı zamanda j'inci parçacık da i'inci parçacığın komşusu olmaktadır. Aksi takdirde komşuluk ilişkileri karşılıklı olmayabilir. Bu durumda t zaman anında i'inci parçacığın j'inci parçacığın komşusu olması j'inci parçacığın da i'inci parçacığın komşusu olacağı anlamına gelmemektedir.

3.1.2. Fonksiyon Uzayında En Yakın Komşular Yöntemi

Yukarıda sunulan komşuluk belirleme yönteminde arama uzayında parçacıklar arasındaki mesafe göz önünde bulundurulmuştur. Benzer şekilde parçacıklar arasındaki komşuluk ilişkileri fonksiyon uzayında parçacıklar arasındaki mesafeler göz önünde bulundurularak matematiksel olarak t zaman anında i'inci parçacığın

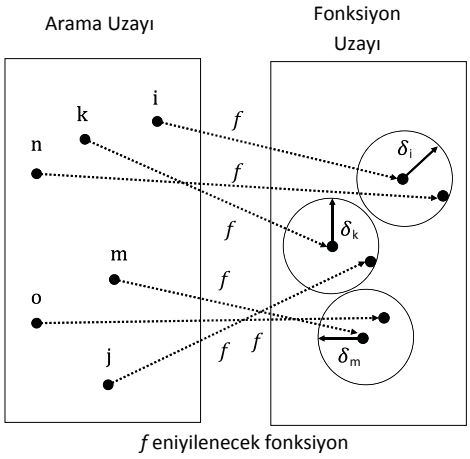


Şekil 3.1. Arama uzayında en yakın komşular yönteminin gösterimi.

komşuluğu

$$\mathcal{N}_i(t) = \{j, j \neq i \mid \|f(x_i(t)) - f(x_j(t))\| \leq \delta_i\} \tag{3.2}$$

biçiminde betimlenebilir. Burada $f(\cdot)$ ele alınan eniyilenecek fonksiyonu göstermektedir. Parçacıkların bu fonksiyondaki değerlerinin arasındaki fark ve parçacıkların algılama/haberleşme alanları olan δ_i göz önünde bulundurularak parçacıkların komşu olup olmadığı belirlenmektedir (bakınız Şekil 3.2). Önceki bölümde bahsedilen komşuluk belirleme yönteminde de olduğu gibi δ_i her parçacık için farklı olabilir ve parçacıklar arasındaki komşuluk ilişkileri karşılıklı olmayabilir. Eğer sürüdeki bütün parçacıklar aynı algılama/haberleşme alanına sahipler ise, yine her i parçacığı için $\delta_i = \delta$ olmakta ve parçacıkların komşuluk ilişkileri karşılıklı olmaktadır.



Şekil 3.2. Fonksiyon uzayında en yakın komşular yönteminin gösterimi.

3.1.3. Rasgele Belirleme Yöntemi

Parçacık komşuluklarının dinamik olarak belirlenmesi için bir diğer yöntem ise rasgele belirleme yöntemidir. Bu amaç ile yöntem uygulanmadan önce $0 < \epsilon < 1$ gibi bütünsel bir eşik olasılık değeri tanımlandığı ve her t anında her (i, j) parçacık çiftinin komşu olup olmadığının belirlenmesi için $\epsilon_{ij} \in [0, 1]$ gibi düzgün[1] rasgele bir sayı üretildiği ve ϵ_{ij} rasgele sayısının ϵ_{ji} rasgele sayısından bağımsız olarak üretildiği düşünülmüştür. Böylece her adımda toplam

$$2 \times \left(\begin{array}{c} N \\ 2 \end{array} \right) = N(N-1) \tag{3.3}$$

düzgün rasgele sayı üretilmiştir. Burada N sürüdeki parçacık sayısını belirtmektedir. Bu durumda i parçacığının komşuluğunun belirlenmesi için

$$\mathcal{N}_i(t) = \{j, j \neq i \mid \epsilon_{ij} \leq \epsilon\}. \tag{3.4}$$

denklemi kullanılmıştır. Bir başka deyişle eğer $\epsilon_{ij} \leq \epsilon$ ise t zaman anında j parçacığı i parçacığının komşusu olmakta ve t zaman anında i parçacığı j parçacığından bilgi alabilmektedir. Yine bu yöntemde ϵ_{ij} ve ϵ_{ji} birbirinden bağımsız düzgün rasgele sayılar oldukları için komşuluk ilişkileri karşılıklı olmak zorunda değildir. Komşuluk ilişkilerinin karşılıklı olabilmesi veya bir başka deyişle t zaman anında j parçacığının da i parçacığının komşusu olabilmesi için rasgele üretilen ϵ_{ji} rasgele sayısının da $\epsilon_{ji} \leq \epsilon$ koşulunu sağlaması gerekmektedir. Sürüdeki tüm parçacıklar için karşılıklı komşuluk ilişkilerinin istendiği durumda ise her adımda toplam

$$\left(\begin{array}{c} N \\ 2 \end{array} \right) = \frac{N(N-1)}{2} \tag{3.5}$$

rasgele sayı üretilir ($N(N-1)$ adet rasgele sayı yerine) ve $\epsilon_{ji} = \epsilon_{ij}$ olarak atanır.

Yukarıdaki yöntemin dışında parçacık komşuluklarının rasgele belirlenmesi için farklı yöntemler de kullanılabilir. Örneğin bir diğer yöntem olarak bütünsel bir eşik olasılık değeri ϵ yerine her i parçacığı için ϵ_i gibi farklı eşik değerleri atanarak her parçacık için farklı ortalama komşu sayısı belirlenebilir. Bu durumda sürüdeki parçacıklar daha sosyal ve ortalama komşu sayısı daha fazla olan parçacıklar veya daha az sosyal ve ortalama komşu sayısı daha az olan parçacıklar olarak betimlenerek sürüdeki parçacık

[1] ing:uniform

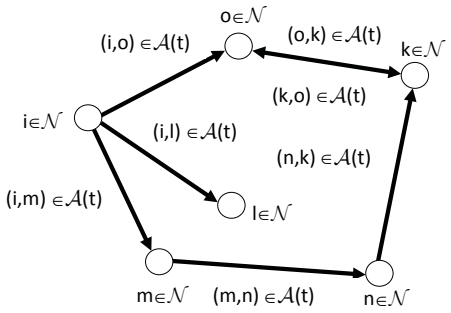
komşulukları farklı biçimde modellenebilir.

Parçacık sürü eniyileme yönteminin dağıtık, paralel ve eşzamansız uygulamalarında parçacık komşuluklarının zaman ile değişmesi uygulama sırasında kendiliğinden ortaya çıkan bir davranış olarak gözlemlenebilir. Bu durumda bu bölümde sunulan parçacık komşuluklarının rasgele belirlenmesi bu tip uygulamalar için daha uygun bir komşuluk belirleme yöntemi olarak görülebilir. Yöntemin eşzamansız biçiminin uygulandığı [90] çalışmasında bu tip bir komşuluk yapısı ele alınarak sunulan yöntemin matematiksel modeli geliştirilmiş ve yöntem bir fonksiyonun eniyilenmesi ile sınanmıştır.

3.1.4. Yönlü Çizgeler ile Komşuluk Gösterimi

Sürüdeki parçacık komşuluk yapısı ve bilgi akış yapısı yönlü çizgeler kullanılarak gösterilebilir. Bu tip bir gösterim sürüdeki bilgi akışı ve komşuluk yapısının görselliğine ve yöntemin yakınsaklık veya diğer özelliklerinin analizi gerçekleştirilirken çizge kuramının sonuçlarının kullanılmasına olanak sağlamaktadır. Bu amaç ile $\mathcal{G}(t) = (\mathcal{N}, \mathcal{A}(t))$ t zaman anında komşuluk (bilgi akışı) çizgesi olarak düşünülmüştür (bakınız Şekil 3.3.). Burada $\mathcal{N} = \{1, 2, \ldots, N\}$ kümesi sabit düğüm kümesini $\mathcal{A}(t) \subset \mathcal{N} \times \mathcal{N}$ ise t zaman anındaki yönlenmiş okları/kirişleri temsil etmektedir. Bu gösterimde i'inci parçacık $i \in \mathcal{N}$ sabit düğümü, $(i,j) \in \mathcal{A}(t)$ yönlü çizgesi/kirişi t zaman anında i parçacığından j parçacığına yönlenmiş olan bilgi akışını temsil etmektedir. Bir başka deyişle eğer $(i,j) \in \mathcal{A}(t)$ ise t zaman anında j parçacığı i parçacığından bilgi alabilmekte ve i parçacığı j parçacığının komşusu olmakta yani $i \in \mathcal{N}_j(t)$ olmaktadır. Ayrıca tekrar belirtmek gerekirse genelde sürüdeki/çizgedeki bilgi akışı tek yönlü olmakta ve $(i,j) \in \mathcal{A}(t)$ olması $(j,i) \in \mathcal{A}(t)$ olacağı anlamına gelmemektedir. Sözü edilen durum Şekil 3.3.'te de görülmektedir. Şekil 3.3.'te m parçacığından n parçacığına yönlenmiş olan $(m,n) \in \mathcal{A}(t)$ okunun/kirişin varlığı m parçacığından n parçacığına bilgi akışı olduğunu belirtmektedir ancak n parçacığından m parçacığına yönlenmiş bir okun/kirişin olmaması n parçacığından m parçacığına herhangi bir bilgisi akışının olmadığını, bu nedenden dolayı m ve n parçacıklarının karşılıklı komşu olmamaktadır. Öte yandan çizgede $(k,o) \in \mathcal{A}(t)$ ve $(o,k) \in \mathcal{A}(t)$ yönlenmiş oklarının/kirişlerinin varlığı o ve k parçacıkları arasında karşılıklı bilgi akışı olduğu ve bu iki parçacığın karşılıklı komşu oldukları anlamına gelmektedir. Karşılıklı komşuluk ilişkilerinin olabilmesi için daha önceki bölümde ele alınan komşuluk yöntemlerinde belirtilen koşulların sağlanması gerekmektedir.

Temel parçacık sürü eniyileme yönteminin uygulamalarında, parçacık komşulukları



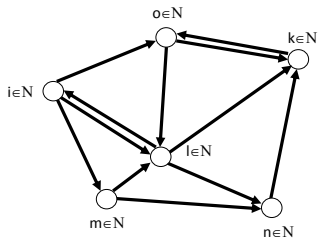
Şekil 3.3. Parçacık komşuluklarının yönlü çizgeler ile gösterimi.

sabit olarak seçilmekte ve yöntemin yinelemeleri boyunca her t anı için $\mathcal{G}(t) = \mathcal{G}$ gibi önceden belirlenmiş sabit bir komşuluk yapısı olmaktadır. Bu bölümde geliştirilen yöntemde parçacık komşuluk yapısının dinamik olarak değişimine izin verilmiştir. Açıkca görüldüğü üzere komşuluk yapısındaki bağlantı özellikleri sistemdeki bilgi akışını belirlemekte ve yöntemin başarımını büyük ölçüde etkilemektedir. Örneğin $\mathcal{G}(t)$ haberleşme/bilgi akışı çizgesi tam bağlı değil ise parçacık sürüsü iki veya daha fazla alt sürüye bölünür ve bu alt sürüler birbirinden bağımsız arama gerçekleştirerek arama uzayında farklı noktalara yakınsayabilirler. Bu durumun engellenmesi veya belirlenmesi için etkileşim/haberleşme çizgelerinin bağlanırlık özelliklerine çeşitli koşullar konulmalı ve bu koşulların sağlanıp sağlanmadığı incelenmelidir. Bu koşullar bölümün ilerleyen kısımlarında tartışılacaktır.

Bu koşulları sunmadan önce çizgenin yapısının daha iyi ifade edilebilmesi için çeşitli tanımlamalara ihtiyaç vardır. Bir çizgede i düğümünden j düğümüne yönlü bir ok/kiriş var ise i düğümü j düğümüne bağlı olmaktadır. Bir başka deyişle i düğümünden j düğümüne $i = i_1$, $j = i_p$ olacak biçimde $(i_1, i_2), (i_2, i_3), \ldots, (i_{p-1}, i_p)$ şeklinde oklar dizisi var ise i düğümü j düğümüne bağlıdır (bakınız Şekil 3.4.). Örneğin Şekil 3.4.'de i düğümü k düğümüne bağlıdır çünkü çizgede (i, o) ve (o, k) okları/kirişleri (veya (i, l) ve (l, k) okları/kirişleri) bulunmaktadır. Eğer çizgedeki her i düğümünden her j düğümüne bir yol var ise çizgeye güçlü şekilde bağlı[2] denir. Burada dikkat edilmesi gereken husus bir çizgenin güçlü bir şekilde bağlı olması için çizgedeki her i düğümünden her j düğümüne doğrudan bir ok/kiriş olması gerekmemesidir. Bu düğümler arasında ara düğümlerin oluşturduğu bir yol olması, bir başka deyişle çizgedeki her i düğümden her j düğümüne ara düğümlerden geçen bir oklar/kirişler dizisinin varlığı yeterlidir. Şekil 3.4.'de belirtilen çizge bu koşulu sağladığı için güçlü bir şekilde bağlı çizgedir. Öte yandan belirtilen çizgede (l, i) oku/kirişi

[2]ing: strongly connected

olmasaydı, çizge güçlü bir şekilde bağlı olmayacaktı çünkü i düğümü çizgedeki diğer düğümlerden bağlı olmayacak ve bilgi alamayacaktı.



Şekil 3.4. Güçlü şekilde bağlı çizge.

Sürüdeki parçacık sayısının, çizgedeki düğüm sayısının ve bu sebeble sınırlı olmasından dolayı olası komşuluk (haberleşme) çizgelerinin sayısı da sınırlıdır. $\bar{\mathcal{G}} = \{\mathcal{G}_1, \ldots, \mathcal{G}_M\}$ çizge kümesi parçacık sürüsünde olası bütün komşuluk (haberleşme) çizgelerini temsil etsin. Bu durumda her t zaman anı için $\mathcal{G}(t) \in \bar{\mathcal{G}}$ dir. Yöntemin başarımı ve sürüdeki bilgi akışı için anlık $\mathcal{G}(t)$ haberleşme/komşuluk çizgesinin özelliklerinden çok haberleşme/komşuluk çizgeleri dizisi olan $\{\mathcal{G}(t)\}$'nin özellikleri daha önemlidir. Bu durumda $\{\mathcal{G}(t)\}$ çizgeler dizisindeki çizgelerin birleşimi önem kazanmaktadır. Aynı düğüm kümesine sahip $\{\mathcal{G}_i = (\mathcal{N}, \mathcal{A}_i)\} \subset \bar{\mathcal{G}}$ kümesindeki çizgelerin birleşimi $\cup\mathcal{G}_i = (\mathcal{N}, \cup\mathcal{A}_i)$ biçiminde tanımlansın. Eğer $\mathcal{I}$ gibi bir zaman aralığında $\cup_{t\in\mathcal{I}}\mathcal{G}(t)$ birleşiminde bir kapsayan ağac[3] mevcut ise $\{\mathcal{G}(t)\}$ çizge dizisinin de $\mathcal{I}$ aralığında kapsayan bir ağacı mevcuttur denir. Dikkat edilirse bu durum $\mathcal{I}$ aralığında en az bir düğümün çizgedeki diğer tüm düğümlere bağlı olduğunu ve bu düğümdeki bilginin diğer tüm düğümlere yayılabileceğini göstermektedir.

3.2. Dinamik Komşuluk

PSO eniyileme problemi bazı açılardan çok erkinli sistemlerin uzlaşma ve çok erkinli sistemlerin toplu olarak beslenme (doğadaki sürüler gibi) problemleri ile benzerlikler taşımaktadır. Dinamik komşuluk doğadaki sürülerde olduğu gibi parçacığın farklı zamanlarda farklı komşuluklarda olmasına olanak sağlamaktadır. Yöntemin ilham kaynağı olan kuş sürüleri gibi doğadaki sürülerde karşılıklı haberleşme olanağı bulunmayabilir ve sadece tek yönlü haberleşme mümkün olabilir. Öte yandan yöntemin tek yönlü haberleşmeli ve değişken komşuluklu gerçeklenmesi yöntemin dağıtık ve eşzamansız uygulamalarında daha uygun olabilir ve yöntemin yeni uygulama alanlarında kullanılmasına olanak tanıyabilir. Örneğin yöntemin bu

[3]ing: spanning tree

biçimde gerçeklenmesi parçacık sürü eniyileme yöntemi tabanlı çok robotlu arama yöntemlerinin geliştirilmesine olanak sağlayabilir. Daha öncede belirtildiği gibi çok robotlu sistemler gibi örneklemeli sistemlerde arama görevi için eğim tabanlı yöntemlerin uygulanmasının zorluğundan ötürü parçacık sürü eniyileme yönteminin bu tip görevler için kullanılmasının daha uygun olduğu görülmüştür. Ayrıca bu tip sistemlerdeki robotlar sınırlı algılama/haberleşme alanlarına sahiplerdir ve bu alan dışındaki robotlar ile haberleşememektedirler. Sistemdeki bir robot sürüdeki bir parçacık olarak düşünülürse robot arama sırasında sistemdeki bazı robotlar ile haberleşebilecektir. Öte yandan robotların algılama/haberleşme alanlarında farklı zaman anlarında farklı robotlar bulunacağından ve robotlar arasındaki haberleşmenin karşılıklı olamayabileceğinden dolayı dinamik komşuluk yapısının daha uygun olacağı düşünülmüştür. Bu bölümde sunulan yöntemin dinamik komşuluklu biçimi yukarıda belirtilen özelliklerin gerçekleşmesine olanak sağlamaktadır.

Parçacık komşuluklarının dinamik olarak fonksiyon uzayında parçacıklar arasındaki mesafelere göre belirlendiği yöntem genetik algoritmalarda bulunan seçkinci[4] çaprazlama yöntemlerine benzemektedir. Bazı problemlerde yakınsamayı hızlandırabilirken bazı problemlerde sürünün alt sürülere bölünerek arama yapmasını ve farklı noktalara yakınsamaya neden olabilir (bu durum bazı problemlerde istenen bir durum olabilir örneğin arama uzayında birden fazla minimum noktayı bulmak gibi). Öte yandan sürünün bu şekilde alt sürülere bölünmesi sözü edilen bütün komşuluk belirleme yöntemlerde olabilecek bir durumdur. Bu durumun belirlenmesi veya engellenmesi için gerekli şartlardan bu bölümün sonraki kısımlarında sunulacaktır.

Parçacık komşuluklarının rasgele belirlendiği durum ise yönteme fazladan bir rasgelelik hali getirmektedir (parçacığın elde ettiği en iyi konumu ve komşuluğun en iyi konumu yönüne rasgele atılan adımlara ek olarak). Dikkat edilirse yukarıda sözü edilen parçacıklar arasındaki mesafelere göre komşuluğun belirlendiği yöntemler belirlenimci özelliktedir (komşuluğun belirlenmesi için herhangi bir rasgelelik kullanılmamıştır). Bu durumda parçacık komşulukları ardışık adımlarda büyük ölçüde değişmemekte ve sürüdeki parçacıklar diğer alt kümelerde bulunan parçacıklar ile bilgi değişimi olanağı azdır. Bu durumun sonucu olarak sürüdeki parçacıklar ardışık adımlarda sınırlı sayıdaki arama yönünde arama yapmaktadır. Her parçacığın her adımda rasgele seçilen parçacık alt kümeleri, böylece farklı parçacık alt kümeleri, ile bilgi değişimi yapmasına izin verilmesi yöntemin değişik arama yönlerinde arama yapmasını sağlarken yöntemin başarımının artmasını sağlayabilir.

[4] ing: elitist

Öte yandan parçacık komşuluklarının belirlenmesi yukarıdaki yöntemlerle sınırlı değildir ve bu amaçla yukarıdakilerden farklı komşuluk belirleme yöntemleri de kullanılabilir. Örneğin yukarıda sözü edilen yöntemlerin birinin veya birkaçının kullanıldığı melez komşuluk belirleme yöntemleri farklı bir alternatif olabilir.

Daha önceki kısımlarda yöntemin dinamik komşuluklu biçimi yöntemin paralel ve eşzamansız gerçeklemelerine daha uygun olduğu ve bu tip gerçeklemelerde yöntemin başarımını artırabileceği belirtilmiştir. Yöntemin bir bilgisayar ağında bulunan birkaç işlemcide paralel olarak gerçeklendiği ve ağdaki işlemcilerin bilgi değişimi yaptığı bir uygulama sözü edilen duruma bir örnektir. Ağda bulunan bütün işlemciler çeşitli nedenler (örneğin ağdaki aşırı veri trafiğinden dolayı) yüzünden her adımda birbirleri veya belirli bir komşu işlemci ile haberleşemeyebilir ve sadece ağdaki bazı işlemciler ile haberleşebilir. Bu durumda işlemciler/parçacıklar diğer işlemciler/parçacıklar ile bağlantı kurulmasını beklemek yerine sahip oldukları komşuluğun en iyi uyumluluk değerini ($evrensel_{eniyi}$) daha önceki anlarda bağlantı kurabildiği ve bilgi alabildiği işlemcilerden elde ettiği bilgileri kullanarak günceller ve yinelemelerine devam eder. Komşuluk yapısının sabit olduğu durumda ise işlemcilerin yinelemelerine devam edebilmeleri için ağdaki bütün komşu işlemciler ile bağlantı kurması ve bilgi paylaşımında bulunması zorunludur. Yöntemin sözü edilen biçimde eşzamansız gerçeklenmesi [90] çalışmasında eşzamansız konum ve hız vektörleri güncellemeleri göz önünde bulundurularak basit bir örnek üzerinden sağlanmıştır ve benzetim sonuçları ile yöntemin başarımı sınanmıştır. Bu tezde ise bu çalışma daha da geliştirilerek çeşitli denektaşı problemler üzerinde yöntemin başarımı sınanacaktır. Bu kısım Bölüm 4.'de daha ayrıntılı olarak anlatılacaktır.

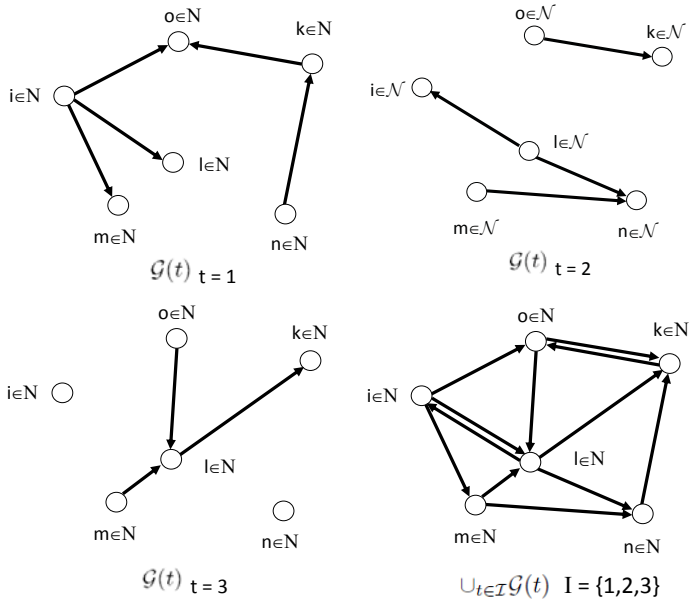
Daha önceden belirtildiği gibi parçacık komşuluklarının zaman ile değişmesi sürünün birden fazla alt sürüye bölünmesine neden olabilir. Sürünün parçalanmaması ve bilgi akışının sürekliliğinin sağlanması için aşağıdaki varsayımların sağlanması gerekmektedir.

Varsayım 1 *Öyle bir sabit $I > 0$ vardır ki $\{\mathcal{G}(t) = (\mathcal{N}, \mathcal{A}(t))\}$ haberleşme çizgeleri dizisi I uzunluğundaki her $\mathcal{I}$ zaman aralığında güçlü bir şekilde bağlıdır.*

Yukarıda belirtilen varsayım çizgenin bağlantılarındaki düzgünlük varsayımı olarak düşünülebilir. Bir başka deyişle bu varsayım haberleşme/etkileşim dizisinin birlikte düzgün şekilde güçlü bağlı[5] olduğunu belirtmektedir. Şekil 3.5. Varsayım 1'in

[5]ing: jointly uniformly strongly connected

yönlü çizgeler ile gösterimini göstermektedir. Bu bölüme ilham kaynağı olan [89] çalışmasında belirtilen $\{\mathcal{G}(t) = (\mathcal{N}, \mathcal{A}(t))\}$ haberleşme çizgeleri dizisinin birleşiminde kapsayan ağacın varlığının sürüdeki bilgi akışının sürekliliğini ve sürünün birden fazla alt sürüye bölünmemesi için yeterli bir koşul (çizgenin bağlı ancak güçlü bir şekilde bağlı olmasına gerek olmamasına) olacağı öne sürülmüştür. Belirtilen bu koşul parçacık komşulukların karşılıklı olması durumunda veya bir başka deyişle çift yönlü haberleşmenin olduğu durumlarda yeterli bir koşuldur. Parçacıklar arasında tek yönlü komşuluk ilişkilerinin olduğu durumda ise haberleşme/etkileşim çizgeleri dizisinin birleşiminde kapsayan ağacın varlığı sürüdeki bütün parçacıkların güncel bilgilere ulaşacağını garanti edemez ve bu durumda bilgi akışının devamlılığı için çizge dizisinin birleşiminin güçlü bir şekilde bağlı olması gerekmektedir. Öte yandan çizgede haberleşme karşılıklı ise, yukarıda belirtilen kapsayan ağaç varsayımı çizge dizisinin birleşiminin güçlü bir şekilde bağlı olması varsayımı ile eşdeğer olmaktadır. Bu nedenden dolayı sürüdeki her parçacığın her diğer parçacığının bilgisine erişmesi için haberleşme/etkileşim çizgeleri dizisinin birleşiminin güçlü bir şekilde bağlı olması gerekmektedir. Haberleşmenin karşılıklı olduğu durumlarda yukarıdaki varsayımlar daha da basitleştirilebilir (çok erkinli sistemlerde olduğu gibi [93, 94]).



Şekil 3.5. Varsayım 1 yönlü çizgeler ile gösterimi.

Daha önceki bölümlerde belirtilen komşuluk belirleme yöntemlerinin başarımlarını etkileyen çeşitli katsayılar ve durumlar bulunmaktadır. Örneğin, arama uzayında en yakın komşular yönteminde arama uzayında bulunan parçacık sayısı ve arama uzayının büyüklüğü, bir başka deyişle arama uzayındaki parçacık yoğunluğu, başarımını

etkileyen önemli unsurlar arasındadır. Arama uzayında parçacık yoğunluğunun az olması, parçacıkların haberleşme/etkileşim çizgesinin güçlü bir şekilde bağlı olma olasılığını parçacıkların algılama alanları içerisinde yeterli sayıda parçacık olmamasından ötürü azaltmaktadır. Arama uzayında parçacık yoğunluğunun artması ile parçacıkların bilgi değişimi yapabilecekleri komşu sayısı artmakta, böylece sürüdeki haberleşme/etkileşim çizgesinin güçlü bir şekilde bağlı olması ve Varsayım 1'in sağlanma olasılığı artmaktadır. Öte yandan fonksiyon uzayında en yakın komşular yönteminde arama uzayında bulunan parçacık sayısı ve arama uzayının büyüklüğü yanında eniyilenmekte olan fonksiyon da başarımını etkilemektedir. Parçacıkların algılama/haberleşme alanlarını belirlenmesi için eniyilenmekte olan fonksiyonun en büyük ve en küçük değerleri önem arz etmektedir. Bir başka deyişle fonksiyon uzayındaki parçacık yoğunluğu yöntemin başarımını etkileyen en önemli unsurdur.

Parçacık komşuluklarının rasgele belirlendiği yöntemde başarım sadece arama uzayındaki parçacık sayısına bağımlıdır ve bu sayı Varsayım 1'in sağlanması için önemli bir parametre olmaktadır. Dikkat edilirse bu yöntemde komşulukların belirlenmesi için sürüdeki her (i, j) parçacık çifti için ϵ_{ij} gibi bir komşu olma olasılığı atanmakta ve bu olasılık ϵ bütünsel eşik değeri olasılığı ile karşılaştırılmaktadır. Sabit bir ϵ değeri için arama uzayında parçacık sayısının artırılması ile sürüdeki parçacık çiftlerinin birbirleri ile komşu olma olasılığı artar ve aralarında bilgi paylaşımı yapan parçacıkların sayısı artma eğilimi gösterir. Böylece sürüdeki haberleşme/etkileşim çizgesinin güçlü bir şekilde bağlı olma olasılığı artmakta ve Varsayım 1 sağlanabilmektedir.

Haberleşme/etkileşim çizgelerinin sabit yapıya sahip olduğu durumlarda, sürüdeki bütünsel bağlanırlık özelliğinin sağlanması için Varsayım 1 aşağıdaki biçimde ifade edilebilir.

Varsayım 2 *Sabit etkileşim çizgesi* $\{\mathcal{G} = (\mathcal{N}, \mathcal{A})\}$ *güçlü bir şekilde bağlıdır.*

Dikkat edilirse Varsayım 2 sürüdeki parçacıkların ortak bir noktaya yakınsamaları için asgari gerekliliktir. Haberleşme/etkileşim çizgesi güçlü bir şekilde bağlı değil ise, sürüde en az bir parçacık diğer parçacıklara bağlı değildir ve bu durum bu parçacığın doğrudan veya dolaylı olarak sürüdeki diğer parçacıklardan bilgi alamayacağı anlamına gelir. Bu gerçek sürüdeki parçacıkların ortak bir konuma yakınsamasını engelleyebilir. Varsayım 1 ve Varsayım 2 çok erkinli sistemlerin

toplanma davranışı için gerekli koşulları sağlamaktadır [93,95]. Bu bağlamda sürüdeki parçacıklar arasındaki bilgi akışı sürekliliği garanti edilmektedir.

Dikkat edilmesi gereken önemli bir husus da sürünün alt sürülere ayrılmasının her zaman yöntemin başarımının aleyhine olan bir özellik olmamasıdır. Arama uzayında birden fazla en küçük/en büyük değerli konumun saptanması için sürünün alt sürülere ayrılıp, arama uzayının değişik bölgelerinde arama yapmaları gerektiği durumlarda bu durum yöntemin başarımının lehine olan bir durum olabilir. Öte yandan bu durum doğrudan eniyilenecek fonksiyonda, dolayısıyla PSO yinelemelerinde, belirtilmemiş diğer en küçük değerli konum tabanlı koşulların kullanıcı tarafından serbestçe seçilmesine olanak sağlayabilir. Bu bakış açısında, yukarıda belirtilen varsayımlar sürünün alt sürülere bölünmeden arama yapmasının istendiği durumlar için gerekli şartları sağlamaktadır.

Yöntemin çok robotlu bir sistemin arama görevinde uygulanmasında ve bir bilgisayar ağında paralel ve eşzamansız olarak gerçeklenmesinde parçacık komşuluklarının dinamik olarak değişmesi kendiliğinden ortaya çıkan bir davranış olarak göze çarpmaktadır. Çok robotlu bir sistemde sistemdeki her robotun sınırlı algılama/haberleşme alanına sahip olması nedeniyle robotlar sadece algılama/haberleşme alanları içerisindeki robot ile haberleşebilmekte ve farklı zaman anlarında farklı robotlar robotların algılama/haberleşme alanlarında bulunabilmektedir. Arama boyunca haberleşen robotların komşuluk yapısının dinamik olarak değişmesi uygulama sırasında ortaya çıkan bir davranış olmakta ve bu tip bir komşuluk yapısı bu sistemler için daha gerçekçi olan karşılıklı olmayan haberleşmeye de olanak sağlamaktadır. Yöntemin bir bilgisayar ağında paralel ve eşzamansız biçimde gerçeklenmesinde ağdaki yoğun veri trafiği yüzünden veya işlemcinin/parçacığın o zaman anında yapması gereken başka işlemlerden ötürü o zaman anında diğer işlemciler/parçacıklar ile bilgi değişimi yapılamayabilir. Yukarıda belirtilen nedenlerden dolayı haberleşebilen işlemciler/parçacıklar farklı zaman anlarında farklı olabilir ve haberleşen işlemcilerin komşuluk yapısının dinamik olarak değişmesi kendiliğinden ortaya çıkan bir davranış olarak gözlemlenebilir. Yöntemin sabit komşuluklu (ağdaki her işlemci/parçacık her diğer işlemci/parçacık ile haberleşmekte ve bilgi değişimi yapmakta) biçiminde ise ağdaki her işlemci/parçacık bir sonraki yinelemeye geçmesi için ağda bulunan her diğer işlemciyi/parçacığı işlemlerini farklı sürelerde bitirmelerinden dolayı beklemek zorundadır. Sözü edilen durum özellikle ağda yoğun veri trafiği olduğu durumlarda istenmeyen bir durumdur.

Parçacık komşulukları önceki bölümlerde belirtilen denklemler göz önünde

bulundurularak yapay olarak belirlendiğinde dinamik komşuluk kendiliğinden ortaya çıkan bir davranış olmamakta ve algoritmaya fazladan hesaplama yükü getirmektedir. Bu hesaplama yükü sürüdeki parçacık sayısına bağlıdır. Örneğin arama uzayında en yakın komşular yönteminde her adımda parçacıklar arası mesafelerinin $\mathbb{R}^n$ uzayında hesaplanması için $N(N-1)/2$ sayıda ve karşılaştırma için $N(N-1)$ sayıda işlem yapılmaktadır. Fonksiyon uzayında en yakın komşular yönteminde her adımda parçacıklar arasındaki mesafelerin $\mathbb{R}$ uzayında hesaplanması için $N(N-1)/2$ sayıda ve karşılaştırma için $N(N-1)$ sayıda işlem yapılmaktadır. Komşulukları rasgele belirleme yönteminde ise $N(N-1)$ adet rasgele sayı üretme ve aynı sayıda karşılaştırma gerçekleştirilmektedir.

