T.C.

SELÇUK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

OPTİMİZASYON PROBLEMLERİNİN ÇÖZÜMÜ İÇİN YAPAY ARI KOLONİSİ ALGORİTMASI TABANLI YENİ YAKLAŞIMLAR

MUSTAFA SERVET KIRAN

DOKTORA TEZİ

Bilgisayar Mühendisliği Anabilim Dalını

MAYIS-2014
KONYA

TEZ KABUL VE ONAYI

Mustafa Servet Kıran tarafından hazırlanan "**Optimizasyon Problemlerinin Çözümü için Yapay Arı Kolonisi Algoritması Tabanlı Yeni Yaklaşımlar**" adlı tez çalışması 28/04/2014 tarihinde aşağıdaki jüri üyeleri tarafından oy birliği ile Selçuk Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda DOKTORA TEZİ olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Başkan Prof.Dr. Şirzat KAHRAMANLI	.....
Danışman Yrd.Doç.Dr. Mesut GÜNDÜZ	.....
Üye Prof.Dr. Ahmet ARSLAN	.....
Üye Doç.Dr. Mehmet ÇUNKAŞ	.....
Üye Yrd.Doç.Dr. İsmail BABAOĞLU	.....

Yukarıdaki sonucu onaylarım.

Prof. Dr. Aşır GENÇ
FBE Müdürü

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

Mustafa Servet KIRAN

20.05.2014

ÖZET

DOKTORA TEZİ

OPTİMİZASYON PROBLEMLERİNİN ÇÖZÜMÜ İÇİN YAPAY ARI KOLONİSİ ALGORİTMASI TABANLI YENİ YAKLAŞIMLAR

Mustafa Servet KIRAN

Selçuk Üniversitesi Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Yrd.Doç.Dr. Mesut GÜNDÜZ

2014, 129 Sayfa

Jüri

Danışman: Yrd.Doç.Dr. Mesut GÜNDÜZ
Prof.Dr. Şirzat KAHRAMANLI
Prof.Dr. Ahmet ARSLAN
Doç.Dr. Mehmet ÇUNKAŞ
Yrd.Doç.Dr. İsmail BABAOĞLU

Optimizasyon problemlerinin çözümü için kullanılan klasik optimizasyon teknikleri genellikle önerildikleri problemler için etkin sonuçlar üretmesine rağmen birçok farklı yapıda optimizasyon problemi olmasından dolayı bazı optimizasyon problemlerinin çözümünde yetersiz kalabilmektedir. Sürü zekâsına dayanan pozisyon güncelleme ile optimal ya da yakın optimal bir çözüm elde etme çabasında olan parçacık sürü optimizasyonu (PSO), karınca kolonisi optimizasyonu (ACO) ve yapay arı kolonisi algoritması (ABC) sadece bir probleme özel olmamakla birlikte birçok farklı optimizasyon probleminin çözümünde başarılı şekilde kullanılmaktadır. Doğadaki sürülerin davranışlarından esinlenilerek oluşturulmuş olan bu teknikler, popülasyon tabanlıdırlar, birden fazla noktadan çözüm uzayını araştırmaya başlarlar ve popülasyonun bireyleri olan yapay ajanlar arasında sıkı bir işbirliği ve etkileşim bulunmaktadır. PSO kuş veya balık sürülerinin yiyecek kaynağına doğru yaptıkları hareketi taklit ederek, ACO karıncaların yuva ile yiyecek kaynağı arasındaki davranışlarını simüle ederek ve ABC gerçek bal arıları kolonilerindeki yiyecek araştırma ve bilgi paylaşımı davranışlarını kullanarak optimizasyon problemi için optimal veya yakın optimal çözüm elde etmeye çalışır. Problemlerin yapılarına göre her yöntem farklı çözüm uzayında hareket edecek şekilde yapılandırılabilir. ACO ayrık çözüm uzayında hareket edebilecek yapıda tasarlanmıştır, ABC ve PSO yöntemlerindeki ajanlar ise sürekli çözüm uzayında araştırma yapma kabiliyetine sahiptirler. Çeşitli problemlerin yapısına uygun olarak bu yöntemlerin geliştirilmesi mümkündür ve ayrık problemler için önerilmiş olan ACO sürekli uzayda hareket edebilecek bir yapıya büründürülürken, ABC ve PSO ise ayrık uzayda çalışacak şekilde modifiye edilmişlerdir. Fakat hâlâ bu yöntemlerin araştırma yeteneklerinin dengelenmesine, yeni araştırma stratejileri ile global ve/veya yerel araştırma kabiliyetlerinin arttırılmasına ve farklı problemler için modifiye edilmelerine ihtiyaç duyulmaktadır.

Bu tez çalışması bu yöntemlerden biri olan ABC algoritmasının sürekli ve ayrık versiyonlarını geliştirmek, iyileştirmek veya güncellemek üzerine kurulmuştur. Geliştirilen yöntemler literatürde sıklıkla kullanılan ayrık ve/veya sürekli kıyas problemlerinin çözümü için uygulanmış ve elde edilen sonuçlar bilinen diğer yöntemler ile kıyaslanmıştır. Her problem türü için farklı sayıda çalışmalar yapılmıştır.

Sürekli problemleri çözmek için ABC algoritması ilk olarak çaprazlama operatörleri ile yerel araştırması ve yakınsama yeteneği arttırılmaya çalışılmıştır. Elde edilen sonuçlar temel ABC algoritması ile kıyaslanmış ve sonuçlar raporlanmıştır. İkinci çalışmada ABC algoritması ve modifiye edilmiş versiyonları Türkiye'nin elektrik enerjisi tahmini için uyarlanmış ve elde edilen sonuçlar PSO ve ACO ile kıyaslanmıştır. Üçüncü çalışmada araştırma kabiliyetlerini arttırmak ve daha iyi sonuçlar elde etmek amacıyla ABC ve PSO yöntemleri hibritleştirilmiştir. Elde edilen sonuçlar ABC ve PSO yöntemlerine dayanan diğer hibrit yöntemler ile karşılaştırılmıştır. Sürekli problemlerin çözümü için önerilen son çalışmada ise farklı güncelleme kuralları bir seçim mekanizması ile birleştirilerek ABC algoritmasında kullanılmış ve literatürde popüler diğer ABC varyantları ve diğer popülasyon tabanlı optimizasyon teknikleri ile karşılaştırılmıştır.

Ayrık problemlerin çözümü için komşuluk operatörleri ile ayrıklaştırılmış ABC algoritmasının performans analizi gezgin satıcı problemi üzerinde yapılmış ve diğerlerine göre iyi olan komşuluk operatörü belirlenmiştir. Bu belirlemeden elde edilen sonuçlara dayanarak ACO yöntemi ile elde edilen en iyi sonuç ABC algoritması ile iyileştirilmeye çalışılmış ve hiyerarşik bir yöntem olarak sunulmuştur. Ayrık problemlerin çözümü için yapılan son çalışmada ise ABC algoritması lojik XOR operatörü kullanılarak ikili optimizasyon problemlerini çözebilecek şekilde değiştirilmiştir ve ABC ajanları ikili uzayda hareket edebilecek bir yeteneğe kavuşturulmuştur. Önerilen yöntem bir ikili optimizasyon problemi olan kapasitesiz tesis yerleşim problemi üzerinde test edilmiş ve ikili PSO, iyileştirilmiş ikili PSO ve DisABC yöntemleri ile kıyaslanmıştır. Yapılan tüm çalışmalardan görülmüştür ki yerel araştırma, global araştırma, durağanlaşma ile mücadele ve yerel araştırmaya karşın global araştırmanın veya tersinin dengelenmesi sürü zekâsına dayanan teknikler ve özellik ABC algoritması için başarılı sonuçlar elde etmenin vazgeçilmez şartlarıdır. ABC algoritmasında küçük değişiklikler yapılarak ABC algoritmasının ayrık optimization problemlerinin çözümü için de uygulanabileceği bu tez çalışmasında ayrıca görülmüştür.

Anahtar Kelimeler: ayrık optimizasyon, ikili optimizasyon, karınca kolonisi optimizasyonu, parçacık sürü optimizasyonu, sürekli optimizasyon, sürü zekâsı yöntemleri, yapay arı kolonisi algoritması.

ABSTRACT

Ph.D THESIS

NOVEL APPROACHES BASED ON ARTICIAL BEE COLONY ALGORITHM TO SOLVE OPTIMIZATION PRONLEMS

Mustafa Servet KIRAN

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE
OF SELÇUK UNIVERSITY
THE DOCTOR OF PHILOSOPHY
IN COMPUTER ENGINEERING

Advisor: Assist.Prof.Dr. Mesut Gündüz

2014, 129 Pages

Jury

Advisor: Yrd.Doç.Dr. Mesut GÜNDÜZ
Prof.Dr. Şirzat KAHRAMANLI
Prof.Dr. Ahmet ARSLAN
Doç.Dr. Mehmet ÇUNKAŞ
Yrd.Doç.Dr. İsmail BABAOĞLU

The classic optimization techniques which are proposed for solving the optimization problems generally produce the effective results for a specific optimization problem, but due to the fact that there are many types of the optimization problems, they cannot be applied to solve optimization problems with different characteristics. Swarm intelligence-based algorithms such as particle swarm optimization (PSO), ant colony optimization (ACO) and artificial bee colony (ABC) algorithm are not specific for an optimization problem, and they are used for solving many types of optimization problems. Being inspired the natural and intelligent behaviors of the swarms, these methods are population-based, start multiple points to search solution space, and there is vigorous collaboration and interaction among artificial agents which are members of population. PSO simulates the movements of fish or bird swarms towards food sources, ACO imitates the behaviors of real ants between nest and food source and ABC models the intelligent behaviors of real honey bee colony, such as information sharing and searching food source. Being simulated these behaviors of swarms, these methods try to obtain optimal or near optimal solution for the optimization problems. The basic version of each method can do search on the search spaces with different structures. While artificial agents of ACO can search discrete solution space, the agents of PSO or ABC can search continuous solution space. But, ACO has been modified for solving the optimization problems with continuous solution space, and PSO and ABC have been modified for solving optimization problems with discrete solution space. Briefly, each method can be modified for solving the optimization problems with different solution spaces. In order to obtain optimal or near optimal solution for the optimization problem and more robust methods, new search strategies for the methods and modification for applying to the many optimization problems are required for these methods.

This thesis is based on improvement, modification or updating of the ABC algorithm, which is one of the prominent members of swarm intelligence, for solving discrete or continuous optimization problems. Newly proposed approaches in this thesis are applied to the discrete or continuous optimization problems and obtained results are compared with the state-of-the-art methods. Both three studies have been completed for solving continuous optimization and three studies have been completed for solving discrete optimization problems. The first study is proposed for solving continuous optimization problems,

and it is based on ABC and crossover operators. The convergence speed and information sharing in the artificial hive are tried to improve by using crossover operators in the ABC algorithm. Obtained results are compared with the basic version of ABC algorithm and reported in the experimental results section. In the second study, the ABC algorithm and its variants proposed in this thesis are modified for applying estimation of electricity energy demand of Turkey. Obtained results are compared with the results of PSO and ACO. In order to improve search capabilities of the methods, ABC and PSO is hybridized in the last study for solving continuous optimization problems. The proposed hybrid method is compared with the basic ABC and PSO, and the other hybrid methods proposed in the literature.

In the first study proposed for solving discrete optimization problems, the discrete ABC algorithm with neighborhood operator is analyzed on solving the travelling salesman problem. Basic or combined nine neighborhood operators are used in discrete ABC, and which neighborhood operator is better than the others is determined. After this determination, the second study is hierarchically established on ACO and ABC algorithms. In the hierarchic approach, the best solution obtained by ACO is given to bee population in ABC algorithm to improve it. When the solution space of the problem is binary structured, the ABC algorithm should be modified for solving this class of optimization problems. In the last study for solving discrete optimization problems in this thesis, the ABC algorithm is modified by using XOR logic operator for solving binary optimization problems. Obtained results are compared with the state-of-art methods in the literature, such as binary PSO, discrete ABC (DisABC).

As seen from the studies done for this thesis, improvement or balancing of local search and/or global search, prevention of stagnation of the population are necessary conditions for obtaining optimal or near optimal solutions for the optimization problems by swarm intelligence-based methods, especially ABC algorithm. This thesis also shows that the ABC algorithm can be easly applied to optimization problems with discrete solution space with a bit modification in the algorithm.

Keywords: ant colony optimization, artificial bee colony algorithm, binary optimization, continuous optimization, discrete optimization, particle swarm optimization, swarm intelligence-based methods

ÖNSÖZ VE TEŞEKKÜR

Yaşanılan dünyada bilim ve teknolojideki hızlı ilerlemeler sayesinde son zamanlarda çözümsüzlük çok az problem için vardır ve birçok problem için ise birden fazla çözümün varlığı karşımıza çıkmaktadır. Bu noktada ise birden fazla çözüm içerisinden en avajtajlı olanın seçilmesi gerekmektedir. Birden fazla çözümden hangisinin diğerlerine göre zaman, maliyet vb. açılardan daha avantajlı olduğunun belirlenmesi karar vericiler için uzun ve yorucu bir süreç teşkil edebilmektedir. Doğal olarak bu da bizi bir tercihe yöneltmektedir. Birden fazla çözüm içerisinden en iyisine tercih etme optimizasyonun çekirdeğini oluşturur. Bu tez çalışması kapsamında da zeki optimizasyon tekniklerinden biri olan yapay arı kolonisi algoritması için yeni yaklaşımlar önerilmekte ve elde edilen çözümlerin kalitesi tespit edilmeye çalışılmıştır. Ayrıca önerilen yöntemlerin var olan diğer yöntemler ile kıyaslamaları yapılarak önerilen metotların deneysel doğrulaması yapılmaya çalışılmaktadır. Tezin dil düzeltmesi için tez, konuya hâkim olan ve olmayan araştırmacılar tarafından okunmuş ve gerekli düzeltmeler yapılmıştır fakat yine de bazı imlâ hatalarına rastlanabilir. Okuyuculara bu gibi hatalar için şimdiden özürlerimi sunarım. Araştırdığı konu itibariyle son yıllarda gündemde olan bu teknikler üzerine yapılmış olan bu tez çalışmasının araştırmacı ve uygulayıcılar için faydalı olmasını temenni ederim.

Yaklaşık 4 yıldır yoğun bir zaman ve emek sarfederek ortaya koyduğumuz bu çalışmada yardımlarını ve rehberliğini eksik etmeyen Danışman Hocam Sayın Yrd.Doç.Dr. Mesut Gündüz Bey'e teşekkür eder ve saygılarımı sunarım. Ayrıca tez izleme komitesinde yer alan Sayın Prof.Dr. Ahmet Arslan ve Doç.Dr. Mehmet Çunkaş Bey'lere önerileri ve eleştirileri için, doktora tez savunmam da yer alan jüri üyeleri Prof.Dr. Şirzat Kahramanlı ve Yrd.Doç.Dr. İsmail Babaoğlu Bey'lere de incelemeleri, tespitleri, eleştirileri ve düzeltmeleri için teşekkür ederim.

Bu yola çıkmamı sağlayan ve her zaman ve her koşulda bana destek olan rahmetli babam Mehmet Kıran'a ve dualarını benden esirgemeyen annem Meziyet Kıran'a teşekkürlerimi, sevgilerimi ve saygılarımı sunmadan bu tezin benim için tamamlanmış olmayacağının da bilinmesini isterim. Maddi ve manevi desteklerini esirgemeyen ve beni sürekli kollayan kardeşlerim Murat ve Ferhat Kıran'a da ayrıca minnettarım.

Elbette her zaman yanımda olan ve zor zamanlarımda desteğini esirgemeyen sevgili eşim Zehra Kıran'a, evde çalışırken en umutsuz ve yorgun zamanlarımda

gülümsemeleri, neşeleri ve sevgileri ile beni yeniden çalışmaya sevkeden oğullarım Mehmet Kıran ve Eren Kıran'a sonsuz sevgilerimi sunarım.

Beraber çalışma yaptığımız Doç.Dr. Harun Uğuz, Doç.Dr. Turan Paksoy, Yrd.Doç.Dr. Ömer Kaan Baykan, Arş.Gör. Eren Özceylan ve Arş.Gör. Hüseyin Haklı hocalarıma da çalışarak geçirdiğimiz zamanlar için teşekkür ederim.

Bu çalışmada bana doğrudan veya dolaylı yardımcı olan mesai arkadaşlarıma ve huzurlu bir çalışma ortamı sağladıkları için Bölüm ve Anabilim Dalı Başkanlarımıza teşekkür etmeyi bir borç bilirim.

Ayrıca tezimi hazırlamam süresince 2211-C Öncelikli Alanlar kapsamında burs imkânı sağlayan Türkiye Bilimsel ve Teknolojik Araştırma Kurumu (TÜBİTAK)'a maddî desteğinden dolayı ve tüm soru ve sorunlarımızı çözmede yardımcı olan değerli personeline teşekkür ederim. Ayrıca TÜBİTAK'ın gelecek yıllarda daha fazla araştırmacıya daha fazla destek sağlamasını da temenni ederim.

Mustafa Servet Kıran

KONYA – 2014

İÇİNDEKİLER

KISALTMALAR

ABC: Yapay Arı Kolonisi Algoritması

ACO: Karınca Kolonisi Optimizasyonu

ACS: Karınca Kolonisi Sistemi

AS: Karınca Sistemi

BABC: En iyi çözüm tabanlı yapay arı kolonisi algoritması

binABC: İki yapay arı kolonisi algoritması

BPSO: İkili Parçacık Sürü Optimizasyonu

CABC: Çaprazlama tabanlı yapay arı kolonisi algoritması

CAP: Kapasitesiz tesis yerleşim test problemi

D: Boyut

DisABC: Ayrık İkili Yapay Arı Kolonisi Algoritması

GDP: Gayrı Safi Milli Hasıla

HA: Hiyerarşik Karınca Kolonisi ve Ayrık Yapay Arı Kolonisi Algoritması

HPA: Hibrid Parçacık Sürü Optimizasyonu-Yapay Arı Kolonisi Algoritması

IABAP: Hibrit Yapay arı kolonisi ve Parçacık sürü optimizasyonu yöntemi

IBPSO: İyileştirilmiş İkili Parçacık Sürü Optimizasyonu

MIN: Maksimum çevrim sayısı

N: Popülasyondaki birey sayısı

Pop: Popülasyon boyutu

PSO: Parçacık Sürü Optimizasyonu

RI: Rastgele nokta ekleme

RIS: Rastgele Dizi Ekleme

RR: Random tersleme

RRIS: Alt dizinin rastgele terslenerek eklenmesi

RRSS: Rastgele Alt dizinin terslenerek yer değiştirmesi

RS: Rastgele nokta tabanlı yer değiştirme

RSS: Rastgele dizi tabanlı yer değiştirme

TSP: Gezgin Satıcı Problemi

UFLP: Kapasitesiz tesis yerleşim problemi

1.GİRİŞ

Hayatın her alanında insanların karşısına birden fazla seçenek çıkmaktadır ve her seçeneğin pozitif ve negatif yanları vardır. Seçeneklerle karşılaşan kişi, bu seçeneklerden hangisini seçeceğine bireysel olarak karar verebileceği gibi daha tecrübeli olanlardan yardım da alabilir. Problemin çözümü için mevcut yöntemlerden en iyisinin uygulanması, problemin çözümünden elde edilecek maksimum faydanın sağlanması ve/veya minimum zararın edilmesi tamamen optimum seçimle mümkündür. Bu süreçlerin tümü optimizasyonun alanında değerlendirilir ve çözümlenmeye çalışılır. İngilizce "opt" (yeğlemek) fiil kökünden türetilmiş olan optimizasyon kelimesi genel anlamda "daha iyiyi yapma veya elde etme" olarak tanımlanabilmesine rağmen bilgisayar bilimi, yöneylem araştırması, yapay zekâ ve ilgili araştırma alanlarında "*belirli bir optimizasyon problemi için geçerli çözümlerden kabul edilebilir bölge içerisinde en iyi olanı bulma*" olarak tanımlanabilir (Akay, 2009). Tanımda birkaç önemli nokta vardır ve bunların üzerinde durulması gerekir. Optimizasyon, optimizasyon problemleri için yapılır ve belirli sınırlar dahilinde amacı en küçük yapma (minimizasyon) ve/veya en büyük yapma (maksimizasyon) olarak ele alınmalıdır. Eğer amaç bir kâr fonksiyonu ise amaç kârın maksimize edilmesidir veya amaç bir maliyet fonksiyonu ise amaç maliyetin minimize edilmesidir. Bir diğer nokta ise "*geçerli çözümlerden kabul edilebilir bölge içerisinde olanı"* bulma ifadesidir. Burada anlatılmak istenen bir optimizasyon problemi için geçerli ve geçersiz çözümlerin mümkün olabilmesidir ve bu geçerli bölge içerisindeki bir çözümün elde edilmesidir ve mümkünse bu çözümünde diğer çözümlerin tümünden daha iyi olmasıdır. Optimizasyonun tanımından da anlaşılacağı üzere optimizasyonun iki önemli noktası vardır ve bunlar optimizasyon problemleri ve optimizasyon yöntemleridir. Bir optimizasyon problemi birden fazla olası çözümü olan bir problemdir ve bir optimizasyon yöntemi bu çözümlerden en iyi olanı bulma sürecidir.

Optimizasyon problemlerinin çözümü için optimizasyon teknikleri uygulanır. Optimizasyon teknikleri genel olarak iki ana grupta toplanabilir: *klasik* ve *sezgisel* teknikler. Klasik optimizasyon metotları bir optimizasyon problemi için kesin çözümü garanti etmesine rağmen bazı dezavantajlarından dolayı gerçek dünya problemleri için az bir uygulama alanı bulabilmektedir. Bu dezavantajlar şu şekilde sıralanabilir:

i) Problemin formülasyonundaki değişimler yöntemin çalışmamasına neden olabilmektedir.

ii) Probleme özgü olmalarından dolayı bir problem için çok iyi sonuç veren bir tekniğin bir başka probleme uygulanmasında zorluklar ortaya çıkabilmektedir.

iii) Çok sayıda yerel (local) optimuma sahip bir problemin çözümünde klasik teknikler yerel optimumlara takılabilmektedir.

iv) Probleme veya problemin boyutuna bağlı olarak klasik teknikler için hesaplama maliyeti (süre ve hafıza) çok fazla artabilmektedir.

Ayrıca, problem boyutunun artması, lineer olmayan problem yapısı gibi probleme özgü nedenlerden dolayı klasik teknikler bir çok optimizasyon probleminin çözümünde yetersiz kalmaktadır. Bu dezavantajlardan dolayı tavlama benzetimi (Kirkpatrick ve ark., 1953), tabu araştırma (Glover, 1989;1990), genetik algoritma (Holland, 1975), karınca kolonisi optimizasyonu (Dorigo ve ark., 1991), parçacık sürü optimizasyonu (Eberhart ve Kennedy, 1995), yapay arı kolonisi algoritması (Karaboğa, 2005) gibi sezgisel yöntemler optimizasyon problemlerinin çözümü için önerilmiştir. Sezgisel teknikler global optimum çözümü garanti edememelerine rağmen, makul bir hesaplama maliyeti ile optimum veya optimuma yakın bir çözümü elde edebilirler. Klasik tekniklerin dezavantajlarına karşılık sezgisel tekniklerin avantajları da aşağıda maddelenmiştir.

i) Bir probleme özgü değildirler ve birçok probleme kolaylıkla uyarlanabilirler.

ii) Türev vb. ağır matematiksel hesaplamaları gerektirmezler.

iii) Kâr veya maliyet amaç fonksiyonları ile çözümlerin kalitesini ayırdedebilirler.

iv) Problemin boyutuna bağlı olmaksızın makul sürelerde optimum veya optimuma yakın bir çözümü elde edebilirler.

v) Kesin matematiksel formüllerle yapılan tanımlamalar için gerçek dünya problemlerinde amaçlar, kısıtlar, problem verisinin toplanması gibi zorluklar mevcuttur. Model için giriş parametrelerinin belirlenmesinde kullanılan verinin hatalı olması, sezgisel yöntemle elde edilebilecek alt optimal çözümden daha fazla hataya neden olabilir (Karaboğa, 2011).

vi) Hesaplama maliyetleri düşüktür.

Yukarıdaki avantajlarından dolayı literatürde birçok sezgisel yöntem önerilmiştir ve önerilmeye devam edilmektedir. Bu tez kapsamında bu sezgisel tekniklerin bir alt dalı olan "*Sürü Zekâsı*" metotları irdelenmiş, eksik/fazla yönleri araştırılmış ve

iyileştirmeler yapılmıştır. Bu tezin amacı, hali hazırda mevcut olan yöntemlerin iyileştirilmesi, özellikle yapay arı kolonisi algoritması, bu yöntemler tabanında yeni hibrit yöntemlerin geliştirilmesi ve bazı deneysel/pratik çalışmaların yapılmasıdır. Bu bağlamda düşünüldüğünde tezin planı ve organizasyonu aşağıda belirtildiği şekilde yapılmıştır.