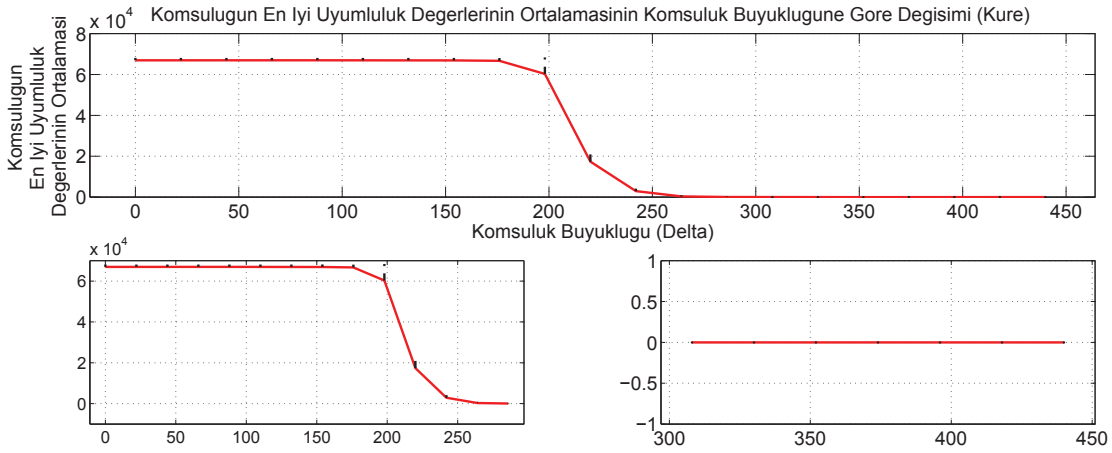
3.3. Benzetim Sonuçları

Benzetimlerde bütün denektaşı fonksiyonlar için arama uzayının boyutu $n = 20$ (arama uzayı $\mathbb{R}^{20}$ kümesinin bir alt kümesi) ve uzaydaki parçacık sayısı 100 olarak belirlenmiştir. Her denektaşı fonksiyon için belirlenen arama alanları göz önünde bulundurularak 30 adet ilk koşul üretilmiştir (bakınız Çizelge 2.2.). Her adımın sonunda komşuluğun en iyi uyumluluk değeri kaydedilmiştir. Çalışmada [20] çalışmasında geliştirilen yöntemin kısıtlama katsayılı biçimi kullanılmış (bakınız Denklem 2.3) ve öğrenme katsayılarının üst sınırları olan $\bar{\varphi}_1$ ve $\bar{\varphi}_2$ parametre değerleri 2.05 alınarak kısıtlama katsayısı Denklem 2.3'e göre $\chi = 0.7298$ olarak hesaplanmıştır. (Elde edilen sonuçlar Çizelge 3.1., Çizelge 3.2. ve Çizelge 3.3.'te belirtilmiş ve e-100 mertebesinde ve e-100 mertebesinden daha küçük değerler 0 olarak kabul edilmiştir. Çizelgelerde 1, 2, 3, 4 olarak numaralandırılan fonksiyonlar sırasıyla küre, Griewank, Rastrigin ve Rosenbrock fonksiyonlarıdır.)

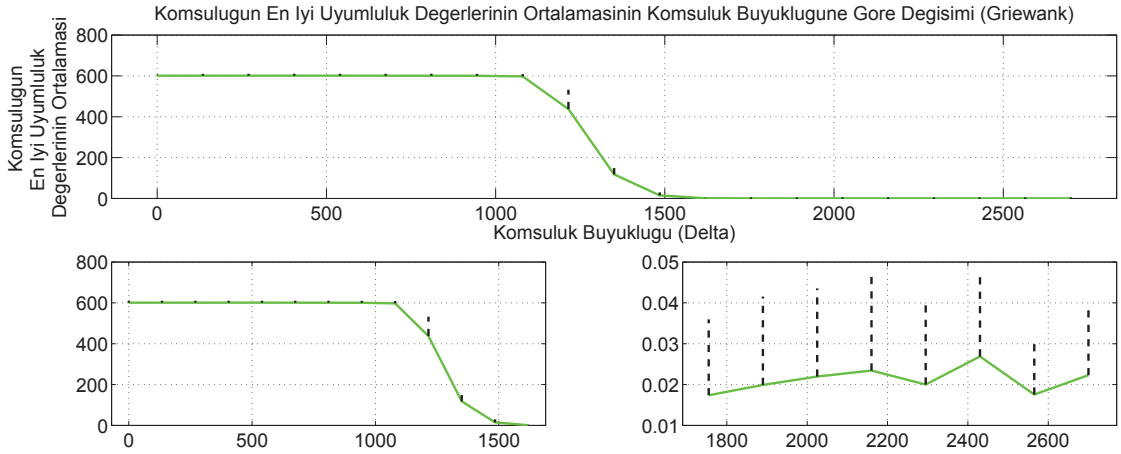
Şekil 3.6., Şekil 3.7., Şekil 3.8. ve Şekil 3.9. ele alınan dört denektaşı fonksiyon için komşuluğun en iyi uyumluluk değerlerinin ortalamalarının, δ_i komşuluk büyüklüğüne (parçacıkların algılama/haberleşme alanlarına) göre değişimini göstermektedir. Bu benzetimlerde parçacık komşulukları, $\mathcal{N}_i(t)$, arama uzayında parçacıklar arasındaki mesafelerin ele alındığı Denklem 3.1 kullanılarak belirlenmiştir. Benzetimler için her parçacık için aynı büyüklükte algılama/haberleşme alanı seçilmiş, her i parçacığı için $\delta_i = \delta$ olmaktadır. Yöntemin başarımını parçacıkların değişik algılama/haberleşme alanlarında, farklı komşuluk dinamiklerinde, test edilmesi için parçacıkların algılama/haberleşme alanları $\delta = \delta_i$, 0 dan arama uzayındaki en büyük mesafenin yarısına kadar değiştirilmiştir. Arama uzayındaki en büyük mesafe küre fonksiyonu için $200\sqrt{20} \approx 894$, griewank fonksiyonu için $1200\sqrt{20} \approx$

5366, rastrigin fonksiyonu için $10.24\sqrt{20} \approx 46$ ve rosenbrock fonksiyonu için $4.096\sqrt{20} \approx 19$ olarak belirlenmiştir. Belirtilen algılama/haberleşme alanları aralığında 21 veri noktası alınmıştır. Her benzetimin sonunda (10000 adım sonunda) birbirinden farklı 30 başlangıç koşulu için elde edilen komşuluğun en iyi uyumluluk değerlerinin ortalamalarının komşuluğun büyüklüğüne ($\delta_i = \delta$) göre değişimi çizilmiştir. Grafiklerdeki dikey doğrular 30 farklı başlangıç koşulu için elde edilen sonuçların standart sapmasını göstermektedir. Ana grafiğin altında yer alan iki küçük grafikler ana grafiğin farklı komşuluk alanları aralıklarında büyütülmüş biçimleridir. Ayrıca benzetim sonuçları Çizelge 3.1. üzerinde belirtimiştir. Tablodan ve grafiklerden görüldüğü üzere parçacık algılama/haberleşme alanlarının küçük olduğu durumlarda algoritma istenen başarımı gösterememektedir. Bu durumun nedeni küçük parçacık algılama/haberleşme alanlarında sürünün güçlü bir şekilde bağlı olmaması ve sürüdeki coğu parçacığın aralarında bilgi paylaşımı yapmadan kendi bilgileri ile arama yapmalarıdır. Parçacıkların algılama alanları büyüklüğünün (komşuluk büyüklüğünün) yaklaşık arama uzayındaki en büyük mesafenin dörtte biri olduğu durumlarda, haberleşme çizgesi daha bağlı bir hale gelmekte ve algoritma daha iyi bir başarım sergilemekte ve bu değerden daha büyük algılama alanlarında algoritma etkin bir başarım göstermektedir. Etkin ve etkin olmayan başarım arasındaki eşik değeri yaklaşık arama uzayındaki en büyük mesafenin dörtte biri olmaktadır. Öte yandan ele alınan komşuluk belirleme yönteminin başarımının parçacık yoğunluğuna (dolayısıyla parçacık sayısı ve arama uzayının boyutu) bağlı olması nedeni ile elde edilen bu sonuç genel bir sonuç olarak kabul edilemez. Çizelge 3.1.'de "Standart" satırında, algoritmanın tam bağlı zamanla değişmeyen komşuluk yapısında dört denektaşı fonksiyonun 30 farklı başlangıç koşulu için elde edilen komşuluğun en iyi uyumluluk değerlerinin ortalamaları ve standart sapmaları bulunmaktadır. Yöntemin dinamik komşuluklu ile yöntemin tam bağlı zaman ile değişmeyen komşuluklu biçimlerinin başarımları karşılaştırıldığı, dinamik komşuluklu biçimde parçacıkların algılama alanlarının yeterince büyük verildiği durumda yöntemin tam bağlı zaman ile değişmeyen komşuluklu biçimi ile karşılaştırılabilir sonuçlar verdiği gözlemlenmiştir.

Şekil 3.10., Şekil 3.11., Şekil 3.12. ve Şekil 3.13. komşuluğun en iyi uyumluluk değerlerinin ortalamalarının fonksiyon uzayında parçacıklar arası mesafeye göre değişimi göstermekte, ayrıca bu sonuçlar Çizelge 3.2. üzerinde belirtilmektedir. Bir başka deyişle bu benzetimler sırasında parçacık komşulukları Denklem 3.2 göz önünde bulundurularak belirlenmiştir. Öte yandan ele alınan denektaşı fonksiyonlarının fonksiyon uzayındaki değerlerinin birbirlerinden farklı olmasından dolayı karşılaştırılabilir sonuçlar elde etmek için parçacıkların algılama alanları δ_i,

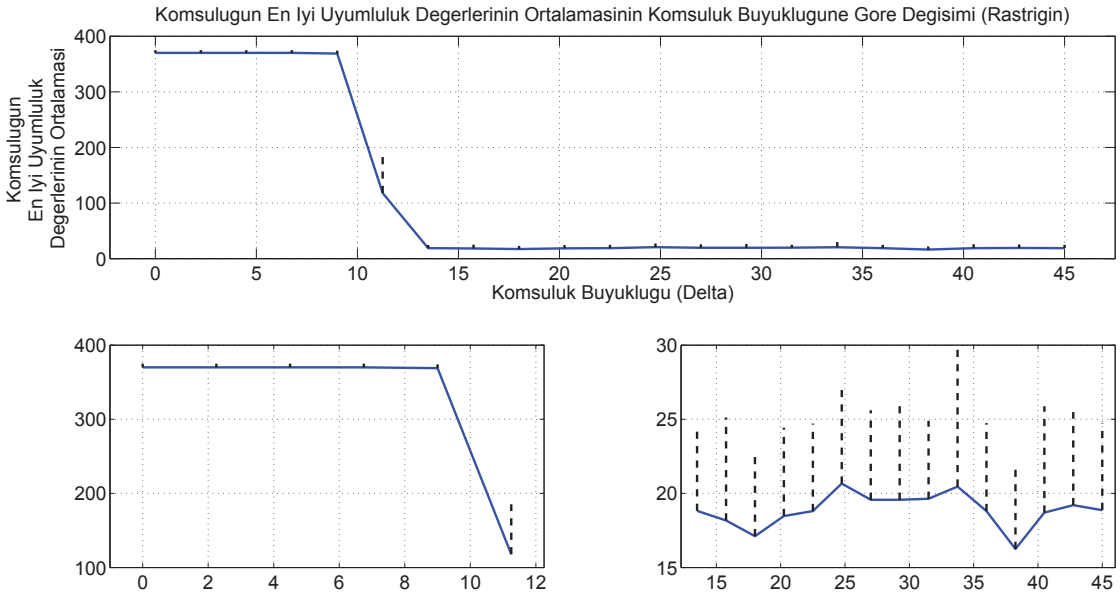


Şekil 3.6.: Komşuluğun en iyi değerlerinin ortalamalarının arama uzayında parçacıkların algılama alanları büyüklüğüne göre değişimi (Küre).

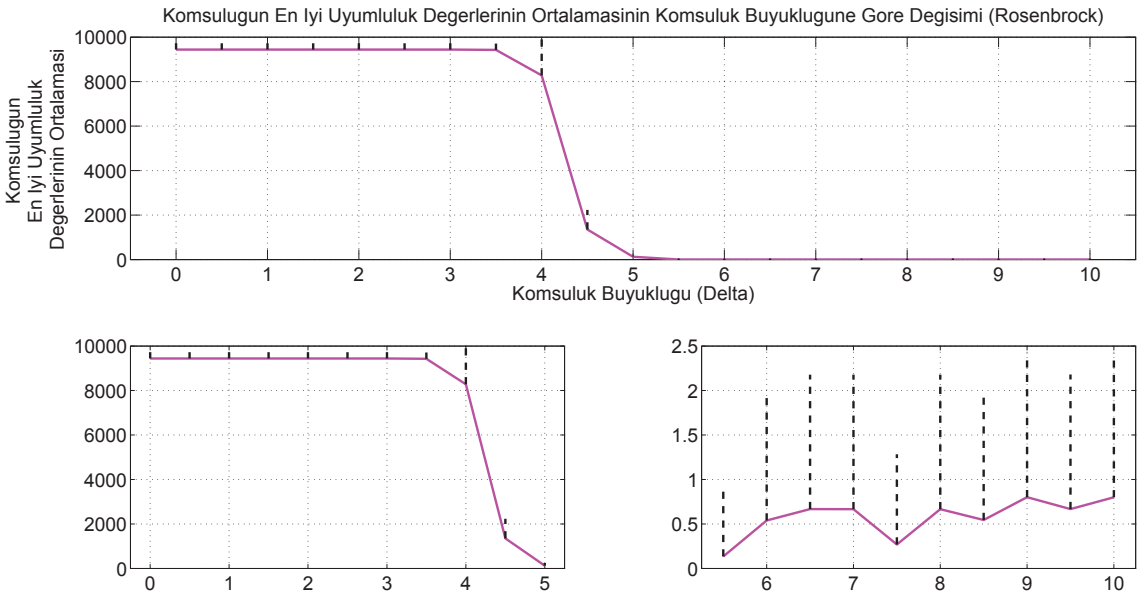


Şekil 3.7.: Komşuluğun en iyi değerlerinin ortalamalarının arama uzayında parçacıkların algılama alanları büyüklüğüne göre değişimi (Griewank).

her denektaşı fonksiyon için ayrı aralıkta düşünülmüştür. Daha önceki bölümlerde de belirtildiği gibi kullanılan komşuluk belirleme yöntemi genetik algoritmalarda kullanılan seçkinci çaprazlama yöntemi ile benzerlikler göstermektedir. Birbirine yakın başarım gösteren parçacıklar (aynı fonksiyon değerlerine sahip) arama sırasında birbirleri ile bilgi paylaşımında bulunmaktadır ve bu nedenden dolayı sürüdeki bağlanabilirlik ele alınan denektaşı fonksiyonların (bakınız Çizelge 2.2.) en büyük ve en küçük değerleri arasındaki farka ve aynı zamanda sürüdeki parçacık sayısına bağlıdır. Parçacık algılama alanları küçük seçildiğinde, sadece benzer başarıma sahip (benzer fonksiyon değerlerine sahip parçacıklar) az sayıdaki parçacık birbirleri ile haberleşmesinden dolayı algoritma tatmin edici bir başarım sergileyemez.



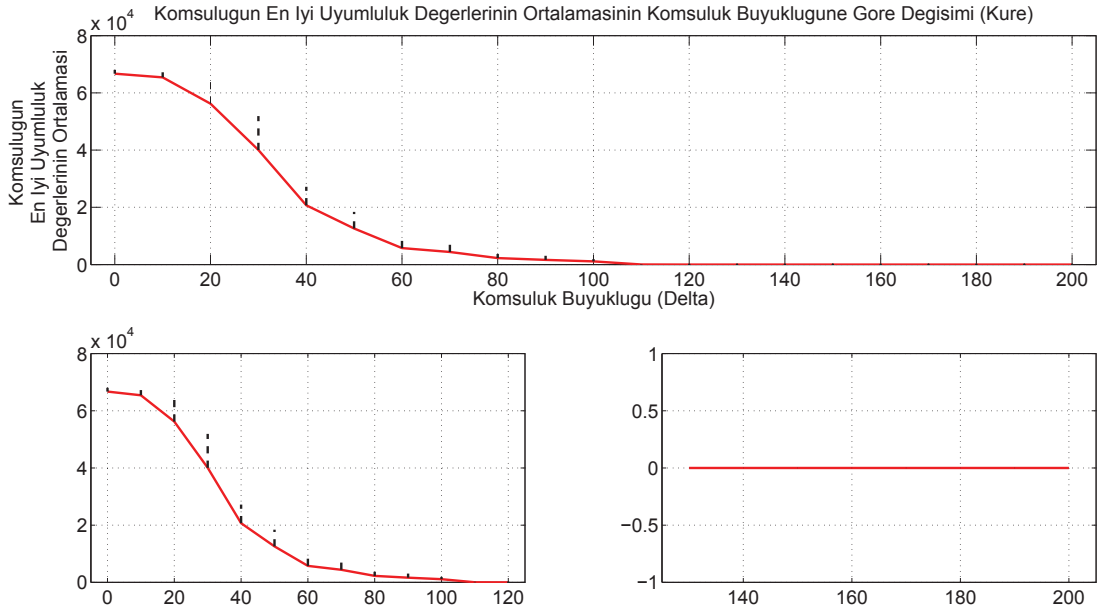
Şekil 3.8.: Komşuluğun en iyi değerlerinin ortalamalarının arama uzayında parçacıkların algılama alanları büyüklüğüne göre değişimi (Rastrigin).



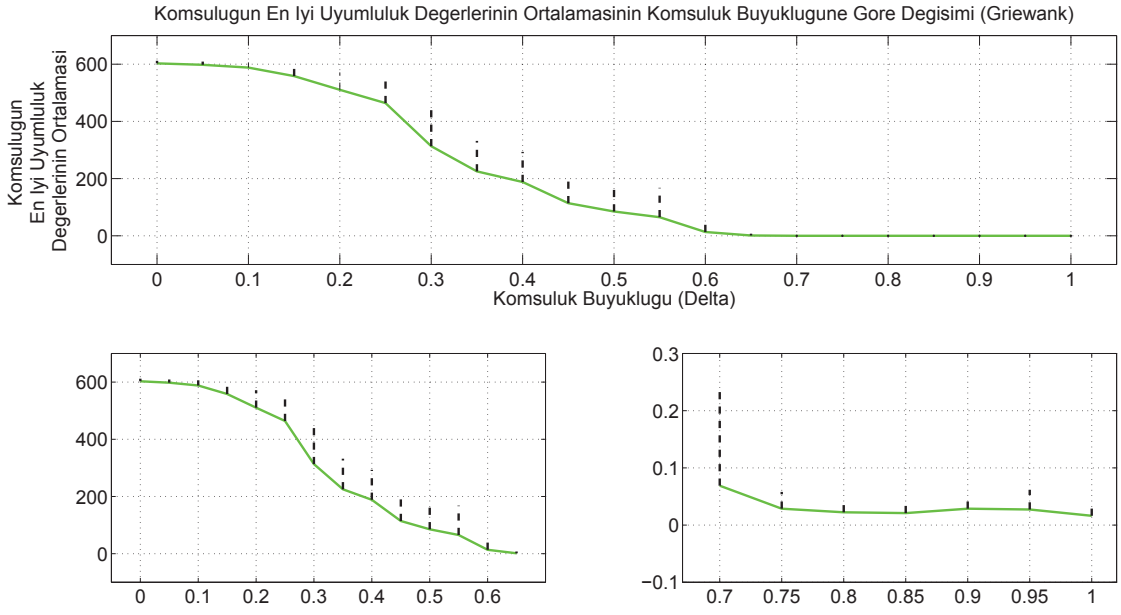
Şekil 3.9.: Komşuluğun en iyi değerlerinin ortalamalarının arama uzayında parçacıkların algılama alanları büyüklüğüne göre değişimi (Rosenbrock).

Parçacıkların algılama alanlarının büyütülmesi ile birbirlerine yakın başarım gösteren daha çok sayıda parçacığın bilgi paylaşımında bulunması sağlanır, böylece yöntem daha iyi bir başarım gösterir. Elde edilen sonuçlar yöntemin tam bağlı zaman ile değişmeyen komşuluklu biçimi ile karşılaştırılarak dinamik komşuluklu biçimin ile

karşılaştırılabilir bir başarım sergilediği gözlemlenebilir (bakınız Çizelge 3.2. son satır).

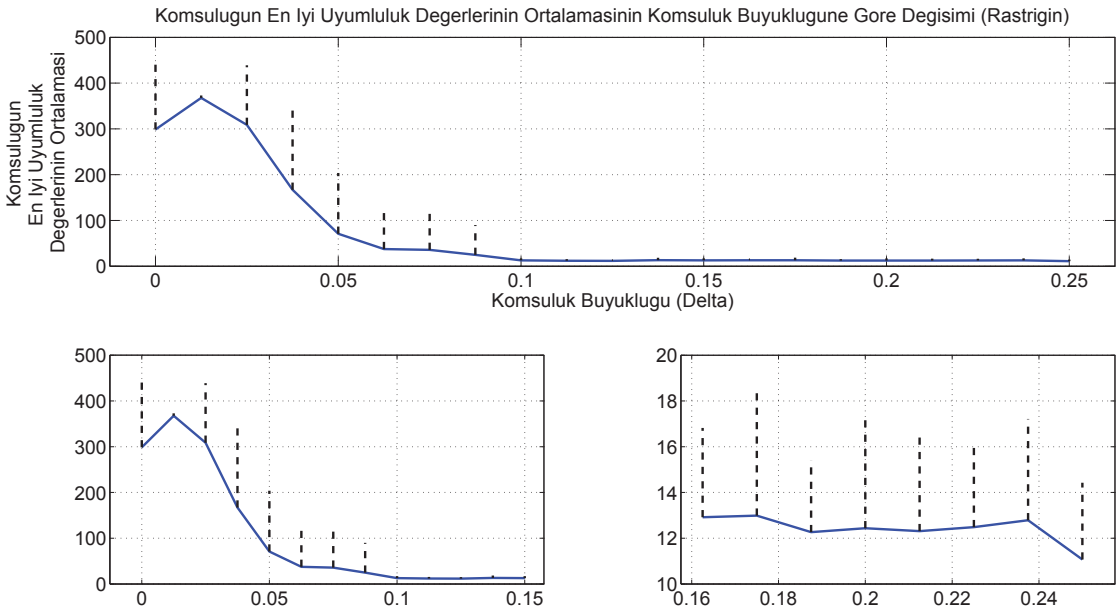


Şekil 3.10.: Komşuluğun en iyi değerlerinin ortalamalarının fonksiyon uzayında parçacıkların algılama alanları büyüklüğüne göre değişimi (Küre).

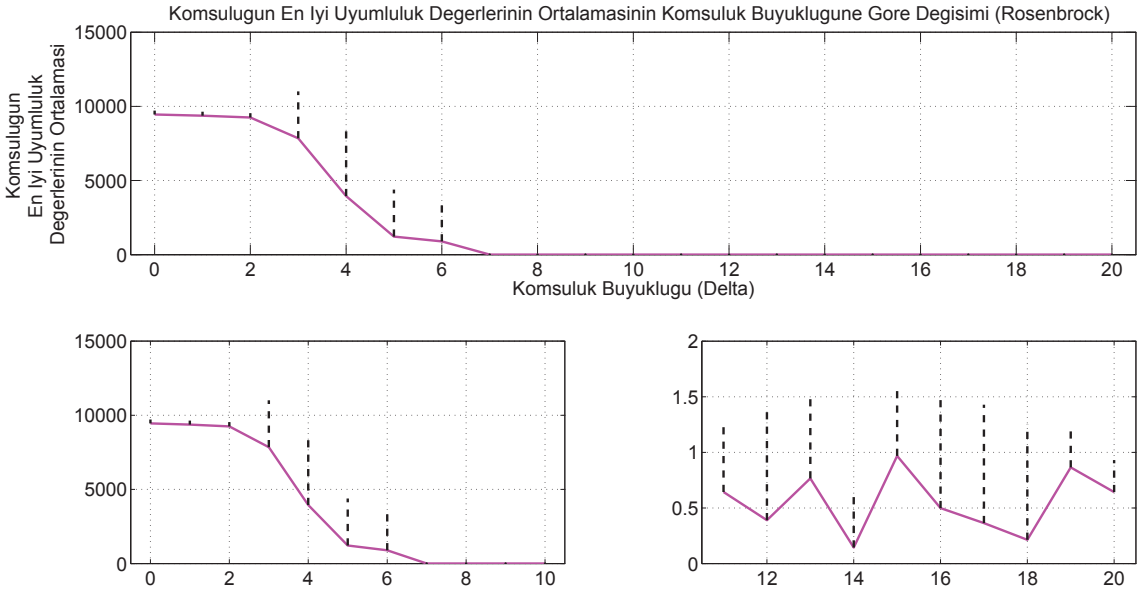


Şekil 3.11.: Komşuluğun en iyi değerlerinin ortalamalarının fonksiyon uzayında parçacıkların algılama alanları büyüklüğüne göre değişimi (Griewank).

Şekil 3.14., Şekil 3.15., Şekil 3.16., Şekil 3.17. ve Çizelge 3.3. parçacık komşuluklarının rasgele belirlendiği durumda elde edilen sonuçları göstermektedir.



Şekil 3.12.: Komşuluğun en iyi değerlerinin ortalamalarının fonksiyon uzayında parçacıkların algılama alanları büyüklüğüne göre değişimi (Rastrigin).



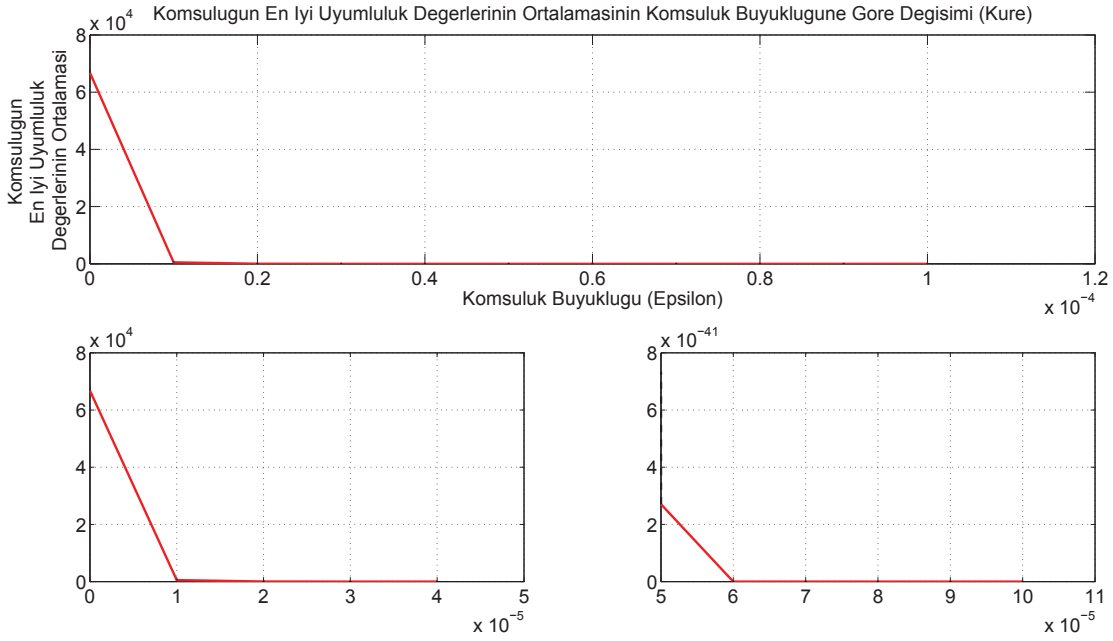
Şekil 3.13.: Komşuluğun en iyi değerlerinin ortalamalarının fonksiyon uzayında parçacıkların algılama alanları büyüklüğüne göre değişimi (Rosenbrock).

Bu benzetimlerde parçacık komşulukları Denklem 3.4 ele alınarak belirlenmiştir. Grafiklerden de görüldüğü üzere parçacıkların komşu olma olasılıklarının çok düşük olduğu durumlarda yöntemin başarımı istenen düzeyde olmamıştır. Bunun nedeni ise sürünün Varsayım 1'i sağlayamamasından ve sürüde bilgi paylaşımında

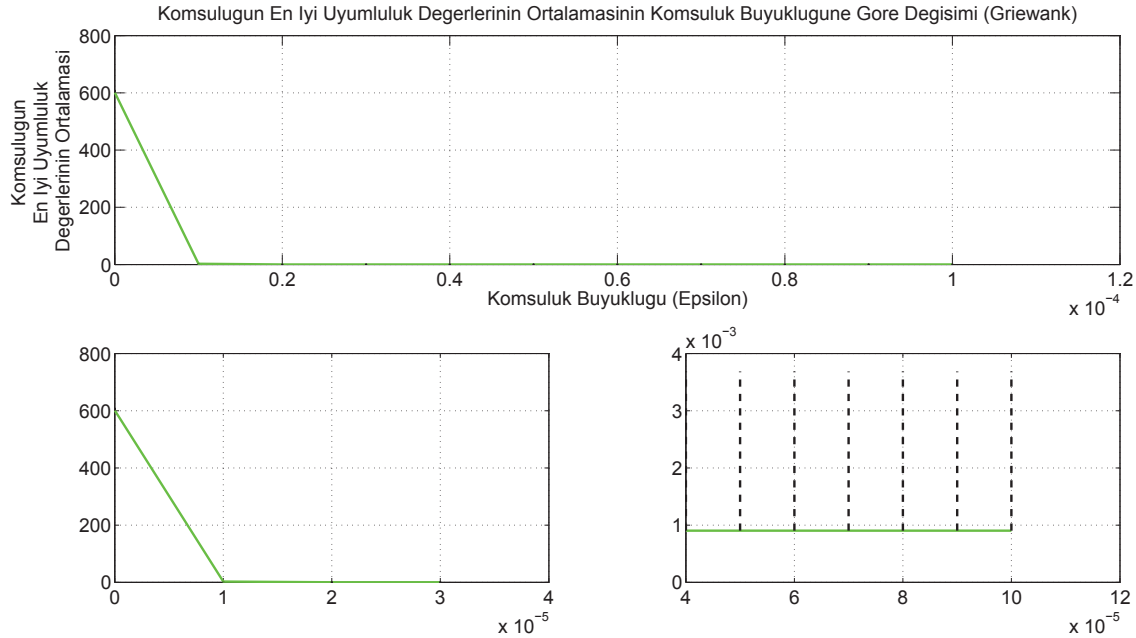
bulunan parçacık sayısının az olmasından kaynaklanmaktadır. Parçacıkların komşu olabilme olasılığının artmasıyla daha çok parçacık arama sırasında bilgi paylaşımında bulunmakta ve yöntemin başarımı artmaktadır. Şekil 3.18., Şekil 3.19., Şekil 3.20. ve Şekil 3.21. komşu olma olasılıkları daha büyük değerler alınarak elde edilen sonuçlar göstermektedir. Benzetimler sırasında parçacıkların komşu olabilme eşik değerinin, ϵ, 0.01 alınması, bir başka deyişle herhangi iki parçacığın komşu olma şansının $\%1$ olması, sürüdeki her parçacığın ortalama bir komşusu olması (sürüde 100 parçacık olmasından ötürü) anlamına gelmektedir. Ayrıca daha önceki bölümlerde belirtildiği gibi rasgele komşuluk belirleme yönteminde parçacık komşulukları her yinelemede rasgele belirlendiği için parçacıkların farklı arama yönlerinde arama yapma sanşları artmaktadır. Bu durumun aksine daha önce belirtilen iki komşuluk belirleme yönteminde parçacıklar belirli bir arama algılama alanına sahip olması nedeniyle parçacıklar ardaşık yinelemlerde daha az yönde arama gerçekleştirmektedir. Öte yandan yukarıda belirtilen iki komşuluk belirleme yönteminde komşuluk ilişkileri karşılıklı olmasına karşın, rasgele komşuluk belirleme yönteminde parçacık komşulukları genellikle karşılıklı değildir çünkü herhangi bir zaman anı ve herhangi i ve j parçacığı için büyük olasılıkla $\epsilon_{ij} \neq \epsilon_{ji}$ dir. Elde edilen sonuçlardan görüldüğü üzere, parçacıklarının komşu olma olasılığının $\epsilon = 0.01$ gibi küçük bir değer olmasına karşın yöntem etkin bir başarım sergilemiştir. Yeniden belirtilmelidir ki bu yöntem diğer iki komşuluk belirleme yönteminin aksine, sürünün bağlanırlık yeterliliği yalnızca komşu olma olasılığının eşik değerine ve arama uzayındaki parçacık sayısına bağlıdır.

Şekil 3.22., Şekil 3.23., Şekil 3.24., Şekil 3.25. ve Çizelge 3.3. komşu olma olasılığının $\epsilon = 1$ (bu durumda tam bağlı haberleşme çizgesine) değerine kadar 0.1 artışlarla elde edilen sonuçları göstermektedir. Parçacıkların komşu olabilme eşik değeri, ϵ, 0.01 alınması ile yöntemin etkin bir başarım sergilediğinin belirlenmesine karşın sürünün sonraki komşuluk büyüklüklerinde de tutarlı sonuçlar verip vermediğinin denetlenmesi için bu benzetimler yapılmıştır. Elde edilen sonuçların tam bağlı zaman ile değişmeyen parçacık komşulukları kullanılarak (bakınız Çizelge 3.3. son satır) karşılaştırılmasıyla bir kez daha yöntemin dinamik komşuluklu biçiminin yöntemin tam bağlı zaman ile değişmeyen komşuluklu biçimi ile benzer sonuçlar verdiği gözlemlenmiştir.

Daha önceki bölümlerde de belirtildiği üzere yöntemin dinamik komşuluklu biçiminin çeşitli uygulamalarında yararlı olabileceği ve benzetim sonuçları doğrultusunda dinamik komşuluklu yöntemin komşuluk parametrelerin doğru seçilmesi durumunda, yöntemin istenen başarımı sergileyeceği belirlenmiştir. Bu bölümde elde edilen

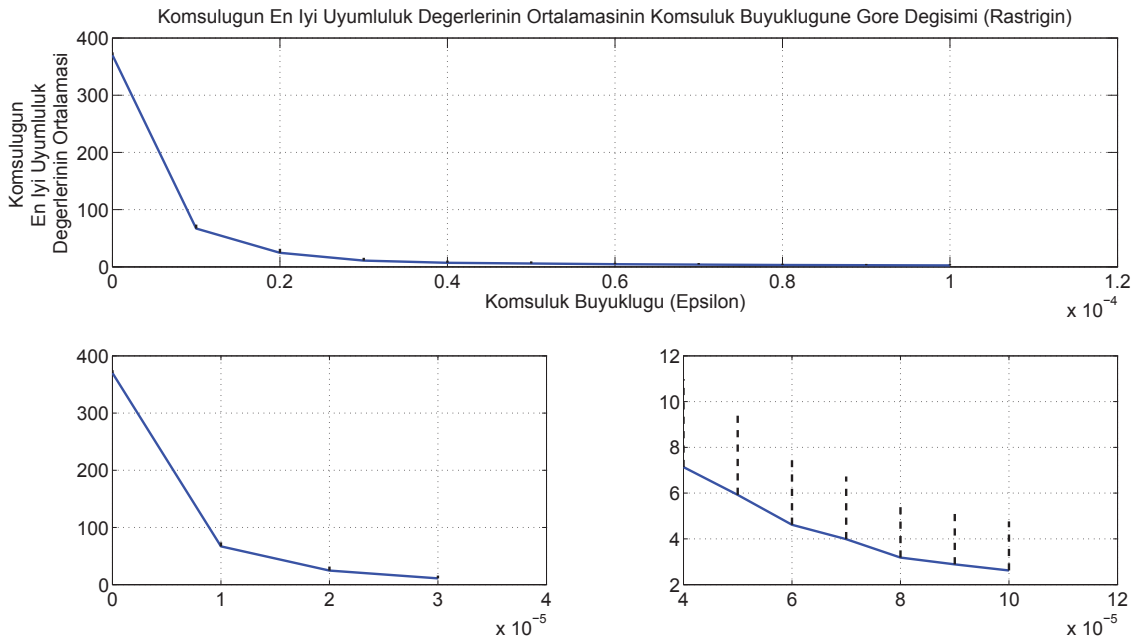


Şekil 3.14.: Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 0.0001]$) (Küre).

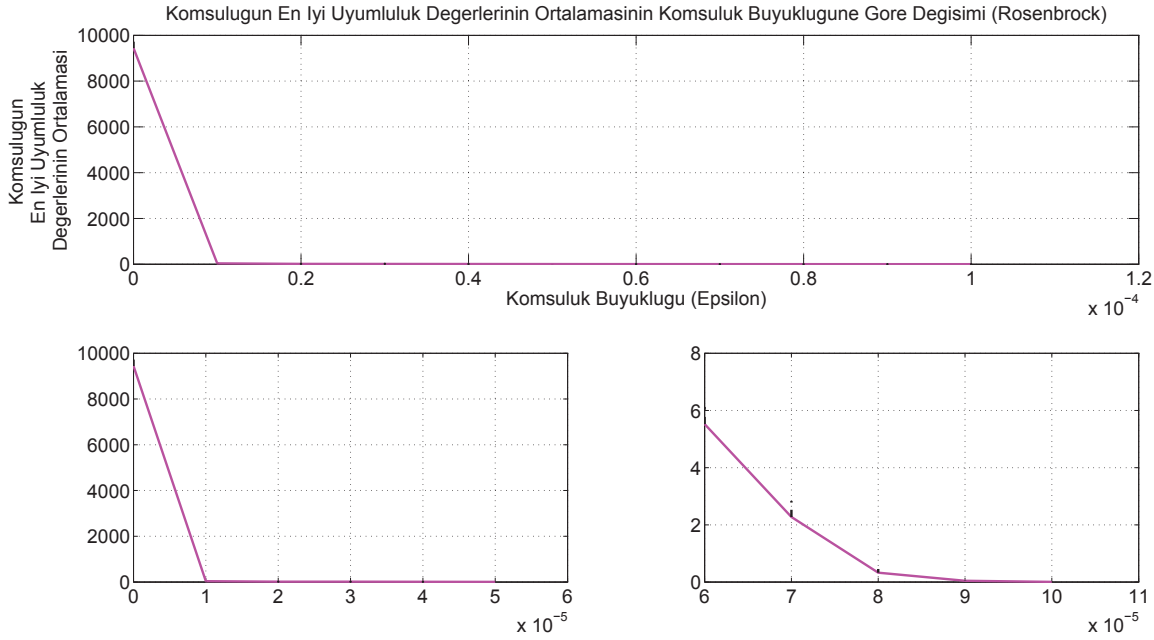


Şekil 3.15.: Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 0.0001]$) (Griewank).

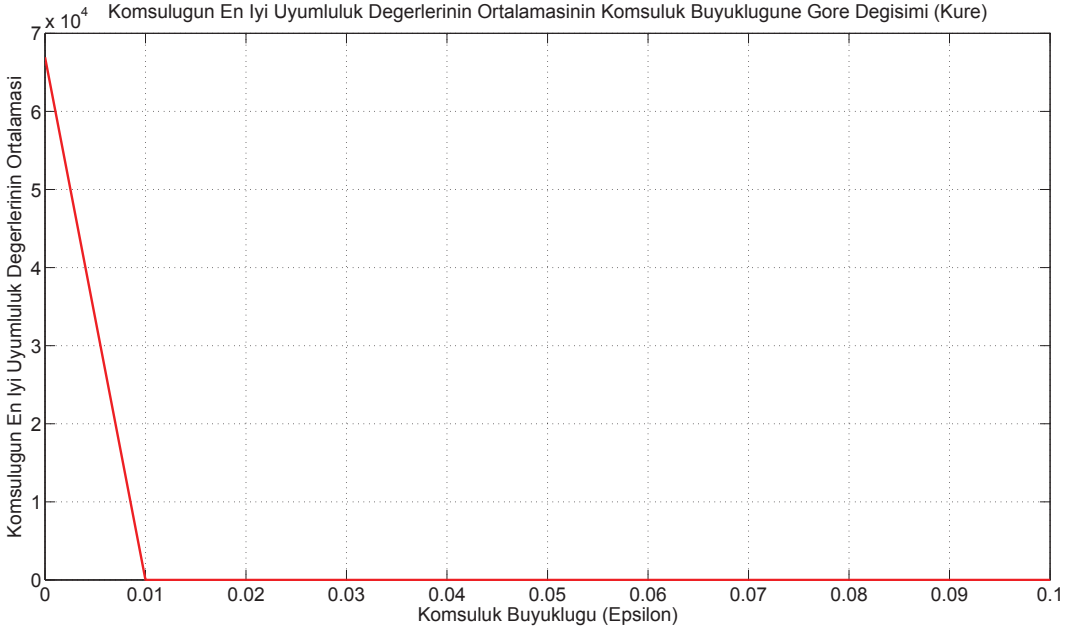
sonuçlara göre yöntemin dinamik komşuluklu biçiminin yöntemin uygun bir biçimi olduğu gözlenmiştir.



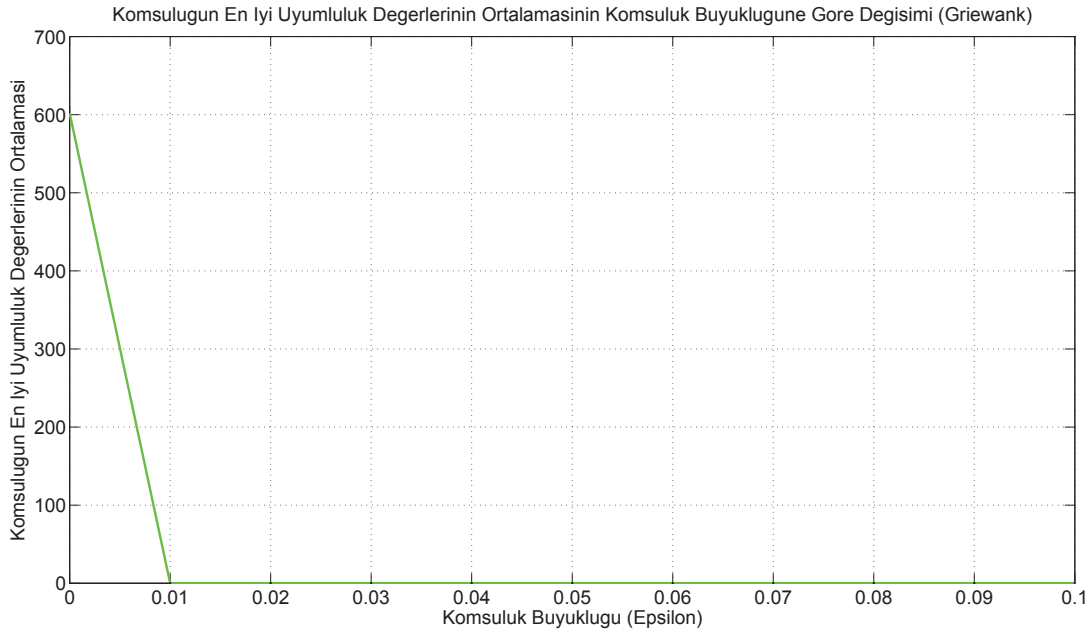
Şekil 3.16.: Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 0.0001]$) (Rastrigin).



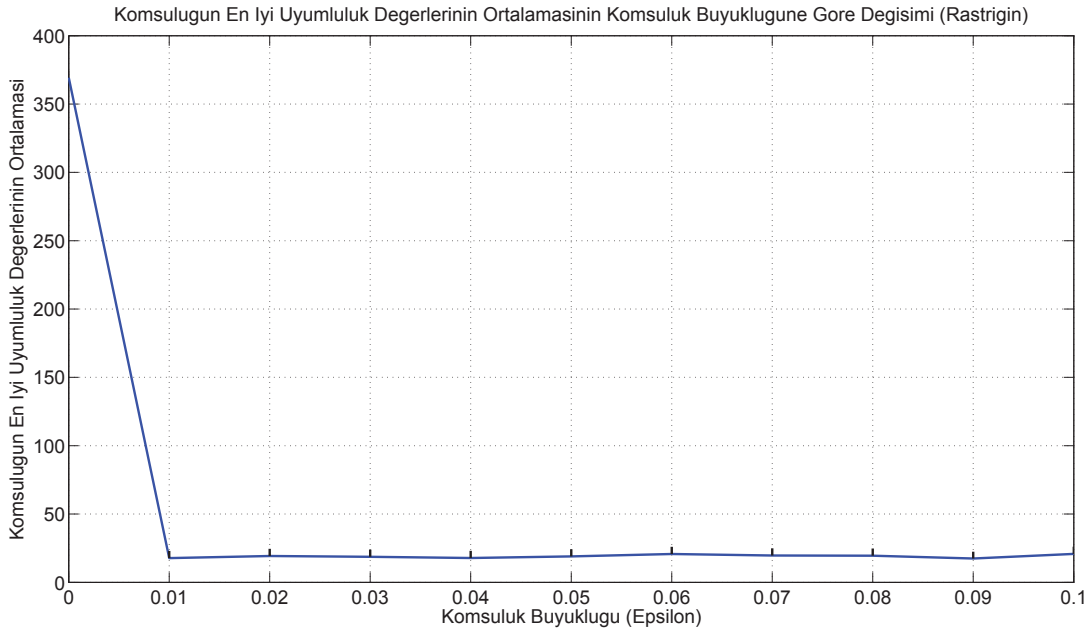
Şekil 3.17.: Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 0.0001]$) (Rosenbrock).



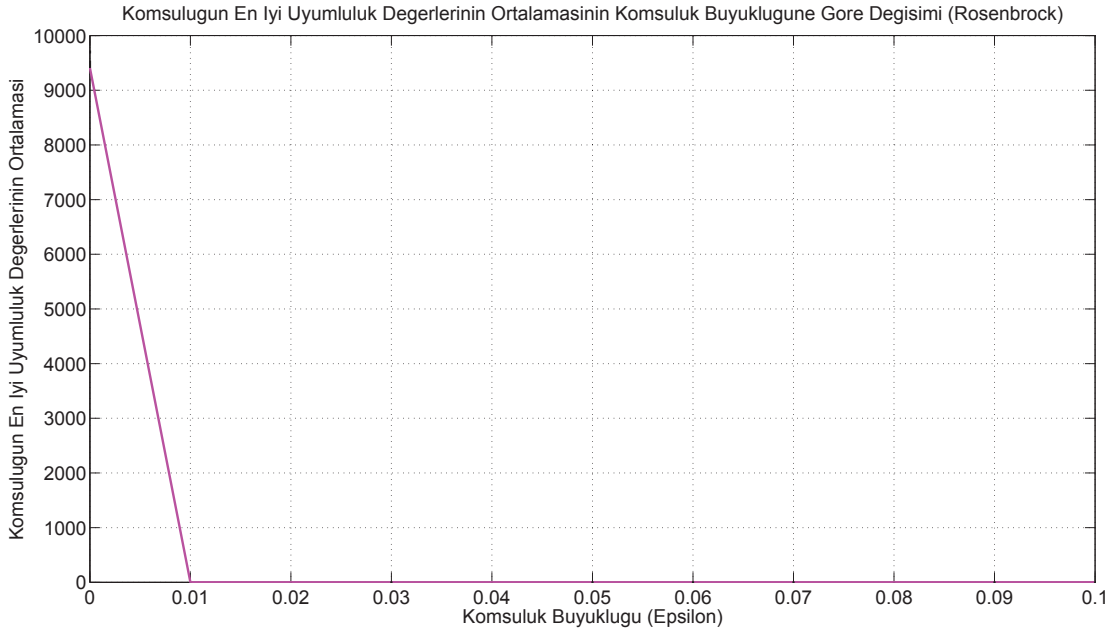
Şekil 3.18.: Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 0.1]$) (Küre).



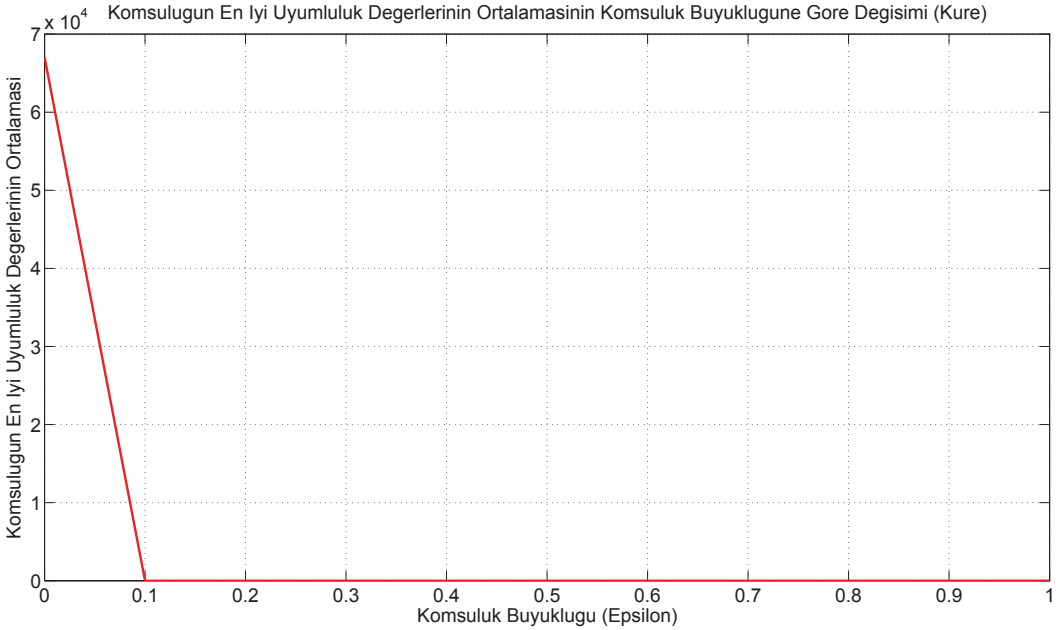
Şekil 3.19.: Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 0.1]$) (Griewank).



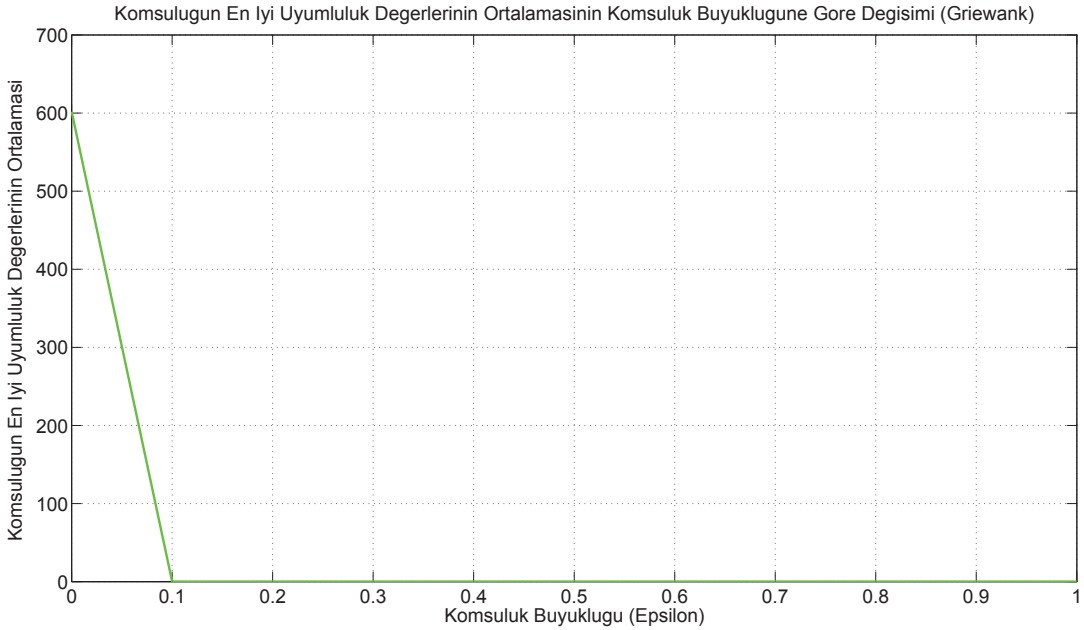
Şekil 3.20.: Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 0.1]$) (Rastrigin).



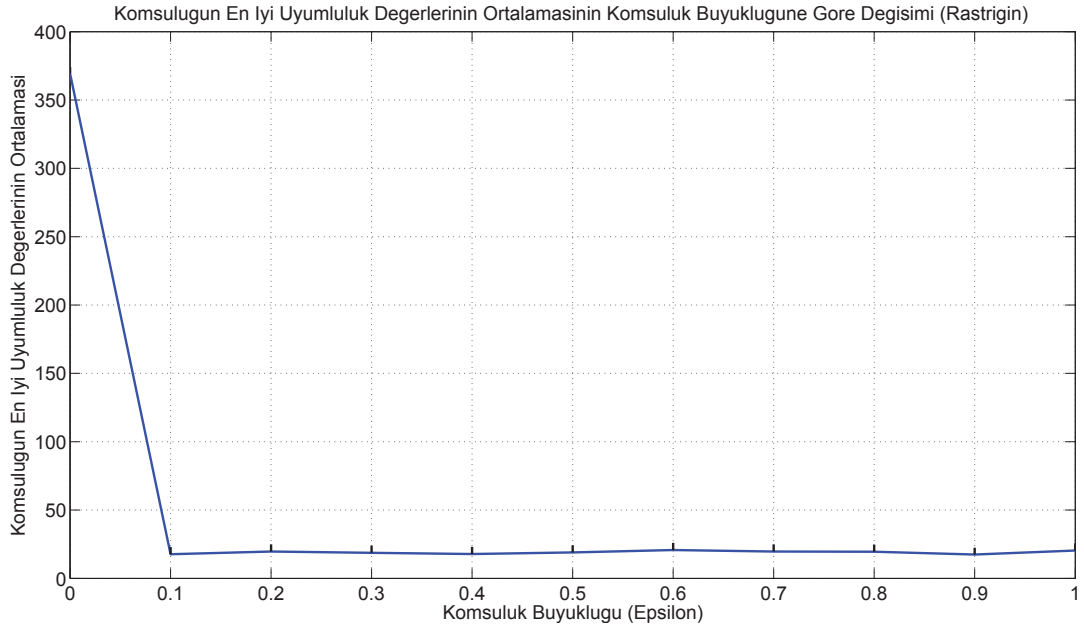
Şekil 3.21.: Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 0.1]$) (Rosenbrock).



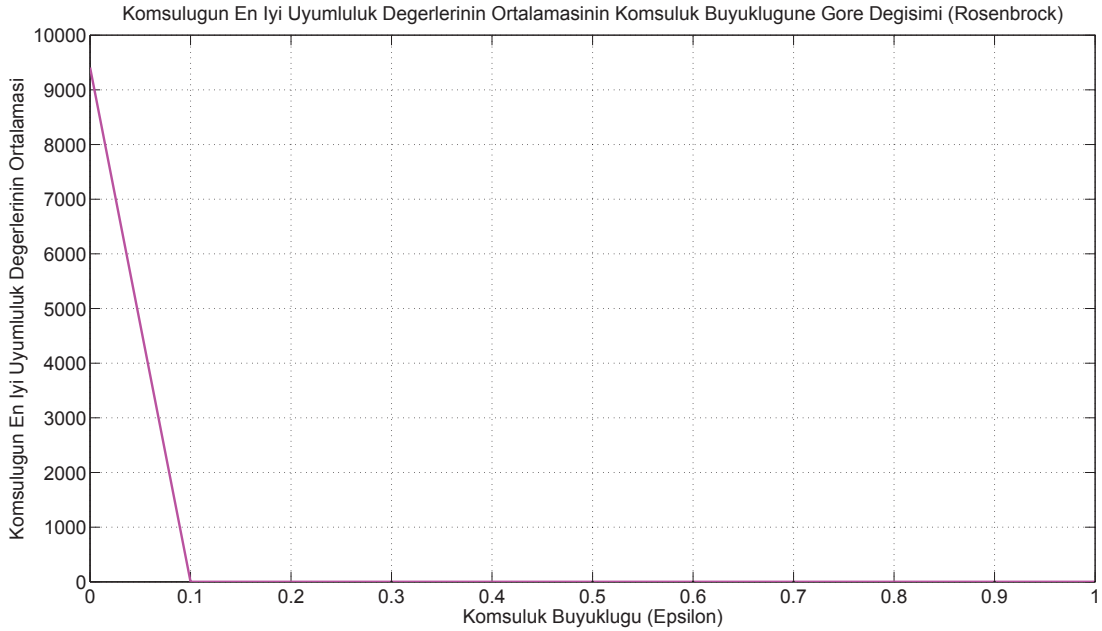
Şekil 3.22.: Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 1]$) (Küre).



Şekil 3.23.: Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 1]$) (Griewank).



Şekil 3.24.: Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 1]$) (Rastrigin).



Şekil 3.25.: Komşuluğun en iyi değerlerinin ortalamalarının parçacıkların komşu olma olasılığına göre değişimi ($\epsilon \in [0, 1]$) (Rosenbrock).

Çizelge 3.1. Arama uzayında en yakın komşular kuralı için benzetim sonuçları

No	1	1		2	2		3	3		4	4
δ_i	Ort.[a]	S S.[b]	δ_i	Ort.[a]	S S.[b]	δ_i	Ort.[a]	S S.[b]	δ_i	Ort.[a]	S S.[b]
0	66600	1200	0	600	9.28	0	370	4.94	0	9440	281
22	66600	1200	135	600	9.28	2.25	370	4.94	0.5	9440	281
44	66600	1200	270	600	9.28	4.50	370	4.94	1.0	9440	281
66	66600	1200	405	600	9.28	6.75	370	4.94	1.5	9440	281
88	66600	1200	540	600	9.28	9.00	368	5.15	2.0	9440	281
110	66600	1200	675	600	9.28	11.25	117	67.40	2.5	9440	281
132	66600	1200	810	600	9.28	13.50	18.80	5.68	3.0	9440	281
154	66600	1230	945	600	8.92	15.75	18.10	6.94	3.5	9440	281
176	66600	1360	1080	597	11.80	18.00	17.10	5.63	4.0	8280	1630
198	66600	8090	1215	437	94.80	20.25	18.40	5.96	4.5	1350	879
220	17300	5100	1350	117	30.06	22.50	18.80	5.90	5.0	122	122
242	2900	1240	1485	14	14.60	24.75	20.60	6.64	5.5	0.13	0.72
264	322	572	1620	0.67	2.48	27.00	19.50	6.03	6.0	0.53	1.37
286	39.20	214	1755	0.01	0.01	29.25	19.50	6.77	6.5	0.66	1.51
308	0.00	0.00	1890	0.01	0.02	31.50	19.60	5.26	7.0	0.66	1.51
330	0.00	0.00	2025	0.02	0.02	33.75	20.04	9.32	7.5	0.26	1.01
352	0.00	0.00	2160	0.02	0.02	36.00	18.80	5.93	8.0	0.66	1.51
374	0.00	0.00	2295	0.02	0.02	38.25	16.20	5.64	8.5	0.54	1.37
396	0.00	0.00	2430	0.02	0.01	40.50	18.70	7.16	9.0	0.80	1.62
418	0.00	0.00	2565	0.01	0.01	42.75	19.20	6.28	9.5	0.66	1.51
440	0.00	0.00	2700	0.02	0.01	45.00	18.80	5.85	10	0.70	1.62
Stan[c]	0.00	0.00		0.02	0.02		17.90	7.39		0.26	1.01

[a] Ortalama [b] Standart Sapma [c] Standart

Çizelge 3.2. Fonksiyon uzayında en yakın komşular kuralı için benzetim sonuçları

No	1	1		2	2		3	3		4	4
δ_i	Ort.[a]	S S.[b]	δ_i	Ort.[a]	S S.[b]	δ_i	Ort.[a]	S S.[b]	δ_i	Ort.[a]	S S.[b]
0	66600	1340	0	600	9.28	0	298	145	0	9440	281
10	65300	1810	0.05	597	11.10	0.0125	370	49.40	1	9370	266
20	56200	8930	0.10	588	18.20	0.0250	308	130	2	9240	293
30	40000	11800	0.15	558	37.30	0.0375	166	175	3	7840	3160
40	20600	6470	0.20	510	61	0.0500	70.80	132	4	3940	4590
50	12500	5820	0.25	463	85.60	0.0625	37.50	90.20	5	1220	3160
60	5730	2710	0.30	313	134	0.0750	35.80	89.50	6	891	2710
70	4370	3140	0.35	225	106	0.0875	24.70	65.20	7	8.28	1.13
80	2250	1450	0.40	188	105	0.1000	12.70	4.51	8	1.70	1.83
90	1630	1400	0.45	114	97.90	0.1125	11.80	3.63	9	1.18	1.17
100	1100	788	0.50	85	97.70	0.1250	11.60	3.41	10	0.88	0.54
110	29	164	0.55	65.10	102	0.1375	13.10	5.36	11	0.64	0.64
120	0.00	0.00	0.60	13.30	31.30	0.1500	12.70	4.33	12	0.38	0.99
130	0.00	0.00	0.65	1.19	6.41	0.1625	12.90	3.90	13	0.76	0.76
140	0.00	0.00	0.70	0.06	0.17	0.1750	12.90	5.64	14	0.14	0.47
150	0.00	0.00	0.75	0.02	0.02	0.1875	12.20	3.14	15	0.96	0.64
160	0.00	0.00	0.80	0.02	0.01	0.2000	12.40	4.78	16	0.49	0.98
170	0.00	0.00	0.85	0.02	0.01	0.2125	12.30	4.12	17	0.36	1.06
180	0.00	0.00	0.90	0.02	0.02	0.2250	12.40	3.74	18	0.23	1.03
190	0.00	0.00	0.95	0.02	0.03	0.2375	12.70	4.42	19	0.86	0.38
200	0.00	0.00	1.00	0.01	0.01	0.2500	11.00	3.35	20	0.64	0.28
Stan[c]	0.00	0.00		0.02	0.02		17.90	7.39		0.26	1.01

[a] Ortalama [b] Standart Sapma [c] Standart

Çizelge 3.3. Rasgele komşuluk belirleme kuralı için benzetim sonuçları

No	1	1	2	2	3	3	4	4
ϵ_i	Ort.[a]	S S.[b]	Ort.[a]	S S.[b]	Ort.[a]	S S.[b]	Ort.[a]	S S.[b]
0	66600	1200	600	9.28	370	4.94	9440	281
0.00001	513.40	295.18	2.69	0.97	66.95	6.95	36.49	8.81
0.00002	0.0001	0.0001	0.01	0.01	24.55	6.95	15.59	0.66
0.00003	4.29e-14	5.39e-14	0.002	0.004	10.99	4.89	13.44	0.55
0.00004	1.55e-15	1.82e-26	0.001	0.002	7.13	3.84	11.10	0.62
0.00005	2.70e-41	4.62e-41	0.001	0.002	5.93	3.58	8.43	0.58
0.00006	9.23e-59	1.22e-58	0.001	0.002	4.61	2.82	5.52	0.60
0.00007	4.07e-78	5.20e-78	0.001	0.002	3.98	2.74	2.27	0.56
0.00008	5.86e-99	1.06e-98	0.001	0.002	3.18	2.34	0.32	0.56
0.00009	0.00	0.00	0.001	0.002	2.88	2.34	0.04	0.01
0.0001	0.00	0.00	0.001	0.002	2.62	2.14	0.52	0.01
0.01	0.00	0.00	0.01	0.01	17.70	5.73	0.83	1.61
0.02	0.00	0.00	0.02	0.01	19.20	6.36	0.86	1.61
0.03	0.00	0.00	0.01	0.01	18.60	6.03	0.28	1.02
0.04	0.00	0.00	0.02	0.02	17.80	5.83	0.54	1.37
0.05	0.00	0.00	0.02	0.02	18.90	5.83	0.40	1.21
0.06	0.00	0.00	0.02	0.01	20.70	7.56	0.13	0.72
0.07	0.00	0.00	0.02	0.02	19.60	7.72	0.13	0.72
0.08	0.00	0.00	0.01	0.02	19.50	5.69	0.53	1.37
0.09	0.00	0.00	0.02	0.02	17.40	4.95	0.40	1.21
0.1	0.00	0.00	0.02	0.02	20.80	8.42	0.40	1.21
0.2	0.00	0.00	0.02	0.01	19.63	5.99	0.82	1.61
0.3	0.00	0.00	0.01	0.01	18.67	6.03	0.28	1.02
0.4	0.00	0.00	0.01	0.02	17.84	5.83	0.54	1.37
0.5	0.00	0.00	0.02	0.02	18.97	5.83	0.40	1.21
0.6	0.00	0.00	0.02	0.01	20.76	7.56	0.14	0.72
0.7	0.00	0.00	0.02	0.02	19.66	7.72	0.13	0.72
0.8	0.00	0.00	0.01	0.02	19.53	5.69	0.53	1.37
0.9	0.00	0.00	0.02	0.01	17.47	4.95	0.40	1.21
1.0	0.00	0.00	0.02	0.01	20.36	8.12	0.40	1.21
Stan[c]	0.00	0.00	0.02	0.02	17.90	7.39	0.26	1.01

[a] Ortalama [b] Standart Sapma [c] Standart

BÖLÜM 4

4. EŞZAMANSIZ VE DAĞINIK PARÇACIK SÜRÜ ENİYİLEME YÖNTEMİ

Parçacık sürü eniyileme yöntemi ile ilgili çalışmalarda genel olarak yöntemin eşzamanlı çalışan biçimi ele alınmıştır. Bu tür uygulamalarda sürüdeki bütün parçacıklar konumlarını/tahminlerini, $p_i(t)$, aynı anda günceller ve ardından aynı anda komşuluğun en iyi uyumluluk değerini ($evrensel_{eniyi}$) belirlemek için bilgi alışverişinde bulunurlar ve bütünsel bir komşuluk yapısında bulundukları için her zaman anında birbirleri ile komşulardır. [23] çalışmasında yöntemin sıralı[1] eşzamansız biçimi sunulmuştur. Yöntemin bu biçiminde parçacıklar önceden yinelemelerini tamamlayan parçacıkların güncellenmiş bilgilerini ve diğer parçacıkların eski bilgilerini kullanarak $evrensel_{eniyi}$ değerlerini belirlemektedir ve yinelemeleri sıralı bir biçimde gerçekleştirmektedir. Bu durumda parçacıklar güncellemelerini aynı anda gerçekleştirememekte ve sürüdeki parçacıklar $evrensel_{eniyi}$ değerlerini güncellerken bazı parçacıkların en güncel bilgilerini kullanamamaktadır. Öte yandan yöntemin bu biçimi tam olarak eşzamansız olmamaktadır çünkü parçacıklar tam olarak birbirinden bağımsız işlemler gerçekleştirememektedir. [76] çalışmasında karmaşık bir mühendislik probleminin çözümü için usta yamak ilişkisi benimsenmiş, yamak işlemciler uyumluluk fonksiyonun değerlerini hesaplarken usta işlemci parçacıkların konum güncellemelerini hesaplayarak yamak işlemcilere iletmektedir. Sistemdeki yamak işlemcilere eşit olmayacak şekilde dağılan iş yükünün başarıma olan olumsuz etkisini en aza indirgemek ve paralel verimliliğinin artması için usta işlemci hesaplanan parçacık güncellemelerini eşzamansız bir biçimde yamak işlemcilere iletmektedir. Böylece yamak işlemcilerin birbirlerini bekleme ihtiyacı ortadan kalkmıştır. [77] çalışmasında da benzer bir usta yamak ilişkisi kullanılarak yamak işlemcilerin usta işlemci tarafından atanan bir nokta için uyumluluk değerini hesaplayarak usta işlemciye iletilmektedir. Bütün yamak işlemciler eşzamanlı olarak usta işlemciye bilgi iletmedikleri için usta işlemci elde ettiği bilgileri kullanarak belirli bir noktanın konum vektörünü parçacığın o zaman anına kadar elde ettiği en iyi uyumluluk değerini kullanarak hesaplamaktadır. Öte yandan eşzamanlı bir sistemde bütün parçacıkların (bütünsel bir komşuluk kabul edilirse) bilgileri kullanılarak komşuluğun en iyi uyumluluk değeri belirlenmektedir. Yukarıda belirtilen iki çalışmada da yöntemin paralel ve eşzamansız biçimleri işlemciler arasında usta yamak ilişkisi benimsendiği

[1] ing: sequential

için merkezi olamayan karar verebilme ve dağıtık çalışma özellikleri görülmemektedir.

Bu tez çalışmasının ikinci amacı parçacık sürü eniyileme yönteminin eşzamansız, dağıtık, paralel ve dinamik komşuluklu biçiminin matematiksel modelinin geliştirilmesi ve çeşitli benzetimler ile sınanmasıdır. Yöntemin geliştirilen biçimi yöntemin ilhamını aldığı doğadaki sürülerin dağıtık/merkezi olmayan çalışması ve dinamik ve karşılıksız komşuluklara sahip olması gibi benzer felsefelere sahip olmaktadır. Ayrıca yöntemin belirtilen biçimi parçacık sürü eniyileme algoritmasının daha farklı uygulama alanlarında kullanılmasına olanak sağlamaktadır.