İlk bölümde sürü zekâsı yöntemleri optimizasyon problemlerini çözmek için önerildiğinden dolayı optimizasyon hakkında genel bilgiler verilmiştir ve genel bir giriş yapılmıştır.

İkinci bölüm sürü zekâsını ve onun içerdiği yöntemleri kapsar. Bu tezin odak noktasını yapay arı kolonisi algoritması oluşturmasına rağmen, sürü zekâsının iki temel tekniği de bazı hibritleştirmelerde veya hiyerarşik olarak kullanıldığı için bu iki teknik ve literatür özetleri de tez kapsamına alınmıştır. Sürü zekâsının en popüler üç yöntemi olan karınca kolonisi optimizasyonu (ant colony optimization –ACO), parçacık sürü optimizasyonu (particle swarm optimization –PSO) ve yapay arı kolonisi algoritması (artificial bee colony algorithm –ABC) literatür özetleri ikinci bölümde ayrı ayrı sunulmuştur. Üçüncü bölümde sürü zekâsı tanımlanmaya çalışılmış ve ACO, PSO ve ABC yöntemlerin kavramsal yapıları ve sözdekodları detaylandırılarak verilmiştir.

Dördüncü bölüm yöntemler üzerinde yapılan iyileştirmeleri kapsar. Tez kapsamında geliştirilen hibrit ve hiyerarşik metotlar da bu bölümde anlatılmıştır. Ayrıca iyileştirmeler sadece açıklanma kalmayıp deneysel çalışmaları, sonuçları ve diğer yöntemler ile karşılaştırmaları da dördüncü bölüme alınmıştır.

Dördüncü bölümde deneysel sonuçlar her iyileştirme ve/veya uygulama için verilmiş olmasına ve yapılan iyileştirmenin tartışılmış olmasına rağmen genel bir sonuçlandırma ve tartışma beşinci bölümde verilmiştir. Elbette okuyucu/araştırmacı için geleceğe yönelik bir takım planlar da bulunmaktadır. Analize, iyileştirmeye ve uygulamaya yönelik bir gelecek senaryosu da altıncı bölümde detaylı olarak açıklanmıştır.

1.1. Optimizasyon

Optimizasyon, bir optimizasyon probleminde olası çözümler içerisinde en iyisini bulmak demektir. Daha teknik bir ifadeyle U araştırma uzayı, $K \subseteq U$ kabul edilebilir çözümler kümesi ve F amaç fonksiyonu olmak üzere optimizasyon Denklem 1 (minimizasyon) ve Denklem 2 (maksimizasyon) ile tanımlanır.

$$\forall \vec{y} \in K \; için \;\; F(\vec{x}) \leq F(\vec{y}) \tag{1.1}$$

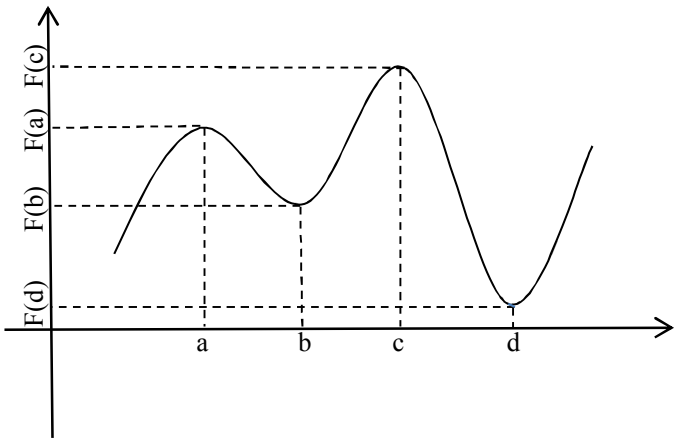
$$\forall \vec{y} \in K \; için \;\; F(\vec{x}) \geq F(\vec{y}) \tag{1.2}$$

Denklem 1.1 bir minimizasyon sürecini tanımlar ve kabul edilebilir bölge içerisinde fonksiyonu minimum yapan $\vec{x}$ karar değişkenlerinin bulunması işlemidir. Bu işlemin tersi yani fonksiyonu maksimum yapan $\vec{x}$ tasarım parametrelerinin bulunması sürecine maksimizasyon denir ve Denklem 1.2 ile tanımlanır. Buradan elde edilen çözümler global optimum çözümlerdir ve bir diğer çözüm türü olan yerel (bölgesel veya lokal) optimumlardan en az birine eşittir. Yerel optimum, yeteri kadar küçük *h* değerleri için Denklem 1.3 (yerel minimum) ve Denklem 1.4 (yerel maksimum) ile tanımlanır.

$$F(\vec{x}) \leq F(\vec{x} + \vec{h}) \tag{1.3}$$

$$F(\vec{x}) \geq F(\vec{x} + \vec{h}) \tag{1.4}$$

Denklem 1.1, 1.2, 1.3 ve 1.4 ile verilen tanımlar Şekil 1.1'de açık şekilde ifade edilmiştir.



Şekil 1.1. Kritik noktalar

Şekil 1.1'den görüldüğü üzere fonksiyonun *a (F(a)), b (F(b)), c(F(c)) ve d (F(d))* noktaları üzerinde yerel optimumları bulunmaktadır. Yerel maksimum *a* noktası üzerindedir ve küçük bir *h* komşuluğundaki tüm *(a+h)* karar değişkeni değerlerinden *F(d)* daha büyüktür. Aynı durumun tersi *b* noktası için de geçerlidir ve b noktası üzerinde bir yerel minimum bulunmaktadır. Fonksiyonun iki adet daha yerel optimum noktası bulunmaktadır fakat bu noktalar tüm komşuluklar için daha küçük/büyük değerlere sahiptir. Bundan dolayı karar parametresinin *c* değeri amaç fonksiyonu maksimum yaparken *d* değeri fonksiyonun minimum yapar ve bu iki nokta amaç

fonksiyonun global optimum noktalarıdır. Global ve yerel optimum hakkında detay verilmesinin sebebi sezgisel yöntemin araştırma kabiliyetine vurgu yapmak içindir ve bu konu ilerleyen bölümlerde detaylı olarak ele alınacaktır.

1.1.1. Karar Seti

Optimizasyonun belki de en önemli noktası karar setinin veya tasarım parametrelerinin belirlenmesidir. Tasarım değişkenlerinin belirlenmesini önemli yapan nokta sistemin bu tasarım değişkenlerini kullanarak çalışacak olmasıdır ve tasarım değişkenleri için sınırların veya sınırlamaların belirlenmesi gerekir. Karar setinin büyüklüğü optimizasyon probleminin boyutluluğuna denk gelir ve boyutun artması optimizasyon yöntemi için bir zorluk oluşturur (Boyer ve ark., 2005; Karaboğa ve Akay, 2009). Dikkat edilmesi gereken bir diğer önemli nokta karar seti için elde edilen değerlerin amaç fonksiyon için geçerli olabilmesidir. Geçerli olmayan değerler için sistemin çalışması imkânsızdır ve karar seti için geçerli çözümlerin elde edilmesi gerekir. Elde edilen çözümler fonksiyonun global optimum noktaları olmayabilir fakat sistemin tutarlı çalışabilmesini sağlar. Bundan dolayı karar seti veya tasarım değişkenlerinin alabileceği değerlerin yani sınırlarının dikkatli belirlenmesi gerekir.

Tasarım parametrelerinin alabileceği değerler sürekli değerler olabileceği gibi ayrık değerler de olabilir. Tasarım değişkenleri verilen bir kümeden belirli değerler alabiliyorsa bu tasarım değişkenine ayrık (kesikli, discrete) denir. Ayrıca tasarım değişkenleri sadece tam sayı değerleri veya ikili değerler alıyor olabilir. Sürekli değerlerin yuvarlanması, belirli bir tabana göre modunun alınması vb. çözümler ile bu problem türleri ifade edilebilir. Fakat bu durumda tasarım parametrelerinin kendi aralarındaki bağımlılığının dikkate alınması gerekebilir. Farklı tip değerler alabilen karar değişkenleri denklem 1.5, 1.6 ve 1.7'de gösterilmiştir.

$$-5 \leq x \leq 5 \tag{1.5}$$

$$x \in \{A, B, C, D\} \tag{1.6}$$

$$x \in \{0,1\} \tag{1.7}$$

Denklem 1.5 karar değişkeninin [-5,5] aralığında sürekli değerleri alabileceğini, Denklem 1.6 karar değişkenin A, B C veya D yani ayrık değerler alabileceğini, Denklem 1.7 karar değişkenin sadece 0 veya 1 değerini alabileceğini ifade eder. Denklem 1.5'te karar değişkeni sürekli iken Denkle 1.6 ve 1.7'de karar değişkeni ayrık/kesikli'dir. Yalnız şurası da unutulmamalıdır ki bir optimizasyon probleminin

karar seti hem sürekli hem ayrık değerlerden oluşabilir. Böyle bir durumda karar değişkenlerine ait tanımlamalar ayrı ayrı ifade edilmelidir ve optimizasyon problemi için bir çözümü oluşturacak karar setine doğru (uygun, feasible) değerler atanabilmelidir.

1.1.2. Amaç (Objective) ve uygunluk (Fitness) fonksiyonu

Amaç fonksiyonu herhangi bir optimizasyon probleminde karar değişkenlerinin durumunun değerlendirilmesi için kullanılan fonksiyondur. Girişleri karar değişkenleri olmakla birlikte çıkış değeri çözümün uygunluğunu değerlendirmek için kullanılır. Yani bu fonksiyon karar değişkenlerinin amaç üzerindeki etkilerini gösterir. Sezgisel yöntemler ile elde edilen çözümlerin birbiri ile kıyaslanması gerekir ve hangisinin veya hangilerinin diğerlerine göre daha uygun olduğuna bu fonksiyondan elde edilen değer ile karar veriler.

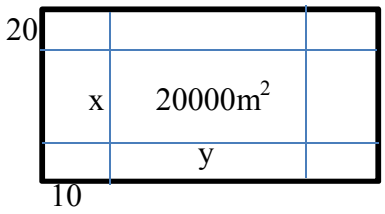
Amaç fonksiyonu ile uygunluk fonksiyonu benzer kıyaslamalar için kullanılabilirler fakat aynı anlamda kullanılmamaktadırlar. Amaç fonksiyonu karar setine bağlı olarak çözümün minimizasyon problemleri için maliyetini, maksimizasyon problemleri için kârını verirken uygunluk fonksiyonu çözümün uygun olup olmadığı konusunda bilgi verir. Bu ayrım kısıtlı optimizasyon problemleri üzerinde oldukça açıktır. Minimizasyon problemlerini ele aldığımızda kısıtların sağlanmadığı bir çözümün amaç fonksiyon değeri küçük olmasına rağmen çözümün uygunluğu düşüktür çünkü kısıtlardan bir veya birkaçı sağlanamamıştır.

Bir optimizasyon probleminde birden fazla amaç fonksiyonu da bulunabilir. Örneğin bir aracın hedefe daha az zamanda gitmesi gerekirken daha az yakıt tüketmesi de istenebilir. Bu durumda iki amaç ortaya çıkmaktadır; birincisi daha az sürede hedefe varmak ve ikincisi daha az yakıt tüketmek. Aynı karar değişkeninin optimum değerinin bulunmasını temel alan bu tip birden fazla amacın olduğu optimizasyon problemlerine çok amaçlı (multi-objective) optimizasyon problemleri denir. Çok amaçlı optimizasyon problemlerinde amaçlar birbirleriyle çelişebilir. Örneğin amaçlardan biri minimize edilmeye çalışılırken, diğeri maksimize edilmeye çalışılabilir. Bu noktada amaçların ödünleşmesi söz konusu olabilir. Örnek üzerinden açıklamak gerekirse hızı arttırdığımızda süre kısalmasına rağmen yakıt tüketimi artacaktır, tersine hızı azalttığımızda ise yakıt tüketimi azalacak fakat süre uzayacaktır. Bu noktada iki amaç için de optimum değerin bulunabilmesi için karar değişkeninin değerinin iki amacı da

mümkün olduğunca sağlayacak şekilde tespit edilmesi gerekir. Çok amaçlı optimizasyon problemlerinin çözümü için farklı metotlar önerilebilir. İki amacın ağırlıklandırılmasıyla tek bir amaç elde edilebilir ve çok amaçlı optimizasyon problemi tek amaçlı bir hale dönüştürülebilir veya bir amaç diğeri için sınırlama olarak kullanılabilir.

1.1.3. Sınırlamalar

Karar değişkenlerinin alabileceği değerler kümesi kısıtlanabilir. Kısıtlar amaç fonksiyonu üzerinde etkiye sahip olmalıdır ve bunun için karar değişkeninin değerini etkilemelidirler. Eşitlik ve eşitsizlik sınırlaması olmak üzere iki tip sınırlama vardır. Optimizasyon problemi için optimal çözüm eşitlik ve/veya eşitsizlik kısıtlamalarını sağlayan çözümdür. Bazı kısıtlamalar fonksiyon olarak sunulabilirken bazıları karar değişkeninin alamayacağı değerler olarak verilebilir. Bir uzunluk veya ağırlık ölçüsü biriminin negatif değerler alamayacağı açıktır. Bir örnek vererek kısıtlamaları açıklamakta fayda vardır. Dikdörtgen biçimli bir odanın alanı 20000 m^2 olmak kaydıyla odanın dört tarafında 10x20m'lik koridorlar bulunmaktadır. Bu şartlar altından toplam alanı minimize edecek dikdörtgen kenar uzunluklarını bulunmak istendiğinde bu problem şu şekilde formüllenebilir.



Şekil 1.2. Optimizasyon problemi için sınırlamalar

Dikdörtgenin alanı iki dik kenarın çarpımına eşit olduğundan dolayı minimize edilmek istenen alan aşağıdaki şekilde bulunabilir:

$$\min(A) = (x + 40) \times (y + 20) \quad (1.8)$$

Yalnız alan minimize edilirken aşağıdaki şartında sağlanması gerekir:

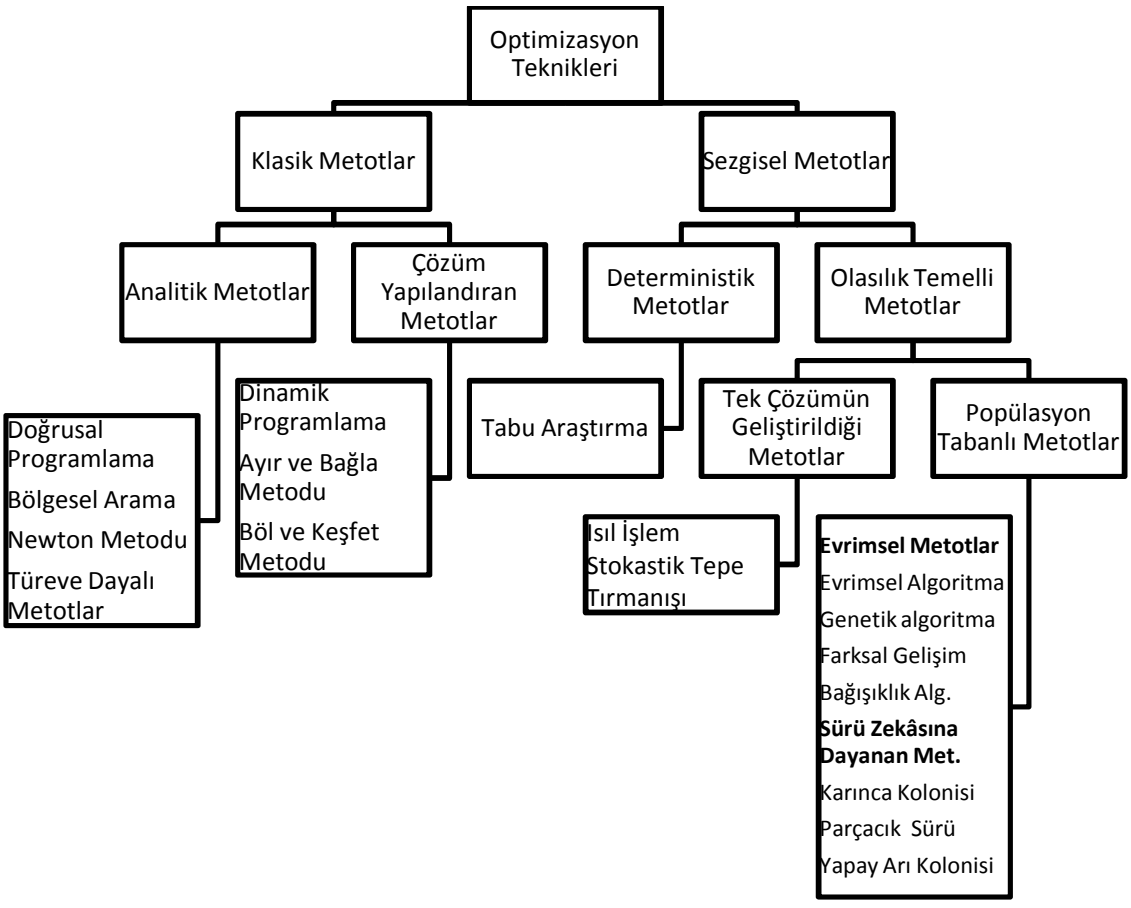
$$x \times y = 20000 \quad (1.9)$$

Denklem 1.8 amaç fonksiyonunu, Denklem 1.9 ise amaç fonksiyonunu karar değişkenleri üzerinden kısıtlayan eşitlik sınırlamasıdır. Ayrıca böyle bir durumda doğal olarak kenar uzunluklarının negatif olmaması gerekir. Bu kısıt da eşitsizlik kısıtı olarak düşünülebilir.

1.2. Optimizasyon yöntemleri

Bir optimizasyon yöntemi bir optimizasyon problemini çözen/çözebilen bir algoritma veya tekniktir. Optimizasyon yöntemi sadece belirli bir grup optimizasyon probleminin çözümüne yönelik olabileceği gibi geniş problem uzayına etkili bir yöntem de olabilir. Uygulayıcı veya araştırmacı elindeki optimizasyon problemini çözmek için uygulayacağı yöntemi kendine göre kıstaslarla belirlemek zorundadır. Ayrık bir problemi olan uygulayıcı tüm olası çözümleri deneyerek optimal sonucu elde edebileceği gibi problem boyutuna, süreye vb. durumlara bağlı olarak ihtimale dayanan bir tekniği kullanmak zorunda da kalabilir. Yani uygulayıcının optimizasyon problemini çözmek için karşılaması gereken bedeller vardır.

Optimizasyon yöntemlerini klasik teknikler ve sezgisel teknikler olmak üzere iki ana gruba ayırmıştık. Optimizasyon metotları alt dallarda da birçok sınıfa ayrılabilir olmasından dolayı burada tek tek açıklanmamıştır, fakat klasik optimizasyon metotları, evrimsel hesaplama teknikleri, sürü zekâsına dayanan yöntemler ve diğer yöntemlerin sınıflandırılması Şekil 1.3'de (Akay, 2009) verilmiştir.



Şekil 1.3. Optimizasyon metotlarının sınıflandırılması (Akay, 2009)

1.2.1. Klasik yöntemler

Klasik optimizasyon teknikleri sürekli ve türevlenebilir fonksiyonların optimum çözümlerini bulmada kullanışlı metotlardır. Fakat genellikle pratik uygulamalarda amaç fonksiyonu sürekli ve/veya türevlenebilir olmadığından dolayı bu teknikler kısıtlı bir uygulama sahasına sahiptirler fakat diğer optimizasyon metotları için de altyapı oluşturmaktadırlar. Klasik optimizasyon metotlarında elde edilen sonuçların optimum nokta olup olmadığına gerek ve yeter şartlar sınanarak karar verilir. Klasik yöntemlerin pratik uygulama sahalarının dar olmasının yanısıra aşağıdaki nedenlerden dolayı evrimsel hesaplama stratejileri veya sürü zekâsına dayanan yöntemlere ilgi artmıştır.

- Klasik optimizasyon tekniklerinin aksine evrimsel hesaplama veya sürü zekâsına dayanan metotlar yapılarında fazla bir değişikliğe ihtiyaç duymaksızın bir çok probleme uyarlanabilir.
- Konsept olarak basittirler ve türev vb. ağır matematiksel hesaplamalara ihtiyaç duymazlar.
- Herhangi bir uzman deneyimi olmayan problemlerin çözümü için uygulanabilirler. Fogel'e (1995) göre evrimsel teknikler problemin nasıl çözüleceğini bilmeden problemi çözerler.
- Multi-start yapıdadırlar ve aynı anda bir çok çözümden başlayarak çözüm uzayını araştırırlar ve paralel hesaplamaya olanak sağlarlar.
- Klasik tekniklerin aksine çevresel değişimlere hızlı adapte edilebilirler.
- *"Anlaşılabilirlik açısından sezgisel teknikler karar vericiler için çok daha basit olabilirler.*
- *Sezgisel teknikler kesin çözümü elde etmenin bir parçası olarak kullanılabilirler.*
- *Matematik formülleriyle yapılan tanımlamalarda genellikle gerçek dünya problemlerinin en zor tarafları (hangi amaçlar ve hangi sınırlar kullanılmalı, hangi alternatifler test edilmeli, problem verisi nasıl toplanmalı) ihmal edilir. Model parametrelerinin belirleme aşamasında kullanılan verinin hatalı olması, sezgisel yaklaşımın üretebileceği alt optimal çözümden daha büyük hatalara sebep olabilir."* (Karaboğa, 2011).

1.2.2. Evrimsel hesaplama teknikleri

Bu teknikler en uygun olan hayatta kalır prensibine dayanır ve evrim sürecinin bazı gerçek dünya problemlerini çözmek için modellenmesidir. Evrimsel hesaplama teknikleri formülizasyonlarına göre "Genetik Algoritmalar", "Evrimsel Programlama", "Evrim Stratejileri" ve "Genetik Programlama" gibi farklı gruplara ayrılır. Evrimsel metotlar tek bir çözüm yerine bir çözüm popülasyonu ile araştırma uzayının taranmasına başlar ve her çevrimde diğer çözümlere göre iyi olan çözümler bazı genetik operatörler kullanılmasıyla yeni neslin oluşturulması için kullanılır.

1.2.3. Sürü zekâsına dayalı yöntemler

Sürü zekâsı birlikte yaşayan bireylerin veya organizmaların bir problemi çözmek için ortaya koyduğu tecrübe ve bilgi birikimidir. Sürü zekâsını oluşturan bireyler merkezi bir hiyerarşi ile kontrol olunmazlar ve kollektif tecrübeye katkıda bulunurlar. Nitekim herhangi bir karıncanın geçtiği yol üzerine feromon adı verilen bir kimyasal madde bırakması diğer karıncaları etkileyerek ya yiyecek kaynağının ya da yuvanın bulunmasına yardımcı olması bakımından önemlidir. Birey olarak zeki olmayan canlıların sürü olarak hayatlarını idame ettirebilmeleri için ortaya koydukları iş bölümü, bilgi paylaşımı vb. zeki davranışları araştırmacıların ilgisini çekmiş ve çeşitli sürülerin bu davranışlarını temel alan yöntemler ile gerçek dünya problemlerini çözmeye çalışmışlardır. Karıncaların yiyecek kaynağı ve yuva arasındaki zeki davranışları (feromon bırakma, feromonu takip etme gibi) karınca kolonisi optimizasyon algoritmasının, kuş veya balık sürülerinin sosyal davranışları parçacık sürü optimizasyonunun ve bal arısı kolonilerinin yiyecek toplama ve bilgi paylaşımı davranışı da yapay arı kolonisi algoritmasının temelini oluşturur.

Sürü zekâsına dayanan yöntemler pozisyon güncellemeyi temel alırlar ve evrimsel hesaplama tekniklerindeki yeni bir nesil oluşturmak için gerekli operatörleri kullanmazlar. Bundan dolayı sürü zekâsı yöntemlerinde genetik operatörler (çaprazlama ve mutasyon) bulunmamaktadır. Bu operatörler yerine sürü zekâsına dayanan yöntemlerde uzayın verimli şekilde araştırılmasını sağlayacak pozisyon güncelleme teknikleri/kuralları/denklemleri bulunmaktadır.