4.1. Eşzamanlı Parçacık Sürü Eniyileme Algoritması

Parçacık sürü eniyileme algoritmasının Denklem 2.3 kullanılarak gerçekleştirilen uygulamaların birçoğunda parçacık komşulukları önceden belirlenmekte ve algoritmanın yinelemeleri boyunca sabit kalmaktadır. Parçacık komşuluklarının belirlenmesi için çeşitli yöntemler kullanılabilir. Olası bir yöntem komşuluk yapısının çember şeklinde bir kafes yapısı olarak tanımlanması ve her parçacığın sanal sağındaki ve solundaki parçacıklar ile komşu olmasının sağlanmasıdır. Bir başka yöntemde komşuluk yapısını bütünsel olarak tanımlayarak sürüdeki her parçacığın her diğer parçacık ile komşu olmasını sağlamaktır. 3. bölümde parçacık komşuluklarının dinamik olarak belirlenmesi için rasgele komşuluk yöntemi ve parçacıklar arasındaki mesafelere dayalı arama uzayında ve fonksiyon uzayında en yakın komşular yöntemleri ele alınmış ve çeşitli benzetimler ile başarımları sınanmıştır. Ayrıca sürünün alt sürülere bölünmeden arama yapması ve tek bir konuma yakınsaması için dinamik olarak değişen parçacık komşulukları için belirli bağlanırlık koşulları belirlenmiştir. Sürünün bu koşulları sağlamaması sürünün alt sürülere bölünerek arama yapmasına ve arama uzayında farklı noktalara yakınsamasına neden olabilir.

Yöntem ile ilgili yapılan başka bir genel varsayım da parçacıkların komşularının en güncel bilgilerini mevcut zaman anında bilmeleri (parçacıklar komşularının o zaman anında kadar elde ettiği en iyi konumu bilmektedirler), bu bilgileri kullanarak komşuluğunun en iyi konumunun $g^i(t)$ hesaplamaları ve Denklem 2.3'e göre konum ve hız vektörlerinin güncellenmesi için kullanmalarıdır. Yöntemin bu biçimi tek bir işlemcide çalışma durumu için gerçekçidir. Ancak yöntemin paralel uygulamalarında algoritmanın belirtilen biçiminin uygulanması için parçacıkların işlemlerini eşzamanlı olarak gerçekleştirme ihtiyacından dolayı çalışma zamanı artmaktadır. Bu durum özellikle işlem yükünün fazla olduğu uygulamalarda istenmeyen bir durum olabilir.

Daha öncede belirtildiği üzere literatürde algoritmanın eşzamanlı ve eşzamansız çalışan biçimleri ele alınmıştır. Algoritmanın eşzamanlı çalışan biçiminde bütün parçacıklar komşuluğun en iyi uyumluluk değerlerini ve komşuluğun en iyi konum vektörünü belirledikten sonra eşzamanlı olarak hız ($v^i(t)$) ve konum ($p^i(t)$) vektörlerini belirlemektedir. Parçacık komşuluklarının tam bağlı olduğu durumda bütün parçacıklar için $g^i(t) = g(t)$ olmaktadır, oysa farklı komşuluk yapıları için her parçacık için bu değer farklı olabilir. Yöntemin eşzamansız ancak sıralı uygulamalarında parçacıkların $g^i(t)$ vektörünü kendinden önce yinelemelerini bitiren parçacıkların güncel bilgilerini ve/veya kendinden sonra yinelemelerini gerçekleştirecek parçacıkların eski bilgilerini kullanarak güncellemektedir. Dikkat edilirse bu durumda bütün parçacıklar aynı anda komşuluğun en iyi konumunu güncellememektedir.

Parçacık sürü eniyileme algoritmasının paralel biçiminin eniyilenecek fonksiyonun hesaplanmasını zaman alıcı olduğu ve fazladan işlem yüküne neden olduğu uygulamalarda kullanılmasının birçok avantajı görülmektedir. [75] çalışmasında algoritmanın paralel uygulanmış olmasına rağmen parçacıklar hız ($v^i(t)$) ve konum ($p^i(t)$) vektörlerini aynı zamanda güncelledikleri için algoritmanın eşzamanlı özelliği korunmuştur ve bu durumda algoritmanın eşzamansız uygulanması zor gözükmektedir. Paralel olan algoritmanın eşzamanlı çalışması fazladan işlem yükü ve zaman gecikmesi getirmektedir. [76, 77] çalışmalarında algoritmanın merkezi paralel ve eşzamansız biçimi ele alınmıştır. Parçacıkların komşuluğun en iyi değeri ile ilgili bilgileri toplanmakta ve güncellemeler tek bir işlemcide (usta işlemcide) gerçekleştirilmekte ve paralel olarak eniyilenecek fonksiyonun bir başka deyişle parçacıkların bulunduğu noktalardaki değerlerinin hesaplanması diğer işlemciler (yamak işlemciler) tarafından yapılmaktadır. Usta işlemci yamak işlemcilere $p^i(t)$ konum vektörünü ve diğer gerekli bilgileri iletmekte ve yamak işlemcilere eniyilenecek fonksiyonun hesaplamasını söylemektedir. İşlemlerini bitiren her yamak işlemci (dikkat edilmelidir ki her yamak işlemci işlemlerini örneğin sistemdeki çok türelilik[2] yüzünden farklı anlarda bitirebilir) bilgilerini usta işlemciye iletmektedirler. Usta işlemci komşuluğun en iyi konumunu ($g(t)$) ve parçacığın eniyilenmekte olan fonksiyonun en küçük/en büyük noktası hakkında o andaki tahminini ($p^i(t)$) günceller ve diğer yamak işlemcileri beklemeden o anda hangi yamak işlemci boş ise bir parçacığın uzaydaki konumuna karşılık gelen uyumluluk değeri (yeni bir j parçacığı olabilir) hesaplamasını ister. Böylece algoritmanın çalışma hızı artırılmış ve işlem yapılmayan zamanlar engellenmiştir. Öte yandan herhangi bir anda sadece tek bir yamak işlemci usta işlemci ile

[2]ing: heterogenity

haberleşebilmektedir. Diğer bir dezavantaj da bu yapıda usta işlemcinin bozulması veya yamak işlemciler ile herhangi bir nedenle haberleşememesi durumunda eniyileme işleminin durmasıdır.

4.2. Eşzamansız ve Dağıtık Parçacık Sürü Eniyileme Algoritmasının Matematiksel Modeli

Önceki bölümlerde belirtildiği gibi bu çalışmanın amaçlarından biri algoritmanın eşzamanlılık ve merkezi karar verme mekanizmasını ihtiyaç olamadan işlemlerin farklı işlemcilerde gerçekleştirildiği merkezi olmayan ve eşzamansız biçiminin geliştirilmesidir. Bu amaçla ilk olarak her işlemcinin yinelemelerini Denklem 2.3 kullanarak gerçekleştirdiği ve $g^i(t)$ vektörünün eşzamansız olarak her işlemci tarafından hesaplandığı varsayılmıştır. Buna ek olarak parçacık komşuluklarının zaman ile dinamik olarak değiştiği kabul edilmiştir. Bir başka deyişle bir önceki bölümde detaylı anlatıldığı gibi her zaman adımında, her parçacık (işlemci veya erkin) sadece sürünün bir alt kümesi (komşuları) ile haberleşebilmekte ve bu alt küme zaman ile değişmektedir. Gerçek uygulamalar göz önünde bulundurulduğunda dinamik olarak değişen bu alt küme ele alınan uygulamaların fiziksel düzeni, eniyileme/hesaplama olanaklarının yapısı, işlemler sırasında haberleşme/internet trafiği veya arama/fonksiyon uzayında parçacıklar arası mesafeler göz önünde bulundurularak belirlenebilir. Ayrıca bu çalışmada parçacıklar arası haberleşmede zaman gecikmelerine de izin verilmektedir. Bu bağlamda bir parçacık başka parçacıktan bilgi aldığı zaman bu bilgi gecikme ile gelmiş eski bilgi olabilir.

i parçacığının t zaman anındaki komşularını $N_i(t)$ kümesi temsil etsin. Burada biz her parçacığın t zaman anında komşuluğun en iyi konumunu $g^i(t)$ vektörünü

$$g^i(t) = \operatorname{argmin} \left\{ f_{N_i}(t), f(p^i(t)), f(g^i(t-1)) \right\} \tag{4.1}$$

denklemini kullanılarak hesapladığını varsayıyoruz. Burada

$$f_{N_i}(t) = \begin{cases} \min_{j \in N_i(t)} \{ f(p^j(\tau_j^i(t))) \}, & \forall t \in T_1^i, \\ f_{N_i}(t-1), & \forall t \notin T_1^i, \end{cases} \tag{4.2}$$

olmaktadır. Denklem 4.2'de $p^j(\cdot)$, $j \in N_i(t)$ komşu parçacığının en iyi tahminidir. Denklem 4.2 i parçacığının komşuluğun en iyi konumu $g^i(t)$'yi komşuluğun en iyi konumunun bir önceki değeri $g^i(t-1)$'yi o zaman anına kadar elde edilen kendi en iyi konumu $p^i(t)$'yi ve o andaki komşularından elde ettiği en iyi konumlarını

$p^j(\tau_j^i(t))$ karşılaştırarak ve fonksiyon uzayında en küçük/en büyük uyumluluk değerine sahip olanını seçerek belirlemektedir. Denklem 4.2'de $T_1^i \subseteq \{0, 1, 2, \ldots\}$ kümesi i parçacığının bir veya birden fazla komşusundan bilgi alabildiği ve bu bilgileri $g^i(t)$ vektörünü hesaplarken kullanabildiği zaman anlarını belirtmektedir. Bu kümenin dışındaki zaman anlarında, $t \notin T_1^i$, i parçacığı/işlemcisi o andaki komşularında bilgi alamamaktadır. Ancak bu durum i parçacığının/işlemcisinin bu zaman anlarında $g^i(t)$ vektörünü güncellemediği anlamına gelmemektedir. Denklem 4.1 ve Denklem 4.2'den de görülebileceği üzere her t zaman anında parçacıkların $g^i(t)$ vektörü ile ilgili bilgileri olmasından dolayı, parçacıklar güncellemelerine devam edebilmektedirler. Parçacık/işlemci komşularından bilgi alamadığı durumda $g^i(t)$ vektörü parçacığın o zaman anına kadar elde ettiği kendi en iyi konumu $p^i(t)$'yi ve komşuluğun en iyi konumunun eski değeri $g^i(t-1)$'yi kullanarak belirlenmektedir.

Denklem 4.2'de bulunan $\tau_j^i(t), j \in N_i(t), i = 1, \ldots, N$ değişkenleri i parçacığının komşusu olan j parçacığından, $j \in N_i(t)$, bilgi alabildiği zaman dizilerini göstermektedir. Bu zaman dizileri $t \in T_1^i$ zamanında $0 \leq \tau_j^i(t) \leq t$ koşulunu sağlamaktadır. Burada $\tau_j^i(t) = 0$ olması i parçacığının komşusu olan j parçacığından herhangi yeni bir bilgi alamadığını göstermekte (i parçacığın sadece j parçacığının ilk değerleri olan $p^j(0)$ ve $f(p^j(0))$ değerlerini bilmektedir), $\tau_j^i(t) = t$ olması ise i parçacığının komşusu olan j parçacığının o andaki bilgisine sahip olduğu, bir başka deyişle i parçacığı $p^j(t)$ ve $f(p^j(t))$ bilgilerine sahip olduğunu göstermektedir. $(t - \tau_j^i(t)) \geq 0$ farkı i parçacığının/işlemcisinin komşusu olan j parçacığından/işlemcisiden bilgi aldığı zaman anlarında ortaya çıkan haberleşme gecikmesi olarak düşünülebilir. Bir başka deyişle $p^j(\tau_j^i(t))$ i parçacığının o anda komşusu olan j parçacığından elde ettiği en iyi konumu bilgisini (bu bilgi güncel olmayabilir) temsil etmektedir. Bahsedilen bu haberleşme gecikmesi bilgisayar ağındaki yoğun veri trafiğinden veya diğer nedenlerden dolayı olabilir. Ayrıca bu matematiksel model işlemcilerin belirli bir süre güncellemelerini gerçekleştirmelerini ve değişik zaman anlarında komşuları ile bilgi alışverişinde bulunmalarını (ağdaki haberleşme yükünün azaltılması için bir yöntem olabilir) temsil etmektedir. Dikkat edilirse $N_i(t)$ komşuluk kümesi i parçacığının $(t-1)$ ve t zaman anlarında bilgi paylaşımında bulunduğu bütün parçacıklar belirtmektedir.

Yukarıda tanımlanan matematiksel modelde i parçacığı sadece komşuları olan parçacıklar ile bilgi alışverişinde bulunmaktadır (bir başka deyişle parçacık sadece $j \in N_i(t)$ parçacıkları ile haberleşebilmektedir). Bu modele ek olarak bu bölümde bir önceki bölümde olduğu gibi parçacıkların komşuları ile tek yönlü (karşılıksız)

haberleşme durumları da ve dolayısıyla komşuluk ilişkilerini karşılıklı olmadığı durumlar da ele alınmıştır. Bir başka deyişle t anında $j \in N_i(t)$ olması $i \in N_j(t)$ olacağı anlamına gelmemektedir. Buna ek olarak bu çalışmada eşzamansızlık da düşünüldüğü için $j \in N_i(t)$ ve $i \in N_j(t)$ aynı anda sağlanmış olsa dahi $\tau_j^i(t) = \tau_i^j(t)$ olamayabilir. Bunun anlamı eğer i ve j gibi iki parçacık karşılıklı olarak birbirlerinin komşuları olsa bile birbirlerinden aldıkları en iyi konum bilgileri güncel veya eşit miktarda gecikmeli olmak zorunda değildir. Yukarıda bahsedilen durum parçacıkların aynı anda haberleşemeyebileceklerini belirtmektedir. Dahası bu çalışmada parçacıkların konumlarını ve diğer değerlerini güncelleyen işlemcilerin sadece bu görevler için ayrılmış işlemciler olmadığı düşünülerek paralel olarak başka işlemleri de yapabilecekleri varsayılmıştır. Örneğin PSO algoritması tabanlı çok robotlu arama görevinde sistemdeki bir robot sürüdeki bir parçacık olarak düşünülebilir ve algoritma sistemdeki her bir robot tarafından gerçekleştirilerek üst seviyede robotların hareketlerin belirlenmesi (sonraki zaman anlarında hangi ara noktalara gitmelerinin belirlenmesi) ve aynı anda alt seviyede robotların gezinimlerinin denetlenmesi ve çeşitli algılayıcıların değerlerinin okunması sağlanıyor olabilir. Böyle bir uygulamada PSO algoritması robotların istenen ara noktalara ulaştığı anda gerçekleştirilmekte ve diğer anlarda alt seviye denetim algoritması (engellerden sakınma, gezinim belirlenmesi ve çeşitli algılayıcıların değerlerinin okunması) gerçekleştirilmektedir. Başka bir örnek olarak genel kullanıma açık bir bilgisayar laboratuarındaki bilgisayarlarda, diğer kullanıcılar bu bilgisayarlarda başka işlemler gerçekleştirirken, algoritmanın aynı bilgisayar üzerinde paralel ve alt seviye işlem[3] olarak gerçekleştirilmesidir. Bu tür bir sistemin temsil edilmesi için i işlemcisinin $t \in T_2^i \subseteq \{0, 1, 2, \ldots\}$ zaman anlarında Denklem 4.1 kullanarak güncellemeler yaptığı ve diğer zaman anlarında, $t \notin T_2^i$, PSO işlemlerinin "dondurulduğu" veya basitce

$$\begin{aligned} v^i(t+1) &= v^i(t), \\ x^i(t+1) &= x^i(t), \end{aligned} \tag{4.3}$$

denkleminin geçerli olduğu varsayılmıştır. Bir başka deyişle işlemciler tahminlerini

$$\begin{pmatrix} v^i(t+1) \\ x^i(t+1) \end{pmatrix} = \begin{cases} \text{Denklem 2.3'ü kullan}, & \forall t \in T_2^i, \\ \text{Denklem 4.3'ü kullan}, & \forall t \notin T_2^i. \end{cases} \tag{4.4}$$

[3]ing: batch processing

denklemine göre güncellemektedirler. Denklem 4.4 işlemcilerin bazı zaman anlarında güncellemeler gerçekleştirdiği ve diğer zaman anlarında diğer işlemler ile meşgul oldukları durumları temsil etmektedir.

Çalışmada işlemcilerin komşularından tamamen bağımsız bilgi alabildikleri gibi güncellemelerini de tamamen bağımsız anlarda gerçekleştirebildikleri kabul edilmiştir. Bir başka deyişle herhangi bir i parçacığı için T_1^i ve T_2^i zaman anları kümelerinin birbirlerinden bağımsız olduğu kabul edilmiştir. Buna ek olarak bu kümelerin, T_1^i ve T_2^i, $j \neq i$ için T_1^j ve T_2^j zaman anları kümelerinden de bağımsız oldukları kabul edilmektedir. Öte yandan $i, j = 1, \ldots, N$ ve $\ell 1, \ell 2 = 1, 2$ için $T_{\ell 1}^i \cap T_{\ell 2}^j \neq \emptyset$ olması gerçekleşebilecek bir durumdur (bir başka deyişle bazı zaman anlarında iki veya daha fazla işlemci aynı anda tahminlerini güncelleyebilir). Buradaki $T_\ell^i, i = 1, \ldots, N, \ell = 1, 2$ kümelerinin bileşenleri ayrık olay sistemlerinde olayların zamanı gibi güncellemelerin gerçekleştiği fiziksel zamanlara karşılık gelen indeks dizini olarak görülebilir. Bir başka deyişle bu zaman dizinleri fiziksel zaman değerleri ile eşleşmekte ve ardışık zaman dizinleri arasındaki fiziksel zaman farklarının değerleri eşit/düzgün olmak zorunda değildir.

4.3. Eşzamansız ve Dağıtık Parçacık Sürü Eniyileme Algoritması

Çizelge 4.1. bu bölümde geliştirilen algoritmanın eşzamansız ve dağıtık biçiminin sahte kodunu göstermektedir. Geliştirilen algoritmanın literatürde ele alınan eşzamanlı, sıralı eşzamansız ve merkezi paralel eşzamanlı/eşzamansız parçacık sürü eniyileme algoritmalarından önemli farkları bulunmaktadır. İlk olarak geliştirilen algoritma tamamen dağıtık/merkezi olmayan bir biçimde çalışmakta ve merkezi bir işlemcinin bütün verileri toplaması ve yinelemelerini gerçekleştirmesine ihtiyaç duyulmamaktadır. Dikkat edilmelidir ki anlatılan dağıtık yapıya sahip sistem merkezi yapıya sahip sistem ile karşılaştırıldığında hatalara karşı (bilgisayar/robot bozulması, haberleşmenin kopması gibi) daha gürbüz olabilmektedir. Bu nedenden dolayı algoritmanın eşzamansız ve dağıtık biçimi paralel uygulamalar ve dolayısıyla çok robotlu sistemler ile gerçekleştirilen uygulamalar için uygun olduğuna inanılmaktadır.

Algoritmanın dağıtık ve eşzamansız olan biçiminde parçacıklar özerk ve bağımsızlardır dolayısıyla parçacıklar farklı komşuluk gruplarına girebilir yada bu gruplardan çıkabilir hatta eniyileme işleminden işlemin çalışmasını engellemeden ayrılabilir veya eniyileme işlemine katılabilir. Bu nedenden dolayı sürüdeki parçacık sayısının sabit kalma ihtiyacı ortadan kalkmaktadır ve bu durum geliştirilen yöntemin

Çizelge 4.1.: Bir parçacık için eşzamansız ve dağıtık parçacık sürü eniyileme algoritmasının sahte kodu

Algoritmadaki parametrelerin ilklendirilmesi
Parçacığın ilk tahminlerinin $x^i(0)$ ve $v^i(0)$ vektörlerinin ilklendirilmesi
Eniyilenen fonksiyonun ilk değerinin hesaplanması $f(x^i(0))$
Parçacığın en iyi uyumluluk değerinin ve en iyi konumunun $p^i(0) = x^i(0)$ ve $f_p^i(0) = f(x^i(0))$ şeklinde ilklendirilmesi
Komşuluğun en iyi konumunun ilklendirilmesi $g^i(0)$.
Komşuluğun en iyi uyumluluk değerinin ilk değerinin $f_g^i(0) = f(g^i(0))$ şeklinde atanması
while (Durma koşuluna ulaşılmamışsa) **do**
  Komşulardan bilgi alınması için komşuların dinlenmesi
  if (Komşulardan bilgiler alınabiliyorsa ($t \in T_1^i$)) **then**
    Komşuluğun en iyi konumunun ($g^i(t)$) alınan bilgilere göre güncellenmesi
  end if
  if (Güncelleme gerçekleştirilebiliyorsa ($t \in T_2^i$)) **then**
    Yeni fonksiyon değerinin hesaplanması, $f(x^i(t))$
    if ($f(x^i(t)) < f_p^i(t-1)$) **then**
      En iyi konumunun ve uyumluluk değerinin $p^i(t) = x^i(t)$ ve $f_p^i(t) = f(x^i(t))$ olarak atanması
      $p^i(t)$ ve $f_p^i(t)$ değerlerinin diğer parçacıklara yayımlanması
    end if
    if $f_p^i(t) < f_g^i(t)$ **then**
      Komşuluğun en iyi konumunun ve uyumluluk değerinin $g^i(t) = p^i(t)$ ve $f_g^i(t) = f_p^i(t)$ olarak atanması
    end if
    Bir sonraki tahminin (2.3) denklemine göre hesaplanması
  else
    Bir sonraki tahminin hesaplanmaması (bakınız denklem (4.3))
  end if
end while

ölçeklendirilmesini[4] kolaylaştırmaktadır. Ayrıca parçacıkların/işlemcilerin geçici veya kalıcı haberleşme hatalarına karşın diğer parçacıklar/işlemciler işlemlerini sürdürdükleri için sistem eniyileme işlemine devam edebilmektedir. Bu avantajların dışında geliştirilen algoritma fonksiyon değerlerinin ($f(x^i(t))$ fonksiyon değerlerinin) belirlenmesi fonksiyonun her adımda hesaplanması yerine birtakım ölçümler ile belirlendiği (dış ortamdan elde edildiği) uygulamalar için daha uygun olduğu düşünülmektedir. Algoritmanın robotların çeşitli kimyasal algılayıcılar ile donatıldığı çok robotlu sistemler ile kapalı bir alanda bulunan zararlı kimyasal kaynakların

[4]ing: scability

aranması ve belirlenmesi için kullanılması bu tür bir uygulama örnek olabilir.

Önerilen eşzamansız algoritmada önemli bir unsur da algoritmanın durma koşulunun belirlenmesidir. Bu durumda algoritmanın birçok uygulamasında olduğu gibi bu koşul önceden belirlenebilir. Algoritmanın başında sabit bir azami yineleme sayısının seçilmesi belirtilen duruma bir örnek olabilir. Bir başka deyişle parçacıklar/erkinler/işlemciler belirli bir sayıda yineleme gerçekleştirdikten sonra işlemlerini bitirmeleri sağlanabilir. Başka bir koşul olarak da parçacıklar belirli bir süre veya yineleme boyunca fonksiyon değerlerinde yeterli artış/azalış gözlemleyemez ise parçacıkların işlemlerini durdurması olabilir. Parçacıklar arası bir müzakereye dayanan bir durma koşulu da belirlenebilir. Bu tip bir durma koşulunda sürüdeki herhangi bir parçacık diğer parçacıklara durma isteği iletebilir ve diğer parçacıklar da durma veya devam etme konusunda müzakere yapabilirler. Yukarıda belirtilen durma koşulları ve diğer koşullar bir arada kullanılarak değişik durma koşulları da belirlemek mümkündür. Öte yandan parçacıklara özel durma koşulları da belirlenebilir (farklı parçacıkların farklı durma koşulları olabilir). Örneğin sürüdeki bir parçacığın 500 yineleme sonrası, diğer bir parçacığın 1000 yineleme sonrası durması belirlenebilir.

Algoritmanın önerilen biçiminde her parçacık kendi yinelemesini gerçekleştirmesinden dolayı farklı parçacıklar için farklı parametreler (χ ve öğrenme katsayıları φ_1^i ve φ_2^i gibi) kolayca ayarlanabilir. Bu bağlamda sürüdeki bazı parçacıklar daha kabataslak arama (büyük $\bar{\varphi}_1$ ve $\bar{\varphi}_2$ değerlerine sahip parçacıklar), bazı parçacıklar ise daha ayrıntılı arama (küçük $\bar{\varphi}_1$ ve $\bar{\varphi}_2$ değerlerine sahip parçacıklar) gerçekleştirerek farklı arama yönelimleri olan parçacıklar elde edilebilir.

Burada belirtmek istediğimiz bir husus da algoritmada her parçacık komşuluğun en iyi konumunu $g^i(t)$ ile ilgili farklı zaman anlarında algıları olduğudur. Bu gerçekten dolayı herhangi bir zaman anında farklı parçacıkların komşuluğun en iyi konumu ile ilgili farklı algıları bulunmaktadır (parçacıklar farklı zaman anlarında farklı parçacıklar ile etkileşim içinde olduğu için).

Algoritmada kullanılan güncelleme denklemi yöntemin temel biçiminde kullanılan denklemler ile aynı olmasına karşın komşuluğun en iyi konumu olan $g^i(t)$'nin hesaplanması farklı bir felsefeye göre gerçekleştirilmekte, böylece algoritmanın eşzamansız ve dağıtık biçiminin uygulanmasına olanak sağlanmıştır. Ayrıca geliştirilen yöntemde zaman gecikmeleri veya diğer nedenlerden ötürü oluşan belirsizliklere de izin verilmektedir. Bütün bu özellikler parçacıkların farklı işlemcilerde programlanarak ve komşuluğundaki/ortamdaki parçacıklar ile

haberleşerek algoritmanın gerçekleştirilmesi gibi dağıtık işlemci/parçacık/erkin tabanlı uygulamaların (doğadaki sürülerde olduğu gibi) yapılmasına olanak sağlamaktadır. Algoritmada işlemlerin eşzamanlılığı için herhangi bütünsel bir saat kullanılmamakta ve ağdaki gecikmeler geliştirilen matematiksel modelde temsil edilebilmektedir.

Yukarıdaki tartışmaların ışığında zaman gecikmelerinin ele alındığı dinamik komşuluklu, merkezi olmayan/dağıtık ve eşzamansız PSO algoritmasının çalışma felsefesi yöntemin ilhamını aldığı doğadaki dağıtık çok erkinli sistemlerin (balık, kuş, arı sürüleri gibi sosyal sürüler) davranışları birçok benzerlik taşımaktadır. Ayrıca yöntemin önerilen biçimi yöntemin birçok yeni uygulama alanında kullanılmasına olanak sağlamaktadır. Daha önceden sözü edilen PSO yöntemi kullanılarak çok robotlu bir sistemin örneklenmiş bir ortamda belirli bölgelerin veya hedeflerin aranması yeni bir uygulama alanı olarak görülebilir. Örneklenmiş ortamlarda eğim tabanlı yöntemlerin uygulanmasının zorluğundan dolayı PSO gibi eğim tabanlı olmayan doğrudan arama yöntemlerinin kullanılması daha verimli olmaktadır. Çok robotlu sistemlerde robotların işlemlerini eşzamanlı olarak gerçekleştirmesi, işlemlerinin bütünsel bir saate göre gerçekleşmesi nedeniyle sistemin dağıtık çalışma özelliğini zayıflatmaktadır. Bu nedenden dolayı geliştirilen yöntemin bu tip uygulamalarda kullanılması sistemin başarımını artırmaktadır.

4.4. Benzetim Sonuçları

Eşzamansız, dağıtık ve dinamik komşuluklu parçacık sürü eniyileme yönteminin sınanması için çeşitli denektaşı fonksiyonların eniyilenmesi problemi ele alınmıştır. Benzetimler için Bölüm 2.8.'de belirtilen küre, Griewank, Rastrigin ve Rosenbrock fonksiyonları kullanılmıştır (fonksiyonların matematiksel ifadeleri için bakınız Çizelge 2.2.).

Benzetimlerde her fonksiyon için arama uzayının boyutu $n = 20$ olarak seçilmiş ($\mathbb{R}^{20}$) ve sürüdeki parçacık sayısı, $N = 100$ olarak belirlenmiştir. Algoritmanın durma koşulu azami yineleme sayısının geçilmesi olarak tanımlanmış ve azami yineleme sayısı 10000 olarak belirlenmiştir. Yöntemin kısıtlama katsayılı biçimi (bakınız Denklem 2.3) kullanılmış, öğrenme katsayılarının üst sınırları $\bar{\varphi}_1, \bar{\varphi}_2$ = 2.05 ve kısıtlama katsayısı $\chi = 0.7298$ olarak belirlenmiştir.

4.4.1. Tek bir İşlemci Üzerinde Gerçekleştirilen Benzetimler

Benzetimler ilk olarak tek bir işlemcide sıralı olarak gerçekleştirilmiştir. Bu nedenden dolayı bilgi değişimindeki ve güncellemelerdeki eşzamansızlık, olasılıklı yöntemler kullanılarak yapay olarak yöntemde tanımlanmıştır (bakınız Çizelge 4.2.). Ayrıca parçacık komşulukları arama uzayında en yakın komşular kuralı kullanılarak dinamik olarak belirlenmektedir. Her denektaşı fonksiyon için belirlenen arama alanları göz önünde bulundurularak 30 adet ilk koşul üretilmiştir (bakınız Çizelge 2.2.)

Çizelge 4.2.'den de görülebileceği üzere her zaman anında eşzamansızlığı elde etmek için parçacıkların komşularını algılaması (bilgi değişiminde bulunması) ve durumlarını Denklem 2.3'e göre güncellemeleri için belirli olasılıklar atanmıştır. Bu amaçla $0 < \bar{p}_{sense} < 1$ ve $0 < \bar{p}_{move} < 1$ gibi iki adet olasılık değeri tanımlanmıştır. Her t zaman adımında, i parçacığı için $[0, 1]$ aralığında p^i_{move} ve $p^{ij}_{sense}(t), j = 1, \ldots, N_i(t)$ gibi toplamda ($N_i(t)$+1) adet düzgün rasgele sayı üretilmektedir. Düzgün rasgele üretilen $p^{ij}_{sense}(t) > \bar{p}_{sense}$ sayıları i parçacığının komşusu olan $j \in N_i(t)$ parçacığından bilgi alıp alamayacağını belirlemektedir. Eğer $p^{ij}_{sense}(t) > \bar{p}_{sense}$ olursa, i parçacığı o anda komşusu olan j parçacığından bilgi alabilmekte ve bu bilgiyi komşuluğun en iyi konumunu belirlerken kullanabilmektedir. Benzer şekilde eğer $p^i_{move}(t) > \bar{p}_{move}$ ise, i parçacığı tahminini Denklem 2.3 kullanarak güncellemekte, parçacık "uyanık" durumda olmaktadır. Aksi takdirde parçacık güncelleme yapmadan önceki tahmini kullanarak bir sonraki yinelemeye geçmektedir (bakınız Denklem 4.3). Dikkat edilirse bu tip bir uygulama gerçek anlamda eşzamansız bir uygulama değildir ancak yöntemin istenen şartlar (eşzamansızlık) altında nasıl bir başarım sergileyeceği hakkında fikirler vermektedir.

Önceden de belirtildiği gibi benzetimlerde parçacık komşuluklarının dinamik olarak değiştiği kabul edilmiştir. Her i parçacığının $\delta_i > 0$ gibi bir algılama alanına sahip olduğu düşünülmüş ve parçacığın sadece algılama alanı içerisindeki parçacıklar ile bilgi değişimi yapabildiği kabul edilmiştir. Parçacıkların komşuluk kümelerinin belirlenmesi için arama uzayında en yakın komşular yöntemi benimsenmiştir ve bu yöntem matematiksel olarak önceki bölümde anlatılan Denklem 3.1 ile betimlenmektedir. Benzetimlerde parçacıkların algılama alanları eşit, $\delta_i = \delta$, olarak seçilmiş, böylece karşılıklı (çift yönlü) komşuluk ilişkileri düşünülmüştür (herhangi bir i parçacığı herhangi bir j parçacığı ile komşu ise j parçacığı da i parçacığının komşusudur).

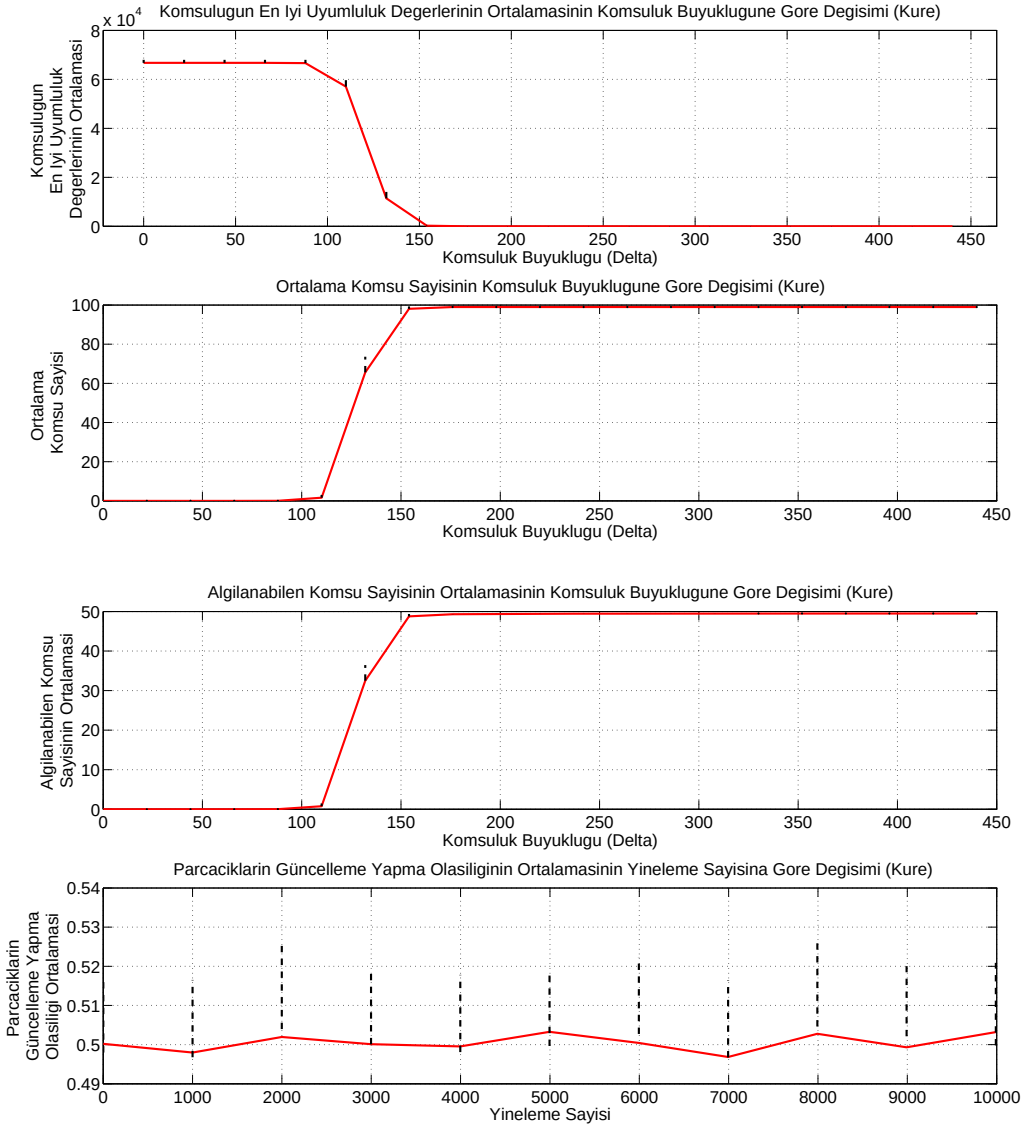
Şekil 4.1.'den Şekil 4.4.'e kadar olan grafiklerde her dört denektaşı fonksiyon için

Çizelge 4.2.: Tek bir işlemci ile gerçekleştirilen benzetimlerde kullanılan algoritmanın sahte kodu

Algoritmadaki parametrelerin ilklendirilmesi
Parçacığın ilk tahminlerinin $x^i(0)$ ve $v^i(0)$ vektörlerinin ilklendirilmesi
Eniyilenen fonksiyonun ilk değerinin hesaplanması $f(x^i(0))$
Parçacığın en iyi uyumluluk değerinin ve en iyi konumunun $p^i(0) = x^i(0)$ ve $f_p^i(0) = f(x^i(0))$ şeklinde ilklendirilmesi
Komşuluğun en iyi konumunun ilklendirilmesi $g^i(0)$
Komşuluğun en iyi uyumluluk değerinin ilk değerinin $f_g^i(0) = f(g^i(0))$ şeklinde atanması
$\bar{p}_{sense}$, $\bar{p}_{move}$ ve δ_i değerlerinin ilklendirilmesi
while (Yineleme sayısı < 10000) **do**
  for her i parçacığı için **do**
    Denklem 3.1 kullanılarak komşuluk kümesinin, $N_i(t)$, belirlenmesi
    for her $j \in N_i(t)$ parçacığı için **do**
      $p_{sense}^{ij}(t)$ olasılığının üretilmesi
      if $p_{sense}^{ij}(t) > \bar{p}_{sense}$ **then**
        Gecikme miktarının (τ_j^i) rasgele belirlenmesi
        $p^i(\tau_j^i)$ vektörü kullanılarak komşuluğun en iyi konumu $g^i(t)$'nin güncellenmesi
      end if
    end for
    $p_{move}^i(t)$ olasılığının üretilmesi
    if $p_{move}^i(t) > \bar{p}_{move}$ **then**
      Fonksiyonun yeni değerinin hesaplanması, $f(x^i(t))$
      if ($f(x^i(t)) < f_p^i(t-1)$) **then**
        $p^i(t) = x^i(t)$ ve $f_p^i(t) = f(x^i(t))$ olarak atanması
      end if
      if $f_p^i(t) < f_g^i(t)$ **then**
        $g^i(t) = p^i(t)$ ve $f_g^i(t) = f_p^i(t)$ olarak atanması
      end if
      Denklem 2.3 kullanılarak konum ve hız vektörlerinin ($x^i(t)$ ve $v^i(t)$) güncellenmesi
    else
      Hız ve konum vektörlerinin ($x^i(t)$ ve $v^i(t)$) güncellenmemesi (bakınız Denklem 4.3)
    end if
  end for
end while

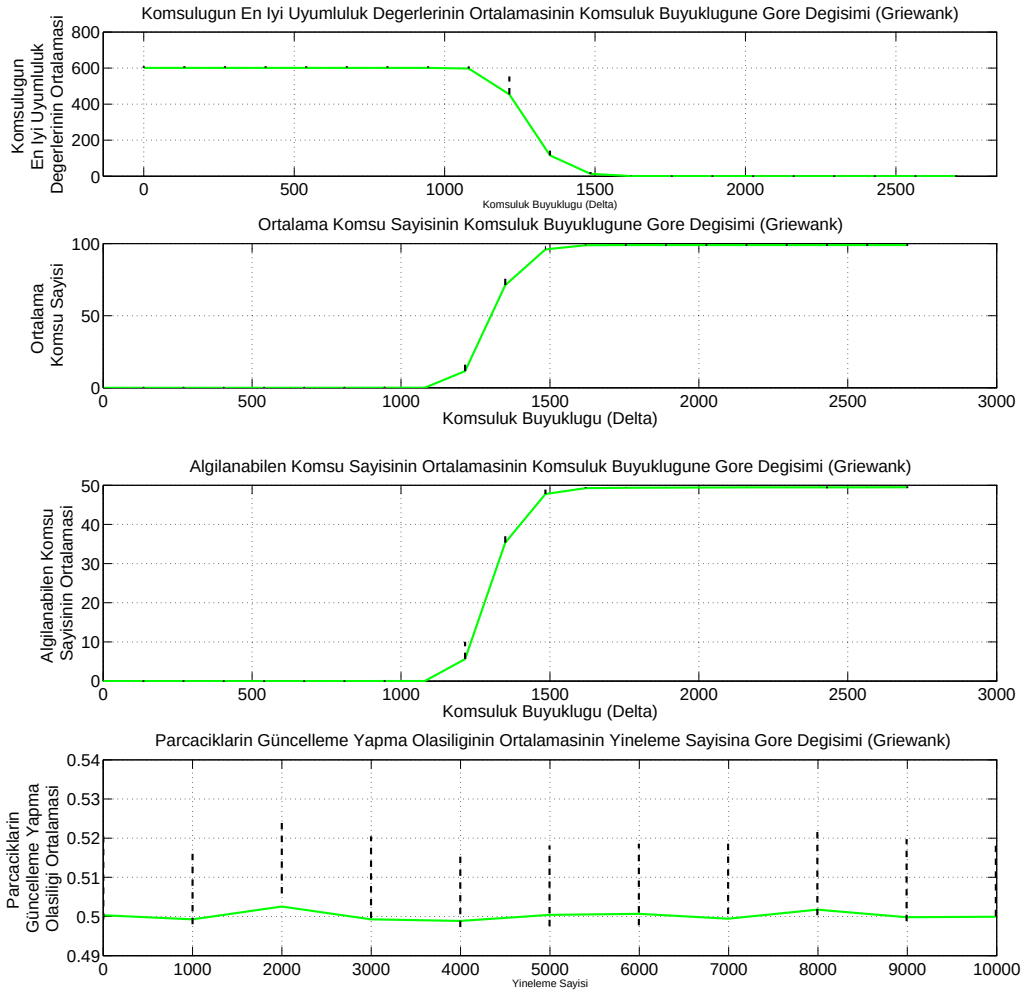
komşuluğun en iyi uyumluluk değerinin ortalamasının, parçacıkların komşu sayısının ortalamasının ve algılanabilen komşu parçacıkların sayısının ortalamasının komşuluk büyüklüğüne (δ_i) göre değişimi ve parçacıkların güncelleme yapma olasılığının

ortalamasının yineleme sayısına göre değişimi gösterilmektedir. Benzetimler için $\bar{p}_{sense}$ ve $\bar{p}_{move}$ olasılık değerleri 0.5 olarak seçilmiştir.



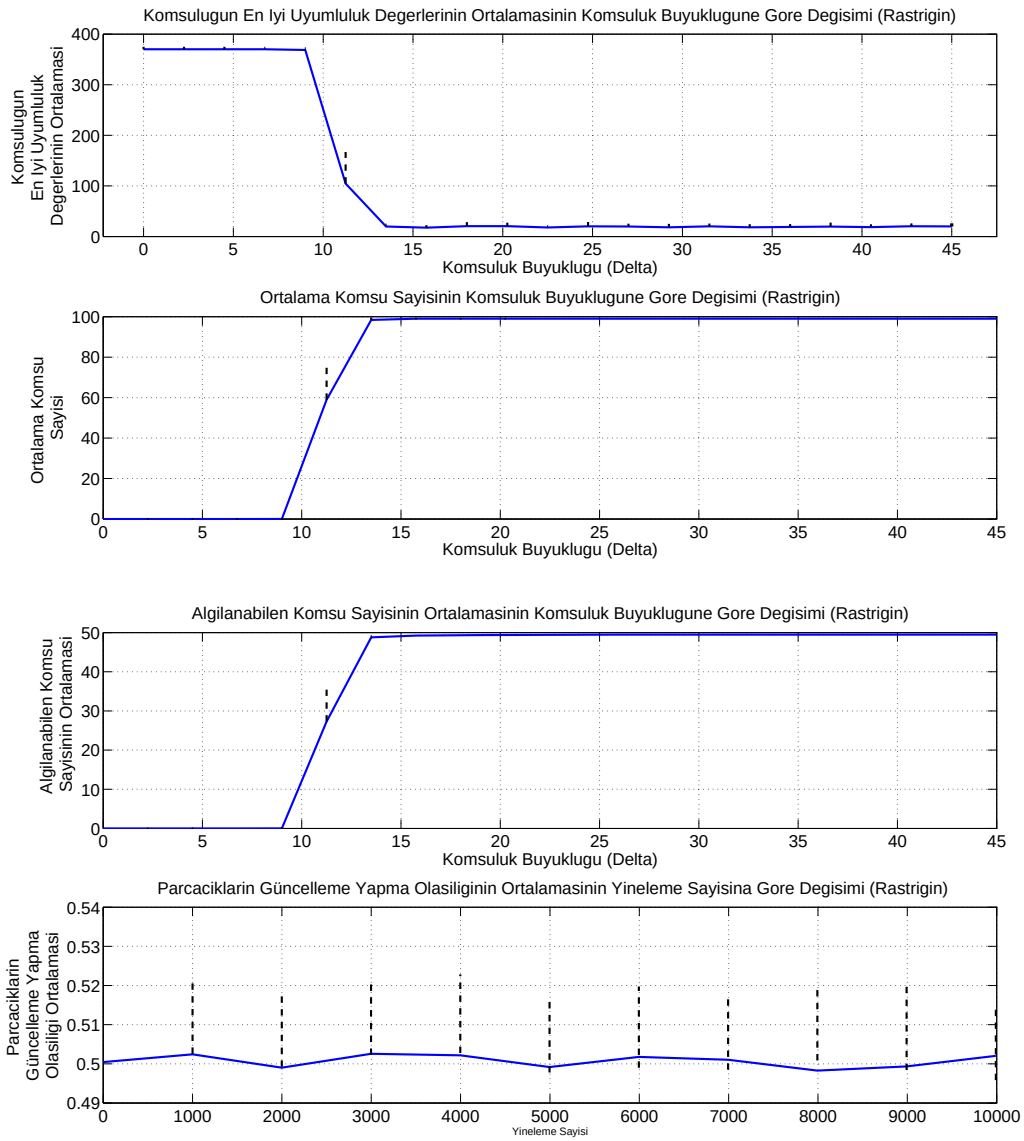
Şekil 4.1.: Komşuluğun en iyi uyumluluk değerinin ortalamasının, parçacıkların komşu sayısının ortalamasının ve algılanabilen komşu parçacıkların sayısının ortalamasının komşuluk büyüklüğüne göre değişimi ve parçacıkların güncelleme yapma olasılığının ortalamasının yineleme sayısına göre değişimi (Küre).

Geliştirilen yöntemin başarımı 21 farklı komşuluk büyüklüğünde incelenmiştir (bakınız Şekil 4.1., Şekil 4.2., Şekil 4.3., Şekil 4.4. ilk üç grafik). Her fonksiyon için komşuluğun en iyi uyumluluk değeri ($evrensel_{eniyi}$) benzetimin sonunda bir başka deyişle $t = 10000$ anında grafiklere çizilmiştir. Buna karşın komşu



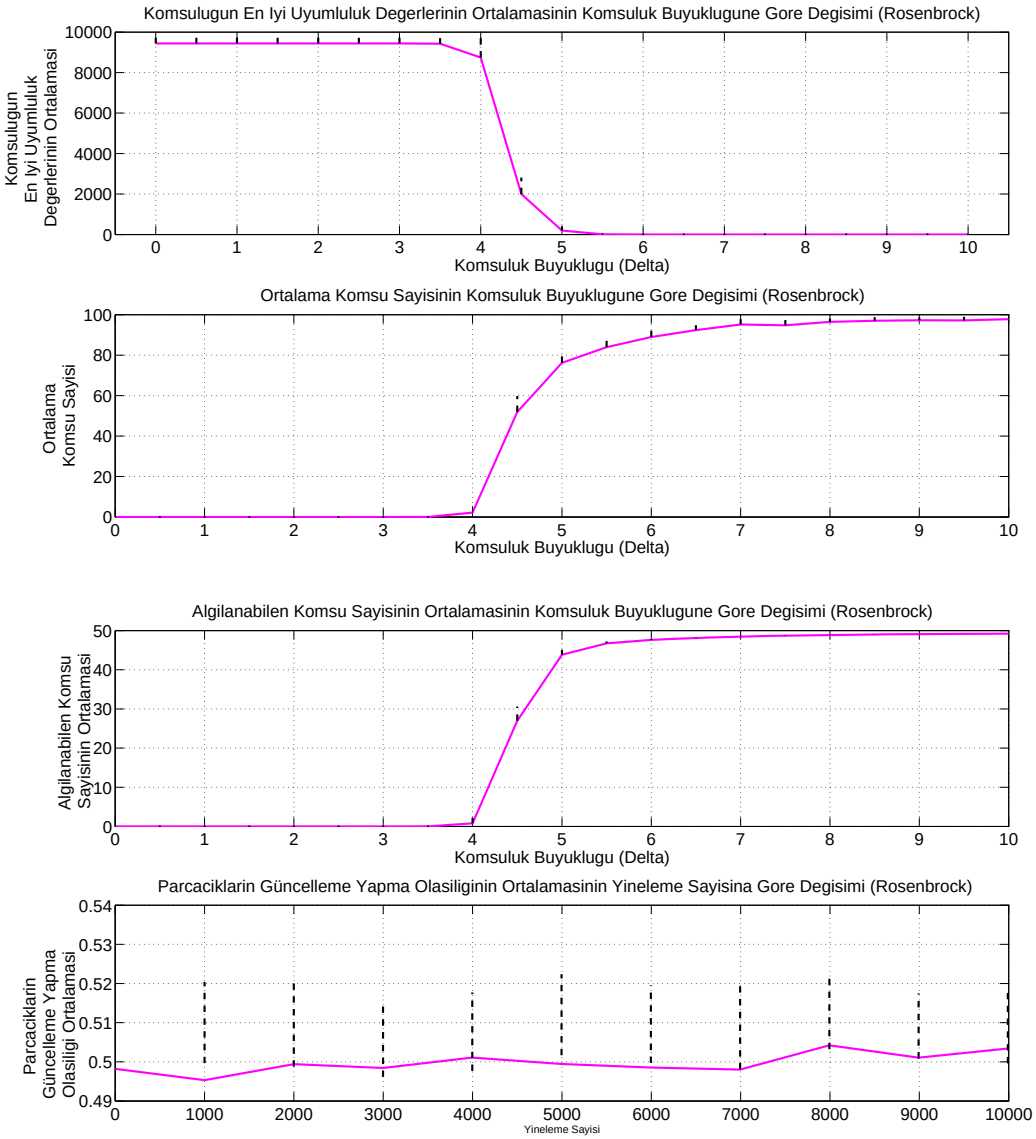
Şekil 4.2.: Komşuluğun en iyi uyumluluk değerinin ortalamasının, parçacıkların komşu sayısının ortalamasının ve algılanabilen komşu parçacıkların sayısının ortalamasının komşuluk büyüklüğüne göre değişimi ve parçacıkların güncelleme yapma olasılığının ortalamasının yineleme sayısına göre değişimi (Griewank).

parçacık sayısının ortalaması ve algılanabilen komşu parçacık sayısının ortalaması bütün yinelemeler boyunca kaydedilmiştir ($t = 0$ anından $t = 10000$ anına kadar). Ayrıca her adımda parçacıkların güncelleme yapabilme olasılıklarının ortalaması kaydedilmiş ve yineleme sayısına göre grafiklerde çizilmiştir. Her denektaşı fonksiyon için arama alanının boyutu farklı alındığı için komşuluk büyüklükleri de farklı değerlerde alınmıştır. Bölüm 3.'te aynı koşullar altında parçacıkların algılama alanlarının yaklaşık arama uzayındaki en büyük mesafenin dörtte biri olarak seçilmesinin parçacık sürüsünün Varsayım 1'in ve sürüdeki bilgi akışının devamlılığının sağladığı belirlenmiştir. Bu gerçek komşu parçacık sayısının



Şekil 4.3.: Komşuluğun en iyi uyumluluk değerinin ortalamasının, parçacıkların komşu sayısının ortalamasının ve algılanabilen komşu parçacıkların sayısının ortalamasının komşuluk büyüklüğüne göre değişimi ve parçacıkların güncelleme yapma olasılığının ortalamasının yineleme sayısına göre değişimi (Rastrigin).

ortalamasının komşuluk büyüklüğüne göre değişiminde de görülmektedir. Bir parçacığın komşu sayısı o parçacığın algılayabileceği ve haberleşebileceği parçacık sayısını göstermektedir. Öte yandan her parçacığın t zamanında herhangi bit komşusunu algılaması için bir olasılığın ($p^{ij}_{sense}(t)$) olması nedeniyle parçacık t zamanındaki bütün komşuları ile haberleşemeyebilir ve bu durum yönteme bir eşzamansızlık getirmektedir. Algılanabilen komşu sayısının ortalamasının komşuluk



Şekil 4.4.: Komşuluğun en iyi uyumluluk değerinin ortalamasının, parçacıkların komşu sayısının ortalamasının ve algılanabilen komşu parçacıkların sayısının ortalamasının komşuluk büyüklüğüne göre değişimi ve parçacıkların güncelleme yapma olasılığının ortalamasının yineleme sayısına göre değişimi (Rosenbrock).

büyüklüğüne göre değişiminin gösterildiği grafiklerde komşuluk büyüklüğünün küçük olduğu durumlarda algılanabilen parçacık sayısının az olmasından ötürü yöntemin istenen başarımı sergileyemediği görülmektedir. Parçacıkların algılama alanlarının genişlemesi ile algılanabilen komşu parçacık sayısı artmakta böylece parçacık sürüsündeki bağlanırlık sağlanmakta ve istenen başarımı sergilemektedir. Parçacıkların komşu parçacıkları algılama olasılıkları olan $\overline{p}_{move}$ değerinin 0.5

alınması parçacıkların her adımda toplam komşularının yaklaşık yarısından bilgi alabilme şansı olduğunu göstermektedir. Parçacıkların güncelleme yapma olasılığının ortalamasının yinelemelere göre değişiminin gösterildiği grafiklerde her yinelemede sürüdeki parçacıkların yaklaşık yarısının ($\%50$) güncelleme yapıldığı görülmektedir ve bu durumun nedeni her parçacık için $\bar{p}_{move}$ olasılığının 0.5 olarak seçilmesidir. Bu durum yönteme bir eşzamansızlık daha getirmektedir.

Gerçekleştirilen ilk benzetimler sonucunda geliştirilen yöntemde bilgi değişimindeki ve güncellemelerdeki eşzamansızlığa, zaman gecikmelerine ve komşuluk yapılarının dinamik olarak değişmesine karşın parçacıkların algılama alanlarının (δ_i) yeterince büyük seçilmesi sonucunda sürüdeki bilgi akışının sürekliliği sağlanmış ve önerilen yöntemin temel PSO yöntemi ile karşılaştırabilir bir başarım sergilediği gözlemlenmiştir.

4.4.2. Bir Bilgisayar Ağında Bulunan İşlemciler ile Gerçekleştirilen Benzetimler

Geliştirilen yöntem daha sonra bir bilgisayar ağında bulunan birçok işlemcinin kullanıldığı benzetimler ile sınanmıştır. Daha doğrusu benzetimlerde 4 adet işlemci kullanılmaktadır. Önceki bölümdeki benzetimlerin aksine bu benzetimlerde birden çok işlemci kullanılması ve her işlemcinin işlemleri farklı zaman anlarında gerçekleştirmesi nedeniyle yöntemin eşzamansız çalışması için yönteme herhangi bir ekleme yapılmamıştır. Ayrıca ağdaki haberleşme yoğunluğu ve yukarıda da belirtildiği gibi işlemcilerin işlemlerini farklı zaman anlarında gerçekleştirmelerinden dolayı işlemciler arasında eşzamansız bilgi paylaşımı ortaya çıkmaktadır. Bunlara ek olarak yöntemin birden çok işlemci tarafından paralel olarak gerçekleştirilmesi, yöntemin merkezi olmayan/dağıtık özelliği kendiliğinden ortaya çıkmasına neden olmuştur.

Her fonksiyon için Bölüm 2.8.'de belirtilen arama alanlarını göz önünde bulundurularak bir adet başlangıç koşulu belirlenmiştir (bakınız Çizelge 2.2.). Benzetimler için sürüdeki parçacık sayısı 100 olarak alınmıştır, ancak yöntemin dağıtık doğasından dolayı parçacıklar kullanılan 4 işlemci arasında paylaştırılmış böylece her işlemciye 25 parçacık verilmiştir. Bu durum 100 parçacıktan oluşan bir parçacık sürüsünün ilk başta alt sürülere bölünerek farklı konumlardan aramaya başlaması; arama sırasında alt sürülerde bulunan parçacıkların komşuluk yapılarının dinamik olarak değişmesi; alt sürülerin eşzamansız olarak bilgi paylaşımında bulunarak eşzamansız arama gerçekleştirmesi ve arama işleminin sonunda aynı konuma yakınsamaları olarak düşünülebilir. Her işlemcide bulunan her parçacığının Çizelge 4.1.'de belirtilen algoritmayı gerçekleştirmesi istenmektedir. Parçacık komşuluklarının

belirlenmesi için rasgele belirleme kuralı kullanılmıştır. Bu kural matematiksel olarak önceki bölümlerde sunulan Denklem 3.4 ile betimlenmiştir. (Elde edilen sonuçların sunulduğu çizelgelerde işlemcilerin ilk yinelemeler boyunca nasıl bir başarım izlediği ve işlemcilerin uzayda aynı konuma nasıl yakınsadıklarını belirtmek için çizelgelerin y ekseni olan yineleme sayısı ekseni log_{10} tabanı düşünülerek çizilmiştir.)

Benzetim sırasında farklı işlemcilerde bulunan parçacıklar arası haberleşme TCP/IP protokolü kullanılarak gerçekleştirilmiştir. Bu amaçla işlemci diğer işlemcilere bağlanmak için önceden belirlenen bir bağlantı noktasında[5] her diğer işlemci için birer TCP soketi oluşturmakta ve oluşturulan soketler üzerinden diğer işlemciler ile bağlanmaktadır. İşlemci diğer işlemcilerden bilgi alabilmek amacıyla aynı bağlantı noktasında bir TCP soketi daha oluşturmakta ve açılan bu soketi belirlenen bağlantı noktasına bağlamaktadır. Bağlanan bu soket kullanılarak ağ dinlenmekte ve önceden bağlantı sağlanan diğer işlemciler ile başarılı bir kabul işlemi gerçekleştikten sonra bilgi alınabilmektedir. İşlemci diğer işlemcilere bilgi göndermek için daha önceden bağlantı sağlanan her diğer işlemci için oluşturulan TCP soketlerini kullanmaktadır.

Geliştirilen yöntemin paralel çalışması için işlemcilerin bilgi alması ve bilgi göndermesi işlemleri alt süreçler[6] olarak tanımlanmıştır. Bu durumda PSO algoritması üst süreç olarak tanımlanmış ve işlemcilerin bilgi alma ve gönderme işlemleri üst sürecin özkaynaklarını kullanan ve üst süreçle birlikte alt seviyede çalışan işlemler olarak benimsenmiştir. Bu durumun bir sonucu olarak farklı işlemcilerde bulunan parçacıklar eşzamansız olarak birbirlerine bilgi yollayabilmekte ve birbirlerinden bilgi alabilmektedir. Eşzamansız bilgi paylaşımı geliştirilen yöntemde kendiliğinden oluşması beklenen bir davranış[7] olduğu daha önce de belirtilmiştir. Herhangi bir işlemciler bulunan her parçacık yinelemesi sırasında komşuluğun en iyi uyumluluk değeri olan $evrensel_{eniyi}$ ve en iyi konumu olan $g^i(t)$'yi belirlemek amacıyla diğer işlemcilerde bulunan parçacıklardan elde ettiği bilgileri kullanmaktadır. Daha sonra parçacıklar kendi en iyi uyumluluk değeri olan $kisisel_{eniyi}$ ve kendi en iyi konumu olan $p^i(t)$'yi güncelledikten sonra bu bilgiyi ve aynı işlemcide bulunan komşu parçacıklardan elde ettiği bilgileri de kullanarak komşuluğun en iyi uyumluluk değerini ve konumunu belirlemektedir. Parçacıklar yinelemenin sonunda elde ettikleri kendi en iyi uyumluluk değerlerini ve konumlarını diğer işlemcilerde bulunan parçacıklara iletmektedir. Farklı işlemcilerde bulunan parçacıklar yineleme farklı zamanlarda bitirmelerinden dolayı bilgi gönderme işlemlerini farklı zaman

[5] ing: port
[6] ing: threads
[7] ing: emergent behaviour

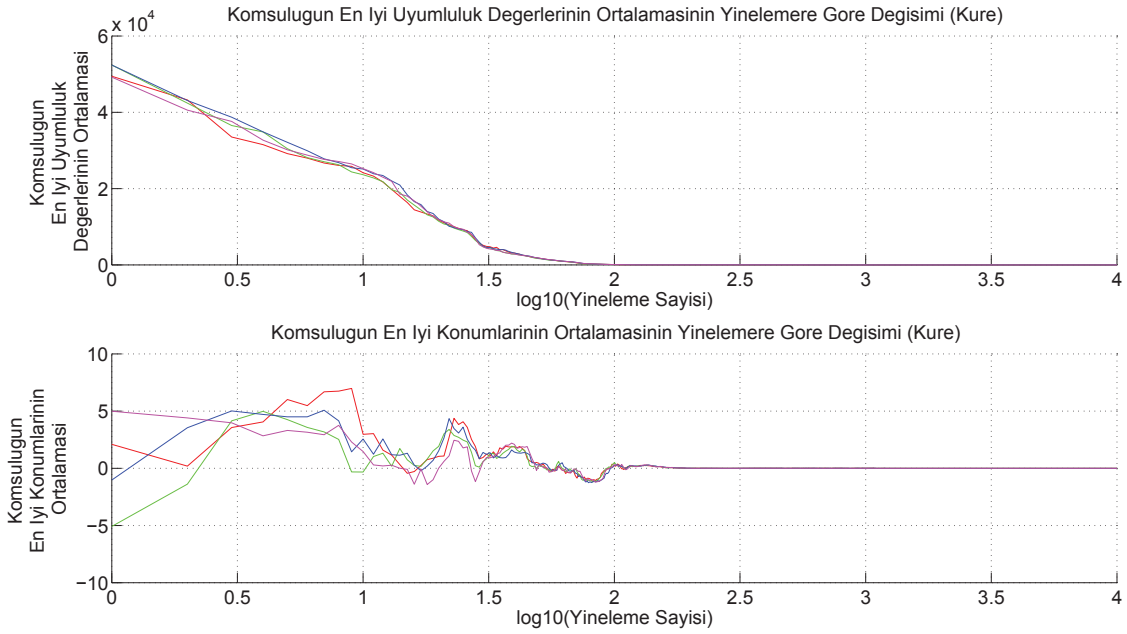
anlarında gerçekleştirmekte böylece diğer işlemcilerdeki parçacıklar tarafından alınan bilgilerde zaman gecikmeleri mevcut olabilir. Bir başka deyişle farklı işlemcilerde bulunan parçacıklar birbirlerinin en güncel bilgilerini kullanmadan komşuluğun en iyi uyumluluk değerini ve konumunu belirleyebilir. Öte yandan farklı işlemcilerde bulunan parçacıkların farklı zaman anlarında bilgi paylaşımında bulunmasından dolayı parçacıklar arasındaki komşuluk yapısının zamana göre değişmesi kendiliğinden oluşan bir davranış olmaktadır. Farklı işlemcilerde bulunan parçacıklar yinelemelerine farklı zaman anlarında bitirmeleri ve farklı bir işlemcide bulunan bir parçacığın diğer işlemcilerdeki bütün parçacıklar ile aynı anda bilgi paylaşımında bulunamaması sebebiyle işlemciler arasındaki komşuluk ilişkileri dinamik olarak değişmektedir.

Benzetimler için ilk olarak işlem güçleri birbirine eş işlemciler kullanılmış (bakınız Çizelge 4.3.) ve Linux işletim sisteminde çalışılmıştır. Başlangıç konum vektörü dörde bölünerek her işlemciye atanmıştır. Şekil 4.5.'ten Şekil 4.8.'e kadar olan grafiklerde dört denektaşı fonksiyon için her işlemcide bulunan alt sürünün elde ettiği komşuluğun en iyi uyumluluk değerlerinin ortalamaları ve komşuluğun en iyi konumunun değerlerinin ortalamaları gösterilmiştir. Her işlemcideki alt sürü farklı konumlardan arama işlemine başladığı için ilk yinelemelerde alt sürülerin komşuluğun en iyi uyumluluk değerleri ve komşuluğun en iyi konumlarının ortalama değerleri farklı olmaktadır. Yinelemeler devam ettikçe ve daha iyi uyumluluk değerlerine ulaşan alt sürüler diğer alt sürüler ile bilgi paylaşımında bulundukça bütün alt sürüler aynı konuma yakınsayabilmektedir. Öte yandan grafiklerde gösterilen komşuluğun en iyi konumun ortalama değerlerinin dört denektaşı fonksiyon için 2.8. bölümde belirtilen bütünsel en küçük konumlara yakın konumlara ulaştığı görülmektedir.

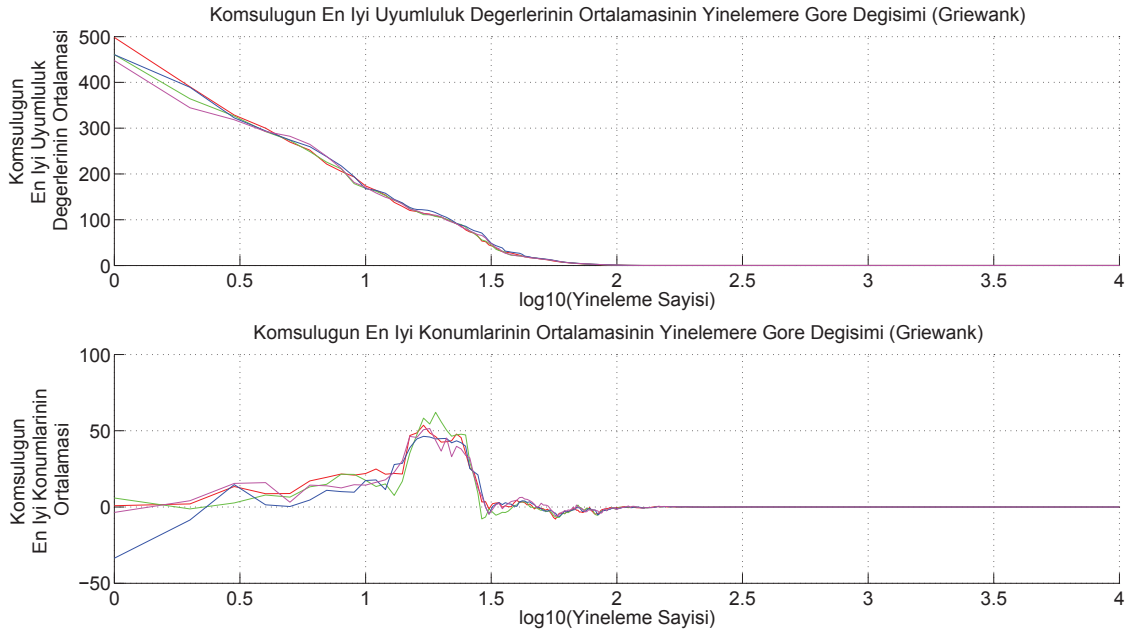
Çizelge 4.3. Türdeş ağda gerçekleştirilen benzetimlerde kullanılan işlemciler

No	CPU Hızı	CPU Tipi	Hafıza	Ağ
1	2.4 Ghz	Intel(R) Core(TM)2	1.98 GB	Gigabit Ethernet
2	2.4 Ghz	Intel(R) Core(TM)2	1.98 GB	Gigabit Ethernet
3	2.4 Ghz	Intel(R) Core(TM)2	1.98 GB	Gigabit Ethernet
4	2.4 Ghz	Intel(R) Core(TM)2	1.98 GB	Gigabit Ethernet

Benzetimler daha sonra farklı işlem gücü ve ağ hızlarına sahip işlemciler ile gerçekleştirilmiştir (bakınız Çizelge 4.4.). Bu işlemciler ile yapılan benzetimlerde işlemciler farklı özelliklere sahip oldukları için yönteme daha çok eşzamansızlık gelmiştir. Ayrıca türdeş ağda yapılan benzetimler ile aynı başlangıç koşulu kullanılarak eşzamansızlığın yöntemin başarımına olan etkisi saptanmaya çalışılmıştır. Dört denektaşı fonksiyon için elde edilen sonuçlar Şekil 4.9., Şekil 4.10., Şekil 4.11.