1.2.3.1. Çözümün gösterimi

Probleme ait olası çözümlerin evrimsel veya sürü zekâsına dayanan yöntemlerin hesap yapabileceği şekilde kodlanması gerekmektedir. İkili kodlama, tam sayı kodlama, ağaç gösterimi, sürekli kodlama formları çözümün gösterimi için kullanılabilir. Optimizasyon probleminin yapısına bağlı olmak şartıyla bu kodlamalar arasında da dönüşüm yapılabilir. Örneğin ikili bir optimizasyon problemi için problemin uygun çözüm kodlaması ikili kodlama olarak görülmektedir fakat sürekli kodlama kullanılarak algoritma işlemlerini sürdürebilir ve amaç fonksiyonu değerlendirilmeden hemen önce ikili kodlamaya geçilebilir. Aynı şekilde sürekli bir gösterim yerine ikili gösterim kullanılarak (özellik genetik algoritma için) genetik operatörler koşturulabilir fakat amaç fonksiyon değerlendirilmeden önce kodlama sürekli hale getirilebilir. Kısacası çözümün gösterimi optimizasyon problemine bağlı olduğu kadar, yönteme ve geliştiricinin seçimine de bağlıdır. Bu noktada seçici yöntemin performansı iyileştiren kodlamayı tercih etmeye dikkat etmelidir.

1.2.3.2. Çözümün uygunluğu

Belirli bir gösterimle kodlanmış bireylerin çözüm kalitesini ölçmek amacıyla uygunluk fonksiyonu kullanılır. Optimizasyon probleminin amaç fonksiyonu çözümün uygunluğunu belirlemek amacıyla kullanılabileceği gibi yönteme özel bazı durumlardan (seçim mekanizmalarının çalışabilmesi gibi) dolayı amaç fonksiyonunu da kullanan farklı uygunluk fonksiyonları geliştirilebilir.

1.2.3.3. Popülasyon

Birden fazla uygun veya uygun olmayan çözümlerin oluşturduğu kümeye popülasyon adı verilir. Popülasyon kromozomlardan, parçacıklardan, yapay karıncalardan, yapay arılardan veya yiyecek kaynaklarından oluşabilir. Popülasyon yöntem içerisinde çözümlerin (karar setinin) tutulması için kullanılır. Popülasyon için bazı durumlar söz konusudur ve popülasyon yöntemin verimli çalışabilmesi için bu durumlardan kurtarılmalıdır. Örneğin popülasyon yeni çözüm üretemeyecek duruma (durağanlaşma – stagnation) gelebilir veya yerel minimumlara takılabilir. Etkili bir

yöntemde bu durumlarla başa çıkabilecek ve popülasyonun durumunu kontrol edecek mekanizmalar bulunmalıdır.

1.2.3.4. Seçim mekanizmaları

Araştırmanın sürdürülebilmesi yani aday çözümlerin veya yeni nesillerin oluşturulabilmesi için ebeveyn çözümlere ihtiyaç vardır. Yeni çözümlerin oluşturulabilmesi için hangi ebeveyn veya aktüel çözümlerin kullanılacağının belirlenmesi gerekmektedir. Bunun için seçim mekanizmaları uygulanır. Bu seçim mekanizmaları çözümün kalitesine bağlı olabileceği gibi rastgele de olabilir. Ayrıca bu seçim mekanizmaları komşu çözümlerin belirlenmesinde de kullanılabilir.

1.2.3.5. Yeni çözümün üretilmesi

Yeni çözümün elde edilmesi evrimsel tekniklerde çaprazlamayla, sürü zekâsına dayanan yöntemlerde ise pozisyon güncellemesiyle sağlanır. Evrimsel tekniklerde ise iyi çözümlerin çaprazlanmasıyla daha iyi çözümlerin elde edilmesi amaçlanır. Sürü zekâsına dayanan yöntemlerde ise popülasyondaki çözümlerin kullanılmasıyla (popülasyonun en iyi çözümü, o ana kadar elde edilmiş en iyi çözüm, birey tarafından elde edilmiş en iyi çözüm vb.) yeni çözümler elde edilir. Bunun için genellikle farka dayalı yöntemler kullanılmaktadır ve rastsallık da bu fark tabanlı işlemin içerisine entegre edilmiştir.

1.2.3.6. İlklendirme

Sezgisel tekniklerde ilklendirme genellikle rastgele yapılır ve basitçe hesaplanabilir. Yöntemlerin yetenekleri test edilirken başlangıç şartlarına bağlılığı da test edilmektedir. Bu bağlamda düşünüldüğünde verimli ve etkili bir yöntemin başlangıç durumlarına karşı duyarsız olması beklenmektedir, yani farklı başlangıç durumlarında da iyi bir çözümün elde edilebilmesini yöntem sağlamalıdır.

1.2.3.7. Sonlandırma

Sezgisel yöntemler algoritma olarak adlandırılmaktadır ve algoritmalar sonlu bir işlem kümesini ifade ederler. Bu bağlamda kara kutu optimizasyon (black-box optimization) problemleri için belirli bir çevrim sayısı veya amaç fonksiyonun değerlendirilme sayısı durdurma kriteri olarak kullanılabilir. Sezgisel yöntemler kullanılarak eğitilen bir sınıflandırıcıda ise sezgisel yöntemin durdurma kriteri olarak sınıflandırıcının eğitimdeki başarısı veya hatası kullanılabilir.

1.2.4. Performans değerlendirme kıstasları

Sezgisel yöntemlerle elde edilen sonuçların diğer yöntemlere göre daha iyi sonuç elde ettiği iddia edilmektedir. Elbette yeni üretilen bir yöntemin belirli kıyas testleri neticesinde diğerlerinden daha iyi sonuç vermesi gerektiği aşikârdır. Fakat daha iyi olmanın hangi kıstaslara göre olduğu da net olarak ortaya konulmalıdır. Bu kıstaslar aşağıda maddeler halinde verilmiştir.

Zaman Verimliliği: Sezgisel yöntemlerin kesin yöntemlere nazaran iki önemli avantajı vardır. Bunların birincisi bir çok problem türüne kolaylıkla uyarlanabilmeleri, ikincisi ise optimum veya optimuma yakın bir çözümü kısa zamanda elde edebilmeleridir. Süre ya da çalışma zamanı aynı zamanda metotların birbirleri ile kıyaslamalarında kullanılan temel ölçütlerden biridir.

Başarı: Diğer yöntemlere nazaran optimum veya optimuma ne kadar yakın bir çözümün elde edildiği, yani çözümün kalitesi yöntemlerin birbirleri ile kıyaslanması için kullanılmaktadır.

Gürbüzlük: Sezgisel yöntemin araştırma stratejisi ve kavramsal tasarımı gürbüz olmalıdır ve yöntem başlangıç şartlarına duyarsız olduğu kadar çevresel değişikliklere de uyum sağlayabilmelidir. Ayrıca yöntemin farklı problemler için başarılı sonuçlar elde etmesi ve çok sayıda çalıştırmadan elde edilen sonuçların birbirine yakın olması gürbüzlük kapsamında değerlendirilmektedir (Dorigo ve ark., 1996; Akay 2009).

Basitlik: Önerilen yöntem basit olmalıdır. Basitlik uygulama kolaylığı anlamına da gelmektedir ve yöntemin literatürde kabul görmesi için pozitif bir etkendir.

Adaptasyon: Yöntem birçok problem türüne uyarlanabilmelidir. Bu kıstas gürbüzlük performans kriteri altında da değerlendirilebilir.

Temel olarak performans değerlendirilmesinden sonra bilinmek istenen ve cevaplanması gereken sorular aşağıda sıralanmıştır (Akay, 2009).

- Algoritmanın koşturulması sonucu elde edilen çözüm ne kadar kaliteli?
- Bu çözüm ne kadar zamanda elde edildi?
- Yöntemin gürbüzlüğü nedir?
- Çok sayıda koşturma sonucunda elde edilen çözümler arasındaki uzaklık nedir?
- Çözümün kalitesi ile kabul edilebilir çözüm arasındaki ödünleşme nasıldır?

1.2.4.1. Çözüm kalitesi

Sezgisel yöntemler optimum çözümü garanti edemezler fakat belirli bir zamanda optimum veya optimuma yakın bir çözümü garanti edebilirler. Bu bağlamda düşünüldüğünde elde edilen çözümün veya çözümlerin kalitesi ölçülmelidir. Optimum çözüm biliniyorsa optimum çözüme olan uzaklık yüzde cinsinden (göreceli hata) verilir. Optimum çözüm yoksa problemin alt sınırına olan yakınlık çözümün kalitesine değerlendirmek için kullanılabilir.

1.2.4.2. Gürbüzlük

Sezgisel yöntemler için gürbüzlük konusu birkaç noktadan ele alınmıştır. Bunlar aşağıda sıralanmıştır.

Dorigo ve ark. (1996) önerdikleri karınca kolonisi optimizasyon algoritmasını gezgin satıcı ve karesel atama problemleri üzerinde test etmişlerdir. Bu problemlerin çözümüyle elde ettikleri sonuçların başarılı olduğunu ve yöntemin farklı problemler için başarısının iyi olmasından dolayı yöntemlerini gürbüz olarak tanımlamışlardır.

Karaboğa (2012) yapay arı kolonisi algoritmasının MATLAB kodunda yöntemin gürbüzlüğünü test etmek için algoritmanın birçok defa çalıştırılması gerektiğini ifade etmiştir. Yani yöntemin gürbüzlüğünü birçok çalıştırmadan (rastgele başlamak şartıyla) elde edilen sonuçların birbirine benzerliğine endekslemiştir.

Sezgisel yöntemler üzerine gürbüzlük düşünüldüğünde ise öncelikle gürbüzlüğün neye karşı olduğuna karar verilmelidir. Bu bağlamda Karaboğa'ya (2012) göre gürbüzlük farklı başlangıç şartlarına karşı yöntemin duyarsız olması ve iyi sonuçlar

elde edebilmesiyken, Dorigo ve arkadaşlarına (1996) göre ise yöntemi farklı problemlere uygulayabilme ve yöntemle başarılı sonuçlar elde edebilme olarak tanımlanmıştır. Bir diğer açıdan gürbüzlük amaç fonksiyondaki çevresel etkilere karşı yöntemin başarısı olarak da ele alınabilir. Sonuç olarak gürbüzlük, yöntemin birçok problem için birçok defa çalıştırılmasından elde edilen sonuçlar değerlendirildiğinde başarılı olup olmaması olarak düşünülebilir.

2. KAYNAK ARAŞTIRMASI-LİTERATÜR ÖZETİ

Sürü zekâsı terimi ilk olarak Beni ve Wang (1989) tarafından "Hücresel Robotlarda Sürü Zekâsı" adlı çalışmada kullanılmıştır. Sürü zekâsı kavramı 1989 yılında literatüre önerilmesine rağmen sosyal canlıların davranışlarını açıklamaya çalışan ilk ciddi çalışmalar Fransız zoolog Grassé tarafından yapılmıştır (Grassé, 1959; Aydın, 2011). Grassée çalışmasında sürülerin davranışını açıklarken *Stigmergy* kavramını kullanmıştır. Sosyal canlının rehberi yaptığı iştir yani işin kendisi işi icra eden (işçi) için bir yol göstericidir ve işin kendisinden sosyal canlı ek görevleri yerine getirmesi gerektiğini çıkarabilmektedir (Wilson, 1971). *Stigmergy* olarak adlandırılan bu kavram yuva inşasında çalışan karıncalar için işlerin koordinasyonu ve düzenlemesi karıncalar ve aralarındaki etkileşimden ziyade inşa edilen yuvanın o anki durumuna (şekline) bağlıdır. Stigmergy işçinin çevresini değiştirmesi ve değişen çevreden etkilenmesi olarak da açıklanabilir (Theraulaz ve Deneubourg, 1992). Aydın'a göre (2011) iş koordinasyonundaki mükemmellik işin bir plana bağlı olduğu sonucunu dahi doğurabilmektedir. Bu kavramın anlamı daha sonraları sürü davranışı olarak genişlemiştir (Theraulaz ve Deneubourg, 1992; Millonas, 1994; Krink, 2012).

Literatürde sürülerin çeşitli davranışlarını modelleyerek gerçek dünya problemleri için çözüm üreten sürü zekâsına dayanan bir çok optimizasyon yöntemi vardır. Örneğin, karıncaların yuva ile yiyecek kaynağı arasındaki hareketi – yiyecek arama davranışı üzerine kurulu karınca kolonisi optimizasyonu (Dorigo ve ark. 1991; 1996; Dorigo, 1992), kuş veya balık sürülerinin yiyecek araştırma ve bireyler arasındaki etkileşimler üzerine kurulu parçacık sürü optimizasyonu (Kennedy ve Eberhart, 1995), E.Coli bakterisinin besin arama davranışından esinlenilerek ortaya konulan bakteriyel besin arama optimizasyonu (Passino, 2002), bal arılarının yiyecek araştırması, bilgi paylaşımı, üreme süreci gibi zeki davranışları üzerine kurulu bal arısı evlilik optimizasyon yöntemi (Abbass, 2001), yapay arı kolonisi optimizasyon algoritması (Karaboğa, 2005), sanal arı algoritması (Yang, 2005), arı algoritması (Pham ve ark., 2005), arı kolonisi optimizasyonu (Teodorovic, 2005), kedi hareketlerine dayalı kedi sürüsü optimizasyonu (Chu ve ark., 2006; Chu ve Tsai, 2007), ateş böceklerinin davranışları üzerine kurulu ateş böceği sürü optimizasyonu (Krishnanand ve Ghose, 2005) ve ateş böceği algoritması (Yang, 2009), balık davranışlarını taklit eden yapay balık sürüsü algoritması (Jiang ve ark., 2009), kurt sürülerinin avlanma davranışı üzerine kurulu kurt kolonisi algoritması (Liu ve ark., 2011), guguk kuşlarının üreme ve

yumurta bırakma davranışından esinlenilerek geliştirilen guguk kuşu optimizasyon algoritması (Yang ve Deb, 2009), meyve sineklerinin yiyecek arama davranışından esinlenilerek oluşturulan meyve sineği optimizasyon algoritması (Pan, 2012) sürü zekâsına dayanan metotlardan bazılarıdır.

Hiç şüphesiz ki araştırmacılar veya uygulayıcılar tarafından en çok incelenen, iyileştirilen ve/veya uygulanan yöntemler karıncaların yiyecek arama davranışından esinlenilerek Dorigo ve ark. (1996) tarafından önerilen karınca kolonisi optimizasyonu, Eberhart ve Kennedy (1995) tarafından kuş ve balık sürülerinin sosyal davranışları üzerine kurulu parçacık sürü optimizasyonudur. Ayrıca son zamanlarda araştırmacıların ilgisini çeken bir diğer yöntem de Karaboğa tarafından geliştirilen yapay arı kolonisi algoritmasıdır. Arıların sürü davranışlarını temel alan son önerilen algoritma yapay arı kolonisi optimizasyonu olmasına rağmen arı davranışlarını temel alan çalışmaların %54'ü yapay arı kolonisi algoritmasının iyileştirilmesi ve/veya uygulaması üzerinedir (Karaboğa ve ark., 2012).

Bu tez çalışmasında aşağıda sayılan sebeplerden dolayı 3 yöntem ve bu yöntemlerin iyileştirilmesi üzerinde durulmuştur. Bu üç yöntem karınca kolonisi, yapay arı kolonisi ve parçacık sürü optimizasyon tekniklerdir.

i) Karınca kolonisi ayrık optimasyon problemleri için iyi çözümler üreten nadir sürü zekâsına dayanan optimizasyon metotlarından biridir.

ii) Parçacık sürü optimizasyonu kolay uygulanabilir ve uyarlanabilir bir yöntemdir.

iii) Hibritleştirme için sürüler arasındaki etkileşimi sağlayacak mekanizmalar bulunmaktadır.

iv) Literatürde en çok ilgi çeken yöntemlerdendir.

v) Yapay arı kolonisi algoritması diğer sürü zekâsı yöntemlerine göre sürekli optimizasyon problemlerini çözmekte oldukça başarılıdır.

vi) Yapay arı kolonisi algoritması diğer yöntemlere nispeten yeni olduğu için temel yöntemin iyileştirilmeye ve farklı tür optimizasyon problemlerini çözümü için uyarlanmaya ihtiyacı vardır.

vii) Karınca kolonisi algoritmasının yapıcı (constructive) olduğu baz alındığında farklı teknikleri ile birlikte kullanılarak hızlandırılmaya ihtiyacı vardır.

viii) Parçacık sürü optimizasyonunun lokal minimumlara takılmasının engellenmesi (her ne kadar üzerine bir çok iyileştirme yapılmış olmasına rağmen hâlâ bazı problemler için iyileştirilmesi) gerekmektedir.

ix) Bu yöntemlerin hibritleştirilmesi ve/veya hiyerarşik olarak kullanılması problemler için daha iyi sonuçlar elde edilmesini sağlayacaktır.

x) Basit yapıda olmalarından dolayı yapılacak iyileştirilmeler birçok uygulayıcının da işine yarayacaktır ve diğer araştırmacıların da dikkatini çekecektir.

Yukarıda sayılan 10 maddeden dolayı tez çalışmasında tüm sürü zekâsı algoritmalarının ki bu zaten doktora tez konusuna sığmayacak kadar geniş olurdu, yerine bu üç yöntem üzerine yoğunlaşma tercih edilmiştir.

Kaynak araştırması bu 3 yöntem üzerine yapılmış çalışmaları tarih sırasına göre özetleyecek şekilde verilmek yerine konu bütünlüğünü sağlamak amacıyla 5 bölüme ayrılmıştır. Yukarıda verilen birinci bölümünde sürü zekâsı ve yöntemleri üzerine kısa bir literatür özeti verilmiştir. İkinci bölümde karınca kolonisi optimizasyonu üzerine yapılan iyileştirmeler sunulacaktır. Üçüncü bölümde parçacık sürü optimizasyonun iyileştirilmesi için önerilen temel çalışmalar verilecektir. Dördüncü bölüm kapsamlı bir yapay arı kolonisi kaynak araştırması içerir ve beşinci bölümde bu yöntemlerin birbirleri ile ve diğer yöntemlerle hibritleştirilmesi için yapılmış bazı çalışmalar sunulmuştur.

2.1. Karınca Kolonisi Optimizasyonu Kaynak Araştırması

Karınca kolonisi optimizasyonu (ant colony optimiztion-ACO) karınca sistemi, karınca algoritması, karınca kolonisi sistemi vb. yöntemlerin genel adıdır (Aydın, 2011) ve genel karınca davranışını temel alan algoritmalar Tablo 2.1'de kronolojik sırada verilmiştir. Karınca kolonisi optimizasyonundaki karıncalar gerçek karıncalardan biraz farklıdırlar. Yapay karıncalar belirli bir hafızaya sahiptirler ve gerçek karıncalar gibi tamamen kör değildirler. Zaman doğadakinin aksine karınca kolonisi optimizasyonunda ayrıktır yani her bir adımda karıncalar bir noktadan diğer noktaya geçiş yapabilirler. Bu bağlamda genel karınca kolonisi optimizasyonu algoritması Algoritma 2.1'de sunulmuştur.

İlk önerilen karınca sistemi (Dorigo, 1992; Dorigo, Maniezzo ve Colorni, 1991; 1996) üç farklı model olarak önerilmiştir. Bunlar *karınca-yoğunluk (ant-density)*,

karınca-miktar (ant-quantity) ve *karınca-çevrim (ant-cycle)* modelleridir. Karınca-yoğunluk ve karınca-miktar modellerinde feromon güncellemesi kısmi çözüm parçaları oluşturulduktan hemen sonra yapılırken karınca-çevrim modelinde karınca tüm çözümü oluşturduktan sonra feromon güncellemesi yapılır. Karınca-çevrim modeli diğer iki modelden daha üstün performans gösterdiğinden dolayı karınca sistemi dendiğinde akla gelen yöntem karınca-çevrim modelini içeren metottur ve Algoritma 2.1'de gösterilen genel modelde karınca-çevrim modelidir (Dorigo ve Stützle, 2004).

Tablo 2.1. Karınca kolonisi optimizasyon yöntemleri kronolojik özeti

ACO Algoritması	**Yazarlar ve Yıl**
Ant System (AS)	Dorigo, 1992; Dorigo, Maniezzo ve Colorni, 1991; 1996
Elitist AS	Dorigo, 1992; Dorigo, Maniezzo ve Colorni, 1991; 1996
Ant-Q	Gambardella ve Dorigo, 1995; Dorigo ve Gambardella, 1996
Ant Colony System (ACS)	Dorigo ve Gambardella, 1997a; 1997b
Max-Min AS	Stützle ve Hoos, 1996; 2000
Rank-based AS	Bullnheimer, Hartl ve Strauss, 1997; 1999
ANTS	Maniezzo, 1999
Best-worst AS	Cordon, Vianna ve Moreno, 2000
TACO	Hiroyasu ve ark., 2000.
HC-ACO	Blum, Roli, ve Dorigo 2001; Blum ve Dorigo, 2004
P-ACO	Guntsch ve Middendorf, 2002a, 2002b
Beam-ACO	Blum, 2005
ACO_R	Socha ve Dorigo, 2008

Elitist karınca sistemi (Dorigo, 1992, Dorigo ve ark. 1991;1996) karınca çevrim modelini temel alır ve elitist stratejiyi kullanır. Bu modeldeki temel fikir o ana kadar elde edilen en iyi karınca çözümüne fazladan feromon depolanmasıdır. Elitist karınca sistemindeki feromon arttırımı yukarıda ifade edildiği gibi elitist stretajiye dayanırken feromon azaltımı sürecinde herhangi bir değişiklik yapılmamıştır. Elitist strateji ile karıncalar en iyi çözüme doğru güdülenerek en iyi çözümün etrafında bir araştırmaya sevkedilmekte ve bu sayede daha az iterasyon ile iyi çözümler elde edilmesi sağlanmaktadır.

Ant-Q algoritması Q-öğrenme'deki Q-değerlerinden etkilenilerek oluşturulmuş bir geçiş kuralına sahiptir. Belirli bir ihtimalle karıncaların yüksek kazançlı tarafa doğru yönelmeleri sağlanır. Bu ihtimal gerçekleşmediği takdirde karıncalar karınca sisteminin

geçiş kuralı ile hareket eder (Gambardella ve Dorigo, 1995; Dorigo ve Gambardella, 1996; Machado ve Schirru, 2002).

Algoritma 2.1. Genel Karınca Kolonisi Optimizasyon Algoritması

```
KKO için sözde kod
    Başlangıç parametrelerini ayarla ve yollara bir miktar feromon bırak
    while (Durdurma kriteri ile karşılaşılmadıysa) do
        Karıncaların turlarını oluştur.
        Karıncalar geçtiği yollardaki feromonu güncelleştir
    End While
```

Karınca koloni sisteminin (ant colony system) karınca sisteminden, geçiş kuralı, lokal feromon güncelleme ve o ana kadar ki en iyi çözüme ait feromon güncelleme olmak üzere üç temel farkı vardır. Karınca kolonisi sistemi geçiş kuralı olarak Ant-Q modelinin geçiş kuralını kullanır. Lokal feromon güncelleme karıncanın seçtiği yoldaki feromonu buharlaştırması ve bir miktar feromon bırakması olarak tanımlanabilir. Global feromon güncellemesi ise o ana kadar ki elde edilen en iyi çözüme göre yapılır. Temel olarak karınca koloni sistemi Ant-Q modeli üzerine kuruludur (Dorigo ve Stützle, 2004). İki yöntem arasındaki fark ise başlangıç feromon miktarının ayarlanmasındadır.

Max-Min karınca sistemi diğer karınca sistemi modellerindeki durağanlaşma davranışını (karıncaların sürekli en kısa görünen çözümü seçerek yeni farklı çözüm elde edememesi) engellemek için karıncanın çözümlerine bırakılacak feromon miktarı için alt ve üst sınır koyar. Alt ve üst sınırlar probleme göre değişiklik göstermekle birlikte gezgin satıcı problemi için kenar uzunluğu ile orantılı olarak belirlenir (Aydın, 2011). Ayrıca feromon ekleme işlemi sadece iterasyonun en iyi çözümüne veya o ana kadarki elde edilmiş en iyi çözüme göre yapılabilir. Stützle ve Hoos'un (2000) gezgin satıcı problemi üzerindeki deneysel çalışmalarından dinamik karışık stratejinin (belirli bir süre iterasyonun en iyi çözümü üzerine, belirli bir süre o ana kadar ki elde edilmiş en iyi çözüm üzerine) en iyilerden sadece bir tanesine feromon eklenmesinden daha performanslı olduğu görülmüştür.

Sıra tabanlı karınca sisteminde (Rank-based Ant System) karıncaların elde ettiği çözümler en iyiden en kötüye doğru sıralanır. Sadece belirli sayıda karınca çözümüne feromon ekleme işlemi yapılır ve bu ekleme işlemi de karıncanın sırasıyla orantılıdır. Ayrıca sıra tabanlı karınca sisteminde de o ana kadar ki elde edilen en iyi çözüm üzerine

en fazla feromon depolaması yapılmaktadır. (Bullnheimer, Hartl ve Strauss, 1997; 1999).

Karınca sisteminde kollektif zekâ ürünü olan yol üzerindeki feromona ve görülebilirlik (sezgisel) faktörlerine verilen önem değerleri üstel olarak ifade edilirken, Maniezzo (1999) tarafından önerilen ANTS algoritmasında çarpan haline getirilip birbiri ile ödünleşen (a ve 1-a şeklinde) bir faktör olarak görülmektedir. Ayrıca feromon güncelleme işleminde ANTS doğrusal dinamik ölçekleme ile yollara bırakılacak feromon miktarını hesaplar.

En iyi-en kötü (Best-worst) karınca sisteminde feromon güncellemesi en iyi yol üzerine feromon bırakılması ve en kötü yol üzerinden feromon buharlaştırılmasını esas alır (Cordon ve ark., 2000). En iyi ve en kötü çözümlerin çakıştığı kenarlarda ise feromon buharlaştırılması yapılmaz. Bu davranış iyi çözümler oluşturulmasını sağlamakla birlikte popülasyonun durağanlaşmasına ve farklılaşmanın sağlanmasına engel olur. Bu yan etkiyi önlemek amacıyla yollar için Genetik algoritmadaki mutasyon işlemine benzer bir süreç kenarlar için uygulanır. Ayrıca en iyi-en kötü karınca sistemi karınca kolonisi sisteminin geçiş kuralını kullanır.

Sürekli fonksiyonların optimizasyonu için Hroyasu ve ark. (2000) tarafından gezici karınca kolonisi optimizasyonu (touring ant colony optimization-TACO) yöntemi önerilmiştir. TACO yönteminde karar değişkenleri bit dizisi ile ifade edilmektedir ve karıncalar sadece feromon izini temel alarak hareket etmektedirler ve TACO'da sezgisel faktör yok sayılmıştır. HC-ACO algoritmasında da feromon değerlerinin bir vektör olarak ifade edilmesi ve [0,1] aralığına normalize edilmesi esas alınır (Blum, Roli, ve Dorigo 2001; Blum ve Dorigo, 2004). Bir diğer karınca kolonisi optimizasyonu iyileştirmesi de popülasyon tabanlı ACO (P-ACO) algoritmasıdır (Guntsch ve Middendorf, 2002a; 2002b). P-ACO temel ACO işlevlerinin yanı sıra bir popülasyon barındırır ve yeni çözümlerin bu popülasyonda bulunup bulunmamasına göre feromon güncellemesi yapılır. Popülasyona eklenen yeni bir çözüm feromon arttırırken popülasyondan çıkarılan bir çözüm feromon azaltılmasına neden olmaktadır (Angus, 2006, 2007). Karınca kolonisi optimizasyonu ile beam arama algoritması hibritleştirilerek Beam-ACO yöntemi önerilmiştir (Blum, 2005). Bu yöntemde her bir karınca her bir iterasyonda birden fazla çözümü ihtimalli beam araştırmayı kullanarak üretmektedirler. Sürekli optimizasyon problemlerinin çözümü için de çeşitli karınca tabanlı algoritmalar önerilmiştir. Bunlardan birisi Toksarı (2006) tarafından karıncaların en iyi karıncanın feromon bıraktığı çözüm etrafında araştırma yapmasını sağlayan bir

sürekli karınca kolonisi algoritmasıdır. Bir diğer sürekli karınca kolonisi optimizasyon yöntemi de ACO_R algoritmasıdır (Socha ve Dorigo, 2008). Karınca kolonisi optimizasyon ayrık türdeki problemlerin çözümü için önerildiğinden dolayı ACO_R algoritması için temel problemlerden birisi çözümün oluşturulması bir diğeri ise karıncalar tarafından problem komponentleri arasına bırakılan feromondur. Feromon matrisi yerine bir çözüm tablosu tutulmak suretiyle karıncaların bu tablodaki çözümleri kullanarak yeni çözümler elde etmesi ve çözümleri iyileştirmesi hedeflenmiştir.

Karınca sistemleri hakkında daha detaylı bilgi edinmek için Dorigo ve Stützle (2004) tarafından yazılmış "Ant Colony Optimization" kitabı incelenebilir. Ayrıca karınca kolonisi optimizasyonundaki parametre ayarlaması (Stützle ve ark., 2012), karınca kolonisi optimizasyonu ve veri madenciliği (Michelakos ve ark., 2011), son gelişmeler ve uygulamaları (Dorigo ve Stützle, 2010) hakkında bilgi almak için de ilgili referansları verilen çalışmalar incelenebilir.

Literatür taramasından görüldüğü üzere karınca kolonisi optimizasyon algoritmalarının temel problemi kollektif zekânın popülasyonun davranışını durağan hale getirmesidir. Literatüre önerilen birçok önemli iyileştirme (ACS, Max-Min ACS, Best-worst AS vb.) feromonun eklenmesi ve/veya artttırılması yani güncellenmesini düzenlemeye çalışmaktadır. Bir diğer önemli noktada ACO algoritmalarının yapıcı bir nitelikte olması (çözüm bileşenlerinin adım adım oluşturulması) çözüm iyileştirici (local araştırma metotları gibi) metotlara nazaran daha uzun bir çalışma süresi gerektirmektedir.

2.2. Parçacık Sürü Optimizasyonu Kaynak Özeti

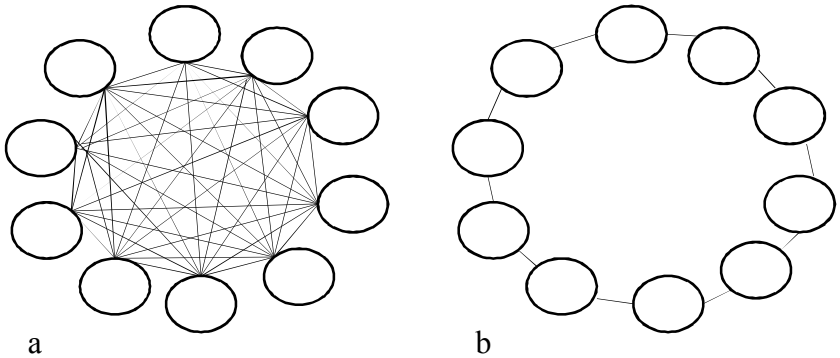
Parçacık sürü optimizasyonu (particle swarm optimization-PSO) sürekli optimizasyon problemleri (karar setinin sürekli değerler alabildiği) için efektif çözümler sunan popülasyon tabanlı iteratif bir sürü zekâsı algoritmasıdır ve kuş ve balık sürülerinin sosyal davranışlarının modellenmesiyle geliştirilmiştir (Eberhart ve Kennedy, 1995; Kennedy ve Eberhart, 1995). PSO yönteminde parçacık adı verilen potansiyel çözümler çözüm uzayında hareket ederek optimizasyon probleminin optimum çözümünü bulmaya çalışır. Parçacıkların her iterasyonda gideceği yer hızına göre belirlenir. Hız parçacığın geçmişte elde ettiği en iyi çözüm (*Pbest*) ile popülasyonun o ana kadar elde edilen en iyi çözümü (*Gbest*) kullanılarak hesaplanır (Eberhart ve Kennedy, 1995). Parçacığın her bir iterasyondaki pozisyonu optimizasyon

problemi için potansiyel bir çözümdür. PSO tekniğinin genel çerçevesi Algoritma 2.2'de sunulmuştur. *Pbest* ve *Gbest* değerleri güncellenirken parçacığın bulunduğu pozisyon yani çözümün kalitesi kullanılarak daha iyi olan *Pbest* ve *Gbest* olarak tayin edilir. Basit bir anlatımla minimizasyon problemi için daha küçük amaç fonksiyon değerine sahip olan çözüm diğerine tercih edilir.

Temel olarak iki PSO modeli vardır: global bilgiyi kullanan *gbest*-PSO ve bölgesel bilgiyi kullanan *lbest*-PSO (Eberhart ve Kennedy, 1995) ve bu iki model tam PSO olarak adlandırılır. İki model arasındaki temel farklılık komşuluk topolojilerinden kaynaklanmaktadır. Algoritma 2.2 tam PSO'nun algoritmasını vermektedir ve *gbest*-PSO için komşuluk topolojisi *yıldız* (*star)* topolojisi (Şekil 2.1-a) iken *lbest*-PSO *halka (ring)* topolojisini (Şekil 2.1-b) kullanmaktadır. Ayrıca iki boyutlu vektör uzayında bir parçacığın hareketini etkileyen bir önceki hızı, sosyal bileşen *Gbest* ve bilişsel bileşen *Pbest* Şekil 2.2'de gösterilmiştir.

Algoritma 2.2. Parçacık Sürü Optimizasyonu Algoritmasının Genel Çerçevesi

```
Başlangıç parametrelerini ayarla ve parçacıkları uzaya dağıt
Popülasyon için Gbest ve her parçacığın kendisi için Pbest değerlerine karar ver
while (Durdurma kriteri ile karşılaşılmadıysa) do
        Her bir parçacık için
                Gbest ve parçacığın Pbest değerini ile yeni pozisyonu hesapla
                Parçacığı yeni pozisyonuna konumlandır.
                Parçacıkların Pbest değerini güncelle
        Popülasyonun Gbest değerini güncelle
End While
```

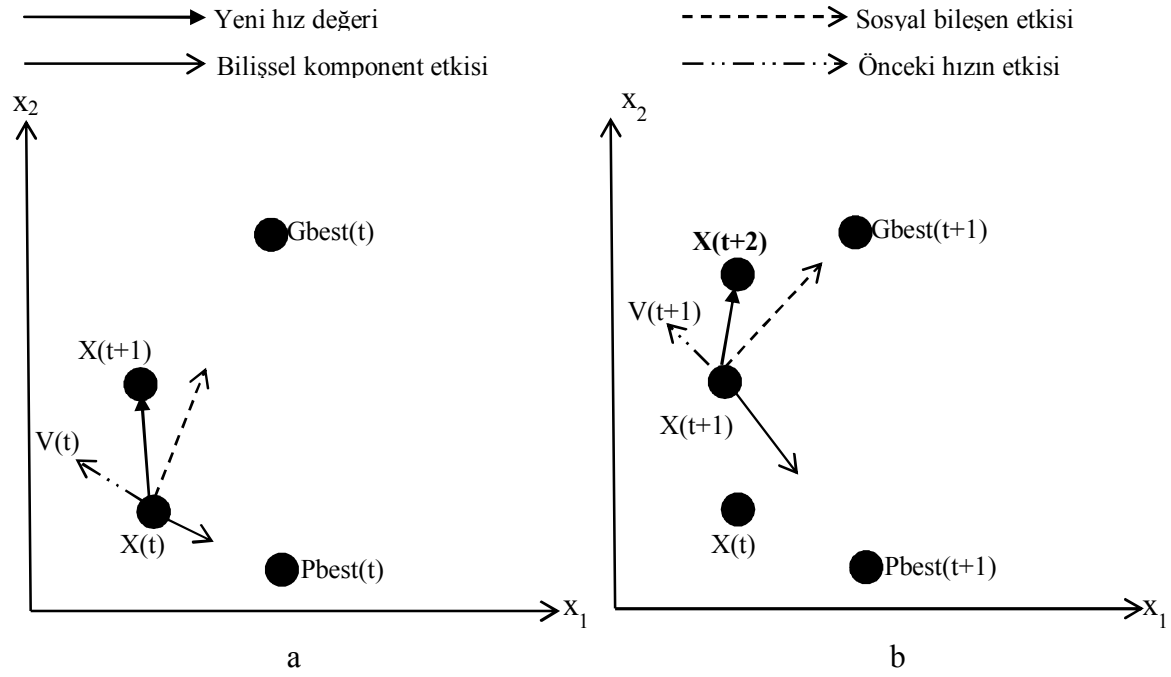


a b

Şekil 2.1. Tam PSO'nun Komşuluk Topolojileri

a) Gbest topolojisi b) LbestTopolojisi

Algoritmanın genel çerçevesinde bir değişim yoktur çünkü *gbest*-PSO'da *Gbest* olarak belirlenen sosyal parametre *lbest-PSO'da lbest* olarak işaretlenmektedir. PSO'nun lokal ve global varyantları araştırma stratejisi bakımından farklı özelliktedir. Lokal varyant bölgesel araştırmayla tüm çözüm uzayını verimli şekilde araştırmayı hedeflerken global varyant daha hızlı şekilde iyi bir çözüme yakınlaşmaktadır. Global ve lokal varyantın bu özelliklerini birleştiren bir PSO modeli de literatürde önerilmiştir (Parsopoulos ve Vrahatis, 2004).



Şekil 2.2. Parçacık hareketinin iki boyutlu uzayda geometrik gösterimi

a) t zamanında b) t+1 zamanında

Literatür özetindeki kavramların açıklanması amacıyla PSO yönteminin hız denklemindeki bileşenler ve isimleri İngilizce karşılıklarıyla aşağıda verilmiştir.

$$v_{i,j}(t) = \omega \times v_{i,j}(t) + \phi_1 \times r_{1,j}(t) \times \left[pbest_{i,j}(t) - x_{i,j}(t) \right] + \phi_2 \times r_{2,j}(t) \times \left[gbest_j(t) - x_{i,j}(t) \right]$$

$\omega \times v_{i,j}(t)$: Atalet terimi (Inertia term)

$v_{i,j}(t)$: t zamanındaki atalet

ω : atalet ağırlığı

$\phi_1 \times r_{1,j}(t) \times \left[pbest_{i,j}(t) - x_{i,j}(t) \right]$: Bilişsel bileşen terimi (Cognitive component)

ϕ_1: Bilişsel bileşenin ivme katsayısı (Accelaration coefficient of cognitive component)

$r_{1,j}(t)$: Bilişsel etkinin (0,1) aralığında üretilen rastgele sayı bileşeni (stochastic element of cognitive component)

$\left[pbest_{i,j}(t) - x_{i,j}(t) \right]$: Bilişsel etki (Cognitive influence)

$\phi_2 \times r_{2,j}(t) \times \left[gbest_j(t) - x_{i,j}(t) \right]$: Sosyal bileşen terimi (Social Component)

Sosyal bileşenin terimleri bilişsel bileşenin terimleri ile aynı anlamda olduğu için tekrar yazılmamıştır.

PSO algoritmasının hız denkleminden sosyal bileşeni çıkarırsak sadece bilişsel model (cognitive-only) kalır. Benzer şekilde bilişsel komponent çıkarıldığında ise yeni modelin adı sadece sosyal model (social-only) olarak adlandırılır (Kennedy, 1997). Bilişsel model parçacığın hareketine sadece geçmişinde elde edilen tecrübenin etkisini ifade ederken, sosyal model parçacığın hareketine sadece popülasyonun en iyi çözümüne göre karar verileceği anlamına gelir. Ayrıca iyi bir çözüme ulaşmak için sadece bilişsel modelin daha fazla çevrime ihtiyaç duyduğu ve sadece sosyal modelin sadece bilişsel modelden ve tam PSO modellerinden daha verimli ve hızlı olduğu literatürde raporlanmıştır (Kennedy, 1997; Carlisle ve Dozier, 2000).

Parçacıkların hızlarının çabucak büyük değerlere ulaşması hem parçacıkların aşırı farklılaşmasına hem de çözüm uzayının dışına çıkmasına sebep olmaktadır (Engelbrecht, 2005). Parçacıkların uzaydaki araştırmasını kontrol etmek amacıyla hızın etkisini dengelemek amacıyla atalet ağırlığı (*inertia weight)* adı verilen bir parametre PSO'nun araştırma denkleminde hız terimine katsayı olarak eklenmiştir (Shi ve Eberhart, 1998). Bununla birlikte atalet ağırlık parametresinin bir diğer amacı da hız için sıkıştırmanın önemi azaltmaktır. Hızın ayarlanması için bulanık sistem yaklaşımı (Eberhart ve Shi, 2000), rastgele bir aralıkta atalet ağırlığı seçimi yaklaşımı (Eberhart ve Shi, 2001), ataletin arttırılması için bir yaklaşım (Zheng ve ark., 2003), atalet ağırlığının dinamik azaltılması ve hız üst sınırı azaltılması yaklaşımları (Fourie ve Groenwold, 2002) önerilmiştir. PSO algoritması hız değerlerinin belirli bir aralığı kısıtlanmadan koşturulması durumunda çok hızlı bir şekilde araştırma uzayının dışına çıkmaktadır (Poli ve ark., 2007). İvme katsayılarından kaynaklanan bu durum için Clerc ve Kennedy (2002) bilişsel ve sosyal bileşenlerin ivme katsayılarını analiz ederek bilişsel bileşenin ivme katsayısı olarak 0.7298, sosyal bileşenin ivme katsayısı olarak 1.49618 değerlerini

önermişlerdir. Ayrıca Eberhart ve Shi (2000) hız sınırları için araştırma uzayı sınırlarını bağlı dinamik bir aralık önermişlerdir. Bazı sosyal komşuluk yapılarının (topolojiler) PSO'nun performansına etkileri Kennedy (1999), Kennedy ve Mendes (2002) ve Mendes (2004) tarafından analiz edilmiştir. Mendes (2004) sosyal komşuluk yapısının anlamını ve etkilerini parçacıkların etkileşimindeki tarza bağlı olduğunu vurgulamıştır. *Lbest* PSO'nun global araştırma ve *Gbest* PSO'nun hızlı yakınsama özelliklerini birleştirerek PSO'nun performansını iyileştirmek amacıyla dinamik bir komşuluk yapısı Suganthan (1999) tarafından önerilmiştir. Bu komşuluk yapısında halka komşuluk yapısından yıldız komşuluk yapısına yavaşça geçiş temel alınmaktadır. Temel PSO yönteminde parçacık, önceki hız, geçmişte elde ettiği en iyi değer ve popülasyondaki en iyi çözüm bileşenlerinden etkilenmektedir ve diğer parçacıkların bir parçacığın hareketine doğrudan herhangi bir etkisi bulunmamaktadır. Kennedy ve Mendes (2002) geleneksel yöntemin aksine bir parçacığın tüm parçacıklardan etkilenmesini sağlayarak tam haberdar (fully informed) PSO yöntemini önermişlerdir (Mendes ve ark., 2002;2003). Peram ve ark. (2003) parçacıkların tüm parçacıklardan veya sadece bir (*lbest* veya *Gbest*) parçacıktan etkilenmesinin yerine parçacıkların uygunluk değerlerine oranla yakınındaki parçacıklardan etkilenmesi sağlamak amacıyla uygunluk-mesafe-oranlı PSO (Fitness-Distance-Ratio PSO, FDR-PSO) yöntemini geliştirmişlerdir. Liang ve Suganthan (2005) tüm popülasyondan rastgele alt popülasyonlar elde ederek ve bu alt popülasyondaki tüm komşuluk yapılarını rastgele hale getirerek PSO'da farklılaşmayı sağlamışlar ve PSO'nun multimodal problemler üzerindeki başarısını arttırmışlardır. Popülasyondaki parçacıkların uygunluk değerlerini kullanarak bir hiyerarşik ağaç yapısında düzenlenmesi ve alttaki parçacıkların hiyerarşide üstte olan parçacıklardan daha fazla etkilenmesi sağlanmıştır (Janson ve Middendorf, 2005) ve önerilen yöntemin performansı diğer PSO varyantlarıyla kıyas fonksiyonları üzerinde test edilmiştir.

1997 yılında PSO yöntemi basit bir değişiklikle ayrık problemleri de çözecek şekilde modifiye edilmiştir (Kennedy ve Eberhart, 1997). Modifikasyonun temelinde parçacıkların sürekli değerlerin yerine ikili değerleri tutması ve hızın bir geçiş fonksiyonuna tabi tutulması vardır. Parçacığın her karar değişkeni için bir rastgele sayı üretilmekte, rastgele sayı hız (eşik) değerinden küçük ise parçacığın ilgili boyutu 1 değilse 0 değerini almaktadır. Geliştirilen ayrık ikili parçacık sürü optimizasyonunun (discrete binary particle swarm optimization-BPSO) performansı bazı genetik algoritmaların performansı ile kıyaslanmış ve daha iyi sonuçlar elde edildiği

raporlanmıştır (Kennedy ve Spears, 1998). İkili PSO için alternatif bir hız denklemi geliştiren Al-kazemi ve Mohan (2002) çok fazlı ayrık PSO adını verdikleri bir yöntem geliştirmişlerdir (M-DiPSO). M-DiPSO sadece daha iyi çözümleri hafızada tutmaktadır ve parçacığın hızını, parçacığın kendisini ve global en iyi çözümü kullanarak (her birisi için [-1,+1] aralığında rastgele sayılar vardır) parçacık için yeni bir pozisyon üretmektedir. PSO'nun bir başka ikili versiyonu özellik seçimi için kullanılmıştır. Bu versiyonda parçacıklar sürekli değerleri tutmaya devam etmektedir ve ikili vektör daha sonra elde edilmektedir. Parçacıklardaki sürekli değerler ikili vektörü elde etmek için ihtimal değeri olarak kullanılmaktadır ve ihtimal için rulet seçimi önerilmiştir (Cedeño ve Agrafiotis, 2002; Agrafiotis ve Cedeño, 2002).

Erken yakınsama ve lokal minimaya takılma PSO için büyük bir problemdir. Popülasyondaki farklılaşmayı arttırmak yöntemin global araştırma yeteneğini arttıracağından dolayı PSO'nun bir çok yeniden başlatmalı varyantı literatürde önerilmiştir. Bu varyantlar şunlardır; sabit bir süre sonunda PSO'nun yeniden başlatılması (Xie ve ark., 2002a), belirli bir ihtimale dayalı pozisyonların ve hızların yeniden başlatılması (Xie ve ark, 2002b) ve sadece hızların yeniden başlatılması (Schutte ve Groenwold, 2003;2005). PSO n-boyutlu vektörlerin bir popülasyonunu tutmaktadır. Eğer n-boyut 1 boyutlu olarak alt gruplara ayrılırsa o zaman n tane alt sürü elde edilmiş olur ki bu ayrışımı kullanan PSO varyantı işbirlikçi (cooperative PSO-CPSO) PSO modelidir (van den Bergh ve Engelbrecht, 2004). Uygunluk fonksiyonunun değerlendirilmesi için bu alt sürülerin içeriklerinin uygun şekilde birleştirilmesi gerekir. Temel PSO yönteminde 30 boyutlu bir problem için 30 parçacık kullanıldığında bir iterasyonda 30 farklı çözüm elde edilirken CPSO'de 30x30 çözüm elde edilebilmektedir. Bu kombinasyon artışı çözümlerin farklılaşmasını sağlayarak erken yakınsamaya neden olmakta ve yöntemin lokal minimaya takılmasını da engellemektedir (van den Bergh ve Engelbrecht, 2004). PSO'nun hız denkleminde sadece bilişsel faktörü bırakarak Lin ve ark (2006) kapsamlı bir öğrenme stratejisine sahip PSO türevini (comprehensive learning PSO-CLPSO) önermişlerdir. CLPSO yöntemi bilişsel olarak tüm parçacıkların geçmişinde elde ettiği çözümleri diğer parçacıklarla paylaşmayı temel alır ve bunlar arasından belirli bir ihtimalle birini seçerek parçacığın bundan etkilenmesi sağlanır. PSO'nun hız denklemindeki sosyal bileşende global en iyi çözümün yerine sürünün ortasındaki çözümü kullanmayı öneren bir PSO modeli Liu ve ark. (2007) tarafından literatüre sunulmuştur. Ayrıca Cai ve ark. (2008) PSO'da sosyal bileşenin ivme katsayısını çözümün kalitesine bağlı ayarlayarak

yeni bir PSO modeli önermişlerdir. PSO algoritmasında çalışmanın sonuna doğru popülasyondaki farklılık kaybolmaktadır. Bundan dolayı büyük popülasyon boyutunun başlangıçta yüksek sona doğru düşük olması çözüm uzayının farklı alanlarını taramak adına verimlidir. Bu bağlamda popülasyondaki farklılaşmaya göre popülasyon boyutunu otomatik ayarlayan PSO varyantı Chen ve Zhao (2009) tarafından önerilmiştir. Temel PSO'da yüksek hızlı erken yakınsama ve hızlı şekilde farklılaşmanın kaybolmasından dolayı dağıtılmış parçacık sürü algoritması Xinchao (2010) tarafından temel PSO'nun bahsedilen davranışlarını düzenlemek adına önerilmiştir. Ortogonal öğrenmeli (Zhan ve ark., 2011) ve artan sosyal öğrenmeli PSO (de Oca ve ark., 2011) algoritmaları da literatürde son zamanlarda önerilen PSO varyantlarıdır. PSO'nun yakınsama, lokal ve global araştırma kapasitesini iyileştirmek amacıyla Li ve ark. (2012) farklı hız güncellemelerini içeren bir model önermişlerdir. Önerilen modelde 4 farklı hız güncelleme denklemi bulunmaktadır ve her hız güncelleme denklemi yöntem için yakınsama, global araştırma, yerel araştırma gibi avantajlar sağlamaktadır. Bu literatür özetine ek olarak PSO algoritmasının iyileştirmeleri ve uygulamaları için Engelbrecht (2005) ve Poli ve ark. (2007) tarafından hazırlanan çalışmalara bakılabilir.