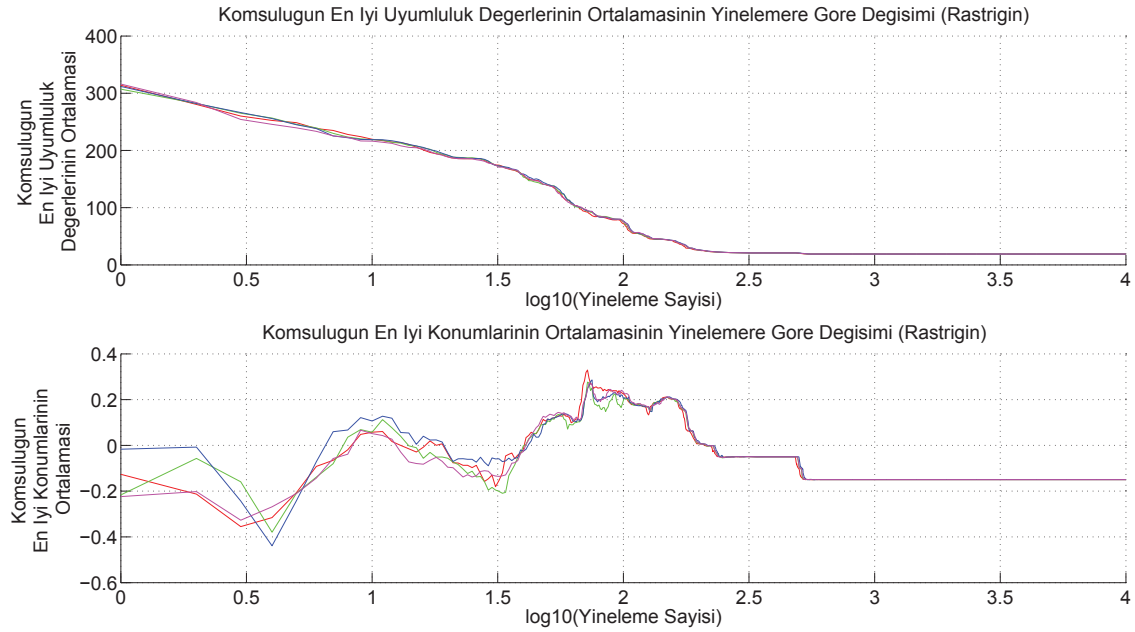


Şekil 4.5.: Türdeş ağda komşuluğun en iyi uyumluluk değerlerin ve komşuluğun en iyi konumlarının yinelemelere göre değişimi (Küre).

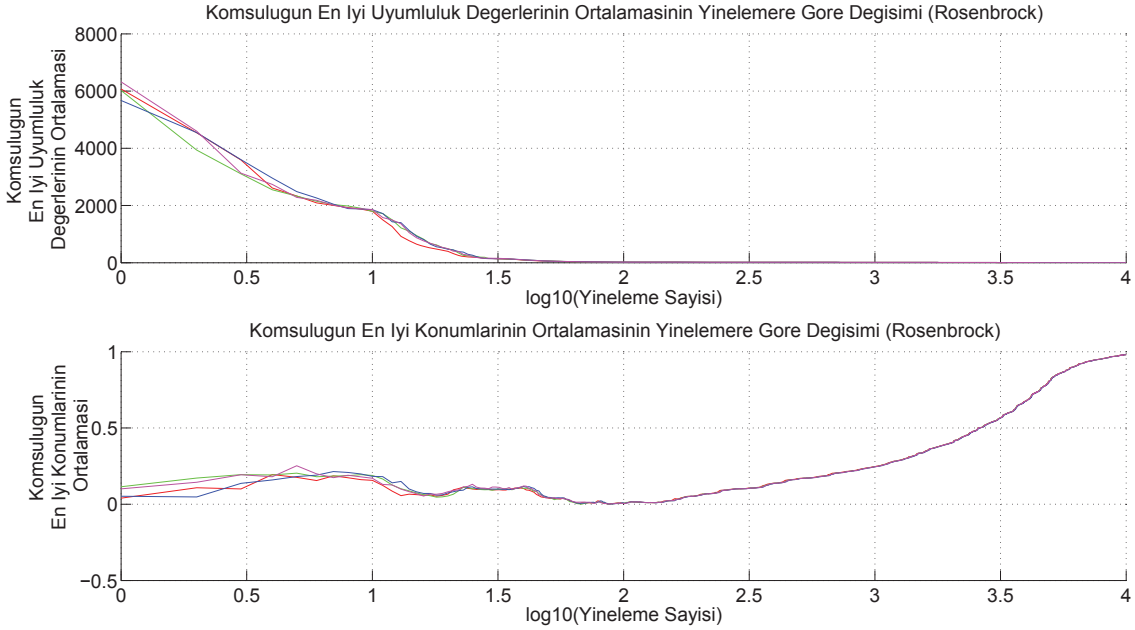


Şekil 4.6.: Türdeş ağda komşuluğun en iyi uyumluluk değerlerin ve komşuluğun en iyi konumlarının yinelemelere göre değişimi (Griewank).

ve Şekil 4.12. de belirtilmiştir. Grafiklerden de görüldüğü üzere sistemdeki artan eşzamansızlığa karşın elde edilen sonuçlar türdeş ağdaki işlemciler ile elde edilen sonuçlar ile karşılaştırılabilir düzeydedir.



Şekil 4.7.: Türdeş ağda komşuluğun en iyi uyumluluk değerlerin ve komşuluğun en iyi konumlarının yinelemelere göre değişimi (Rastrigin).

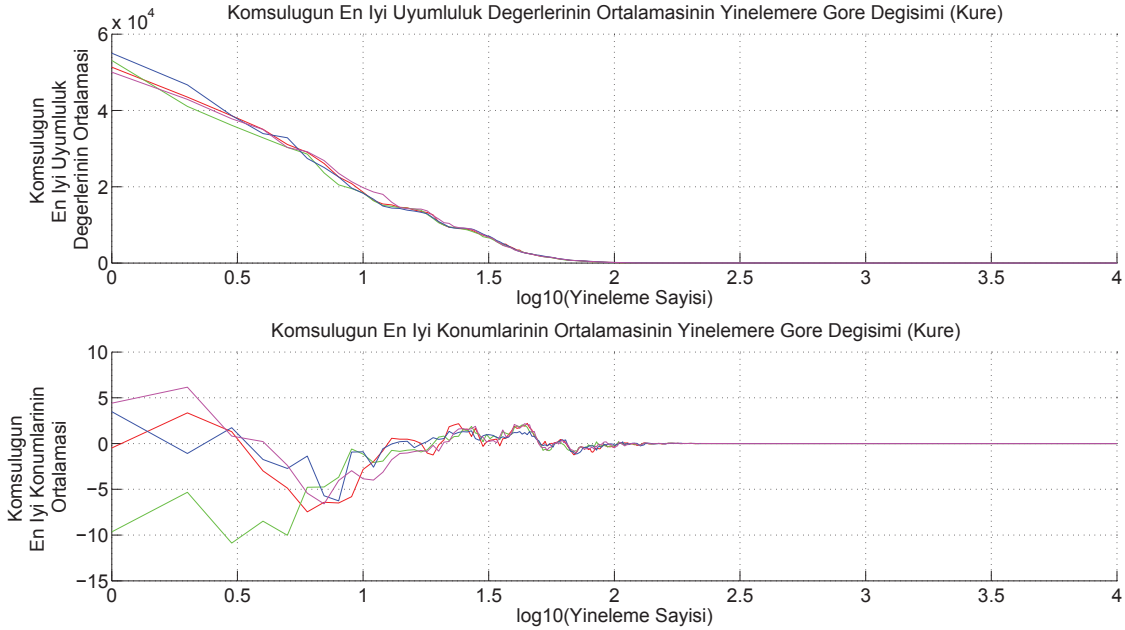


Şekil 4.8.: Türdeş ağda komşuluğun en iyi uyumluluk değerlerin ve komşuluğun en iyi konumlarının yinelemelere göre değişimi (Rosenbrock).

Daha önceki bölümlerde de belirtildiği gibi yöntemin eşzamansız, dağıtık ve dinamik komşuluklu biçiminin yöntemin çeşitli uygulamalarında avantajları olmaktadır. Gerçekleştirilen benzetimler sonucunda komşuluk parametrelerinin doğru seçilmesi

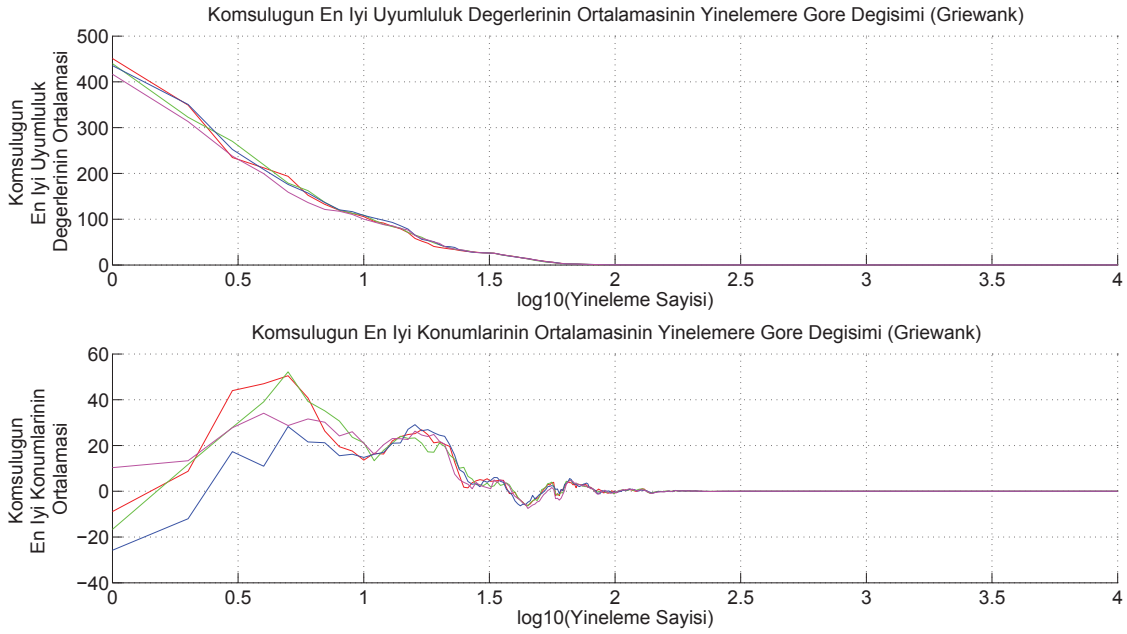
Çizelge 4.4.: Türdeş olmayan ağda gerçekleştirilen benzetimlerde kullanılan işlemciler

No	CPU Hızı	CPU Tipi	Hafıza	Ağ
1	2.4 Ghz	Intel(R) Core(TM)2	1.98 GB	Gigabit Ethernet
2	2.4 Ghz	Intel(R) Core(TM)2	1.98 GB	Gigabit Ethernet
3	3.0 Ghz	Intel(R) Pentium(R)4	1.98 GB	Fast Ethernet
4	3.0 Ghz	Intel(R) Pentium(R)4	1.98 GB	Fast Ethernet

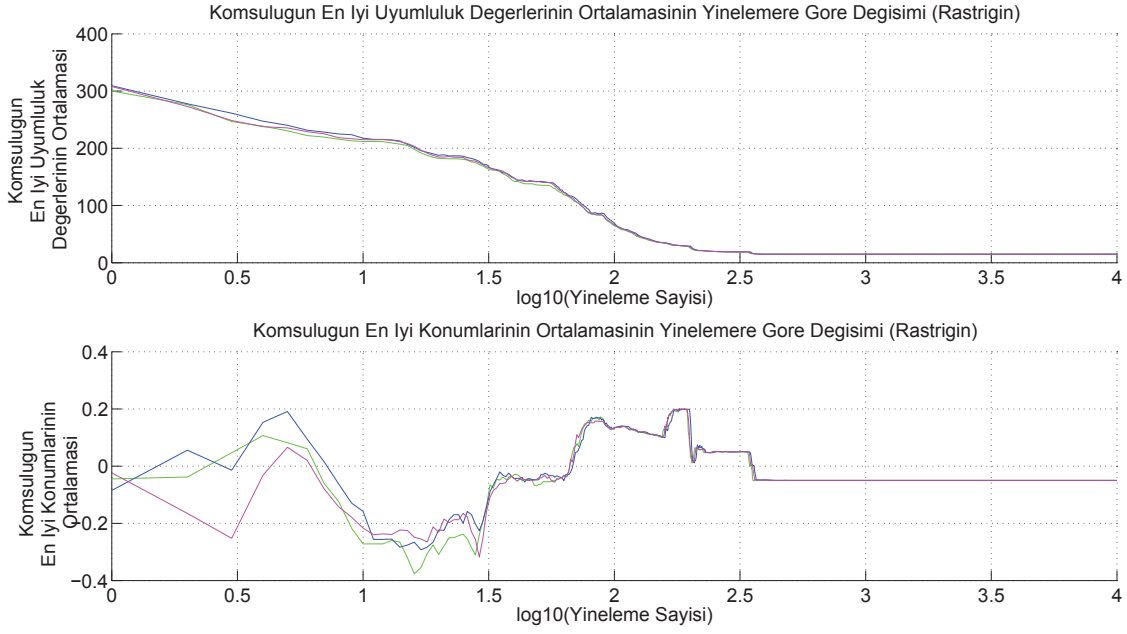


Şekil 4.9.: Türdeş olmayan ağda komşuluğun en iyi uyumluluk değerlerin ve komşuluğun en iyi konumlarının yinelemelere göre değişimi (Küre).

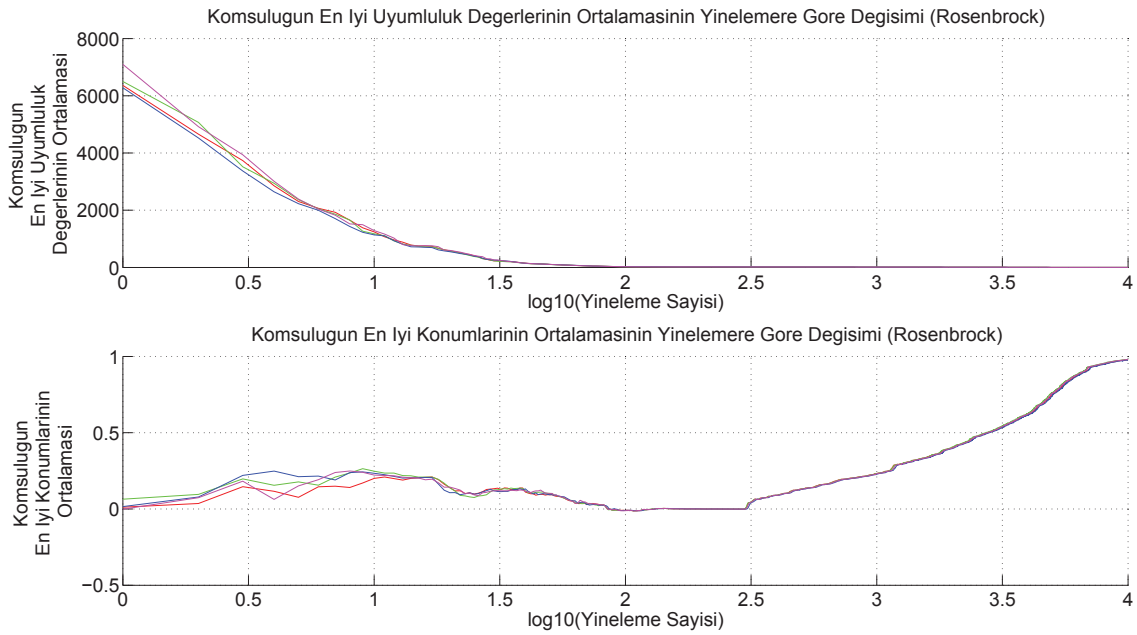
durumunda geliştirilen yöntemin iyi bir düzeyde başarım sergilediği gözlemlenmiştir.



Şekil 4.10.: Türdeş olmayan ağda komşuluğun en iyi uyumluluk değerlerin ve komşuluğun en iyi konumlarının yinelemelere göre değişimi (Griewank).



Şekil 4.11.: Türdeş olmayan ağda komşuluğun en iyi uyumluluk değerlerin ve komşuluğun en iyi konumlarının yinelemelere göre değişimi (Rastrigin).



Şekil 4.12.: Türdeş olmayan ağda komşuluğun en iyi uyumluluk değerlerin ve komşuluğun en iyi konumlarının yinelemelere göre değişimi (Rosenbrock).

BÖLÜM 5

5. EŞZAMANSIZ VE DİNAMİK KOMŞULUKLU PARÇACIK SÜRÜ ENİYİLEME YÖNTEMİNİN ÇOK ROBOTLU ARAMA GÖREVİNDE UYGULANMASI

Bilinmeyen ve insanlar için tehlikeli olabilecek bir ortamda bir veya birden fazla hedefin aranması görevi çok sayıda özerk robot kullanılarak gerçekleştirilebilecek bir görevdir. Robotların (erkinlerin) gerekli algılayıcılar ile donatılması ve etkin gezinim ve işbirlikli arama algoritmalarının geliştirilmesi, arama ortamının daha iyi kaplanmasını ve arama zamanının azaltılmasını sağlar.

Arama görevinde işlem kapasitesi yüksek tek bir robotun veya işlem kapasitesi yüksek bir robotun birincil[1] robot olduğu çok robotlu bir sistemin kullanılması durumları ile karşılaştırıldığında işlem kapasiteleri sınırlı çok sayıda robottan oluşan çok robotlu bir sistemin kullanılmasının birçok avantajları vardır. Bu avantajların en başında sistemin hatalara karşı gürbüzlüğü ve esnekliği gelmektedir [7]. Çok robotlu bir sistemde herhangi bir robotun hatalı bir işlem gerçekleştirmesi, devre dışı kalması veya haberleşme sorunları yaşaması durumunda sistemdeki diğer robotlar istenen işlemleri gerçekleştirebilir ve sistemin arama görevine devam etmesini sağlayabilir. Ayrıca çok robotlu sistemlerin esneklik özelliği, robotların belirli bir zamanda örgütlenerek bir görevi yerine getirebildikleri gibi farklı zaman anlarında farklı şekilde örgütlenerek farklı görevleri yerine getirebilmeleri, arama görevi sırasında gerekirse daha dar bir alanda toplanarak daha titiz ve detaylı bir arama gerçekleştirme veya daha geniş alana yayılarak daha genel arama gerçekleştirme ve birbirleri ile haberleşerek yardımlaşmada bulunarak gerekli olabilecek diğer görevleri yerine getirilmesini sağlar. Bu avantajlarının dışında denetim yöntemlerinin uygun tasarlanması ile sistemin ölçeklenebilirliği (sistemin robot sayısından bağımsız olarak her zaman aynı şekilde çalışması) ve sistemdeki her robotun benzer görevler gerçekleştirmesinden dolayı bu tip sistemler arama görevlerine oldukça uygundur.

Çok robotlu sistemlerin arama görevini hızlı ve etkin bir şekilde gerçekleştirebilmeleri için bu sistemlerin özelliklerine uygun bir arama algoritması belirlenmesi gerekmektedir. Çok robotlu sistemlerin en önemli özelliklerinden olan paralel ve eşzamansız çalışma özelliklerinin seçilen algoritma için uygun olması, algoritmanın

[1] ing: master

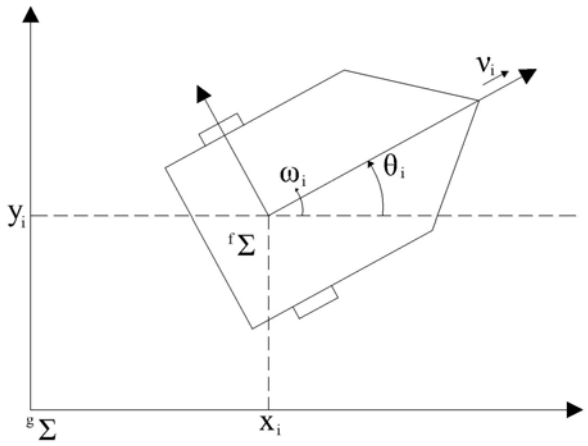
kolayca paralel hale getirilmesi ve eşzamansız olarak gereken işlemleri yerine getirmesi, gerekmektedir. Ayrıca algoritmanın sistemdeki her robot tarafından gerçekleştirilmesi ve arama alanında robotların hareketlerini planlarken sistemdeki bütün robotların bilgisinin kullanılması için dağıtık/merkezi olmayan özelliğe sahip olması gerekmektedir. Öte yandan kullanılacak algoritmanın, sistemdeki robotlar arasındaki etkileşimler zamana bağlı olarak değişmesinden dolayı haberleşen robotların dinamik olarak değişen komşuluk yapısına uygun olması önemli bir hususdur. Son olarak sistemdeki robotların sınırlı işlem kapasitesi ve güç kaynağı olması nedeniyle algoritmanın hesaplama yükünün düşük olması gerekmektedir.

Bölüm 3.'te yapılan benzetimler sonucunda geliştirilen yöntemin dinamik komşuluklu biçiminin parçacık sürü eniyileme yönteminin uygun bir biçimi olduğu ve çalışmanın konusu olan çok erkinli sistemlerin arama görevinde kullanılmasında uygun olacağı belirtilmiştir. Öte yandan Bölüm 4.'te yöntemin paralel olarak çalışan dağıtık sistemler (çok robotlu sistemler gibi) için uygun ve tamamen eşzamansız güncellemelere olanak sağlayan biçiminde işlemlerinin eşzamanlı ve bütünsel bir yinelemede yapılması zorunluluğu ortadan kaldırılarak sistemin olası hatalara (parçacık ve haberleşme hataları) ve parçacıklar arasındaki bilgi değişimindeki zaman gecikmelerine (parçacıklar farklı zaman anlarında haberleşmekte) karşın parçacık sürüsünün arama işlemine devam etmesi sağlanmıştır. Sözü edilen çalışmalardan esinlenilerek bu bölümde bu tezin son amacı olan çok robotlu bir sistemin parçacık sürü eniyileme yönteminin karar verme ve konum güncelleme işlemlerinde kullanılarak bilinmeyen bir ortamda arama görevini gerçekleştirmesi ele alınmıştır. Sistemdeki bir robot sürüdeki bir parçacık olarak düşünülerek robotlar arasındaki bilgi paylaşımında zaman gecikmelerinin de düşünüldüğü, robotların dağıtık ve eşzamansız olarak işlemlerinin gerçekleştirebildiği ve haberleşen robotların komşuluk yapısının zaman ile dinamik olarak değiştiği bir arama yöntemi geliştirilmiştir.

5.1. Problem Tanımı

Bu çalışmada N adet hız kısıtlı[2] erkinden oluşan ve $\mathbb{R}^2$ alanında hareket eden bir robot sürüsü ele alınmıştır. Hız kısıtlı erkinler düzlemdeki bir konumda herhangi bir yönelimde bulunabilir ancak iki ana tekerleğin ortasındaki hız vektörü her zaman yönelime teğettir [97] (bakınız Şekil 5.1.). Sürekli zamanda $\mathbb{R}^2$ alanında hareket eden

[2]ing: non-holonomic



Şekil 5.1. Hız-kısıtlı robot yapısı.

her $i = 1, \ldots, N$ hız kısıtlı robotun dinamikleri

$$\begin{aligned} \dot{x}_i(t) &= \bar{v}_i(t)\cos(\theta_i(t)), \\ \dot{y}_i(t) &= \bar{v}_i(t)\sin(\theta_i(t)), \\ \dot{\theta}_i(t) &= w_i(t). \end{aligned} \tag{5.1}$$

biçiminde gösterilebilir. Denklem 5.1'de $x_i(t)$ and $y_i(t)$ i'inci erkinin kartezyan koordinatları olup, $\theta_i(t)$ i'inci erkinin t zamanındaki yönelim açısı olmaktadır. Denklem 5.1 sadece robotun kinematik kısıtlarına göre elde edilmiş bir denklemdir ve düşük hızlarda geçerlidir. Daha yüksek hızlarda geçerli Denklem 5.1'de bulunan kinematik modele ek dinamikler ilave edilerek elde edilebilir (bakınız [96]). Bu çalışmada bu modelde denetleme girdileri i'inci erkinin doğrusal hızı $\bar{v}_i(t)$ ve açısal hızı $w_i(t)$ olmaktadır.

Robotların bilinmeyen bir ortamda arama yapmaları istenmiştir. Ortamdaki her noktanın deneysel olarak elde edilen gerçekçi etanol gazı yoğunluğuna göre bir potansiyel değeri olduğu kabul edilmiştir. Öte yandan ele alınan probleme göre başka uygulamalarda potansiyel değerler noktalardaki koku, kimyasal parçacık veya duman yoğunluğunu temsil edebilir. Bu potansiyel değerler kaynak fonksiyonu/alanı[3] olarak adlandırılmıştır ve PSO tabanlı eniyileme yöntemi ile robotların alanda izleyecekleri yollar planlanarak, kaynak fonksiyonun buradaki uygulamada en büyük değere sahip bölgesinin bulunması istenmektedir. Ele alınan probleme göre aranan en küçük veya en büyük değerli bölge koku, duman, ısı veya ışık kaynağının yerini temsil

[3]ing: resource profile

edebilir. Robotların arama ortamındaki noktaların potansiyel değerlerini belirlemeleri için gerekli algılayıcılar ile donatıldıkları kabul edilmiştir. Yukarıda da belirtildiği gibi bu çalışmada kaynak fonksiyonu/alanı etanol gazı yoğunluğunu temsil etmektedir ve amaç robotların gaz yoğunluğunun en yoğun olduğu bölgeyi bulmalarıdır.

PSO algoritması tabanlı arama yönteminde sistemdeki i'inci robot PSO açısında parçacık sürüsündeki i'inci parçacık olarak düşünülebilir. Robotun t_k zamanındaki (robotun/parçacığın k'ıncı yinelemesi) k'ıncı ara noktası (bulunduğu konum) $p_i(t_k) = [x_i(t_k), y_i(t_k)] \in \mathbb{R}^2$ olsun; bu durumda PSO algoritması bir sonraki ara noktanın $p_i(t_k + 1)$ bulunması için kullanılmaktadır. Bir başka deyişle PSO algoritması üst seviyede robotların gidecekleri yolların planlanmasını, robotun arama boyunca gitmesi gereken ara noktaların belirlenmesi, için kullanılmaktadır. Robotun k'ıncı ara nokta olan $p_i(t_k)$'dan $k + 1$'inci ara nokta olan $p_i(t_k + 1)$'e gitmesi için yapay potansiyel fonksiyonlar alt seviye denetleme algoritması olarak kullanılmıştır. Yapay potansiyel fonksiyonlar kullanılarak robotlara etkin gezinim sağlandığı birçok çalışma bulunmaktadır [98–100]. Robotun takip etmesi istenen yol, iki ara noktayı birleştiren bir doğru olarak tanımlanmış ve robotun $\bar{p}(t_k) = p_i(t_k) - p_i(t_{k+1})$ vektörü doğrultusunda engellerden sakınarak $p_i(t_{k+1})$ ara noktasına doğru hareket ettirilmiştir. Bu amaçla karesel bir çekim potansiyel fonksiyonu tanımlanmış ve robotun fonksiyonun eğiminin azaldığı yöne doğru hareket etmesi sağlanmıştır. Robotların çarpışmalarını engellemek için robotlar arasında bir itim potansiyel fonksiyonu kullanılmış ve bu fonksiyon iki robot arasındaki mesafenin d gibi önceden belirlenmiş bir mesafeden daha az olduğu durumlarda göz önünde bulundurulmuştur. $G_{ai}(t)$, $t \in [t_k, t_{k+1})$ zamanında çekim potansiyel fonksiyonunun gradyanını, $G_{rij}(t)$ ise i robotunun kendi yakınlığında bulunan j robotu ile arasındaki itim potansiyel fonksiyonunun gradyanını göstersin. Bu çalışmada potansiyel alan olan itim ve çekim potansiyel fonksiyonlarının gradyanı şekilde tanımlanmıştır

$$G_{ai}(t) = -a\bar{p}_i(t),$$

ve

$$G_{rij}(t) = \begin{cases} b\bar{p}_{ij}(t)\left(\frac{1}{\|\bar{p}_{ij}(t)\|^2} - \frac{d}{\|\bar{p}_{ij}(t)\|^3}\right), & \|\bar{p}_{ij}(t)\| \leq d \\ 0, & \|\bar{p}_i j(t)\| > d. \end{cases}$$

Burada $\bar{p}_i(t) = \big(p_i(t) - p_i(t_{k+1})\big)$ ve $\bar{p}_{ij}(t) = \big(p_i(t) - p_j(t)\big)$ olmaktadır. Robotun $t \in [t_k, t_{k+1})$ zaman aralığındaki konumu $p_i(t)$, robotun arama alanındaki bir sonraki ara noktası $p_i(t_k + 1)$ olarak gösterilmiştir. Sabit $a > 0$ katsayısı çekim katsayısı,

$b > 0$ katsayısı ise itim katsayısı olarak alınmıştır. Bu tasarımda robotun gideceği ara noktalar arasında doğrusal bir çekim kuvveti varken, i'inci robotun yakınlığında d mesafesinden daha yakın bir mesafede bulunan j robotunun olduğu durumlarda etkin olan doğrusal olmayan ve sınırsız bir itim kuvveti vardır. Belirtilen potansiyel kuvvetler hesaplanırken robotların yerel koordinat sistemleri dikkate alınmıştır.

Yukarıdaki belirtilen itim ve çekim potansiyel kuvvetler kullanılarak, toplam potansiyel kuvvet (potansiyel alan) $G_i(t)$'nin değeri

$$G_i(t) = G_{ai}(t) + \sum_{j=1}^{N} G_{rij}(t) \tag{5.2}$$

gibi hesaplanabilir. Denklem 5.2'de i'inci robot üzerindeki itim potansiyel kuvvetinin sistemdeki bütün robotlar tarafından oluştuğu gösterilmesine karşın, $G_i(t)$ toplam potansiyel kuvveti hesaplanırken sadece i'inci robotun yakınlığındaki robotlar dikkate alınmaktadır.

Sistemdeki robotların hız kısıtlı olması nedeniyle ve o andaki yönlerinin potansiyel vektör yönünde olmaması olasılığına karşın robotların istenen hareket yönü

$$\theta_{id}(t) = \mathrm{atan2}\Big(G_{yi}(t), G_{xi}(t)\Big), t \in [t_k, t_{k+1}),$$

olarak tanımlanmıştır. Burada $G_{xi}(t)$ ve $G_{yi}(t)$ sırasıyla potansiyel alanın x ve y bileşenleri olarak tanımlanmıştır

Robotun üzerindeki potansiyel kuvvetler yerel koordinatlar göz önüne alınarak hesaplanmasına karşın, robotun bütünsel koordinatlardaki çekim açısı göz önünde bulundurulmuştur. Bu durumda i'inci robotun bütünsel koordinatlara göre arama alanındaki gitmesi gereken yönelim açısı

$$\theta_{ida}(t) = \mathrm{atan2}\Big(G_{aiy}(t), G_{aix}(t)\Big), t \in [t_k, t_{k+1}),$$

olarak hesaplanır. Burada $G_{axi}(t)$ ve $G_{ayi}(t)$ sırasıyla çekim potansiyel alanın x ve y bileşenleri olarak tanımlanmıştır. Robotun yerel koordinatlarına göre arama alanındaki belirli bir konuma olan çekim açısı

$$\theta_{rel}(t) = mod\Big[\Big((\theta_{ida}(t) - \theta_i(t)) + \pi\Big), 2\pi\Big] - \pi, t \in [t_k, t_{k+1}),$$

olarak hesaplanabilir. Burada $\theta_i(t)$ i'inci robotun t anındaki yönelimini, $\theta_{ida}(t)$ ise

robotun t anında bütünsel koordinatlarda arama alanındaki belirli bir konuma olan çekim açısını göstermektedir. Bu çalışmada ele alınan açıların $(-\pi, \pi]$ aralığında değer aldıkları kabul edilmiştir. Bu nedenden dolayı açılar arasındaki toplama veya çıkarma işlemleri mod 2π olarak ve $-\pi$ radyan kayması düşünülerek gerçekleştirilmiştir. Bu işlem ayrıca iki vektör arasındaki küçük açıyı vermektedir. Gerektiği durumlarda robotun üzerindeki itim kuvveti hesaplanırken robotun üzerindeki algılayıcı değerleri dikkate alınmıştır ve kuvvetin x ve y bileşenleri hesaplanırken algılayıcıların robottaki konumlarına göre açıları belirlenmiştir. Bu açılar robotun yerel koordinatlarında belirlendiği için herhangi bir çevrim işlemi yapılmamaktadır.

Robotun yön dinamiklerinin denetlenmesi için

$$w_i(t) = -\alpha\Big(mod(mod((\theta_i(t) - \theta_{id}(t)), 2\pi) + \pi, 2\pi) - \pi\Big), t \in [t_k, t_{k+1}), \quad (5.3)$$

gibi basit bir oransal denetleyici tasarlanmıştır. Burada $\theta_i(t)$ t zamanında robotun yönü ve $\alpha > 0$ oransal denetleyicinin kazanç katsayısı olarak alınmıştır. Daha önceden de belirtildiği gibi iki vektör arasındaki küçük açıyı belirlemek için toplama veya çıkarma işlemleri mod 2π olarak ve $-\pi$ radyan kayması düşünülerek gerçekleştirilmiştir. Doğrusal hızın denetlenmesi için ise

$$\bar{v}_i(t) = \min\{\|G_i(t)\|, v_{max}\}, t \in [t_k, t_{k+1}), \quad (5.4)$$

şeklinde bir denetleyici kullanılmıştır. Burada v_{max} robotun doğrusal hızının üst sınırı olarak belirlenmiştir. Bunlara ek olarak robot bir sonraki ara noktası $p_i(t_k+1)$'in küçük bir komşuluğuna geldiğinde bu ara noktaya ulaştığı varsayılmış ve PSO algoritmasının bir sonraki yinelemesine geçilmiştir.