2.3. Yapay Arı Kolonisi Literatür Araştırması

Yapay arı kolonisi (Artificial bee colony-ABC)) optimizasyon algoritması Karaboğa (2005) tarafından sürekli optimizasyon problemlerinin çözümü için önerilmiştir. ABC algoritması bal arılarının yiyecek arama ve kaynakların pozisyon bilgilerini paylaşmak için kullandığı sallanım dansını temel alır. Optimizasyon probleminin olası çözümleri yiyecek kaynaklarının pozisyonları olarak düşünülür ve görevlerine göre üç tip arı yiyecek kaynaklarının keşfi ve iyileştirilmesi için çalışır. Arama sürecinin başlangıcında arı popülasyonun yarısı kâşif arıdır. Bu kâşif arılara yiyecek kaynağı pozisyonları rastgele atandıktan sonra görevli arı (employed bee) adını almaktadırlar. Görevli arılar iterasyonlar boyunca daha iyi yiyecek kaynakları bulmaya çalışırlar. Popülasyonun diğer yarısı ise gözcü (onlooker bee) arılardan oluşmaktadır ve bu arılar kovanda bekleyerek görevli arıların yiyecek kaynakları hakkındaki pozisyon bilgisini paylaşmasını beklerler. Kaynakların bilgisini alan gözcü arılar yiyecek kaynağının kalitesine bağlı olarak bir görevli arıya atanmış olan kaynağı iyileştirmeye çalışır. Eğer herhangi bir yiyecek kaynağı belirli bir zaman (ki bu süre *limit* olarak tanımlanır ve ABC için bir parametredir) süresince iyileştirilemezse bu kaynağın

görevli arısı kâşif arı olur. Kendisine rastgele bir yiyecek kaynağı atanan kâşif arı tekrar görevli arı olur (Akay, 2009). Arıların yiyecek arama davranışını temel alan ABC algoritması Algoritma 2.3.'te verilmiştir.

Algoritma 2.3. Yapay Arı Kolonisi Algoritmasının Genel Çerçevesi

```
Başlangıç parametrelerini ayarla ve görevli arılar için yiyecek kaynakları üret
Yiyecek kaynaklarının kalitesini hesapla
while (Durdurma kriteri ile karşılaşılmadıysa) do
        Her görevli arı için
                Yeni bir yiyecek kaynağı üret
                Üretilen kaynağın kalitesini hesapla
                Kaynakların kalitesine bağlı olarak aç gözlü seçimi uygula
        Yiyecek kaynaklarının kalitelerini bağlı seçilme ihtimalini hesapla
        Her gözcü arı için
                Bir görevli arının yiyecek kaynağını ezberle
                Bu kaynağının etrafında yeni bir yiyecek kaynağı üret
                Yeni kaynağın kalitesini hesapla
                Kaynakların kalitesini bağlı olarak açgözlü seçimi uygula
        Eğer kolonide kaşif arı varsa
                Rastgele bir yiyecek kaynağı üret
                Kaşif arıya üretilen kaynağı ata
                Kaynağın kalitesini hesapla
        Popülasyondaki o ana kadarki en iyi çözümü kaydet.
End While
```

Yukarıda verilmiş olan arı davranışlarından esinlenilerek oluşturulan ABC algoritmasını Karaboğa ve Baştürk (2007) nümerik fonksiyonların optimizasyonu için kullanmışlardır ve ABC'nin performansını genetik algoritma, parçacık sürü optimizasyonu ve parçacık sürü optimizasyonu ile evrimsel hesaplamanın hibritleştirildiği bir yöntem ile kıyaslamışlardır. Karaboğa ve Baştürk (2008) bir başka çalışmalarında ABC'yi çok boyutlu nümerik fonksiyonlarının optimizasyonu için kullanmış ve yöntemin başarısını farksal gelişim, parçacık sürü optimizasyonu ve evrimsel algoritma ile kıyaslamışlardır. Ayrıca yöntem birçok test fonksiyonunun optimizasyonu için kullanılmış ve diğer yöntemlerle başarısı kıyaslanmıştır (Karaboğa ve Akay, 2009; 2009a). Zho ve Kwong (2010) ABC'nin araştırma stratejisine global en iyi çözümün etkisini ekleyerek yöntemin araştırma yeteneğini arttırmak için GABC yöntemini önermiştir. Yöntemin yakınsama yeteneğini arttırmak amacıyla ölçekleme faktörünün hangi aralıkta olması gerektiği analiz edilmiş ve yönteme değişim parametresi (modification rate-MR) adında bir parametre eklenmiştir (Akay ve Karaboğa, 2010). ABC'nin yavaş yakınsaması ve lokal minimumlara takılmasından

dolayı birden fazla sürünün elde ettiği çözümlerin birleştirilmesine dayalı ABC metodu Wu ve ark. (2011) tarafından önerilmiştir. Levy ihtimal dağılımını kullanan L-ABC temel yöntemin yerel araştırmadaki eksikliğini gidermeyi amaçlar (Rajasekhar ve ark., 2011). El-Abd (2012) ABC ve diğer pozisyon güncellemeye dayalı tekniklerinin evrimsel yöntemlere dayalı teknikler ile arasındaki performans değerlemesini yaparak yöntemlerin karşılaştırmalarını literatüre sunmuştur.

ABC'nin sürekli çözüm uzayına sahip optimizasyon problemlerindeki yüksek performansı araştırmacıların ilgisini çekmiş ve ikili ve/veya ayrık problemlerin çözümü için ABC algoritması uyarlanmıştır. Akay ve Karaboğa (2009) tamsayılı programlama problemlerinin çözümü için ABC'yi modifiye etmişlerdir. İlgili yöntemde ABC'nin ajanları sürekli uzayda pozisyon araştırması yapmaktadır fakat problemin tamsayı kısıtı ile başa çıkmak amacıyla elde edilen yeni çözümlerdeki sürekli değerler en yakın tamsayı'ya yuvarlanmaktadır. İkili optimizasyon problemlerinde karar parametreleri 0 veya 1 değerini alabilmektedir. Bu bağlamda özellik seçme (feature selection) bir ikili optimizasyon problemi olarak ele alınabilir. Wang ve ark. (2010) gerçek zamanlı saldırıları tespit edebilmek amacıyla ABC'yi hem destek vektör makinalarının (support vector machines) parametrelerini optimize etmek amacıyla hem de BABC olarak adlandırdıkları ABC'ye dayanan bir yöntemi destek vektör makinalarının hangi özellikleri kullanarak saldırıyı tespit edeceği konusunda genişletmişlerdir. BABC yöntemi ABC'nin tüm konseptini korumasının yanı sıra sürekli değerlerin ikili değerlere dönüştürülmesi noktasında ikili parçacık sürü optimizasyonunda kullanılan geçiş kurallarını kullanır. Sürekli değerler sigmoid fonksiyonundan geçirilir ve fonksiyonun çıkışında elde edilen sonuç ilgili karar değişkenin 0 ya da 1 olacağı konusunda rastgele üretilen bir sayı için eşik olarak kullanılır. Benzer şekilde Kashan ve ark. (2011) temel ABC'nin çözüm adımlarını korumasına rağmen araştırma ve yeni çözüm elde etme stratejisini yiyecek kaynağı ile popülasyondan rastgele seçilen yiyecek kaynağı arasındaki benzerlik-benzersizlik ve miras alma-mirastan farklı hareket etmeye dayalı bir yöntem geliştirmişlerdir ve geliştirdikleri yöntemi ikili bir optimizasyon problemi olan kapasitesiz tesis yerleşimi problemlerini çözmek için kullanmışlardır. Aynı problemin çözümü için Kıran ve Gündüz (2013) binABC adını verdikleri temel ABC'nin tüm yapısını koruyan bir yöntem önermişlerdir. Önerilen yöntemde ABC'nin sürekli değil ikili çözüm uzayında hareket edecek şekilde XOR lojik operatörüyle araştırma stratejisi güncellenmiştir.

Ayrıca Karaboğa ve Görkemli (2011) ABC algoritmasını mutasyon operatörü tabanlı bir güncelleme yöntemi ile ayrık bir problem olan gezgin satıcı problemini çözecek şekilde uyarlamışlardır. Modifiye edilmiş en yakın komşu tekniği ile *inver-over* (Tao ve Michalewicz, 1998) operasyonu tabanlı bir ABC yöntemi de gezgin satıcı probleminin çözümü için uygulanmıştır (Li ve ark., 2011). Szeto ve ark. (2011) bazı komşuluk operatörlerini kullanarak ABC yönteminin ayrık versiyonunu geliştirmişler ve araç rotalama problemi üzerinde önerdikleri yöntemin başarısını araştırmışlardır. Ayrık ABC metodu Kıran ve ark. (2013) tarafından gezgin satıcı problemini çözmek için kullanmışlar ve bazı operatörlerin kombinasyonu ile yeni operatörler elde ederek algoritmanın farklı şartlar altındaki performans değerlemesini yapmışlardır.

Algoritmanın çeşitli parametrelerinin ve mekanizmalarının incelenmesi için de çeşitli çalışmalar yapılmıştır. Akay ve Karaboğa (2009a) yöntemin parametrelerinin performansına etkisini araştırmışlardır ve gözcü arıları için çeşitli seçim mekanizmalarının incelemesi Bao ve Zeng (2009) tarafından yapılmıştır. Aderhold ve ark. (2010) popülasyon boyutunun ABC'nin çalışmasına etkileri araştırmışlar, ABC'nin araştırma stratejisine global en iyi çözüme ait bilgiyi dahil etmişler ve seçimi uzaklığa dayalı hale getirmişlerdir. ABC'nin araştırma denklemine global en iyi ve arının geçmişte elde ettiği en iyi çözümler eklenerek GABC adında yeni bir yöntem geliştirilmiştir (Guo ve ark., 2011). Bu yöntemde kaşif arılar için üretilen yiyecek kaynakları için de bireyin geçmişinde elde ettiği en iyi çözüme ve iterasyon zamanına bağlı bir strateji bulunmaktadır. Ayrıca yeni bir farklılık stratejisi (Lee and Chai, 2011), yapay arıların pozisyon güncellemesini için yeni yaklaşımlar (Diwold ve ark., 2011), Von Neumann komşuluk topolojisine dayalı yeni ABC varyantları literatürde önerilmiştir (Zou ve ark., 2011). Kıran ve Gündüz (2012) genetik bir operatör olan çaprazlama operatörünü gözcü arılar için komşu seçiminde kullanmışlardır. Yapay arı kolonisi programlama yaklaşımı sembolik regresyon için Karaboğa ve ark. (2012a) tarafından geliştirilmiştir. Gao ve Liu (2012) ABC'nin lokal araştırma yeteneğini ve yakınsama hızını arttırmak için çalışmalar yapmıştır. Popülasyonun başlatılması evrimsel ve sürü zekâsı tekniklerinde yöntemin yakınsama kabiliyetine etkisi oldukça yüksektir. ABC'nin rastgele başlatılması yerine kaotik sistemler ve zıt-tabanlı öğrenme tekniklerini ABC popülasyonunun başlatılması için kullanmışlardır. Lokal araştırma yeteneğini arttırmak amacıyla da farksal gelişim algoritmasının araştırma denklemleri ile ABC'nin araştırma denklemlerinin yerine kullanmışlardır (Gao ve ark., 2012). ABC'nin araştırma denklemi yerine PSO'nun araştırma denklemine benzer bir yapı

kurularak ABC'nin lokal araştırma yeteneği arttırılmaya çalışılmıştır (Li ve ark., 2012). Komşuluk operatörleri ABC'nin araştırma denkleminin yerine mutasyon için kullanılarak ayrık bir ABC algoritması Li ve Yin (2012) tarafından önerilmiş ve çizelgeleme için kullanılmıştır.

Yukarıda verilen ABC yönteminin iyileştirmelerin yanı sıra uygulamalarının sayısı da oldukça fazladır ve detaylı bir özet için (Karaboğa ve ark., 2012) referansına bakılabilir. Literatür taramasında görülmüştür ki ABC'nin optimizasyon problemlerinin çözümü için uygulanması PSO'nun zaman içerisindeki uygulamaları ile paralellik göstermektedir. Buradan da anlaşılmaktadır ki birçok sürü zekâsı tekniği önerilmesine rağmen ABC'nin test fonksiyonları üzerindeki başarısı ve performansı araştırmacılar tarafından kabul görmüştür.

2.4. Hibrit Yöntemler üzerine Bir Kaynak Taraması

Sürü zekâsı tekniklerinin bireysel olarak bir çok optimizasyon problemi için üstün başarı göstermesi araştırmacıların bu tekniklerin kabiliyetlerini birleştirerek yeni hibrit yaklaşımlar geliştirmeleri için bir esin kaynağı oluşturmaktadır. Hibrit yaklaşımlardaki temel amaç yöntemlerin yeteneklerinin (lokal araştırma, global araştırma, yakınsama hızı, başlangıç şartlarına duyarsızlık vb.) birleştirilerek optimizasyon problemi için i) daha iyi sonuçların, ii) benzer sonuçların daha hızlı iii) daha iyi sonuçların daha hızlı elde edilmesidir. Yani yöntemlerin bireysel olarak uygulanmasıyla elde edilenden daha iyi bir sonuca ulaşmak gaye olabileceği gibi aynı sonuca daha kısa zamanlarda ulaşmak da amaç olabilir. Elbette beklenen ve ümit edilen durum daha iyi sonuçlara daha kısa sürelerde ulaşılmasıdır. Bu çalışmanın merkezine yapay arı kolonisi, parçacık sürü optimizasyonu ve karınca kolonisi optimizasyonunu algoritmaları yerleştirildiği için hibrit çalışmalara ayrılan literatür özetinde de bu üç yöntemin birbirleriyle hibritleştirilmesini temel alan çalışmalara yer verilmiştir.

Shi ve ark. (2010) parçacık sürü optimizasyonu ve yapay arı kolonisi yöntemini IABAP adını verdikleri bir yöntemde birleştirmişlerdir. IABAP parçacıkların arı kolonisinden, arı kolonisinin ise parçacıklardan etkilenmesini temel alır. Arı kolonisinin parçacıklardan etkilenmesi kaşif arı mekanizması üzerine kuruludur. Herhangi bir zamanda arı kolonisinde kaşif arı oluştuğu zaman kaşif arıya rastgele bir çözüm üretmek yerine kaşif arı için yeni bir çözüm parçacık sürüsünden elde edilmektedir. Parçacık sürüsünün arı kolonisinden etkilenmesi ise PSO'nın hız denkleminde arıların

çözümlerinin de kullanılması ile sağlanmıştır. ABC ve PSO'yu hibritleştiren bir diğer çalışma ise El-Abd (2011) tarafından yapılmıştır. El-Abd PSO yönteminin ana döngüsünden sonra parçacıkların o ana kadar elde etmiş oldukları kendi en iyi çözümleri arasında ABC'nin güncelleme kuralı uygulanmaktadır. Bu sayede parçacıkların elde ettiği en iyi çözümlerin etrafında da araştırma sağlanmakta ve yöntemin araştırma kabiliyeti arttırılmaktadır.

Shelokar ve ark. (2007) parçacık sürü optimizasyonu ve karınca kolonisi optimizasyonunu PSACO adını verdikleri iki aşamalı bir yapıda hibritleştirmişlerdir. Parçacık sürü optimizasyonu her bir iterasyonda birinci olarak uygulanmış ve parçacıkların çözümleri etrafında lokal araştırmayı güçlendirmek için sürekli ACO yöntemi kullanılmıştır. Aynı şekilde iki aşamalı bir birleşim de Kaveh ve Talatahari (2009) tarafından önerilmiştir. Birinci aşamada parçacıklar ikinci aşamada ise karıncalar çalışmaktadır. Birinci aşamadaki parçacıklar tarafından elde edilen bilgi her iterasyonda karıncalara aktarılmaktadır. Ayrıca yöntemin global araştırma yeteneğini arttırmak amacıyla PSO'nun hız denklemi modifiye edilmiştir. Önerilen yöntem ayrık bir problemin çözümü için uygulandığından dolayı elde edilen çözümler en yakın tam sayıya yuvarlanarak parçacıkların ve karıncaların çözümleri (karar değişkenleri) ayrıklaştırılmaktadır. Kıran ve ark. (2012) parçacık sürü optimizasyonu tarafından her iterasyonda elde edilen en iyi çözüm etrafındaki lokal araştırmayı desteklemek için sürekli bir ACO yöntemini kullanmışlardır. Bu hibrit yaklaşımda sistemin en iyi çözümü hem karıncalar hem de parçacıklar için yol gösterici olarak kullanılmaktadır. Kıran ve ark. (2012a) bu hibrit yöntemi Türkiye'nin enerji talebi tahmini için de kullanmışlardır.

Literatür araştırması sırasında ABC ile ACO'nun hibritleştirildiği bir çalışmaya rastlanmamıştır. PSO ile ABC'nin hibritleştirildiği bazı çalışmaların özeti ise yukarıda verilmiştir. PSO yönteminin genetik algoritma, farksal gelişim gibi diğer yöntemlerle hibritleştirilmesi üzerine yapılmış bir literatür araştırması için Thangaraj ve ark. (2011) referansına bakılabilir. Ayrıca karınca kolonisinin hibrit yapıda kullanıldığı çalışmalar içinse Blum ve ark. (2008) çalışmasına bakılabilir.

3. SÜRÜ ZEKÂSI VE YÖNTEMLERİ

Sürü zekâsını tanımlamadan önce "Sürü" ve "Zekâ" kavramlarının tanımlarına ihtiyaç vardır. Böylelikle sürü zekâsının ne anlama geldiği daha net anlaşılabilir. Türk Dil Kurumu Güncel Türkçe Sözlüğü'nde (TDK, 2012) "sürü" için aşağıda sıralanan 4 anlam verilmiştir.

i) Evcil hayvanlar topluluğu

ii) Bir insanın bakımı altındaki hayvanların tümü

iii) Birlikte yaşayan hayvan topluluğu

iv) Yönlendirilebilen insan topluğu

Biyolojide "sürü" ortak özellikleri olan ve ortak davranışlar sergileyen bir gruptur (Gazi ve Passino, 2011). Sürü terminolojisinde çeşitli türde ve kendi aralarında ortak davranış sergileyen arı veya karıncalar için koloni (Şekil 2.1, Şekil 2.2), antiloplar için topluluk (Şekil 2.3), balıklar için "*shoal*" veya "*school*" (Şekil 2.4), kurtlar için "*packs*" (Şekil 2.5) ve kuşlar için "*flocks*" (Şekil 2.6) kavramları kullanılmaktadır. Farklı dilde yazılan kelimelerin tümünün Türkçe karşılığı sürü olmasına rağmen her biri farklı bir sürüyü tanımlamak için kullanılmışlardır. Mühendislikte ise sürü bağımsız bireysel dinamiklere sahip olan, davranışlarında yakın bir ilişki olan ve bazı görevleri beraber yerine getiren bir ajanlar grubudur (Gazi ve Passino, 2011).

Zekâ Arapça kökenli bir kelimedir ve anlamı Türk Dil Kurumu Güncel Türkçe Sözlüğü'nde (TDK, 2012) "İnsanın düşünme, akıl yürütme, objektif gerçekleri algılama, yargılama ve sonuç çıkarma yeteneklerinin tamamı" olarak anlamlandırılmıştır. Bir kavram olarak ise zekâ zihnin öğrenme, öğrenilen bilgiden yararlanma, yeni durumlara uyma ve yeni çözüm yolları bulabilme yeteneğidir (Yörükoğlu, 2004). Hem tanımından hem de anlamından ortaya çıkan sonuç zekânın bir yetenekten ziyâde yeteneklerin birleşimi olduğudur.

Sürü ve Zekâ hakkında detaylı bilgi verilmesinin sebebi Zekâ kavramının Sürüler için kullanıldığında bir anlam kaymasına sahip olup olmamasını incelemek içindir. Nitekim zekâ bireye ya da zihne has bir olgu iken bireysel ya da zihni olarak zeki olmayan ama toplu olarak zeki davranışlar sergileyebilen sürüler için kullanıldığında tanımındaki bazı ifadeleri kaybetmektedir.

Yukarıdaki bilgilere ve tecrübelerimize dayanarak sürü zekâsı için "bireysel olarak zeki olmayan ama toplu olarak aralarındaki etkileşimler ve birikimler ile zeki davranışlar sergileyen, merkezi bir kontrole sahip olmayan ve kendi kendine organize

olabilen bir canlı grubunun ortaya çıkardığı kolektif bir yetenek" tanımını yapmak mümkündür. Burada sürü zekâsı bir yetenek olarak addedilmektedir ve düşünme, bilginin üretilmesi vb. yeteneklerin tümü için kullanılmamaktadır. Bundan dolayıdır ki Sürü Zekâsına dayalı optimizasyon yöntemleri sürünün tüm davranışlarından ziyade zeki olarak adlandırılan davranışlarını modellemektedirler. Ayrıca belirtmekte fayda vardır ki Sürü Zekâsı (Swarm Intelligence) kavramı yapay zekâ alanında incelenir ve Sürü Zekâsına dayanan tüm yöntemler için de aynı kavram kullanılır.

Bir sürünün zeki olup olmadığına karar verebilmek için Millonas (1994) 5 tanımlayıcı unsur öne sürmüştür. Bu ilkeler ve açıklamaları aşağıda verilmiştir.

Yakınlık Prensibi: Sürü temel ve basit uzay zaman hesaplamalarını yapabilmelidir.

Kalite Prensibi: Sürü temel uzay zaman hesaplamalarının yanı sıra yiyecek kaynağı, güvenlik vb. açılardan kalite faktörlerini de dikkate alabilmelidir.

Dağılım Cevabı Prensibi: Sürü kaynaklarını dar boğaz oluşturacak şekilde kullanmamalıdır. Çevredeki ani değişikliklere tepki verecek şekilde kaynaklarını düzenli dağıtmalıdır.

Kararlılık Prensibi: Sürü çevresel her değişiklikte çalışmasını değiştirmemelidir. Her değişiklik bir enerji sarfiyatına neden olur ve her çevresel değişiklikte çalışmasını değiştiren sürü kayda değer bir kazanım elde edemeyebilir.

Uyarlanabilirlik Prensibi: Sürü kaydadeğer bir enerji kazanımı için davranış modunu değiştirebilmelidir.

Millonas'a (1994) göre birçok karmaşık adaptif sistem yukarıda sayılan prensiplerin tümüne veya bir kısmına dahil olur. Örneğin ekonomide, "zaman paradır", "sadece en iyisini al", "tüm yumurtaları bir sepete koyma", "emniyet üzülmekten iyidir", "eldeki bir kuş daldaki on kuştan iyidir" cümleleri yukarıda prensipler kapsamında açıklanabilir.

Sürü zekâsının tanımında iki önemli özellik göze çarpmaktadır. Birincisi merkezi bir kontrolün olmamasıdır. Merkezi kontrol olmamasından kasıt sürünün herhangi bir çevresel etkiye karşı tepkisini bir emir almaksızın oluşturmasıdır. İkinci önemli özellik ise kendi kendine organize olabilmedir. Kendine kendine organize olabilme dinamik bir davranışlar bütünüdür ve lokal etkileşimlerden global cevaplar oluşturulması işidir. Bonabeau ve arkadaşlarına (1999) göre kendi kendine organize olabilen canlılarda 4 karakteristik davranış bulunmaktadır.

1. Pozitif Geri Besleme: Sürünün bireyleri arasındaki etkileşim çok yiyecek bulunan kaynaklarının diğer kaynaklara nispeten daha fazla işlenmesini amaçlar. Yapılan bir işin sürdürülmesi ve iyi bir kaynağın daha fazla işlenmesi sürü içerisindeki pozitif geri besleme ile meydana gelir.

2. Negatif Geri Besleme: Pozitif geri besleme ile oluşabilecek durağanlaşmayı ve sürünün tüm kaynaklarını dar boğaza sevk etmesini engellemek negatif geri besleme ile meydana gelir.

3. Dalgalanma: Kendi kendine organize olabilen canlılarda dalgalanmalar (rastgele araştırma, hatalar, rastgele görev değişimi) az bir süreliğine de olsa sürünün kararlığını yitirmesi neden olur. Bu sayede sürü bu dalgalanmanın sonucu olarak ortaya çıkan durumdan yeni keşifler yapabilir. Doğada yaşayan sürünün tam bir modeli ortaya konulamayacağından dolayı sürü zekâsında da bu dalgalanmalara ihtiyaç vardır. Örnek vermek gerekirse tüm karıncaların yuva veya kaynak etrafındaki feromon izlerini takip etmesi sürünün durağanlaşmasına sebep olur. Dalgalanma sayesinde bazı karıncalar rastgele yeni yiyecek kaynakları keşfedebilir. Bundan dolayıdır ki dalgalanmalar kendi kendine organize olabilen canlılar için önemli bir iyileştirme faktörü olabilir.

4. Etkileşimler: Sürü içerisinde bireylerin arasındaki etkileşim sayesinde kendi kendine organize olma gerçekleşir. Bireyler arasındaki etkileşim karıncaların kimyasal maddeyi takip etmesi olabildiği gibi arıların dansları da olabilir. Burada önemli bir nokta vardır ki etkileşim için verici kadar alıcının da olması gerektiğidir.

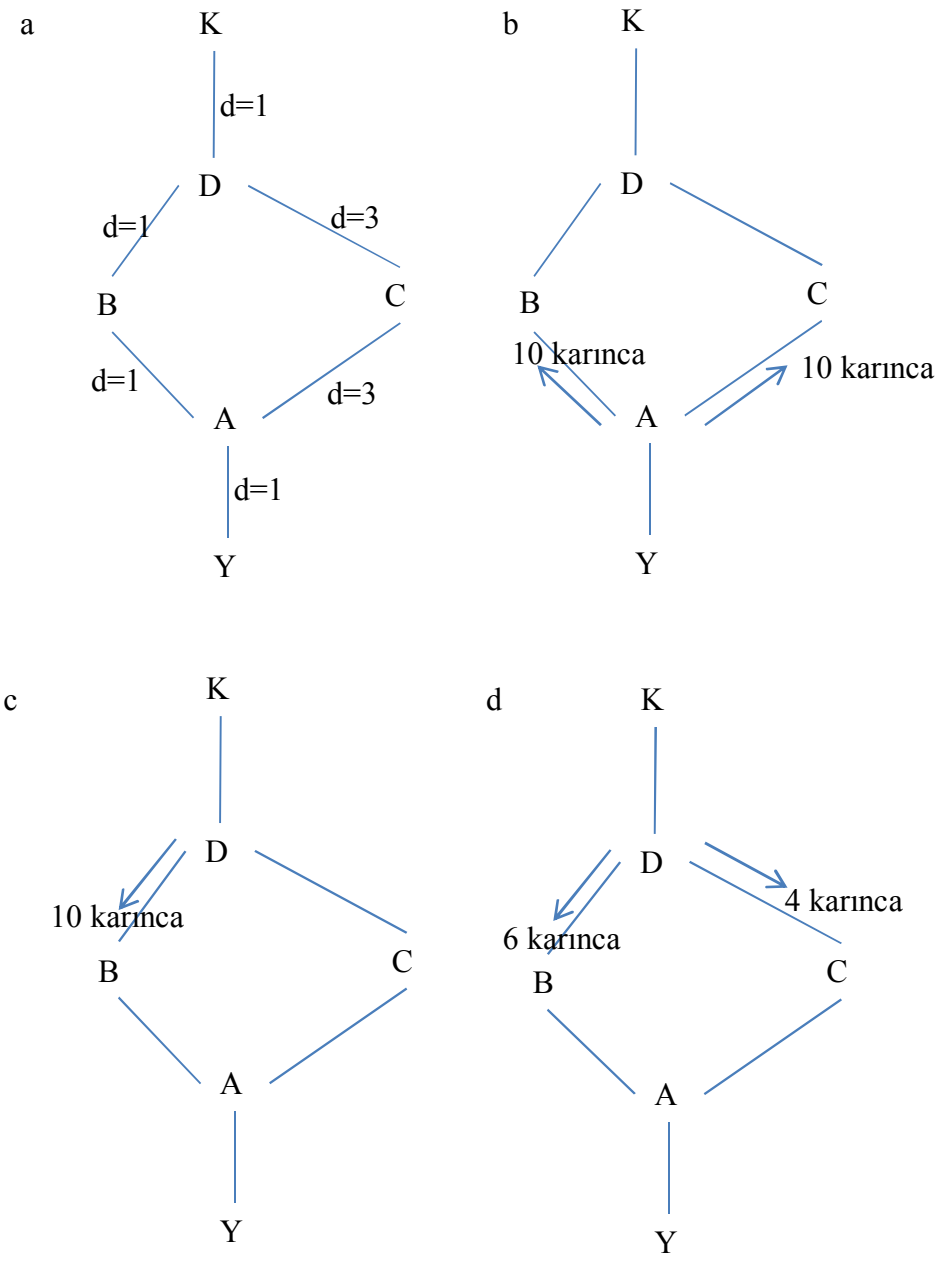
Kendi kendine organize olabilmeye karınca kolonilerden bir örnek vererek sürü zekâsının kavramsal tartışması sonlandırılacaktır. Birden fazla karınca birden fazla yola (yiyecek kaynağı ve yuva arasındaki) feromon bırakır. Daha sonra karıncalar kendi yollarında bir müddet çalışırlar. Açıktır ki kısa yolda uzun yoldan fazla feromon birikir. Daha sonra karıncalar feromonun fazla olduğu yolu tercih etmeye başlarlar ve bu yolda feromon olabildiğince artar (Pozitif Geri Besleme). Çevresel nedenlerle (rüzgar vb.) yollardaki feromon miktarları azalır ve karıncalar yeniden farklı yolları seçmeye başlarlar (Negatif Geri Besleme). Bazı karıncalar yolun bir tarafında hiçbir feromon izi bulamayınca rastgele bir araştırma yapar ve bulduğu kaynak ile yuva arasına feromon bırakır (Dalgalanma). Tüm bu süreçte karıncaların birbirlerine yol göstermeleri feromon üzerinden gerçekleşir (Etkileşimler).

3.1. Karınca Kolonisi Optimizasyonu

Karıncaların yem arama davranışı üzerine kurulu olan ACO yapıcı (constructive) bir yöntemdir. Karıncalar arası iletişim dolaylı yoldan yani çözümün parçaları arasına bırakılan bir değer (doğada feromon maddesi) üzerinde yapılır. Başlangıçta yollarda feromon olmadığı varsayıldığında karınca kolonisinin yiyecek kaynağı ile yuva arasındaki davranışları basit olarak Şekil 3.1'de gösterilmiştir.

Şekli kısaca açıklamak gerekirse; yuvadan (Y) yiyecek aramak için çıkan karıncalar (20 tane olsun) A noktasına geldiklerinde iki yolda da feromon olmadığı için ortalama eşit miktarlarda karınca AB ve AC yollarını seçecektir (Şekil 3.1b). AB yolunu seçen karıncalar yiyecek kaynağına varıp geri D noktasına döndüklerinde DB ve DC yollarından birini seçecektir. Bu seçim feromon maddesine göre yapıldığı için DB yolundaki feromon miktarı 10 birim (her karıncanın geçtiği yola 1 birim feromon bıraktığı varsayıldığında) iken DC yoluna henüz karıncalar tarafından feromon bırakılmamış olacaktır. Bundan dolayı yuvaya dönen 10 karınca DB yolunu seçecektir (Şekil 3.1c). Kaynağa uzun yoldan giden 10 karınca kaynaktan D noktasına ulaştıklarında DB yolundaki feromon miktarı 20 birim iken DC yolundaki feromon miktarı 10 birim olduğundan 10 karıncadan büyük çoğunluğu DB yolunu tercih edecek bir kısmı ise DC yolunu kullanacaklardır (Şekil 3.1d). Bu süreç tüm karıncalar kısa olan yolu (Y-A-B-D-K) seçene kadar devam etmektedir.

Tüm bu süreç literatürde ayrık optimizasyon problemlerinin çözümü için başarılı şekilde uygulanmaktadır. Daha açıklayıcı olması adına algoritmanın geri kalan kısmı gezgin satıcı problem (traveling salesman problem-TSP) üzerinden anlatılacaktır. TSP ayrık bir optimizasyon problemidir ve literatürde ayrık optimizasyon yöntemlerinin başarısı bu problemi çözmedeki başarısına göre analiz edilmektedir. Kısaca problemi tanıtmak gerekirse TSP sonlu sayıdaki şehirlerin tümüne uğramak isteyen ve sonunda başladığı noktaya dönen bir satıcının en kısa yoldan turunu tamamlamasıdır. Simerik TSP'de nodlar arasında çift yönlü ve eşit mesafeli bir ilişki bulunmakta iken asimetrik TSP'lerde bazı nodlar arasında tek yönlü harekete izin verilmektedir. Ayrık bir çok gerçek dünya problemi bu problemin modeli kullanılarak çözümlenmeye çalışıldığı için literatürde çok bilinen problemlerden biridir.



Şekil 3.1. Karıncaların yuva ile yiyecek kaynağı arasındaki davranışı

a) Yuva yiyecek kaynağı arası b) Birinci karar verme

c) İkinci karar verme d) Üçüncü karar verme

ACO'nun başlangıç fazında karıncalar rastgele şehirlere konumlandırılırlar. Her karınca bir sonraki adımda hangi şehre gideceğini belirlemek için ACO'nun geçiş kuralını kullanır ve bu kural Denklem 3.1 ile tanımlanır.

$$p_{ij}^{k}(t)=\begin{cases}\dfrac{\left[\tau_{ij}(t)\right]^{\alpha}\times\left[\eta_{ij}\right]^{\beta}}{\sum_{m\in S_k}\left[\tau_{im}(t)\right]^{\alpha}\times\left[\eta_{im}\right]^{\beta}}, & if\,(m\in S_k)\\ 0 & diğer\ durumda\end{cases} \tag{3.1}$$

Denklem 3.1'deki $p_{ij}^{k}(t)$ t anında i. şehirden j. şehire k. karıncanın gitme ihtimali, $\tau_{ij}(t)$ i-j kenarı üzerindeki feromon miktarı, η_{ij} Denklem 3.2. ile hesaplanan i-j kenarının görülebilirlik veya sezgisel değeri, α ve β feromon miktarının ve görülebilirlik değerinin seçime etkisini kontrol eden parametreler ve S_k, k karıncası tarafından ziyaret edilmeyen şehirlerin bir listesidir. *N* şehirlerin ve $tabu_k$, k. karınca tarafından ziyaret edilen şehirlerin kümesi olmak üzere $S_k=\{N-tabu_k\}$, k karıncası tarafından ziyaret edilmeyen şehirlerin bir kümesidir.

$$\eta_{ij}=\frac{1}{d_{ij}} \tag{3.2}$$

burada, d_{ij} i-j kenarının uzunluğudur.

Denklem 3.1 ile verilen geçiş kuralı kullanılarak başlangıçta rastgele şehirlere konumlandırılan karıncaların tümünün kapalı turlarını tamamlaması sağlanır ve kapalı turların uzunlukları hesaplanır.

Karıncalar tarafından gezilen şehirler arasına feromon güncellemesi Denklem 3.3 ile yapılır. Gezilen kenarlara feromon bırakılmadan hemen önce yollardaki feromon miktarının bir kısmı silinir (buharlaştırma). Buharlaştırmanın temel amacı negatif geri beslemeyi sağlayarak tam çözüm için iyi olmayan kenarların karıncalar tarafından gereksiz gezilmesini engellemektir. Feromon güncelleştirme işlemi Denklem 3.3 ile verilmiştir.

$$\tau_{ij}(t+1)=\rho\times\tau_{ij}(t)+\Delta\tau_{ij} \tag{3.3}$$

Denklemdeki ρ buharlaşma katsayısıdır. Eğer her iterasyonda feromonun %10'u buharlaştırılmak isteniyorsa ρ katsayısının 0.9 olarak seçilmesi gerekir. Bırakılan feromon miktarı denklemdeki sağ terimdir ve Denklem 3.4 ile hesaplanır;

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{3.4}$$

$\Delta\tau_{ij}^{k}$, k karıncası tarafından i-j kenarına bırakılan feromon miktarıdır ve denklem 3.5 ile hesaplanır. Eğer farklı karıncalar aynı kenarlar üzerinde hareket etmişse bu kenarlarda daha fazla feromon birikecektir ve bu kenarın seçilme olasılığı artacaktır.

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{Q}{L_k}, & Eğer\ k.karıncanın\ \ turunda\ i-j\ kenarı\ bulunuyorsa \\ 0, & bulunmuyorsa \end{cases} \tag{3.5}$$

Denklem 3.5'te Q sabit bir değer ve L_k k. karıncanın turunun uzunluğudur. Denklem 3.5'ten görüldüğü üzere çözümün kalitesi (yolun kısalığı) pozitif geri beslemenin kuvvetini arttırmaktadır. Yukarıdaki anlatılan karınca kolonisi optimizasyonu adım adım Algoritma 3.1'de verilmiştir.

Algoritma 3.1. Detaylı Karınca Kolonisi Optimizasyonu Algoritması

1. Başlat
   a. N karınca sayısı ve M şehirlerin sayısı
   b. Maksimum iterasyon sayısına (T) karar ver ve iterasyon sayacını t'yi sıfırla
   c. Her kenara az miktarda feromon bırak.
2. Başlangıç şehirlerine karıncaları konumlandır
   a. s=1
   b. For k=1 to N
      i. $tabu_k(s) = round(rand(M))$
      ii. s=s+1
3. Çözümleri oluştur
   a. s<M iken
      i. For k=1 to N
         1. Denklem 3.1 ile karıncayı bulunduğu noktadan j. noktaya taşı.
         2. $tabu_k(s) = j$
      ii. s=s+1
4. Çözümleri değerlendir ve feromon güncelle
   a. For k=1 to N
      i. $tabu_k(s) = tabu_k(1)$
      ii. Her karıncanın tur uzunluğunu hesapla ($tabu_k$)
   b. Her i-j kenarı için
      i. For k=1 to N
         1. Denklem 3.5 ve 3.4'ü kullanarak bırakılacak feromon miktarını hesapla
   c. Denklem 3.3'ü kullanarak kenarlardaki feromon miktarını güncelle
5. Çevrim hesapları
   a. O ana kadar bulunan en iyi çözümü kaydet
   b. t=t+1
   c. Eğer (t<T)
      i. Her karıncanın tabu listesini boşalt ($tabu_k$)
      ii. 2. Adıma git

      Değilse
      i. Bulunan en iyi çözümü raporla.

3.2. Parçacık Sürü Optimizasyonu

Kuş veya balık sürülerinin yiyecek arama davranışlarından ve yiyeceğe doğru yapılan aksiyondan esinlenilerek sürekli optimizasyon problemlerinin çözümü için geliştirilen parçacık sürü optimizasyonu (PSO) algoritmasında potansiyel çözümler parçacık olarak adlandırılır. Karınca kolonisi optimizasyonuna göre daha kısa ve anlaşılır bir yapıya sahiptir ve parçacıkların yiyecek kaynağına doğru yapılan hareketini esas alır. Hareketin temelinde hız vardır ve parçacıklarının hızının hesaplanmasında iki temel faktör vardır. Birinci faktör bireyin bulunduğu sürüden etkilenmesini sağlayan sosyal bileşendir ve bu bileşen doğrudan parçacığın sürünün en iyi çözüme sahip bireyinin bulunduğu pozisyona doğru hareketini sağlar. İkinci faktör ise bireyin kendi tecrübesidir ve iterasyonlar boyunca elde ettiği en iyi çözümü kısıtlı hafızasında tutar.

Bireyin hızını hesaplamak için Denklem 3.6 kullanılır. Denklem 3.6 üç terimden oluşur, birinci terim önceki hızı, ikinci terim bilişsel komponenti ve son terim ise sosyal komponenti gösterir.

$$\vec{v_i}(t+1) = \vec{v_i}(t) + r_1 \times c_1(\overrightarrow{pbest_i}(t) - \vec{X_i}(t)) + r_2 \times c_2(\overrightarrow{Gbest}(t) - \vec{X_i}(t)) \qquad (3.6)$$

Denklemdeki $\vec{v_i}$, $\vec{X_i}$ parçacığına ait hız vektörü, $\overrightarrow{pbest_i}$, i. parçacık tarafından o ana kadar elde edilen en iyi çözüm (Denklem 3.7), $\overrightarrow{Gbest}$, sürü tarafından o ana kadar elde edilmiş olan en iyi çözüm (Denklem 3.8), c_1, bilişsel komponentin ivme katsayısı, c_2, sosyal komponentin ivme katsayısı, r_1 ve r_2, yöntemin rastsal elementleridir ve [0,1] aralığında rastgele üretilirler. Parçacığın geçmişte elde ettiği en iyi çözüm Denklem 3.7. ile elde edilir.

$$\overrightarrow{pbest_i}(t+1) = \begin{cases} \vec{X_i}(t+1), & Eğer(f(\vec{X_i}(t+1)) < f(\overrightarrow{pbest_i}(t))) \\ \overrightarrow{pbest_i}(t), & değilse \end{cases} \qquad (3.7)$$

$$\overrightarrow{Gbest}(t+1) = \min\left\{ f(\overrightarrow{pbest_i}(t+1)) \right\} \; i = 1,2,...,N \qquad (3.8)$$

Denklem 3.7'deki $\vec{X_i}(t+1)$, t+1 zamanında parçacığın pozisyonudur ve Denklem 3.9 ile hesaplanır.

$$\vec{X_i}(t+1) = \vec{X_i}(t) + \vec{v_i}(t+1) \qquad (3.9)$$

Temelde yapılan iş her iterasyonda Denklem 3.6 kullanılarak parçacığın hızının hesaplanması ve hesaplanan hız ile parçacığın yeni pozisyonunun Denklem 3.9 ile bulunmasıdır. Denklem 3.7 kullanılarak parçacığın elde ettiği yeni çözümün öncekinden iyi olup olmadığına karar verilir ve iyiyse yeni kişisel en iyi çözüm olarak hafızaya alınır. Denklem 3.8 ile de sürünün en iyi çözümüne karar verilir. Denklem 3.8 sürünün t anındaki en iyi çözümünü sürünün önceki en iyi çözümü ile kıyaslamak yerine kişisel en iyi çözümlerden en iyisini sürünün en iyi çözümü olarak tayin etmeyi esas alır, çünkü kıyaslamalar zaten kişisel en iyi çözümler belirlenirken yapılmaktadır. Anlatımda ve denklemlerde özellikle Denklem 3.7 ve 3.8'de dikkat edilirse daha küçük olanın daha iyi olduğu söylenmiştir çünkü dikkate alınan husus minimizasyon işlemidir. Ayrıca detaylı PSO algoritması Algoritma 3.2'de verilmiştir.

Algoritma 3.2. Detaylı Parçacık Sürü Optimizasyonu Algoritması

1. Başlat
   a. N sürüdeki parçacık sayısı ve D optimizasyon probleminin boyutu
   b. Maksimum iterasyon sayısı (T) karar ver ve iterasyon sayacını t'yi sıfırla
   c. Parçacıkları çözüm uzayına rastgele dağıt
   d. Parçacıkların bireysel en iyi çözümlerini ata ($\overrightarrow{pbest}$)
   e. Sürünün en iyi çözümüne karar ver.($\overrightarrow{Gbest}$)
2. Parçacıkların pozisyonlarını güncelle
   For i=1 to N
      a. Denklem 3.6'yı kullanarak parçacığın hızını belirle.
      b. Denklem 3.9'u kullanarak parçacı yeni pozisyonuna hareket ettir.
3. Sürünün ve bireylerin en iyi çözümlerini güncelle
      a. Denklem 3.7'yi kullanarak kişisel en iyi çözüme karar ver.
      b. Denklem 3.8'i kullanarak sürünün en iyi çözümüne karar ver.
4. Çevrim Hesapları
      a. t=t+1
      Eğer (t<T) ise.
         i. 2.Adıma git
      Değilse
         i. Sürünün en iyi çözümünü raporla

N parçacık sayısı ve D optimizasyon probleminin boyutu yani optimize edilecek karar değişkenlerinin sayısı olduğuna göre popülasyon diye adlandırılan olgu NxD boyutlu bir matristir. Bu matriste satırlar parçacıkları ve sütunlar ise karar değişkenlerini gösterir. Bir diğer NxD boyutlu matriste bireylerin kısıtlı hafızasıdır. Bu kısıtlı hafızada o ana kadarki en iyi çözümler ($\overrightarrow{pbest}$) tutulmaktadır. Ayrıca hızın da her karar değişkeni için bir değeri olduğu düşünüldüğünde NxD boyutlu bir matris de her parçacığın her karar değişkeni için hızını tutmak için kullanılmaktadır. Hızın etkisi Denklem 3.6'da görüldüğü üzere sosyal ve bilişsel komponentin etkisini bastıracak

düzeylere ulaşabilmektedir ve yöntemin erken yakınsamasına ve lokal minimumlara takılmasına neden olabilmektedir. Yöntemin lokal ve global araştırma yeteneğini kontrol etmek ve dengelemek amacıyla Shi ve Eberhart (1998) eylemsizlik ağırlığı (Denklem 3.10'daki "ω") olarak adlandırılan bir terimi hızın hesaplanması için kullanmışlardır. Bu sayede önceki hızın yeni hıza etkisi ağırlıklandırılmış ve yöntemin iterasyonlar boyunca en iyiye yaklaşırken yavaşlaması sağlanmıştır.

$$\vec{v_i}(t+1) = \omega \times \vec{v_i}(t) + r_1 \times c_1(\overrightarrow{pbest_i}(t) - \vec{X_i}(t)) + r_2 \times c_2(\overrightarrow{Gbest}(t) - \vec{X_i}(t)) \qquad (3.10)$$

3.3. Yapay Arı Kolonisi Algoritması

Yapay arı kolonisi bal arılarının yiyecek arama davranışı üzerine kurulu bir optimizasyon algoritmasıdır. Her ne kadar literatürde bal arılarının yiyecek arama davranışından esinlenilerek geliştirildiği bildirilmekte ise de görevli arıların yiyecek kaynaklarının pozisyon bilgisinin paylaşmaları için yaptığı sallanım dansı da yöntem içerisinde modellenmiştir. Görevli (employed) arılar gözcü arılarla bilgilerini paylaşmakta ve gözcü arılar da paylaşılan bilgiyi dikkate alarak araştırmalarını yapmaktadırlar. Yapay arı kolonisinde iki tip arı mevcuttur. Birinci tip arılar görevli arılardır ve kendilerine atanmış olan yiyecek kaynağı etrafında araştırma yaparlar ve kaynakların pozisyon bilgilerini koloniye taşırlar. Diğer tip arılarsa görevli olmayan (unemployed) arılardır. Görevli olmayışlarının sebebi kendilerinin sürekli bir çalışma döngüsü içinde olmamalarıdır ve yapacakları işte bir seçim keyfiyetine haiz olmalarındandır. Gözcü arılar birinci tip görevli olmayan arılardır ve görevli arılar tarafından paylaşılan bilgiyi kullanarak araştırmalarını yaparlar. Kaşif arılar da bir diğer görevli olmayan arı çeşididir. Kaşif arıların oluşumu kaynağın durumuna bağlıdır. Herhangi bir kaynak belirli bir (*limit*) zamanda görevli veya gözcü arılar tarafından iyileştirilemezse bu kaynağın görevli arısı kâşif arı olur. Kendisine rastgele bir çözüm atandıktan sonra bu kâşif arı tekrar görevli arı durumuna devam eder.

ABC algoritması bazı kabullere sahiptir.

1. Kolonideki arıların yarısı görevli ve diğer yarısı gözcü arılardan oluşur. Ayrıca kaynak sayısı görevli arı sayısına eşittir.
2. Görevli arıların tümü başlangıçta kâşif arıdır. Kendisine yiyecek kaynağı atandıktan sonra kâşif arı görevli arı olur.

3. Algoritmanın bir ayrık zamanında yalnızca bir tane kaşif arı oluşabilir.
4. Her görevli arı kaynağına ait pozisyon bilgisini hafızasında tutar ve bu bilgiyi her iterasyonda diğer arılarla paylaşır.

Bu dört maddeden ilk üçü literatürde var olmasına rağmen dördüncüsü açıkça bildirilmemiştir. İlerleyen paragraflarda 4. Kabul detaylı olarak açıklanacaktır.