5.2. Eşzamansız Parçacık Sürü Eniyileme Yöntemi Tabanlı Arama Algoritması

Önceki bölümlerde de belirtildiği gibi çok robotlu sistemlerin arama görevini hızlı ve etkin bir biçimde gerçekleştirmeleri için çalışma özelliklerine uygun olan bir arama algoritması belirlenmesi gerekmektedir. PSO algoritmasının çok robotlu sistemlerin bazı çalışma özelliklerine uygun olmasına karşın algoritmanın herhangi bir değişiklik yapılmadan çok robotlu sistemlerin arama görevinde kullanılması sistemin verimsiz çalışmasına neden olabilir. Bunun nedeni robotların arama alanında gidecekleri ara noktalara hemen ulaşamamaları ve robotların ara noktalara farklı zamanlarda ulaşmalarıdır. PSO algoritmasının temel biçiminde (bakınız Çizelge 2.1.) komşuluğunun en iyi uyumluluk değeri $evrensel_{eniyi}$ değerinin güncellenmesi,

robotlar arasındaki bilgi paylaşımının gerçekleşmesi ve parçacık sürü eniyileme yöntemi kullanılarak robotların bir sonraki ara noktalarının belirlenmesi için ara noktalarına erken ulaşan robotlar diğer robotların ara noktalarına ulaşmalarını beklemek zorundadırlar. Ayrıca arama alanının büyük olduğu durumlarda, robotların gidecekleri ara noktalar arasındaki mesafeler robotların algılama/haberleşme alanlarından daha büyük olabilir ve sistemdeki bazı robotlar diğer robotlar ile haberleşememesi olası bir durumdur. Bu durumda robotlar sonraki ara noktalarını belirlemek amacıyla diğer robotlardan bilgi alamaz ve sistem durabilir. Geçici veya kalıcı haberleşme sorunları ve sistemdeki erkinlerin hata yapma veya devre dışı kalma olasılıkları yukarıda belirtilen durumun daha da kötüleşmesine neden olur. Öte yandan sistemdeki robotlar işlemlerini farklı zaman anlarında gerçekleştirdikleri için birbirleri ile paylaştıkları bilgilerde zaman gecikmeleri mevcuttur. Bunun anlamı sistemdeki robotlar diğer robotların o zaman anına kadar elde ettikleri en güncel bilgiyi kullanamayabilir. Yukarıda belirtilen sorunların aşılabilmesi için bu çalışmada parçacık sürü eniyileme algoritmasının çok robotlu sistemlerin arama görevi için uygun bir biçimi Çizelge 5.1.'de verilmiştir.

Çizelge 5.1.: Eşzamansız parçacık sürü eniyileme yöntemi tabanlı arama algoritmasının sahte kodu

$kisisel_{eniyi}$ ve $evrensel_{eniyi}$ (ve diğer değişkenlerin) ilklendir
İlk ara noktaların Denklem 2.3 kullanılarak hesapla
while (Hedef bulunmadıysa veya azami yineleme sayısına ulaşılmamışsa) **do**
  while (Erkin ara noktasına ulaşmamışsa) **do**
    Erkini istenen ara noktaya doğru ilerlet
    $kisisel_{eniyi}$ değerini güncelle
    if (Diğer erkinlerden bilgi elde edilmişse) **then**
      $diger - kisisel_{eniyi}$ değerinin güncelle
    end if
  end while
  Bulunan $kisisel_{eniyi}$ değerini diğer erkinler ile paylaş
  if ($diger - kisisel_{eniyi} > evrensel_{eniyi}$ veya $kisisel_{eniyi} > evrensel_{eniyi}$) **then**
    $evrensel_{eniyi}$ değerinin güncelle
  else
    Önceki $evrensel_{eniyi}$ değerinin kullan
  end if
  Yeni ara noktayı Denklem 2.3 kullanılarak hesapla
end while

Çizelge 5.1.'de bulunan sözde koddan da görüldüğü üzere, robot arama alanında arama noktasına doğru ilerlerken, diğer robotlardan bilgi alabilmek için (diğer

robotlar ara noktalarına erken ulaştıkları zaman, kendi en iyi uyumluluk değerlerini paylaşmaktadırlar) diğer robotları sürekli dinlemekte ve bu bilgiyi daha sonra $evrensel_{eniyi}$ değerinin güncellenmesi için kullanmaktadır. Robot kendi ara noktasına ulaştıktan sonra kendi en iyi uyumluluk değeri olan $kisisel_{eniyi}$ değerini diğer robotlar ile paylaşmaktadır. Daha sonra diğer robotlardan elde ettiği değerleri, kendi hız vektörünü, kendi en iyi konumunu ve komşuluğunun en iyi konumunu (diğer robotlardan elde edilen bilgiler kullanılarak bulunmaktadır) kullanarak arama alanında bir sonraki gitmesi gereken ara noktayı belirlemektedir (bakınız Denklem 2.3). Eğer robot kendi ara noktasına ulaştığı anda sistemdeki diğer robotlar ara noktalarına ulaşmamışsa bir başka deyişle robot sistemdeki diğer robotlardan bilgi almamışsa, bir sonraki ara nokta belirlenirken robot sadece kendi bilgisini ve daha önceki adımlarda haberleştiği robotların bilgilerini kullanmaktadır. Başka robotlardan bilgi alındığı durumda ise bu bilgi de $evrensel_{eniyi}$ değerinin güncellenmesinde kullanılmaktadır. Sürüdeki robotlar arama alanında ara noktalarına farklı zamanlarda ulaştıkları için robotun haberleşebildiği ve bilgi alabildiği robotlar zaman ile değişmektedir.

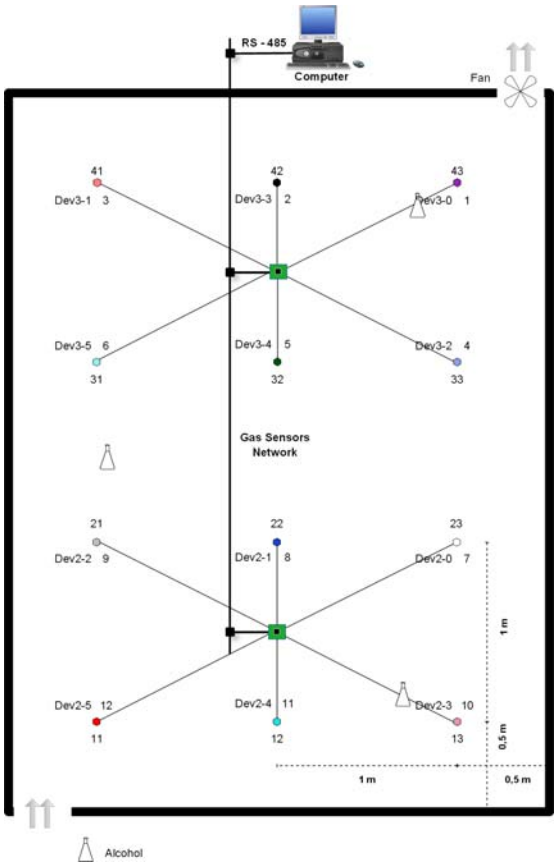
Sistemdeki robotlar birbirlerini beklemek zorunda kalmayıp, eşzamansız ve özerk olarak işlemlerini gerçekleştirerek arama görevini daha hızlı ve etkin bir biçimde gerçekleştirebilecekleri düşünülmüştür. Öte yandan sistemdeki herhangi bir robotun devre dışı kalması durumunda sistemdeki diğer robotlar işlemlerini özerk ve eşzamansız biçimde gerçekleştirdikleri için sistem arama görevine devam edebileceği öngörülmektedir. Ayrıca haberleşen robotların komşuluk yapısının dinamik olarak değişmesinden ötürü sistemdeki bütün robotların her zaman anında haberleşerek hareketlerini planlama ihtiyacı ortadan kaldırılarak, sistem geçici veya kalıcı haberleşme hatalarına karşı daha gürbüz hale geldiğini inanılmaktadır.

5.3. Benzetim ve Uygulama

Benzetimde ve uygulamada kullanılan gerçekçi etanol gazı yoğunluğu, herhangi bir engelin bulunmadığı kapalı bir ortamda elde edilmiştir. Veriler 3 metre boyunda, 4 metre eninde ve 0.5 metre yüksekliğe sahip kapalı bir alanda elde edilmiştir. Ortam sol alt köşede bulunan açıklık ve sağ üst köşede bulunan yapraklarının çapı 12 cm olan bir fan ile zayıf bir şekilde havalandırılmıştır. Fan 0 ile 1500 lpm[4] arasındaki değerlerde hava akışı sağlamaktadır (bakınız Şekil 5.2.). Ortamda konumları G1 (2.25, 0.625), G2 (0.5, 2) ve G3 (2.25, 3.45) olan tavana asılı üç adet gaz kaynağı bulunmaktadır. Gaz kaynaklarından denetlenebilir bir hava akışı ile ortama gaz salınımı yapılmaktadır.

[4]ing: liter per minute

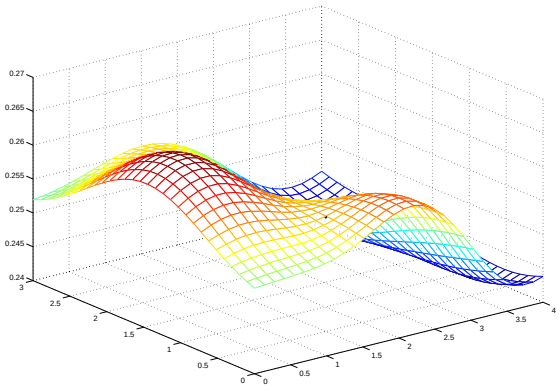
Kaynaklardaki hava akışı SMC oransal vanaları ile denetlenmektedir. Ortamdaki etanol gazı yoğunluğu 12 adet Figaro TGS2600 gaz algılayıcısından oluşan bir algılayıcı ağından elde edilmektedir. Bu gaz algılayıcıları milyonda bir etanol parçacığı bulunan etanol buharını algılayabilmektedir. Her gaz algılayıcısı, gerekli işaret iyileştirme işlemleri yapılmış bir baskı devre plaketi üzerine yerleştirilmiştir. Algılayıcıların çıkışı algılayıcı ağında bulunan iki adet PIC18F4431 mikroişlemcisi tarafından toplanmakta ve RS485 veri yolu ile bir bilgisayara aktarılmaktadır (bakınız Şekil 5.2.). Ortamdaki gaz yoğunluğunun sürekli dağılımı Kriging kestirim yöntemi kullanılarak elde edilmiştir. (Şekil 5.2.'de gösterilen deneysel düzenek kurulması ve yukarıda belirtilen deneysel olarak gerçekci etanol gazı yoğunluğu bilgisinin elde edilmesi Dr. Lino Marques'in katkılarıyla Coimbra Üniversitesi Gömülü Sistemler Laboratuvarında gerçekleştirilmiştir.)



Şekil 5.2. Deneysel düzenek.

Elde edilen deneysel veriler kullanılarak, geliştirilen arama yöntemi ile arama alanında gaz yoğunluğunun en yüksek olduğu bölgenin (hedef) bulunması amaçlanmaktadır. Şekil 5.3.'te çalışmada kullanılan ve deneysel olarak elde edilen gaz yoğunluğunun

sürekli dağılımı görülmektedir. Bu kaynak fonksiyonu sözü edilen kapalı ortamdan belirli bir zaman anında elde edilen anlık etanol gazı yoğunluğunu göstermektedir.



Şekil 5.3. Kaynak fonksiyonu.

Arama işleminin başında robotlar arama alanının girişine yakın, kartezyan düzlemde (0,0) noktasına yakın, bir bölgede yer almaktadırlar. Robotların arama alanında gidecekleri ilk ara noktalar rasgele belirlenmekte ve her robot kendi ara noktasına doğru ilerlemektedir. Bu sırada robotlar mesafe sayaçlarındaki[5] değerleri kullanarak konumlarını belirlemekte, böylece robotların arama boyunca herhangi bir bütünsel konum bilgisi olmadan kendi konumları belirlemesi sağlanmıştır.

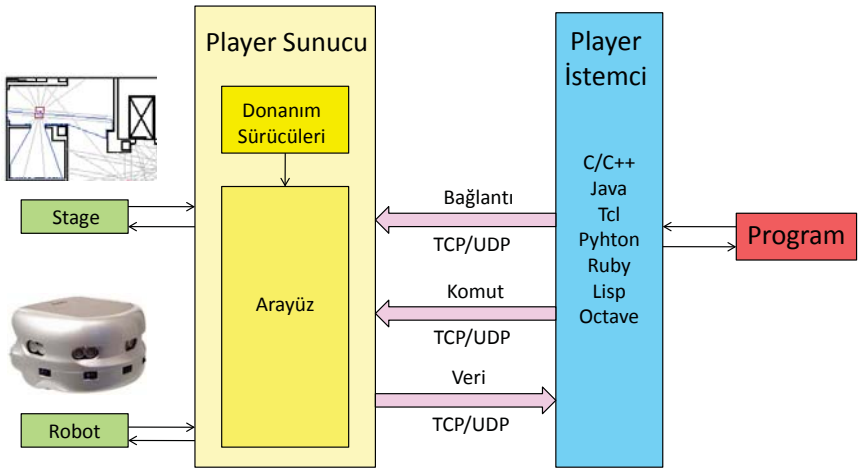
Öte yandan yukarıdaki konumlama arama alanının tek girişi olduğu gerçekçi durumlar için daha uygundur. Robotlar için arama işleminin sonlandığına dair durma koşulu Denklem 2.3'te hesaplanan $v_i(t_{k+1})$ hız vektörünün önceden tanımlanan belirli bir eşik değerinden daha küçük değerlere ulaşması olarak belirlenmiştir. Böylece Denklem 2.3'e göre sonraki adımlarda robotun o andaki ara noktasına, $p_i(t_k)$, yakın ara noktalara, $p_i(t_{k+1})$, gitmesi ve sonunda durması olarak belirlenmiştir. Parçacık sürü eniyileme yönteminin kısıtlama katsayılı biçimi ele alınmış, kısıtlama katsayısı $\chi > 0$, Denklem 2.4 göz önüne alınarak ve $\bar{\varphi}_1 = \bar{\varphi}_2 = 2.05$, $\varphi = 4.1$, $\kappa = 1$ alınarak $\chi = 0.7298$ olarak hesaplanmıştır.

5.3.1. Benzetim Sonuçları

Eşzamansız Parçacık Sürü Eniyileme Yöntemi tabanlı arama yöntemi ilk önce Player/Stage gerçekçi benzetim yazılımında test edilmiştir. Player/Stage gerçekci

[5]ing: odometry

benzetim yazılımı TCP/IP tabanlı bir yazılımdır [101, 102]. Yazılımın Player kısmı soket tabanlı bir sunucu[6] görevi görmekte ve bir IP ağ üzerinden robotun algılayıcı ve eyleyicilerine ara yüz sağlamaktadır. İstemci[7] programlar TCP soketleri üzerinden yazılımın Player kısmına bağlanmakta ve çeşitli komutlar yollayarak çeşitli algılayıcıların ve eyleyicilerin değerlerini elde etmektedir. Player gerçek robotlarda kullanılan birçok donanımı destekleyecek sürücülere sahiptir. Player kısmının dış ara yüzü TCP soketleri olduğu için istemci programlar herhangi bir programlama dilinde yazılabilir. Yazılımın Stage kısmı geliştirilen algoritmaların test edilmesi için benzetim ortamını sağlamaktadır. Benzetim ortamında robotların iki boyutlu bir alanda hareket etmesi ve yazılımda modellenen çeşitli donanımların kullanılması sağlanmıştır. Öte yandan Stage ortamında çalışan programlar Player tarafından sağlanan donanım ara yüzlerini gerçek donanımlar gibi tanıyıp, gerçek donanımlar ile çalıştıkları düşünülmektedir. Bu nedenden ötürü geliştiren algoritmalar herhangi bir değişiklik yapılmadan sadece yazılımın Player kısmında gerekli donanımların sürücüleri belirtilerek yazılımın desteklediği robotlar üzerinde uygulanabilir (bakınız Şekil 5.4.).

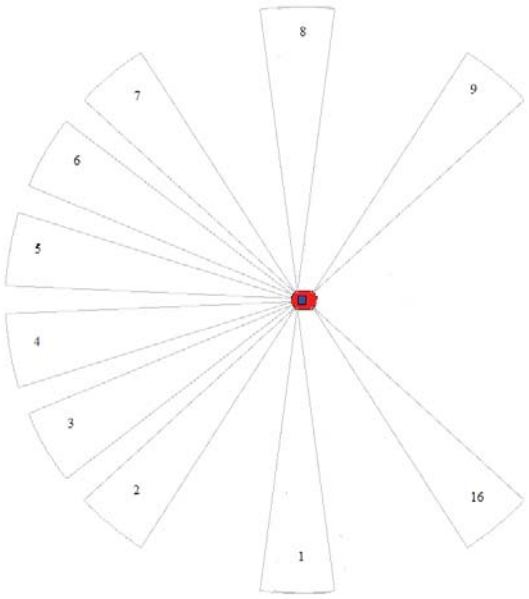


Şekil 5.4. Player/Stage yazılımının blok gösterimi.

Benzetimler için 6 adet Pioneer2dx tipi robot kullanılmıştır.[8] Robotların bütünsel konumlarını bilinmediği ve kendilerini konumlamaları için arama görevi sırasında sadece mesafe sayaçları kullanıldığı kabul edilmiştir. Ancak bütün robotların ilk

[6]ing: server
[7]ing: client
[8]Player/Stage gerçekçi benzetim yazılımında Pioneer2dx robotları modellenmiştir (robot için sürücüler mevcuttur).

Algılayıcı No	Açı (Derece)
1	90
2	50
3	30
4	10
5	-10
6	-30
7	-50
8	-90
9	-130
16	130



Şekil 5.5. İtim potansiyel fonksiyonunda kullanılan ses ötesi algılayıcılar ve açıları.

konumları tam olarak alanın giriş noktasında bir başka deyişle kartezyan düzlemin (0,0) noktasında olmaması ve bazı robotların ilk konumlarının bu noktaya yakın noktalar olması nedeniyle bazı robotların konumlamalarında belirli hatalar olmaktadır. Robotun ön, arka, sağ ve sol taraflarında olmak üzere 16 adet ses ötesi algılayıcı bulunmaktadır. Robotların aralarındaki mesafeleri belirlemeleri için ön, sağ ve sol taraflarındaki ses ötesi algılayıcılar kullanılmış, iki veya daha fazla robotun arasındaki mesafe önceden belirlenen d mesafesinden daha az olduğu durumda itim potansiyel fonksiyonun hesaplanması için bu algılayıcıların okuduğu değerler ve algılayıcıların bulundukları konumlara göre açı değerleri dikkate alınmıştır (bakınız Şekil 5.5.). Ayrıca robotların ortamdaki her noktanın potansiyel değerini elde etmeleri için arama alanı 30x30 adet eşit boyutlardaki kareye bölünmüş ve Şekil 5.3.'teki kaynak fonksiyonu göz önünde bulundurularak bir potansiyeli değer atanmıştır.

Robotların arama boyunca gittikleri ara noktalar ve arama alanında izledikleri gezingeler Şekil 5.6.'da görülmektedir. Burada "X" her robotun rasgele seçilen ilk konumlarını, "O" ise her robotun son konumunu belirtmektedir ve eğriler her robotun arama sırasındaki gezingesini göstermektedir. Robotların gittikleri ara noktalar gaz yoğunluğu kaynak fonksiyonu kontür haritası üzerindeki noktalar ile belirtilmiştir. Robotların fiziksel boyutları, mesafe sayaçlarındaki hatalar ve aralarındaki itim kuvvetleri (robotların arama alanında birbirlerine çarpmamaları için)

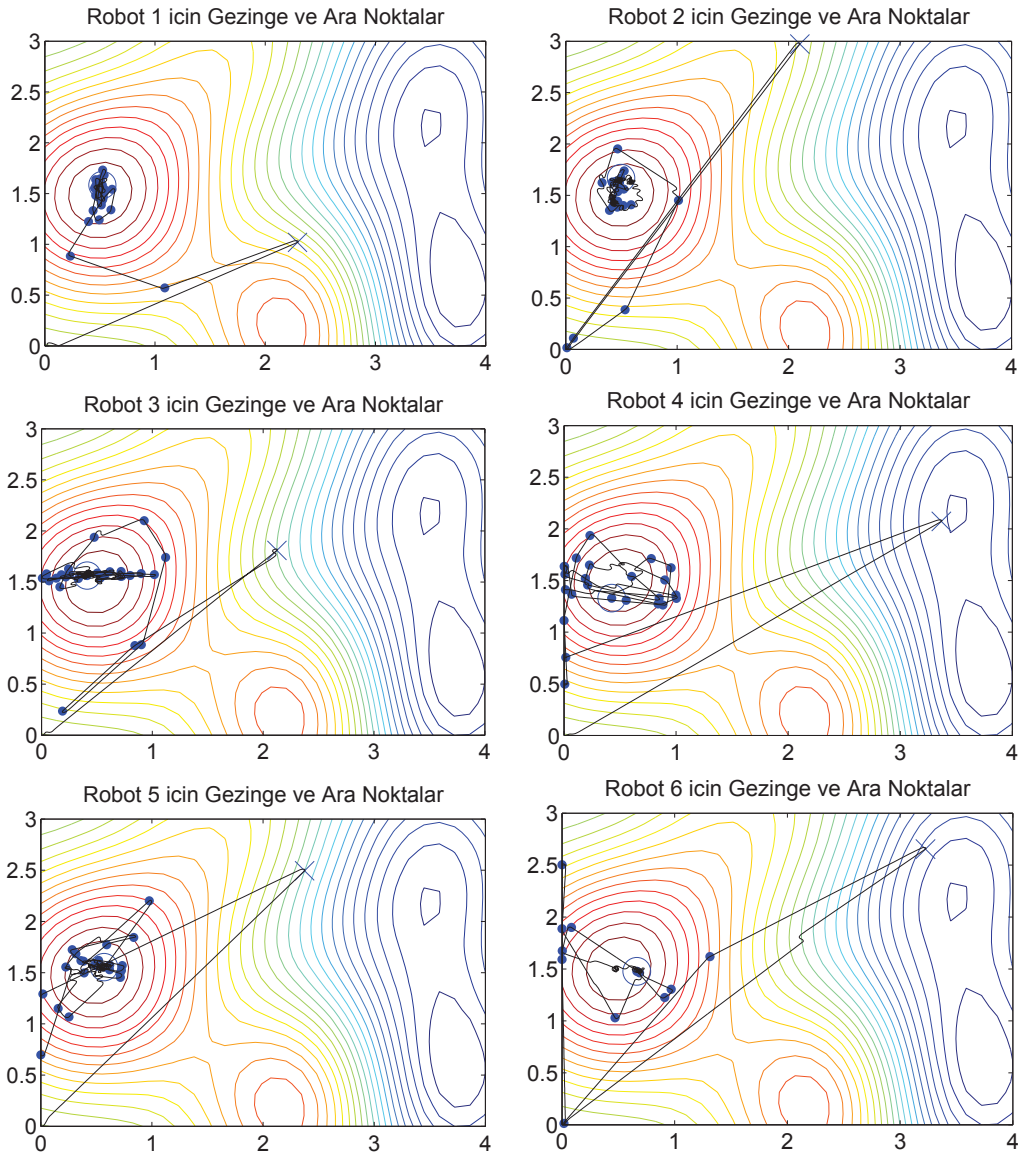
nedeniyle her robotun aynı noktaya yakınsaması beklenmemektedir. Bunun yerine gaz yoğunluğunun en yüksek olduğu bölgenin yakınında toplanmaları ve bu bölgeye yakın konumlara gitmeleri beklenmektedir. Öte yandan robotların hız ve konum vektörleri güncellemelerini eşzamansız olarak gerçekleştirmelerinden dolayı her robotun gaz yoğunluğunun en yüksek olduğu bölgeyi farklı yineleme sayısında bulacaktır. Şekil 5.7.'de robotların her yinelemede gaz yoğunluğunun en yüksek olduğu bölgeye olan ortalama uzaklığı görülmektedir. Burada dikey kesikli çizgiler robotların hedefe olan uzaklığının standart sapmasını belirtmektedir. Gaz yoğunluğunun en yüksek olduğu bölgeye olan ortalama uzaklık ve standart sapma robotların PSO yinelemelerini gerçekleştirmesi ile azalmaktadır. Öte yandan robotlar arama işlemini eşzamansız gerçekleştirmelerinden dolayı arama işlemini erken bitiren robotların daha sonraki yinelemelerde aynı konumlarda kaldıkları kabul edilmiştir. Sistemde istenen bölgeyi en geç bulan robotun 30 yineleme gerçekleştirdiği belirlenmiş, böylece yineleme ekseni, yatay eksen, 30 yinelemeye kadar alınmıştır. Şekil 5.7.'den de görülebileceği üzere yaklaşık 20 yineleme sonunda robotların hedefe olan ortalama uzaklıkları ve standart sapmalarının yakınsadığı gözlemlenmiştir.

5.3.2. Uygulama Sonuçları

Geliştirilen yöntemin robotlar ile uygulaması 3.40mx2.40m boyutlarında engellerin olmadığı bir alanda yapılmıştır (bakınız Şekil 5.8.). Robotların ortamdaki etanol gazı yoğunluğunu ölçebilecekleri/belirleyebilecekleri herhangi bir algılayıcı ile donatılmadıkları için arama alanı 12x12 adet eşit boyutlardaki dikdörtgene bölünmüş ve her dikdörtgenin Şekil 5.3.'teki kaynak fonksiyonu göz önünde bulundurularak bir potansiyeli değeri olduğu kabul edilmiştir.

Yöntemin uygulaması 3 adet Khepera III tipi robotlar ile yapılmıştır. Robotlarda işlemci olarak 400Mhz de çalışan Intel PXA255 işlemcisi bulunmaktadır. Robotların hareketi 2 adet fırçasız DC servo motor ile sağlanmaktadır. Motorlar iki adet PIC18F4432 mikroişlemcisine gömülü PID denetleyicisi ile sürülmektedir ve bu mikroişlemciler motorların mesafe sayaçlarının değerlerinin elde edilmesi için de kullanılmaktadır. Denetleyiciler motorların hız veya konumu denetimini gerçek hız ve sayaçlardan okunan konum bilgisi kullanarak doğru darbe genişlik kiplenimi[9] değerini ayarlayarak yapmaktadırlar. Motor denetim blokları i2c veri yolunda çırak donanım olmakta ve usta işlemci ile i2c veri yolu ile haberleşmektedir. Robotların ön ve yan taraflarında 9 adet, alt tarafında 2 adet olmak üzere toplam 11 adet

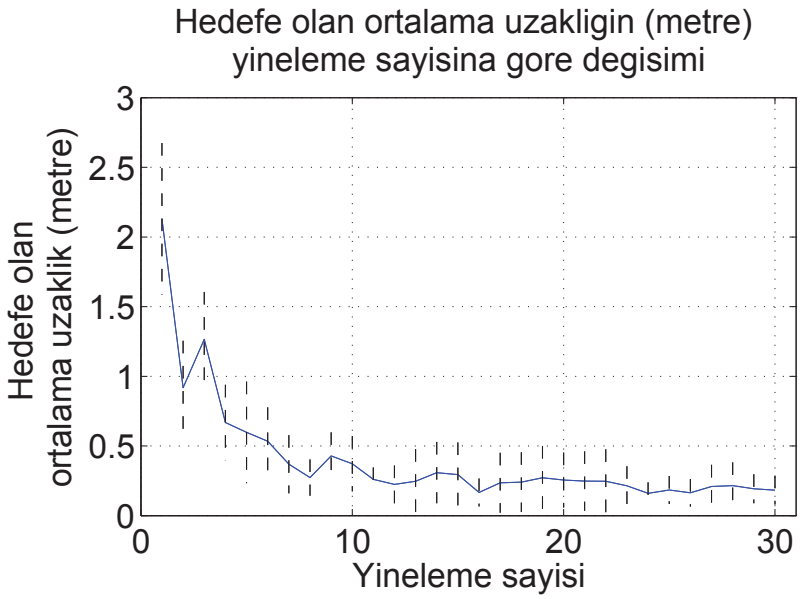
[9]ing: pulse width modulation



Şekil 5.6. Robotların ara noktaları ve gezingeleri.

Vishay Telefunken TCRT5000 kızıl ötesi ve ön ve yan taraflarında 5 adet ses ötesi algılayıcı (verici Midas 400ST100, alıcı Midas 400SR100) bulunmaktadır. Kızıl ötesi ve ses ötesi algılayıcılarda alıcı ve verici modülleri bir arada bulunmaktadır. Algılayıcı okumalarının/ölçümlerinin elde edilmesi 60Mhz'de çalışan DSPIC30F5011 mikroişlemcisi kullanılmaktadır. Algılayıcı değerlerinin elde edildiği DSPIC30F5011 ve motor değerlerinin elde edildiği iki adet PIC18F4432 mikroişlemcileri yamak işlemciler olarak i2c veri yolu usta işlemci olan INTEL PXA255 işlemcisi ile haberleşmektedir [103].

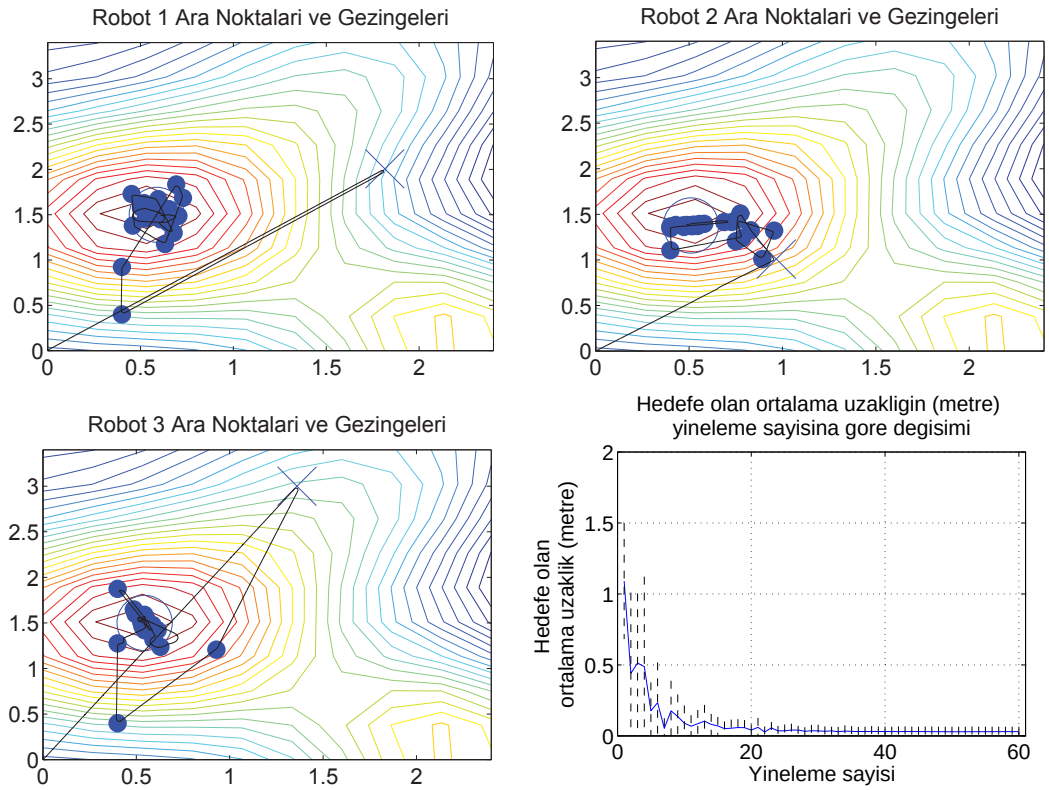
Robotların uygulama sırasındaki dört farklı zaman anındaki, sırasıyla başlangıç anı



Şekil 5.7. Hedefe olan ortalama uzaklık.

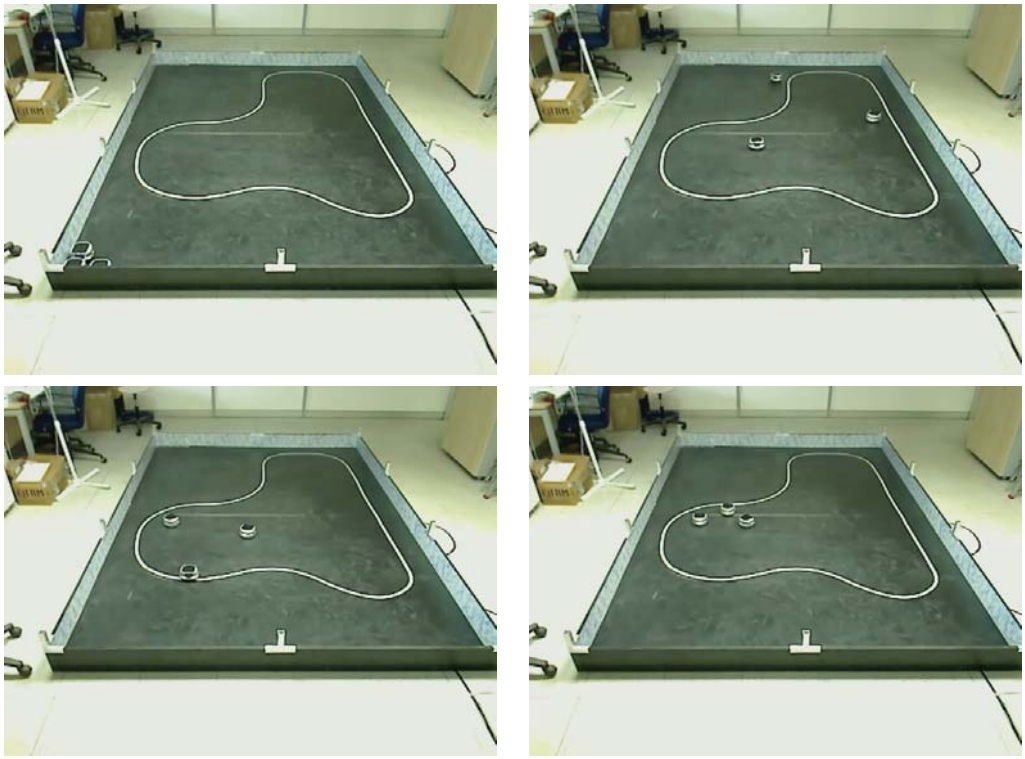
Şekil 5.8. Uygulamada kullanılan alan.

(717'inci kare, 1283'üncü kare ve bitiş anı) konumları Şekil 5.10.'da görülmektedir. Benzetimlerde olduğu gibi uygulamada da arama işleminin başında robotlar arama alanın girişine yakın (kartezyan düzlemde (0,0) noktasına yakın) bir bölgede yer almakta ve arama boyunca konumlarını belirlemek için mesafe sayaçlarının değerlerini kullanmaktadırlar. Robotların ön ve yan taraflarında bulunan kızıl ötesi algılayıcı değerleri robotlar arasındaki mesafenin d mesafesinden ($d = 5cm$) küçük olduğu



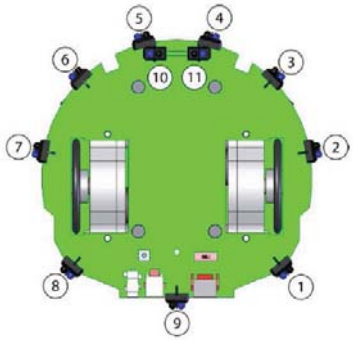
Şekil 5.9.: Khepera III robotların gezingeleri, ara noktaları ve hedefe olan ortalama uzaklık.

durumlarda ele alınan itim potansiyel fonksiyonun hesaplanmasında kullanılmıştır (bakınız Şekil 5.11.). Şekil 5.9. robotların arama alanındaki gezingeleri ve ara noktaları kaynak profilinin kontür haritası üzerinde göstermektedir. Benzetim sonuçlarında olduğu gibi uygulamada da robotların mesafe sayaçların biriken hatalara rağmen gaz yoğunluğunun yüksek olduğu bölgelere yaklaşarak arama yaptıkları, yoğunluğun en yüksek olduğu bölgeyi buldukları ve bu bölgenin etrafında toplandıkları gözlemlenmiştir.