ABC algoritması "Başlangıç", "Görevli Arı Fazı", "Gözcü Arı Fazı" ve "Kâşif Arı Fazı" olmak üzere 4 fazdan oluşmaktadır. Algoritmanın başlangıcında tüm görevli arılar için Denklem 3.11 kullanılarak başlangıç çözümleri üretilir. Ayrıca üretilen her kaynak için de bir deneme sayacı bulunmaktadır. Bu sayaçlardan en büyük içeriğe sahip olan, her iterasyonda limit parametresi ile karşılaştırılarak kaynağın arısının kâşif arı olup olmayacağına karar verilir.

$$X_{ij} = X_j^{\min} + r_{ij}(X_j^{\max} - X_j^{\min}), \quad i = 1,2,...,N \quad ve \quad j = 1,2,...,D \tag{3.11}$$

Denklemdeki X_{ij} i.yiyecek kaynağının j. boyutu veya i. işçi arının hafızasındaki yiyecek kaynağının j. boyutudur. $X_j^{\min}$ ve $X_j^{\max}$, j. karar değişkeni veya boyut için alt ve üst sınırdır. Her karar değişkenin çözüm uzayının sınırları farklı olabileceğinden dolayı bu sınırların içerisinde algoritmayı başlatmak Denklem 3.11 ile garanti edilmiş olur. r_{ij}, [0,1] aralığında üretilen rastgele bir sayıdır. N görevli arı veya yiyecek kaynağı sayısıdır. D ise optimizasyon probleminin boyutluluğudur yani optimize edilecek karar değişkeni sayısıdır. Üretilen kaynaklar görevli arılar tarafından ezberlenir ve yiyecek kaynağının kalitesi Denklem 3.12 ile hesaplanır.

$$fit_i(t) = \begin{cases} \dfrac{1}{1+f_i(t)}, & Eğer(f_i(t) \le 0) \\ 1+abs(f_i(t)), & değilse \end{cases} \tag{3.12}$$

Denklemdeki, fit_i i. yiyecek kaynağının i. işçi tarafından değerlendirilen kalitesi, f_i i. yiyecek kaynağı için optimizasyon problemi için özel olan amaç fonksiyonundan ($f_i = F(\overrightarrow{X_i})$) elde edilen değer ve abs mutlak değer fonksiyonudur.

Görevli arı fazında her görevli arı kendine atanmış olan yiyecek kaynağını pozisyonunu komşu bir yiyecek kaynağı pozisyonunu kullanarak Denklem 3.13 ile iyileştirmeye çalışır.

$$\begin{aligned}
&\overrightarrow{V_i} = \overrightarrow{X_i} && (a)\\
&V_{ij}(t) = X_{ij}(t) + \varphi \times (X_{ij}(t) - X_{kj}(t)) && (b)\\
&\quad i = 1,2,...,N,\ k \in \{1,2,...,N\} && (c)\\
&\quad j \in \{1,2,...,D\} && (d)\\
&\quad i \neq k && (e)
\end{aligned} \qquad (3.13)$$

Denklemdeki $V_{ij}(t)$, t anında i. aday yiyecek kaynağı pozisyonunun j. boyutu, $X_{ij}(t)$, i. yiyecek kaynağının j. boyutu, $X_{kj}(t)$, i. yiyecek kaynağının komşusu olan ve rastgele seçilen yiyecek kaynağının j. boyutu, φ, [-1,1] aralığında rastgele üretilen ölçekleme faktörü, *N*, görevli arıların veya yiyecek kaynaklarının sayısı, D, optimizasyon probleminin boyutluluğudur. Denklem 3.13'ü kısaca açıklamak gerekirse; görevli arı yiyecek kaynağını kopyalar (a) ve kopya yiyecek kaynağı pozisyonunun sadece bir boyutunu görevli arı komşu bir görevli arının yiyecek kaynağını kullanarak günceller (b), bu işlem tüm görevli arılar tarafından yapılır (c), yiyecek kaynağının sadece bir boyutu değiştirilir (d) ve komşu seçilen görevli arının kendisi olmadığından emin olunur (e). Denklem 3.13 ayrıca 4. Kabulün açıklamasını da yapar. Çünkü bir yiyecek kaynağı pozisyonu bir komşu yiyecek kaynağı pozisyonunun kullanılmasıyla ve bu komşu da tüm popülasyondan rastgele seçilerek yapılıyorsa bir görevli arının tüm yiyecek kaynaklarına ait pozisyonları bilmesi gerekir. Bu süreç algoritmanın bilgi paylaşımını sadece görevli arı-gözcü arasında olmadığını aynı zamanda görevli arılar arasında da bilgi paylaşımı olduğunu gösterir.

Görevli arı kaynaklardan sadece birini hafızasında tutmaya devam edecektir. Yeni veya eski yiyecek kaynağı arasındaki tercih çözüm kalitesine bağlı olarak yapılır. Bunun için görevli arı yeni yiyecek kaynağının ($\overrightarrow{V_i}$) kalitesini Denklem 3.12'yi kullanarak değerlendirir ve Denklem 3.14'ü kullanarak hangi yiyecek kaynağını hafızasında tutacağına karar verir.

$$\overrightarrow{X_i}(t+1) = \begin{cases} \overrightarrow{V_i}(t), & Eğer(fit_v \geq fit_i) \\ \overrightarrow{X_i}(t), & değilse \end{cases} \qquad (3.14)$$

Eğer yeni yiyecek kaynağı ezberlenirse yiyecek kaynağına ait sayaç sıfırlanacaktır değilse 1 artırılacaktır.

Algoritmanın gözcü arıları, görevli arılar tarafından pozisyon bilgisi paylaşılan yiyecek kaynaklarını iyileştirmek amacıyla seçerler. Bu seçim mekanizması için rulet tekerleği kullanılır. Bundan dolayı görevli arı fazından gözcü arı fazına geçilmeden

önce her görevli arının yiyecek kaynağının seçilme ihtimali Denklem 3.15 kullanılarak hesaplanır.

$$p_i = \frac{fit_i}{\sum_{j=1}^{N} fit_j}, \quad i = 1,2,...,N \tag{3.15}$$

Denklem 3.15'teki p_i, i. görevli arının yiyecek kaynağının herhangi bir gözcü arı tarafından seçilme ihtimalidir.

Gözcü arı fazında her gözcü arı Denklem 3.15'i kullanarak bir görevli arının kaynağını seçer ve Denklem 3.13'ü kullanarak yiyecek kaynağı pozisyonunun bir boyutunu günceller. Denklem 3.12'yi kullanarak yeni kaynağın kalitesini ölçer ve Denklem 3.14'ü kullanarak eski veya yeni kaynağın pozisyon bilgisini bu kaynağın görevli arısına devreder. Eğer gözcü arı tarafından bulunan çözüm eski çözümden daha iyiyse bu kaynağa ait sayaç değeri sıfırlanır, kötüyse sayaç değeri 1 artırılır.

Kâşif arı fazında, algoritmanın 3. Kabulüne göre her iterasyonda sadece bir kaşif arı oluşabilir. Hangi kaynağın görevli arısının kaşif arı olacağına kaynaklara ait sayaç değerleri ve algoritmanın kendine özgü tek parametresi olan *limit* parametresi kullanılarak karar verilir. En büyük değere sahip sayaç bulunur ve *limit* ile karşılaştırılır. Eğer limit değerinden büyükse bu arı için Denklem 3.11 kullanılarak yeni bir yiyecek kaynağı üretilir ve bu kâşif arı tekrar görevli arı olur. Bu görevli arı Denklem 3.12'yi kullanarak kaynağın kalitesini değerlendirir ve kaynağa ait sayaç değerini sıfırlar. Sayaç değeri limit değerinden büyük değilse bu iterasyonda bir kaşif arı oluşmasına izin verilmez.

ABC algoritması için detaylı algoritma taslağı verilmemektedir çünkü çok detaylı olarak anlatılmıştır. Buna ek olarak algoritmayı hem pasif özne (yiyecek kaynakları) hem de aktif özne (arılar) üzerinden anlatmak mümkündür. Yiyecek kaynakları üretilmekte, iyileştirilmekte ve potansiyel çözümü temsil etmekte iken çözümü iyileştiren, kalitesini değerlendiren de arılardır. Bundan dolayı yukarıdaki açıklamalarda her iki özne de yöntemin açıklanmasında kullanılmıştır.

4. ABC TABANLI YENİ YAKLAŞIMLAR VE VE DENEYSEL ÇALIŞMALAR

Akay (2009)'a göre sezgisel veya sürü zekâsı tekniklerinin performanslarının değerlendirilmesinde çözüm kalitesi, hesaplama maliyeti ve sağlamlık olmak üzere üç temel kriter bulunmaktadır. Bu bakımdan her tekniğin bu kriterlere göre eksiklerinin giderilmesi üzerine çalışmalar yapılmıştır. Çözüm kalitesinin ve sağlamlığın korunmasıyla birlikte çalışmalardan bir kısmı hesaplama maliyetinin süre cinsinden azaltılmasını temel alırken bir kısmı ise çözüm kalitesinin iyileştirilmesi fakat hesaplama maliyetinin kabul edilebilir bölgede tutulmasını temel alır. Önerilen yeni yaklaşımlarımızda sağlamlık birden fazla çalıştırmadan elde edilen sonuçların birbirine yakınlığı yani standart sapma olarak alınmıştır. Ayrıca ABC algoritmasının ayrıklaştırılması ve ikili optimizasyon problemlerini çözecek şekilde yapılandırılması dikkate alındığında sağlamlık farklı çözüm uzaylarında yöntemin iyi sonuçlar elde edebilmesi olarak düşünülebilir.

Tez kapsamında önerilen yaklaşımlarda birinci kriter elbette çözüm kalitesinin kabul edilebilir süreler dahilinde arttırılması idi. Bundan dolayı yöntemlerin yetenekleri birleştirilerek yeni hibrit yöntemler de önerilmiştir. Hibrit olmasa da hiyerarşik düzeyde yöntemlerin dezavantajlarını kaldıracak şekilde yeni yaklaşımlar da bu tez çalışmasının içerisinde değerlendirilmiştir. Yapılan çalışmaların bu bölümde iki başlık halinde sunulması konu bütünlüğü açısından daha uygun bulunmuştur. Optimizasyon yöntemleri genel olarak optimizasyon problemlerine göre sınıflandırıldığı için aşağıdaki başlıklarda yapılan çalışmalar verilmiştir. Birinci başlık sürekli optimizasyon problemlerinin çözümü üzerine yapılan çalışmalardır. Bu konuda iki hibrit yöntem ve ABC yönteminin iki farklı şekilde iyileştirilmesi sunulmuştur. Diğer başlıkta ise ayrık optimizasyon problemlerinin çözümü üzerine çalışmalar yapılmıştır. Bu başlık altında ise ikinci çalışmanın temelini oluşturduğu için komşuluk operatörleriyle ayrıklaştırılmış ABC'nin TSP üzerindeki başarısı analiz edilmiş ve en iyi sonuçları üreten komşuluk operatörleri kaydedilmiştir. İkinci çalışmada ise ACO algoritması ile ayrık ABC algoritmasının hiyerarşik bir yapıda kullanımı sunulmuştur. Ayrık bir problem olmasına rağmen karar değişkenleri sadece 0 veya 1 değerini alabiliyorsa bu tip optimizasyon problemlerini ikili optimizasyon problemi adı verilir. ABC algoritması bu tip problemlerin çözümü için üçüncü çalışmada modifiye edilmiştir.

Ayrıca ifade edilmelidir ki bu bölümde iyileştirilmeler verilmekte deneysel çalışmalar bölümünde ise yöntemleri başarıları farklı problem türleri üzerinde

araştırılmaktadır. Ayrıca tez dönemi boyunca yeni önerilen bir hibrit çalışma Türkiye'nin toplam enerji talep tahmini için uygulanmış ve bir diğer çalışmada ise ABC, PSO, DE (farksal gelişim), GA (genetik algoritma) ve ABC'nin iyileştirilmesi için önerilen bir yaklaşım Türkiye'nin elektrik enerjisi talep tahmini için kullanılmış ve yöntemlerin performans karşılaştırmaları yapılmıştır.

Yukarıda ifade edilen ve yapılan çalışmalardan anlaşıldığı üzere yöntemler hem iyileştirilmiş, hem bazı uygulamalarda kullanılmıştır. Tez konusunun sürü zekâsının en popüler üç algoritmasını kapsaması beraberinde bazı dezavantajlara neden olmaktadır. Araştırılacak konu sayısının fazla oluşu zamanın ve imkânın üzerine çıkabilmekte iken bir diğer problem de konu bütünlüğü içerisinde tezin organizasyonudur. Birinci dezavantajı ortadan kaldırabilmek amacıyla çok popüler olan yöntemler (PSO ve ACO) hibrit yöntemler içerisinde kullanılmıştır. Diğerlerine göre nispeten daha az popüler olan yöntem (ABC) üzerine iyileştirmeler yapılmış, farklı problem türlerini çözecek şekilde yapılandırılmış ve diğer yöntemlerle birlikte hibritleştirilmiş veya hiyerarşik bir çerçevede ele alınmıştır. İkinci dezavantaj olan tezin organizasyonu ise önerilen hibritleştirmelerin veya iyileştirmelerin problem türü tabanında verilmesi ile giderilmeye çalışılmıştır.

4.1. Sürekli Optimizasyon Problemlerinin Çözümü için Yapılan Çalışmalar

4.1.1. ABC'nin Gözcü Arıları için Yeni Bir Komşu Önerisi (CABC)

Temel ABC algoritmasında hem görevli arılar hem de gözcü arılar kendi kaynaklarını iyileştirmek için komşu bir yiyecek kaynağına ihtiyaç duymaktadırlar. Bu komşu yiyecek kaynağının pozisyonu ile kendi yiyecek kaynağının pozisyonu arasında bir araştırma yapmaktadırlar. ABC algoritması bal arılarının yiyecek arama davranışı üzerine kurulmasına rağmen bal arılarının arasındaki bilgi paylaşımı da bu algoritma içerisinde kullanılmaktadır. Görevli arılar hafızalarındaki yiyecek kaynağı pozisyonunun bilgisini kaynaktaki nektar miktarına bağlı olarak (çözümün kalitesine bağlı olarak) gözcü arılarla paylaşmaktadırlar. Gözcü arılar hangi yiyecek kaynağını iyileştirmek amacıyla seçeceklerine paylaşılan bilgiyi (çözümün kalitesi) kullanarak karar vermektedirler (Denklem 3.15). Fakat yiyecek kaynağının komşuluğundaki kaynağı belirlemek tamamen rastgele yapılmaktadır. Çıkarılan sonuç şudur ki gözcü arılar tüm kaynaklar hakkında paylaşılan bilgiye sahip olmasına rağmen komşu

seçiminde tamamen rastgele davranmaktadırlar. Bu rastgeleliği ortadan kaldırmak için birinci yaklaşım popülasyonun en iyi çözümünün komşu olarak verilmesi popülasyonun hızlı bir durağanlaşma davranışına sürüklemektedir. Diğer bir deyişle popülasyon çok hızlı bir şekilde popülasyonun en iyi çözümüne doğru yaklaşmakta ve erken yakınsamaya sebep olmaktadır. Bundan dolayı global en iyi çözümün verilmesi yerine belirli sayıda yiyecek kaynağı pozisyonunun çaprazlanmasıyla elde edilen yeni yiyecek kaynağı pozisyonu (bu pozisyondaki yiyecek kaynağı gerçek anlamda bir arı tarafından değerlendirilmiş bir çözüm olmamasına rağmen) gözcü arılar için komşu olarak kullanılmaktadır (Kıran ve Gündüz, 2012). Çaprazlama operasyonunun detayları ve algoritma üzerinde yapılan değişiklik/yenilik aşağıda sunulmuştur.

4.1.1.1. Çaprazlama Operasyonu

Gözcü arı fazına geçilmeden hemen önce görevli arılarının çözümlerinin seçilme ihtimallerinin hesaplandığına ABC algoritması anlatılırken değinilmişti. Buna ek olarak bir "çiftleşme havuzu" da burada oluşturulmaktadır. Gözcü arılar tarafından yapılan yerel araştırmanın iyileştirilmesi için çiftleşme havuzuna popülasyonun en iyilerinden belirli sayıda görevli arının çözümü yerleştirilmektedir. Bundan sonra yapılan işlem ise seçilen bir gözcü arı için çiftleşme havuzundan yeni bir yiyecek kaynağı pozisyonunun çaprazlama suretiyle elde edilmesidir.

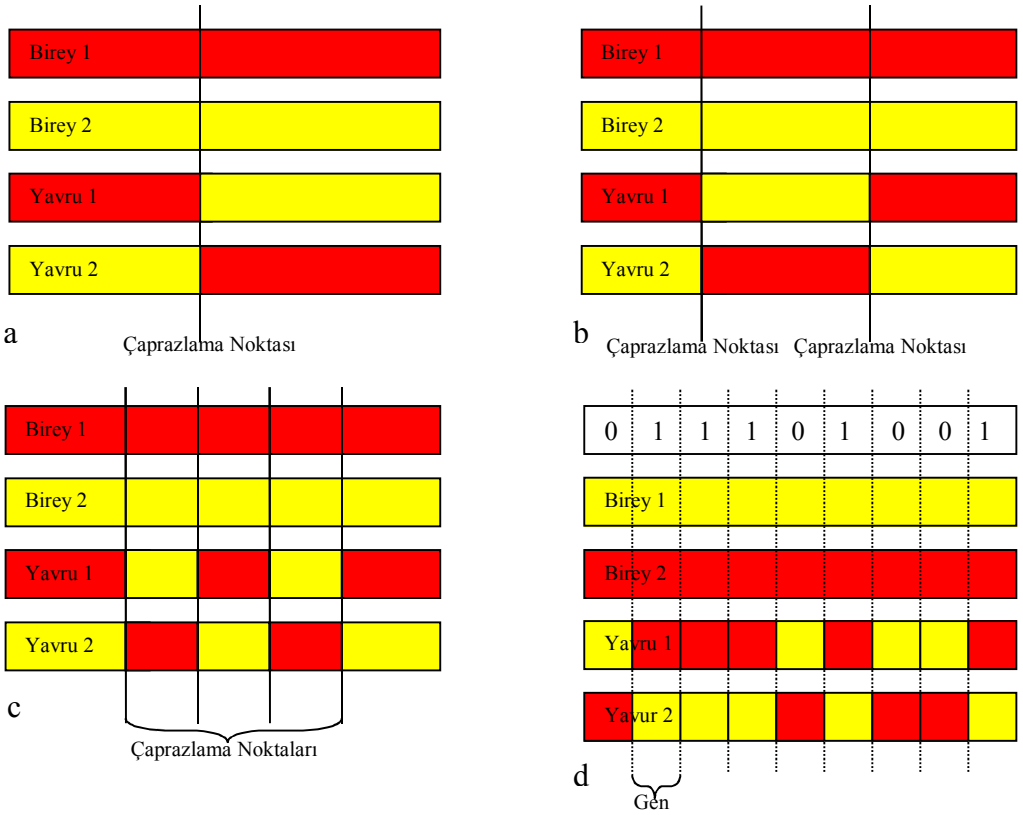
Çaprazlama için sadece bir çaprazlama operatörü kullanılmamış çok bilinen çaprazlama operatörlerinin kullanılmasından ortaya çıkan sonuçlar da çalışmada kıyaslanmıştır. Çaprazlama operatörleri kısaca aşağıda açıklanmıştır ve Şekil 4.1'de gösterilmiştir.

Tek Noktalı Çaprazlama: Seçilen iki yiyecek kaynağı pozisyonu tek noktadan kesilerek elde edilen pozisyon parçaları çapraz şekilde birleştirilir.

İki Noktalı Çaprazlama: Rastgele seçilen iki çaprazlama noktasından kesilerek yiyecek kaynaklarının pozisyonları 3 parçaya ayrılır. Elde edilen yeni birinci pozisyonun birinci ve üçüncü parçası birinci pozisyondan, ikinci parçası ikinci pozisyondan alınır. Aynı şekilde elde edilen yeni ikinci pozisyonun birinci ve üçüncü parçası ikinci pozisyondan, ikinci parçası ise birinci pozisyondan alınır.

Çok Noktalı Çaprazlama: Rastgele seçilen ikiden fazla çaprazlama noktasından yiyecek kaynakları pozisyonları kesilirler. İki yeni yiyecek kaynağı pozisyonu elde edileceğinde çapraz şekilde bu noktalardan ayrılan parçalar birleştirilir.

Uniform Çaprazlama: Çok noktalı çaprazlamaya benzer şekilde çalışır fakat çaprazlama işi pozisyonun boyutu tabanında yapılır. Seçilen uniform çaprazlama noktalarından pozisyonların boyutları yer değiştirilir.



Şekil 4.1. Gözcü arılar için komşu yiyecek kaynağı elde etmek için kullanılan çaprazlama operatörleri

a) tek noktadan (one-point) b) iki noktadan (two-point) c) çok noktadan (multi-points) d) Uniform

4.1.1.2. Algoritmadaki Yenilik

ABC algoritmasında hem görevli arılar hem de gözcü arılar için aynı araştırma denklemini kullanılmaktadır. Görevli arıların global düzeyde araştırma yaptığını gözcü arıların da görevli arıların çözümleri etrafında lokal araştırmaya odaklandığını düşünürsek aynı denklemin kullanılması yöntemin hem lokal araştırma yeteneğini hem de yakınsama yeteneğini zayıflatmaktadır. Bundan dolayı gözcü arıların lokal araştırma yöneltilmesi amacıyla Denklem 3.13 aşağıdaki şekilde geliştirilmiştir.

$$\begin{aligned}
&\overrightarrow{V_i} = \overrightarrow{X_i} && (a)\\
&V_{ij}(t) = X_{ij}(t) + \varphi(X_{ij}(t) - B_{ij}(t)) && (b)\\
&\quad i = 1,2,...,N,\ k \in \{1,2,...,N\} && (c)\\
&\quad j \in \{1,2,...,D\} && (d)\\
&\quad i \neq k && (e)
\end{aligned} \qquad (4.1)$$

Denklemdeki tüm terimleri Denklem 3.13 ile aynı anlamı ifade etmekle birlikte Denklem 3.13'deki $X_{kj}(t)$ komşu yiyecek kaynağı pozisyonu $B_{ij}(t)$ ile değiştirilmiştir. $B_{ij}(t)$, t anında i. gözcü arı için çaprazlama operatörlerinden biri kullanılarak elde edilmiş komşu yiyecek kaynağı pozisyonudur.

4.1.1.3. Deneysel Sonuçlar

Algoritmalar için iki deney tanımlanmıştır. Birinci deneyde Tablo 4.1'de verilen literatürdeki 10 tane test fonksiyonu üzerinde temel algoritma ile önerilen yöntemlerin çözüm kalitesi, sağlamlığı ve süreleri karşılaştırılmıştır. İkinci deneyde ise Türkiye'nin elektrik enerjisi talebi için 1979-2005 arası veriler (Tablo 4.2) kullanılarak bir katsayı belirleme çalışması üzerinde yöntemlerin çözüm kalitesi ve yakınsama hızı test edilmiştir.