Şekil 5.10. KheperaIII robotları ile gerçekleştirilen uygulamadan anlık kareler.

Algılayıcı No	Açı (Derece)
2	-75
3	-45
4	-15
5	15
6	45
7	75



Şekil 5.11.: İtim potansiyel fonksiyonunda kullanılan kızıl ötesi algılayıcılar ve açıları.

BÖLÜM 6

6. SONUÇ

6.1. Yorumlar

Bu tez çalışmasında dinamik komşuluklu eşzamansız dağıtık parçacık sürü eniyileme yönteminin geliştirilmesi ve yöntemin çok robotlu arama görevinde uygulanması çalışılmıştır. Geliştirilen yöntemde parçacık komşuluklarının zaman ile dinamik olarak değişmesi ve parçacıkların eşzamansız olarak bilgi değişiminde bulunması ve güncellemelerini gerçekleştirmesi sağlanmıştır. Böylece geliştirilen yöntem merkezi olmayan/dağıtık bir biçimde çalıştırılmıştır. Ayrıca robotların arama sırasında paylaştıkları bilgilerde zaman gecikmesinin mevcut olabileceği de bir başka deyişle sistemdeki herhangi bir robotun diğer robotların elde ettiği en güncel bilgilere sahip olamayabileceği de düşünülmüştür. Geliştirilen yöntem deneysel olarak elde edilen gerçekci etanol gazı yoğunluğu bilgisinin kullanıldığı doğal olarak merkezi olmayan/dağıtık ve eşzamansız çalışma özelliklerine sahip bir çok robotlu (erkinli) sistem ile gaz yoğunluğunun en yüksek olduğu bölgenin bulunması görevinde kullanılmıştır. Sistemin başarısı benzetimler ve uygulamalar ile sınanmıştır.

Dinamik komşuluklu parçacık sürü eniyileme yönteminde parçacık komşuluklarının zaman ile değişmesinden dolayı parçacıklar farklı zaman anlarında farklı parçacıklar ile etkileşim içindedirler. Belirtilen varsayımlar sağlanması durumunda sürüdeki parçacıkların sınırlı algılama/haberleşme alanları olmasına karşın parçacıkların farklı zaman anlarında farklı parçacıklar ile etkileşim içinde olmaları nedeniyle sürüdeki bilgi akışının devamlılığı sağlanmakta böylece sürü ortak bir konuma yakınsayabilmektedir. Parçacık komşuluklarının dinamik olarak değişmesi yöntemin eşzamansız ve dağıtık uygulamalarında önemli bir unsur olarak ortaya çıkmaktadır. Yöntemin eşzamansız ve dağıtık biçimi parçacıkların birbirlerinden bağımsız biçimde güncellemelerini gerçekleştirmeleri ve bilgi paylaşımında bulunmalarına olanak sağlamaktadır. Parçacık komşuluklarının dinamik olarak değişmesi parçacıklar arası eşzamansız bilgi paylaşımına ayrıca parçacıkların sürüdeki bütün parçacıklar ile haberleşme ihtiyacını ortadan kaldırarak parçacıkların eşzamansız güncellemeler gerçekleştirmelerine imkan sağlamaktadır. Ayrıca paylaşımda bulunan bilgilerde zaman gecikmesinin mevcut olmasına izin verilerek güncel olmayan bilginin de kullanılmasına izin verilmektedir. Geliştirilen yöntemin doğal olarak dağıtık

ve eşzamansız çalışma özelliklerine sahip çok robotlu sistemlerin bilinmeyen bir ortamdaki arama görevinde kullanılması parçacık sürü eniyileme yönteminin farklı bir uygulama alanında kullanılmasına örnek teşkil etmektedir. Geliştirilen yöntemdeki parçacıkların çalışma felsefeleri çok robotlu sistemlerdeki robotların çalışma özellikleri ile uygunluk göstermektedir. Gerçekleştirilen benzetimler ve uygulamalar sonucunda robotların gaz yoğunluğunun en yüksek olduğu bölgeye yakınsadığı gözlemlenmiştir.

İlgili bölümlerdeki benzetimlerde ve uygulamalarında yöntemin eşzamansız, dağıtık ve dinamik komşuluklu biçimi başarıyla geliştirilmiş ve çok robotlu bir sistemin arama görevinde başarıyla uygulanmıştır. Bu anlamda tez çalışması amacına ulaşmıştır.

Parçacık sürü eniyileme yöntemin dinamik komşuluklu ve eşzamansız dağıtık biçimlerinin başarımının sınandığı benzetimlerde geliştirilen yöntemlerden elde edilen sonuçlar temel parçacık sürü eniyileme yönteminden elde edilen sonuçlar ile karşılaştırılmış ve geliştirilen yöntemlerin temel yöntem ile karşılaştırabilir bir başarım sergilediği gözlemlenmiştir. Öte yandan geliştirilen yöntemin çok robotlu bir sistemin bilinmeyen bir ortamdaki arama görevinde robotların sadece gaz yoğunluğunun en yüksek olduğu bölgeyi (dikkat edilirse belirli bir konumu değil) bulmaları istenmiştir. Bu nedenden dolayı geliştirilen yöntemin temel parçacık sürü eniyileme yöntemi ile karşılaştırılabilir bir başarım sergilemesi yeterli bulunmuştur. Yöntemin çok robotlu bir sistemdeki uygulamasında robotların konumlamaları için robotların mesafe sayaçları değerleri göz önünde bulundurulmuştur. Ancak mesafe sayaçlarında biriken hatalar yüzünden bu tür bir konumlama güvenilir olmamaktadır. Robotların güvenilir konumlanmaları için açık çevrim çalışan mesafe sayaçları ile birlikte atalet ölçme birimleri[1] kullanılarak kapalı çevrim bir konumlama sistemine veya belirtilen kapalı çevrim sistemde çeşitli filtreleme teknikleri kullanılmasına ihtiyaç duyulmaktadır.

6.2. Gelecek Çalışmalar

Her çalışma gibi bu tez çalışmasının da geliştirilmesi gereken veya gelecekte çalışılabilecek konular bulunmaktadır. Bu konuların çalışılması ile bu çalışmada önerilen yöntem için daha genel sonuçlar elde edilmesine ve önerilen yöntemin daha gerçekci uygulamalarda kullanılmasına olanak sağlayabilir.

Bölüm 3.'te sürüdeki bilgi akışının devamlılığı için Varsayım 1'in sağlanması üzerinde durulmuştur. Varsayım 1'de ele alınan üç değişik komşuluk belirleme yöntemi için

[1] ing: Inertial Measurement Unit

çok genel kalmaktadır. Yöntemde kullanılan üç değişik komşuluk belirleme yöntemi için sürüdeki bilgi akışının sürekliliğini ve sürünün ortak bir konuma yakınsamasını sağlayan daha belirli ve daha güçlü varsayımlar ortaya konulabilir. Böylece farklı komşuluk yapılarında sürüdeki bilgi akışı ve yöntemin başarımına olan etkisi daha detaylı bir biçimde incelenebilir. Ayrıca Bölüm 3.'te sürüdeki haberleşme/etkileşim yapısının sabit veya değişken yönlü çizgeler ile gösterilmesi yöntemin özelliklerinin gözlemlenmesi için yararlı olabilir. Parçacık sürü eniyileme yöntemi doğrudan bir arama yöntemi olup, $f : \mathbb{R}^n \rightarrow \mathbb{R}$ biçiminde verilen bir f fonksiyonu $x \in X \subset \mathbb{R}^n$ için

$$f(x^*) \leq f(x)$$

koşulunu sağlayan x^* her noktalarının bulunması biçimindeki problemler ile çalışılmaktadır. Burada X arama uzayını belirtmektedir. Yukarıda belirtilen noktalar kritik noktalardır (noktalar uzayın sınırlarında bulunmadığı sürece) ve

$$\nabla f(x^*) = 0$$

koşulunu sağlamaktadır. Birçok eniyileme yöntemi $\nabla f(x^*) = 0$ koşulunu sağlayan bir noktaya yakınsamayı amaçlamaktadır. Ancak bildiğimiz kadarıyla yöntem ile ilgili yapılan çalışmalarda yöntemin kritik noktalara yakınsama davranışı ile ilgili titiz yakınsaklık analizi gerçekleştirilmemiştir. Yöntemin yakınsaklık analizi ile ilgilenilen çalışmalarda genel olarak parçacıkların ortak bir noktaya (bu nokta kritik bir nokta olmak zorunda değil) yakınsayıp yakınsamayacağı araştırılmıştır (bakınız [20, 25, 56]). Yöntemin komşuluk yapısının yönlü çizgeler ile gösterilmesi, literatürde mevcut olan çizge kuramının sonuçlarının kullanılarak yöntemin yakınsaklık özelliklerinin belirlenmesi ve yakınsaklık analizinin gerçekleştirilmesi için önayak olabilir.

Parçacık sürü eniyileme yönteminin önerilen eşzamansız, dağıtık ve dinamik komşuluklu biçimi Bölüm 4.'te çeşitli denektaşı fonksiyonların eniyilenmesi ile sınanmıştır. Yöntemin bu biçiminin daha karmaşık ve daha gerçekci mühendislik eniyileme problemlerinde uygulanması yöntemin verimliliğinin gösterilmesi için uygun olabilir. Yöntemin paralel bir biçimde çalışması, karmaşık mühendislik uygulamalarında kullanılması için önemli bir avantajdır.

Öte yandan geliştirilen yöntemin çok robotlu sistemlerin arama görevinde uygulanmasında kullanılan deneysel olarak elde edilen gerçekci etanol gazı yoğunluğu bilgisi bu çalışmada kabul edildiğinin aksine zamanla dinamik olarak değişmektedir.

Geliştirilen arama yönteminin durağan ortamlarda etkin başarım sergilediği ve çok robotlu sistemlerin çalışma felsefesine uygun olduğu gözlemlenmesine karşın yöntemin dinamik olarak değişen bir ortamda uygulanması daha gerçekci bir durumdur. Bu tip bir uygulama kapalı bir ortamda dinamik olarak değişen bir gaz ortamının yaratılarak ve arama yapan robotlar gaz algılayıcıları ile donatılarak gerçekleştirilebilir. Bu sayede geliştirilen yöntemin bu tip uygulamalar için etkinliği bir kez daha gösterilmiş olur.

KAYNAKLAR

[1] Beni, G., Wang, J., Swarm intelligence in cellular robotic systems, Proceedings of NATO Advanced Workshop on Robots and Biological Systems, 26-30, Tuscany, Italy, Mayıs 1989.

[2] Bonabeau, E., Dorigo, M., Theraulaz, G., Swarm Intelligence: From Natural to Artificial Systems. *Oxford University Press*, New York, 1999.

[3] Kennedy, J., Eberhart, R.C., Shi, Y., Swarm Intelligence. *Morgan Kaufmann Publishers*, San Francisco, CA, 2001.

[4] Dorigo, M., Stützle, T., Ant Colony Optimization. *MIT Press*, Cambridge, MA, 2004.

[5] Pham, D.T., Ghanbarzadeh, A., Koc, E., Otri, S., Rahim, S., Zaidi, M., The Bees Algorithm - A Novel Tool for Complex Optimisation Problems, Proceedings of IPROMS Conference, 454-461, 2006.

[6] Yang, L., Passino, K.M., Swarm Intelligence: Literature Survey. Ohio State University, Mart 2000.

[7] Sahin, E., Swarm robotics: From sources of inspiration to domains of application, Swarm Robotics: State-of-the-art Survey, (E. Sahin and W. Spears, Eds.), Lecture Notes in Computer Science (LNCS 3342), 10-20, Springer-Verlag, Berlin Heidelberg, 2005.

[8] "Swarm-bots: Swarms of self-assembling artifacts." erişim adresi: http://www.swarm-bots.org, erişim tarihi: 11 Şubat 2009.

[9] "Swarmanoid Project." erişim adresi: http://www.swarmanoid.org, erişim tarihi: 11 Şubat 2009.

[10] "Guardians: Group of unmanned assistant robots deployed in aggregative navigation supported by scent detection." erişim adresi: http://www.guardians-project.eu, erişim tarihi: 11 Şubat 2009.

[11] "Viewfinder EU Project." erişim adresi: http://view-finder-project.eu, erişim tarihi: 11 Şubat 2009.

[12] Kennedy, J., Eberhart, R.C., A new optimizer using particle swarm theory, Proceedings of the Sixth International Symposium on Micromachine and Human Science, 39-43, Nagoya, Japan, Ekim 1995.

[13] Kennedy, J., Eberhart, R.C., Particle swarm optimization, Proceedings of IEEE International Conference on Neural Networks (ICNN), 1942-1948, Perth, WA, Australia, Kasım 1995.

[14] Fan, H., Shi, Y., Study on Vmax of particle swarm optimization, Proceedings of IEEE Swarm Intelligence Symposium (SIS), 193-197, Indianapolis, Indiana, USA, 2003.

[15] Shi, Y., Eberhart, R.C., Parameter selection in particle swarm optimization, Proceddings of IEEE Conference on Evolutionary Computation, 591-600, San Diego, USA, Mart 1998.

[16] Shi, Y., Eberhart, R.C., A modified particle swarm optimizer, Proceedings of IEEE Conference on Evolutionary Computation, 69-73, Indianapolis, Indiana,

USA, Mayıs 1998.
[17] Zheng, Y., Ma, L., Zhang, L., Qian, J., On the convergence analysis and parameter selection in particle swarm optimization, Proceedings of International Conference on Machine Learning and Cybernetics, 1802-1807, Hangzhou, China, Kasım 2003.
[18] Shi, Y., Eberhart, R.C., Fuzzy adaptive particle swarm optimization, Proceedings of IEEE Congress on Evolutionary Computation (CEC), (1), 101-106, Seoul, Korea, Mayıs 2001.
[19] Ratnaweera, A., Halgamuge, S.K., Watson, H.C., Self-organizing hierarchical particle swarm optimizer with time varying acceleration coefficients, IEEE Transactions on Evolutionary Computation, 8(3), 240-255, Mayıs 2004.
[20] Clerc, M., Kennedy.J, The particle swarm optmization: explosion, stability, and convergence in a multidimensional complex space, IEEE Transactions on Evolutionary Computation, 6, 58-73, Aralık 2002.
[21] Clerc, M., The swarm and the queen towards a deterministic and adaptive particle swarm optimization, Proceedings of IEEE Congress on Evolutionary Computation (CEC), 1951-1957, Washington, DC, USA, Ekim 1999.
[22] Eberhart, R.C., Shi, Y., Comparing inertia weigths and constriction factors in particle swarm optimization, Proceedings of IEEE Congress on Evolutionary Computation (CEC), (1), 84-88, La Jolla, CA, USA, Temmuz 2000.
[23] Carlisle, A., Dozier, G., An off-the shelf PSO, Proceedings of Workshop on Particle Swarm Optimization, Indianapolis, Indiana, USA, Nisan 2001.
[24] Bratton, D., Kennedy, J., Defining a standard for particle swarm optimization, Proceedings of IEEE Swarm Intelligence Symposium (SIS), Indianapolis, Indiana, USA, Nisan 2007.
[25] Trelea, I.C., The particle swarm optimization algorithm: Convergence analysis and parameter selection, Information Processing Letters, 85(9), 317-325, 2003.
[26] Angeline, P.J., Using selection to improve particle swarm optimization, Proceedings of IEEE Congress on Evolutionary Computation (CEC), 84-89, Anchorage, AK, USA, Mayıs 1998.
[27] Eberhart, R.C., Shi, Y., Comparison between genetic algorithms and particle swarm optimization, Proceddings of IEEE Conference on Evolutionary Programming, 611-616, San Diego, California, USA, Mayıs 1998.
[28] Shi, Y., Eberhart, R.C., An Empirical Study of Particle Swarm Optimization, Proceddings of IEEE Congress on Evolutionary Computing (CEC), 1945-1949, Washington, DC, USA, Mayıs 1999.
[29] Angeline, P.J., Evolutionary optimization versus particle swarm optimization: philosophy and performance differences, Proceddings of IEEE Conference on Evolutionary Programming, 601-610, San Diego, California, USA, Mayıs 1998.
[30] Kennedy, J., Small worlds and mega minds: Effects of neighborhood topology on particle swarm optimization, Proceedings of IEEE Congress on Evolutionary Computation (CEC), (3), 1931-1938, Washington DC, USA, Haziran 1999.
[31] Kennedy, J., Mendes, R., Population structure and particle swarm performance, Proceedings of IEEE Congress on Evolutionary Computation (CEC), (2), 1671-1676, Honolulu, HI, USA, 2002.

[32] Kennedy, J., Mendes, R., The fully informed particle swarm: Simpler, maybe better, IEEE Transactions on Evolutionary Computation, 8(3), 204-210, Haziran, 2004.
[33] Watts, D.J., Strogatz, S.H., Collective dynamics of small world networks, Nature, 393, 440-442, Haziran, 1999.
[34] Løvbjerg, M., Rasmussen, T., Krink, T., Hybrid particle swarm optimizer with breeding and subpopulations, Proceedings of the Third Genetic and Evolutionary Computation Conference (GECCO), (1), 369-476, San Francisco, USA, 2001.
[35] Haupt, R.L., Haupt, S.E., Practical Genetic Algorithms, *Wilwey-IEEE*, 2004.
[36] Løvbjerg, M., Krink, T., Extending particle swarms with self-organized critically, Proceedings of Fourth Congress on Evolutionary Computation (CEC), (2), 1588-1593, New York NY, USA, Mayıs 2002.
[37] Miranda, V., Fonseca, N., New evolutionary particle swarm algorithm (EPSO) applied to voltage/var control, The 14th Power Systems Computation Conference (PSSC), (2), 745-750, Seville, Spain, 2002.
[38] Esquviel, S.C., Coello Coello, C.A., On the use of particle swarm optimization with multimodal functions, Proceedings of IEEE Congress on Evolutionary Computation (CEC), (2), 1130-1136, Honolulu, HI, USA, Aralık 2003.
[39] Higashi, N., Iba, H., Particle swarm optimization with gaussian mutation, Proceedings of IEEE Symposium on Swarm Intelligence (SIS), 72-79, Indianapolis, Indiana, USA, Nisan 2003.
[40] Stancey, T., Jancic, M., Grundy, I., Particle swarm optimization with mutation, Proceedings of IEEE Congress on Evolutionary Computation (CEC), (2), 1425-1430, Canberra, Australia, Aralık 2003.
[41] Verterstørm, J.S., Riget, J., A diversity-guided particle swarm optimizer-the ARPSO, EVALife Technical Report, Department of Computer Science University of Aarhus, Şubat 2002.
[42] Blackwell, T., Bentley, P.J., Don't push me! collision-avoiding swarms, Proceedings of IEEE Congress on Evolutionary Computation (CEC), (2), 1691-1696, Honolulu, HI, USA, Mayıs 2002.
[43] Liang, J.J., Qin, A.K., Suganthan, P.N., Baskar, S., Comprehensive Learning Particle Swarm Optimizer for Global Optimization of Multimodal Functions, IEEE Transactions on Evolutionary Computation, 10(3), 281-295, Haziran 2006.
[44] Thanmaya, P., Veeramachaneni, K., Mohan, C.K., Fitness-Distance-Ratio Based Particle Swarm Optimization, Proceedings of IEEE Swarm Intelligence Symposium (SIS), 174-181, Nisan 2003.
[45] Vrahatis, M., Parsopoulos, K.E., On the computation of all global minimizers through particle swarm optimization. IEEE Transactions on Evolutionary Computation, 8, 221-224, Haziran 2004.
[46] Xie, X., Zhang, W., Yang, Z., A dissipative particle swarm optimization, Proceedings of IEEE Congress on Evolutionary Computation (CEC), (2), 1456-1461, Honolulu, HI, USA, Mayıs 2002.
[47] van den Bergh, F., Engelbrecht, A.P., A cooperative approach to particle swarm

optimization, IEEE Transactions on Evolutionary Computation, 8, 225-239, Haziran 2004.

[48] Monson, C.K., Seppi, K.D., The Kalman Swarm A new approach to particle motion in swarm optimization, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), 140-150, Haziran 2004.

[49] Krink, T., Løvbjerg, M., The lifecycle model: combining particle swarm optimisation, genetic algorithms and hill climbers, Proceedings of Parallel Problem Solving Nature (PPSN), 621-630, Granada, Spain, Ocak 2002.

[50] Hendtlass, T., Randall, M., A survey of ant colony and particle swarm metaheuristics and their application to discrete optimization problems, Proceedings of Inaugural Workshop on Artificial Life, 15-25, Melbourne, Australia, 2001.

[51] Hendtlass, T., A Combined Swarm Differential Evolution Algorithm for Optimization Problem, Proceedings of 14th IAE/AIE, 15-25, 2001.

[52] Verterstørm, J.S., and Thomsen, R., A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems, Proceedings of IEEE Congress on Evolutionary Computation (CEC), (2), 1980-1987, Portland, OR, USA, Haziran 2004.

[53] Ozcan, E., Mohan, C.K., Analysis of a simple particle swarm optimization system, Intelligent Engineering Systems Through Artificial Neural Networks (ANNIE), 8, 253-258, St Louis, Missouri, 1998.

[54] Ozcan, E., Mohan, C.K., Particle swarm optimization: surfing the waves, Proceedings of IEEE Congress on Evolutionary Computation (CEC), (3), 1939-1944, Washington, D.C, USA, Temmuz 1999.

[55] Kadirkamanathan, V., Selvarajah, K., Fleming, P.J., Stability analysis of the particle dynamics in particle swarm optimizer, IEEE Transactions on Evolutionary Computation , 10(3), 245-255, Haziran 2006.

[56] van den Bergh, F., Engelbrecht, A.P., A study of particle swarm optimization particle trajectories, Information Sciences, 176(8), 937-971, Nisan 2006.

[57] Parsopoulos, K.E., Vrahatis, M., Particle swarm optimizer in noisy and continuously changing envoirments, Proceedings of International Conference on Artificial Intelligence and Soft Computing, 289-294, Cancun, Mexico, 2001.

[58] Carlisle, A., Dozier, G., Adapting particle swarm optimization to dynamic envoirments, Proceedings of International Conference on Artificial Intelligence, 429-434, Las Vegas, Nevada, USA, 2000.

[59] Hu, X., Eberhart, R.C., Adaptive particle swarm optimization: detection and response to dynamic systems, Proceedings of IEEE Congress on Evolutionary Computation, (2), 1666-1670, Honolulu, HI, USA, 2002.

[60] Gudise, V.G., Venayagamoorthy, G.K., Comparison of particle swarm optimization and backpropagation as training algorithms for neural networks, Proceedings of IEEE Symposium on Swarm Intelligence (SIS), 110-117, Indianapolis, Indiana, USA, 2003.

[61] Zhang, C., Shao, H., Li, Y., Particle swarm optimisation for evolving neural networks, Proceedings of IEEE Symposium on Systems, Man and Cybernetics,

(4), 2487-2490, Washington DC, USA, 2000.
[62] Van den Bergh, F., Engelbrecht, A.P., Cooperative learning in neural networks using particle swarm optimizers, South African Computer Journal, 26, 84-90, 2000.
[63] Conradie, R.A., Miikkulainen, R., Aldrich, C., Adaptive control utilising 'Neural Swarming', Proceedings of Genetic and Evolutionary Computation Conference (GECCO), 60-67, Honolulu, HI, USA, 2002.
[64] Omran, M., Salman, A., Engelbrecht, A.P., Image classification using particle swarm optimization, Proceedings of 4th Asia-Pacific Conference on Simulated Evolution and Learning, Singapore, 2002.
[65] Talbi, H., Batouche, M.C., Particle swarm optimization for image registration, Proceedings of International Conference on Information and Communication Technologies: Theory to Applications, 397-398, Nisan 2004.
[66] Messerschmidt, L., Engelbrecht, A.P., Learning to play games using a PSO-based competitive learning approach, IEEE Transactions on Evolutionary Computation, 8(3), 280-288, Mayıs 2004.
[67] Franken, N., Engelbrecht, A.P., Comparing PSO structures to learn the game of checkers from zero knowledge, Congress on Evolutionary Computation (CEC), (1), 234-241, Aralık 2003.
[68] Fukuyama, Y., Yoshida, H., A particle swarm optimization for reactive power and voltagecontrol in electric power systems, Proceedings of Congress on Evolutinary Computation (CEC), (1), 87-93, Seoul, South Korea, Mayıs 2001.
[69] Suganthan N., Particle swarm optimiser with neighborhood operator, Proceedings of IEEE Congress on Evolutionary Computation (CEC), 1945-1950, 1999.
[70] Liang, J.J., Suganthan, P.N., Dynamic Particle Swarm Optimizer, Procedings of IEEE Swarm Intelligence Symposium (SIS), 124-129, Mayıs 2005.
[71] Hu, X., Eberhart, R.C., Multiobjective optimization using dynamic neighborhood particle swarm optimization, Proceedings of IEEE Congress on Evolutionary Computation (CEC), (2), 1677-1681, Honolulu, Hawaii, USA, Mayıs 2002.
[72] Eberhart, R.C., Hu, X., Shi, Y., Particle swarm with extended memory for multiobjective optimization, Proceedings of IEEE Swarm Intelligence Symposium (SIS), 193-197, Indianapolis, Indiana, USA, Nisan 2003.
[73] Mohais, A., Mendes, R., Ward, C., Posthoff, C., Neighborhood re-structuring in particle swarm optimization, Proceedings of Australian Conference on Artificial Intelligence, 776-785, Kasım 2005.
[74] Richards, M., Ventura, D., Dynamic Sociometry in Particle Swarm Optimization, Proceedings of International Conference on Computational Intelligence and Natural Computing, 1557-1560, North Carolina, USA, Eylül 2003
[75] Schutte, J.F., Reinbolt, J.A., Fregly, B.J., Haftka, R.T., George, A.D., Parallel global optimization with the particle swarm algortihm, International Journal for Numerical Methods in Engineering, 61, 2296-2315, Haziran 2004.
[76] Koh, B.I., George, A.D., Haftka, R.T., Fregly, B.J., Parallel asynchronous

particle swarm optimization, International Journal for Numerical Methods in Engineering, 67, 578-595, Ocak 2006.
[77] Venter, G., Sobieski, J.S., A parallel particle swarm optimization algorithm accelerated by asynchronous evaluations, Proceedings of 6'th World Congress of Structural and Multidisciplinary Optimization, Rio de Janeiro, Brazil, Haziran 2005.
[78] Chang, F.J., Chu, S.C., Roddick, J.F., Pan, J.S., A Paralel Particle Swarm Optimization Algorithm with Communication Strategies, Journal of Information Science and Engineering, 21, 809-818, Eylül 2005.
[79] Belal, M., El-Ghazawi, T., Parallel Models for Particle Swarm Optimizers, The International Journal of Intelligent Computing and Information Sciences, 4(1), 100-111, Ocak 2004.
[80] Doctor, S., Venayagamoorthy, G.K., Gudise, A.V., Optimal PSO for collective robotic search applications, Proceedings of IEEE Congress on Evolutionary Computation (CEC), (2), 1390-1395, Haziran 2004.
[81] Hereford, J.M., A distributed particle swarm algorithm for swarm robotic applications, Proceedings of IEEE Congress on Evolutionary Computation (CEC), (Vancouver, BC, Canada), 2, 1678-1685, Haziran 2006.
[82] Hereford, J.M., Siebold, M., Nichols, S., Using the particle swarm optimization algorithm for robotic search applications, Proceedings of IEEE Symposium on Swarm Intelligence (SIS), 53-59, 2007.
[83] Hereford ,J.M., Siebold, M., Multi-robot search using a physically-embedded Particle Swarm Optimization, International Journal of Computational Intelligence Research, 4(2), 197-209, 2008.
[84] Pugh, J., Martinoli, A., Inspiring and modelling multi-robot search with particle swarm optimization, roceedings of IEEE Swarm Intelligence Symposium (SIS), Honolulu, Hawaii, USA, Nisan 2007.
[85] Marques, L., Nunes, U., de Almedia, A.T., Particle swarm-based olfactory guided search, Autonomous Robots, 20(3), 277-287, Mayıs 2006.
[86] Jatmiko, W., Sekiyama, K., Toshio, F., A PSO-based Mobile Sensor Network for Odor Source Localization in Dynamic Envoirment: Theory, Simulation and Measurement, IEEE Congress on Evolutionary Computation, 1036-1043, Vancouver, BC, Canada, Temmuz 2006.
[87] Jatmiko, W., Sekiyama, K., Toshio, F., A PSO-based Mobile Sensor Network for Odor Source Localization in Dynamic Advection-Diffusion with Obstacles Environment: Theory, Simulation and Measurement, IEEE Computational Intelligence Magazine, 37-51, Mayıs 2007.
[88] Xue, S., Zeng, J., Sense Limitedly, Interact Locally: the Control Strategy for Swarm Robots Search, IEEE International Conference on Networking, Sensing and Control (ICNSC), 402-407, Nisan 2008.
[89] Gazi, V., Değişken komşuluklu parçacık sürü eniyileme yöntemi, Sinyal İşleme ve İletişim Uygulamaları Kurultayı (SİU), Eskişehir, Haziran 2007.
[90] Gazi, V., Eşzamanlı olmayan parçacık sürü eniyileme yöntemi, Sinyal İşleme ve İletişim Uygulamaları Kurultayı (SİU), Eskişehir, Haziran 2007.
[91] Akat, S.B., Gazi, V., Particle Swarm Optimization with Dynamic

Neighborhood Topology: Three Neighborhood Strategies and Preliminary Results, Proceedings of IEEE Swarm Intelligence Symposium (SIS), St. Louis, Missouri, USA, Eylül 2008.

[92] Akat, S.B., Gazi, V., Decentralized asynchronous particle swarm optimization, Proceedings of IEEE Swarm Intelligence Symposium (SIS), St. Louis, Missouri, USA, Eylül 2008.

[93] Moreau, L., Stability of multiagent systems with time-dependent communication links, IEEE Trans. on Automatic Control, 50(2), 169-182, Şubat 2005.

[94] Ren, W., Beard, R.W., A note on leaderless coordination via bidirectional and unidirectional time-dependent commnication, Proceedings of International Symposium on Mathematical Theory of Networks and Systems, Leuven, Belgium, Mayıs 2004.

[95] Ren, W., Beard, R.W., Consensus seeking in multi-agent systems under dynamically changing interaction topologies, IEEE Trans. on Automatic Control, 50(5), 655-661, Mayıs 2005.

[96] Köksal, M.I., Gazi, V., Fidan, B., Ordonez, R., Tracking a Maneuvering Target with a Non-holonomic Agent Using Artificial Potentials and Sliding Mode Control, Proceedings of 16th Mediterranean Conference on Control and Automation, 1174-1179, Haziran 2008.

[97] Barraquand, J., Latombe, J.-C., On nonholonomic mobile robots and optimal maneuvering, Proceedings of the IEEE International Symposium on Intelligent Control, 340-347, Eylül 1989.

[98] Koren, Y., Borenstein, J., Potential Field Methods and Their Inherent Limitations for Mobile Robot Navigation, Proceedings of the IEEE Conference on Robotics and Automation, 1398-1404, Sacremento, California, Nisan 1991.

[99] McFetridge, L., Ibrahim, Y.M., New technique of mobile robot navigation using a hybrid adaptive fuzzy potential field approach, Computers and Industrial Engineering, 35(4), 471-474, Aralık 1998.

[100] Ge, S.S., Cui, Y.J., New Potential Functions for Mobile Robot Path Planning, IEEE Transactions on Robotics and Automation, 16(5), 615-620, Ekim 2000.

[101] Gerkey, B.P., Vaughan, R.T., Howard, A., The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems, Proceedings of the International Conference on Advanced Robotics (ICAR), 317-323, Coimbra, Portugal, Temmuz 2003.

[102] "The Player Project." erişim adresi: http://playerstage.sourceforge.net/, erişim tarihi: 5 Kasım 2008.

[103] "Khepera3 Robot User Manual 2.2." erişim adresi: http://ftp.k-team.com/KheperaIII/Kh3.Robot.UserManual.2.2.pdf, erişim tarihi: 5 Kasım 2008.

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı,Adı	: Akat, Salih Burak
Uyruğu	: Türkiye Cumhuriyeti
Doğum tarihi ve yeri	: 25.04.1985 Ankara
Medeni hali	: Bekar
Telefon	: 0 (312) 223 46 94
Faks	: 0 (312) 292 40 91
e-mail	: sbakat@etu.edu.tr

Eğitim

Derece	Eğitim Birimi	Mezuniyet Tarihi
Lisans	Koç Üniversitesi	2006

İş Deneyimi

Yıl	Yer	Görev
2007-2008	TOBB ETÜ	Araştırma Görevlisi

Yabancı Dil

İngilizce

Yayınlar

Akat, S.B., Gazi, V., Particle Swarm Optimization with Dynamic Neighborhood Topology: Three Neighborhood Strategies and Preliminary Results, Proceedings of IEEE Swarm Intelligence Symposium (SIS), St. Louis, Missouri, USA, Eylül 2008.

Akat, S.B., Gazi, V., Decentralized Asynchronous Particle Swarm Optimization, Proceedings of IEEE Swarm Intelligence Symposium (SIS), St. Louis, Missouri, USA, Eylül 2008.

Akat, S.B., Gazi, V., Marques, L., Asynchronous Particle Swarm Optimization Based Search with a Multi-Robot System, Proceedings of Workshop/Summer School on Evolutionary Computation (WSSEC), 64-67, Londonderry , Northern Ireland, UK, Ağustos 2008.

Akat, S.B., Gazi, V., Marques, L., Eşzamansız Parçacık Sürü Eniyileme Yönteminin Çok Robotlu Arama Görevinde Uygulanması, Türkiye Otomatik Kontrol Konferansı (TOK), İstanbul, Türkiye, Kasım 2008.

Akat, S.B., Gazi, V., Marques, L., Asynchronous Particle Swarm Optimization Based Search with a Multi-Robot System: Simulation and Implementation on a Real Robotic System, Robotics for risky interventions and Environmental Surveillance (RISE), Brussels, Belgium, Ocak 2009.