Tablo 4.1. Kıyas Fonksiyonları

Fonksiyon	Boyut (D)	C*	Araştırma Uzayı	Formül	Minimum
f_1 (SixHumpCamelBack)	2	MN	[-5,5]	$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	-1.03163
f_2 (Matyas)	2	UN	[-10,10]	$f(x) = 0.26(x_1^2 + x_2^2) - 0.48(x_1x_2)$	0
f_3 (Booth)	2	MS	[-10,10]	$f(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$	0
f_4 (Schaffer)	2	MN	[-100,100]	$f(x) = 0.5 + \frac{\sin^2(\sqrt{x_1^2 + x_2^2}) - 0.5}{(1 + 0.001(x_1^2 + x_1^2))^2}$	0
f_5 (Trid)	10	UN	[-D^2,D^2]	$f(x) = \sum_{i=1}^{D}(x_i - 1)^2 - \sum_{i=2}^{D} x_i x_{i-1}$	-210
f_6 (SumSquares)	10,30,50	US	[-10,10]	$f(x) = \sum_{i=1}^{D} i x_i^2$	0
f_7 (Sphere)	10,30,50	US	[-100,100]	$f(x) = \sum_{i=1}^{D} x_i^2$	0
f_8 (Rastrigin)	10,30,50	MS	[-5.12,5.12]	$f(x) = \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i) + 10]$	0
f_9 (Schwefel)	10,30,50	MS	[-500,500]	$f(x) = \sum_{i=1}^{D} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	-418.9829*D
f_{10} (Rosenbrock)	10,30,50	UN	[-30,30]	$f(x) = \sum_{i=1}^{D-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + \left(x_i - 1\right)^2\right]$	0

* U: Unimodal, M: Multimodal, S: Ayrıklaştırabilir, N:Ayrıklaştırılamaz

Tablo 4.2. Türkiye'nin 1979-2005 arası Enerji Talebi, Gayrisafi Milli Hasıla (GDP), Nüfus, İthalat, İhracat verileri

Yıl	Enerji Talebi (MTOE)	GDP ($\$10^9$)	Nüfus (10^6)	İthalat ($\$10^9$)	İhracat ($\$10^9$)
1979	30.71	82.00	43.53	5.07	2.26
1980	31.97	68.00	44.44	7.91	2.91
1981	32.05	72.00	45.54	8.93	4.70
1982	34.39	64.00	46.69	8.84	5.75
1983	35.70	60.00	47.86	9.24	5.73
1984	37.43	59.00	49.07	10.76	7.13
1985	39.40	67.00	50.31	11.34	7.95
1986	42.47	75.00	51.43	11.10	7.46
1987	46.88	86.00	52.56	14.16	10.19
1988	47.91	90.00	53.72	14.34	11.66
1989	50.71	108.00	54.89	15.79	11.62
1990	52.98	151.00	56.10	22.30	12.96
1991	54.27	150.00	57.19	21.05	13.59
1992	56.68	158.00	58.25	22.87	14.72
1993	60.26	179.00	59.32	29.43	15.35
1994	59.12	132.00	60.42	23.27	18.11
1995	63.68	170.00	61.53	35.71	21.64
1996	69.86	184.00	62.67	43.63	23.22
1997	73.78	192.00	63.82	48.56	26.26
1998	74.71	207.00	65.00	45.92	26.97
1999	76.77	187.00	66.43	40.67	26.59
2000	80.50	200.00	67.42	54.50	27.78
2001	75.40	146.00	68.37	41.40	31.33
2002	78.33	181.00	69.30	51.55	36.06
2003	83.84	239.00	70.23	69.34	47.25
2004	87.82	299.00	71.15	97.54	63.17
2005	91.58	361.00	72.97	116.77	73.48

Parametre Ayarlaması: Deneysel çalışmada yöntemlerin popülasyon boyutu 100 olarak alınmıştır ve popülasyonun yarısı görevli ve diğer yarısı da gözcü arılardan oluşmaktadır. Kâşif arı oluşumunu kontrol eden limit parametresi $Limit = (N/2) \times D$ olarak elde edilmiştir ve her çevrimde bir tane kâşif arı oluşumuna izin verilmiştir. Durdurma kriteri olarak maksimum çevrim sayısı kullanılmıştır ve bu değer optimize edilecek karar değişkenlerinin sayısına göre ayarlanmıştır. İki boyutlu problemler için 200, 10 boyutlu problemler için 1000, 30 boyutlu problemler için 3000 ve 50 boyutlu problemler için 5000 olarak alınmıştır. Önerilen yöntemin kendine özgü kontrol parametreleri ise işçi arı sayısının %10'unun çiftleşme havuzuna yerleştirilmesi ve çok noktalı çaprazlama ve uniform çaprazlama için 5 noktanın rastgele seçilmesidir.

Yukarıdaki parametreler ile yöntemler kıyas fonksiyonlarının minimum değerlerini bulmak amacıyla 30'ar defa çalıştırılmış ve elde edilen ortalama sonuçlar ve standart sapmaları Tablo 4.3'de ve ortalama çalıştırma süreleri Tablo 4.4.'de verilmiştir.

Tablo 4.3. ABC ve CABC algoritmaları ile elde edilen sonuçların karşılaştırılması

Fonksiyon			**Orijinal ABC**		**$ABC_{one\text{-}point}$**		**$ABC_{two\text{-}point}$**		**$ABC_{multi\text{-}point}$**		**$ABC_{uniform}$**	
No	D	Minimum	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
f_1	2	-1.03163	**-1.03163**	**0**	**-1.03163**	**0**	-	-	-	-	-	-
f_2	2	0	1.20E-04	1.00E-04	**5.19E-05**	**7.35E-05**	-	-	-	-	-	-
f_3	2	0	3.29E-11	4.99E-11	**3.95E-14**	**6.81E-14**	-	-	-	-	-	-
f_4	2	0	8.02E-05	1.81E-04	**7.14E-05**	**2.14E-04**	-	-	-	-	-	-
f_5	10	-210	-209.77	0.16	-209.78	0.51	-209.75	0.36	-209.81	0.43	**-209.93**	**0.08**
f_6	10	0	7.89E-017	1.67E-017	8.24E-17	1.46E-17	**7.74E-17**	**1.6E-17**	7.84E-17	1.41E-17	7.98E-017	1.57E-017
	30		4.90E-016	4.57E-017	4.657E-16	7.50E-17	**4.22E-16**	**8.81E-17**	4.40E-16	8.25E-17	4.66E-016	6.92E-017
	50		9.59E-016	1.23E-016	8.68E-16	9.55E-17	8.75E-16	1.06E-16	**8.49E-16**	**1.15E-16**	8.76E-016	1.12E-016
f_7	10	0	**7.85E-017**	**1.88E-017**	8.10E-17	1.48E-17	8.70E-17	1.96E-17	8.30E-17	1.14E-17	8.38E-017	2.87E-017
	30		4.75E-016	5.92E-017	4.62E-16	8.80E-17	4.55E-16	7.69E-17	**4.50E-16**	**8.58E-17**	4.73E-016	5.52E-017
	50		9.57E-016	1.10E-016	8.94E-016	1.23E-016	8.33E-16	1.07E-16	8.71E-16	1.17E-16	**8.28E-016**	**9.97E-016**
f_8	10	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	30		**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	50		**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_9	10	-4189.829	**-4189.829**	**0**	**-4189.829**	**0**	**-4189.829**	**0**	**-4189.829**	**0**	**-4189.829**	**0**
	30	-12569.487	**-12569.487**	**0**	**-12569.487**	**0**	**-12569.487**	**0**	**-12569.487**	**0**	**-12569.487**	**0**
	50	-20949.145	**-20949.144**	**0**	**-20949.144**	**0**	**-20949.144**	**0**	**-20949.144**	**0**	**-20949.144**	**0**
f_{10}	10	0	0.0428	0.0535	0.0470	0.08994	0.0473	0.0788	**0.0240**	**0.0314**	0.0379	0.0648
	30		0.0343	0.0409	0.0367	0.04213	0.0838	0.1373	0.0584	0.1299	**0.0285**	**0.0515**
	50		0.0656	0.1132	0.0699	0.10065	0.0982	0.1290	0.0842	0.1271	**0.0539**	**0.0933**

Tablo 4.4. Yöntemlerin ortalama çalışma süreleri

Fonksiyon	**Boyut**	**Geçen Süre (Saniye)***				
		Temel ABC	$ABC_{onepoint}$	$ABC_{two\text{-}point}$	$ABC_{multi\text{-}point}$	$ABC_{uniform}$
f_1 (Six Hump Camel Back)	2	1.3948	1.8534	-	-	-
f_2 (Matyas)	2	1.4125	1.76211	-	-	-
f_3 (Booth)	2	1.4187	1.7656	-	-	-
f_4 (Schaffer)	2	1.4564	1.8731	-	-	-
f_5 (Trid)	10	7.7739	9.6446	10.5986	10.9867	10.9577
f_6 (SumSquares)	50	38.8197	53.5744	53.7117	55.4224	54.8307
f_7 (Sphere)	50	37.9113	51.5048	52.0082	54.1913	53.4573
f_8 (Rastrigin)	50	38.4297	57.3003	58.3818	63.3698	59.2113
f_9 (Schwefel)	50	38.4641	69.6321	72.6918	74.7356	74.7984
f_{10} (Rosenbrock)	50	38.5371	56.3754	56.7983	57.7755	56.6189

* Çalışma süreleri Matlab v.7.0.4 platformundaki tic ve toc fonksiyonları ile elde edilmiştir.

Temel ABC ve tek noktadan çaprazlamalı ABC algoritması aynı zamanda enerji talep tahmini için lineer denklemin (Denklem 4.2) katsayılarını bulmak amacıyla da uygulanmıştır. Lineer denklemin katsayılarını bulmak için çözümün kalitesi Denklem 4.3 kullanılarak değerlendirilmiştir.

$$E_{linear} = w_1 \times X_1 + w_2 \times X_2 + w_3 \times X_3 + w_4 \times X_4 + w_5 \quad (4.2)$$

Denklemdeki w'ler optimize edilmek istenen katsayıları X'ler lineer model için girişleri temsil etmektedir. Sırasıyla X_1 gayrı safi milli hasılayı, X_2 nüfusu, X_3 ithalatı ve X_4 ihracat değerlerini göstermektedir.

$$\min f(w) = \sum_{k=1}^{R} (E_k^{observed} - E_k^{predicted})^2 \quad (4.3)$$

Arılar tarafından elde edilen bir çözüm Denklem 4.2 kullanılarak her yıl için tahmini değerler elde edilir. Denklem 4.3 ise tahmin edilen ile gerçekleşen arasındaki farkların karesinin toplamıyla elde edilen çözümün kalitesini belirlemede kullanılır. Çözümün kalitesi Denklem 3.12 ile tespit edilmektedir.

Bu problem için önerilen yöntemlerle elde edilen lineer denklemin katsayıları aşağıda verilmiştir. Denklem 4.4, 4.5, 4.6, 4.7, 4.8 sırasıyla temel ABC, tek noktalı çaprazlamalı ABC, iki noktalı çaprazlamalı ABC, çok noktalı çaprazlamalı ABC ve uniform çaprazlamayı kullanan ABC yöntemleriyle elde edilen en iyi katsayılarla kurulmuş denklemleri göstermektedir. Ayrıca denklemlerin altında Denklem 4.3 ile elde edilen hata değerleri de bulunmaktadır.

$$E_{linear} = 0.065622 \bullet X_1 + 1.72044 \bullet X_2 + 0.212867 \bullet X_3 - 0.41529 \bullet X_4 - 50.4559 \quad (4.4)$$
$$f(w) = 84.31805$$

$$E_{linear} = 0.003759 \bullet X_1 + 1.912559 \bullet X_2 + 0.373371 \bullet X_3 - 0.48326 \bullet X_4 - 55.9091 \quad (4.5)$$
$$f(w) = 41.70295$$

$$E_{linear} = 0.003762 \bullet X_1 + 1.912553 \bullet X_2 + 0.373361 \bullet X_3 - 0.48325 \bullet X_4 - 55.9091 \quad (4.6)$$
$$f(w) = 41.70295$$

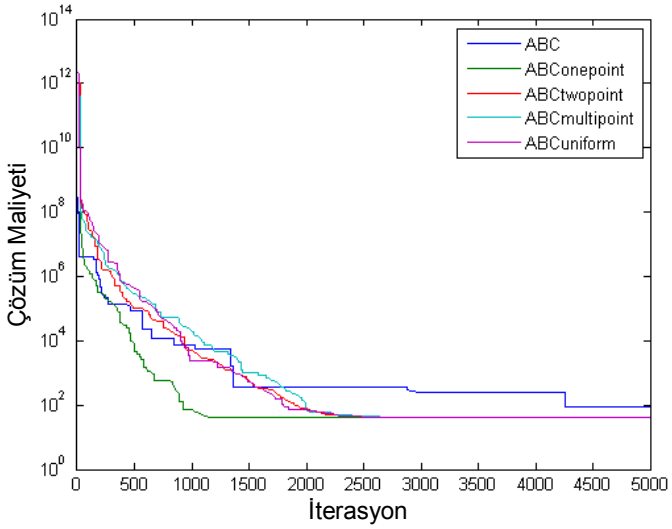
$$E_{linear} = 0.003756 \bullet X_1 + 1.912578 \bullet X_2 + 0.373404 \bullet X_3 - 0.4833 \bullet X_4 - 55.9099 \quad (4.7)$$
$$f(w) = 41.70295$$

$$E_{linear} = 0.003768 \bullet X_1 + 1.912523 \bullet X_2 + 0.373315 \bullet X_3 - 0.48319 \bullet X_4 - 55.908 \quad (4.8)$$
$$f(w) = 41.70295$$

Literatürde elde edilen katsayılarla son yıllara ait girişler kullanılarak modelin geçerliliği kontrol edilmektedir. Bu doğrulama yukarıda verilen katsayılar ve denklemler kullanılarak elde edilen modellerin geçerliliğini kontrol etmek amacıyla Tablo 4.5'te verilmiştir. Ayrıca temel ABC'nin ve önerilen yöntemlerin yakınsama grafiği de Şekil 4.2'de gösterilmiştir.

Tablo 4.5. Modellerin Doğrulaması

Yıl	Gerçekleşen Enerji Talebi (MTOE)	Orijinal ABC		$ABC_{onepoint}$	
		Tahmini Enerji Talebi (MTOE)	Hata Değeri (%)	Tahmini Enerji Talebi (MTOE)	Hata Değeri (%)
1996	69.86	69.08	-1.12	**69.71**	**-0.21**
1997	73.78	71.37	-3.27	**72.31**	**-1.99**
1998	74.71	**73.53**	**-1.58**	73.3	-1.89
1999	76.77	73.72	-3.97	**74.18**	**-3.37**
2000	80.50	78.73	-2.2	**80.71**	**0.26**
2001	75.40	72.55	-3.78	**75.72**	**0.42**
2002	78.33	76.65	-2.14	**79.13**	**1.02**
2003	83.84	81.19	-3.16	**82.36**	**-1.77**
2004	87.82	86.1	-1.96	**87.18**	**-0.73**
2005	91.58	93.12	1.68	**93.1**	**1.66**



Şekil 4.2. Yöntemlerin tahmin probleminin çözümüne yakınsaması

4.1.2. Yapay Arı Kolonisi ile Türkiye'nin Elektrik Enerjisi Talep Tahmini

Elektrik enerjisi yaşamın her alanında sıklıkla kullanılmaktadır. Geleceğe dönük ülkenin elektrik talebinin belirlenmesi politika üreticiler için ülkenin ihtiyaçlarına karşılık alınacak tedbirlerin ortaya konulması açısından oldukça önemlidir. Gelecekte ne kadar enerjiye ihtiyaç olacağı tespit edilebilirse karar vericiler için bugünden önlem almak ve değişik enerji kaynaklarının (Rüzgar, güneş, nükleer vb.) faaliyete alınması ülkenin hayatını idame ettirmek açısından faydalı olacaktır. Karınca kolonisi optimizasyonu (Toksarı, 2009) ve parçacık sürü optimizasyonu yöntemi (Ünler, 2008) literatürde enerji talebinin tahmini için uygulanmış yöntemlerdendir. ABC'nin test fonksiyonları üzerindeki PSO'ya nispeten göstermiş olduğu üstünlük bu çalışmanın da yapılmasını ve diğer yöntemlerle karşılaştırılmasını ABC açısından önemli kılmaktadır.

Ayrıca elektrik enerjisi talep tahmini için daha fazla yöntemin uygulanarak sonuçların elde edilmesi karar vericiler açısından da oldukça faydalı olacaktır.

Bu çalışmada (elektrik enerjisi tahmini) temel ABC algoritmasının yanı sıra bir önceki bölümde verilen çaprazlamaya dayalı ABC yöntemi ve yeni bir komşuluk önerisine dayalı (popülasyonun en iyi çözümünün komşu olarak alınması) ABC algoritması problemin çözümü için uygulanmıştır. ABC ve varyantları ile elde edilen sonuçlar PSO ve ACO ile elde edilen sonuçlar ile kıyaslanmıştır (Kıran ve ark., 2012b).

4.1.2.1. Elektrik Enerjisi Talebinin Modellenmesi

Tahmin geleceğe yönelik olduğundan dolayı geçmiş veriler kullanılarak lineer ve kuadratik iki model oluşturularak modellerin katsayıları belirlenmeye çalışılmıştır. Lineer model Denklem 4.2 ile verilmiştir ve kuadratik model ise Denklem 4.9'da tanımlanmıştır.

$$\begin{aligned} E_{quadratik} = {} & w_1 \times X_1 + w_2 \times X_2 + w_3 \times X_3 + w_4 \times X_4 + w_5 \times X_1 \times X_2 + \\ & w_6 \times X_1 \times X_3 + w_7 \times X_1 \times X_4 + w_8 \times X_2 \times X_3 + w_9 \times X_2 \times X_4 + \\ & w_{10} \times X_3 \times X_4 + w_{11} \times X_1^2 + w_{12} \times X_2^2 + w_{13} \times X_3^2 + w_{14} \times X_4^2 + \\ & w_{15} \end{aligned} \tag{4.9}$$

Denklem 4.9'daki w_i $(i = 1,2,...,15)$ optimize edilmek istenen katsayıları göstermektedir. X_j $(j = 1,2,...,4)$ ise modelin girişleridir. Giriş olarak literatürde sıklıkla ortak kullanılan gayrı safi milli hasıla (X_1), popülasyon (X_2), ithalat (X_3) ve ihracat (X_4) verileri kullanılmıştır ve elektrik talebinin en çok bu göstergelerden etkilendiğine inanılır (Toksarı, 2007, 2009; Ünler, 2008). Bu girişlere karşılık 1979-2013 yılları arasında Türkiye'nin elektrik enerjisi tüketimi Tablo 4.6'de gösterilmiştir. Tablodaki 1979-2006 arası veriler ile model katsayıları belirlenmektedir ve 1997-2006 arası veriler ile modellerin doğrulaması yapılmaktadır. Sistemlerin doğrulamasından sonra ise üretilen senaryolar ve 2007-2013 yılları arası gerçekleşen verilen ile literatürdeki benzer çalışmalardan farklı olarak modellerin başarıları test edilmeye çalışılmıştır.

Amaç bu göstergeler kullanılarak lineer ve kuadratik denklemdeki optimum katsayıların belirlenmesidir. Her bulunan katsayı kümesinin veriye uygunluğu Denklem

4.3 ile verilen amaç fonksiyonu ile değerlendirilir. Giriş verilerine en uygun modelin bulunması için Denklem 4.3 ile verilen fonksiyonun minimize edilmesi gerekmektedir.

4.1.2.2. Algoritmaların Probleme Uyarlanması

Denklemlerde (4.2 ve 4.9) verilen w değerleri modelin çalıştırılabilmesi veya denklemlerden değer elde edilebilmesi için gerekli olan katsayılardır. ABC'nin yiyecek kaynağı pozisyonu w değerlerinin bir kümesidir ve bu kümenin eleman sayısı lineer model için 5 ve kuadratik model için 15'tir. Yani ABC'nin arıları 4 ve 15 boyutlu uzayda hareket edeceklerdir.

Tek ve iki noktalı çaprazlamayı kullanan ABC yöntemlerinin yanı sıra ABC'deki gözcü arıların popülasyonun en iyi çözümünden etkilenmesini sağlayacak şekilde gözcü arılar için kullanılan araştırma denklemi aşağıdaki şekilde değiştirilmiştir.

$$\begin{aligned} &\overrightarrow{V_i} = \overrightarrow{X_i} && (a)\\ &V_{ij}(t) = X_{ij}(t) + \varphi(X_{ij}(t) - Best_j(t)) && (b)\\ &\qquad i = 1,2,...,N,\ k \in \{1,2,...,N\} && (c)\\ &\qquad j \in \{1,2,...,D\} && (d)\\ &\qquad i \neq k && (e) \end{aligned} \qquad (4.10)$$

Denklemdeki $Best_j$ popülasyon tarafından o ana kadar bulunmuş olan en iyi yiyecek kaynağının j. boyutudur.

Görevli arılar kendilerine atanmış olan yiyecek kaynakları etrafında araştırma için Denklem 3.13'ü kullanmakta iken gözcü arılar görevli arılar tarafından paylaşılan bilgiyi daha verimli olarak Denklem 4.10'da kullanırlar.

Aynı şekilde tek ve iki noktalı çaprazlama ile elde edilen çözümler de gözcü arılar için komşu yiyecek kaynağı pozisyonu olarak kullanılmaktadır.

4.1.2.3. Elde Edilen Sonuçlar ve Karşılaştırmalar

Katsayıların en uygun değerlerinin bulunması için ABC ve önerilen iyileştirmeler problemin çözümü için uygulanmıştır. Popülasyon büyüklüğü kıyaslanan tüm yöntemler için 200 olarak seçilmiştir. Algoritmanın durdurma koşulu olarak ise maksimum çevrim sayısı (10,000) kullanılmıştır. Ayrıca ABC ve varyantları için limit değeri lineer denklemin katsayılarının bulunmasında 500, kuadratik denklemin

katsayılarının bulunması deneyinde ise 1500 olarak ayarlanmıştır. Elde edilen katsayılar lineer ve kuadratik denklemlerde aşağıdaki şekildedir. Denklemlerdeki ABCEE temel ABC yönteminin, BABCEE gözcü arılar için en iyi çözümü komşu olarak öneren yöntemin, $CABCEE_{onepoint}$ tek noktalı çaprazlamayı kullanan yöntemin ve $CABCEE_{twopoint}$ iki noktalı çaprazlamayı kullanan yöntemin kısaltması olarak kullanılmıştır.

Tablo 4.6. Türkiye'nin 1979-2013 yıllarındaki elektrik talebi, gayrisafi milli hasılası, nüfusu, ithalat ve ihracat verileri

Yıllar	Elektrik Talebi(GWh)	GSMH (10^9)	Nüfus(10^6)	İthalat(10^9)	İhracat(10^9)
1979	23.566	82	43.530	5.07	2.26
1980	24.617	68	44.438	7.91	2.91
1981	26.289	72	45.540	8.93	4.70
1982	28.325	64	46.688	8.84	5.75
1983	29.568	60	47.864	9.24	5.73
1984	33.267	59	49.070	10.76	7.13
1985	36.361	67	50.306	11.34	7.95
1986	40.471	75	51.433	11.10	7.46
1987	44.925	86	52.561	14.16	10.19
1988	48.430	90	53.715	14.34	11.66
1989	52.602	108	54.893	15.79	11.62
1990	56.812	151	56.203	22.30	12.96
1991	60.499	150	57.305	21.05	13.59
1992	67.217	158	58.401	22.87	14.72
1993	73.432	179	59.491	29.43	15.35
1994	77.783	132	60.576	23.27	18.11
1995	85.552	170	61.644	35.71	21.64
1996	94.789	184	62.697	43.63	23.22
1997	105.517	192	62.480	48.56	26.26
1998	114.023	207	63.459	45.92	26.97
1999	118.485	187	64.345	40.67	26.59
2000	128.276	200	67.461	54.50	27.78
2001	126.871	146	68.618	41.40	31.33
2002	132.553	181	69.626	51.55	36.06
2003	141.151	239	70.712	69.34	47.25
2004	150.018	299	71.789	97.54	63.17
2005	160.794	361	72.065	116.77	73.48
2006	174.637	400	72.974	139.58	85.53
2007	198.085	649	70.58	170.06	107.27
2008	194.079	742	71.52	201.96	132.03
2009	210.434	617	72.56	140.93	102.14
2010	230.306	736	73.72	185.54	113.88
2011	242.369	774	74.72	240.84	134.91
2012	245.501	786	75.63	236.55	152.46
2013	198.085	820	76.67	251.65	151.80

$$ABCEE^{linear} = -108.7298X_1 + 3938.2350X_2 + 1019.4049X_3 - 618.0275X_4 - 156138.2314$$
$$R^2 = \%99.5$$

$$BABCEE^{linear} = -112.4510X_1 + 3957.1359X_2 + 1040.3041X_3 - 643.5624X_4 - 156832.8892$$
$$R^2 = \%99.5$$

$$CABCEE_{onepoint}^{linear} = -112.4534X_1 + 3957.1259X_2 + 1040.3234X_3 - 643.5912X_4 - 156833.7461$$
$$R^2 = \%99.5$$

$$CABCEE_{twopoint}^{linear} = -112.4534X_1 + 3957.1555X_2 + 1040.3234X_3 - 643.5912X_4 - 156833.7468$$
$$R^2 = \%99.5$$

$$\begin{aligned} ABCEE^{quadratic} = & -574.5468X_1 + 763.1509X_2 + 746.6424X_3 - 3519.5084X_4 + \\ & 12.0684X_1X_2 - 9.6536X_1X_3 + 27.5581X_1X_4 - 20.2666X_2X_3 + \\ & 111.2317X_2X_4 - 87.3341X_3X_4 - 0.7591X_1^2 - 6.7619X_2^2 + \\ & 52.9190X_3^2 - 74.4363X_4^2 + 6.9937 \end{aligned}$$

$$R^2 = \%99.58$$

$$\begin{aligned} BABCEE^{quadratic} = & -437.2762X_1 + 66.4875X_2 - 1900.3087X_3 + 2343.5533X_4 + \\ & 8.1060X_1X_2 - 9.5992X_1X_3 + 3.2229X_1X_4 + 14.5197X_2X_3 + \\ & 42.0556X_2X_4 - 52.3947X_3X_4 + 0.9148X_1^2 + 3.4128X_2^2 + \\ & 53.1459X_3^2 - 54.5552X_4^2 + 9226.8846 \end{aligned}$$

$$R^2 = \%99.75$$

$$\begin{aligned} CABCEE_{onepoint}^{quadratic} = & 94.1060X_1 + 332.2221X_2 - 2686.1968X_3 + 830.1608X_4 - \\ & 2.7873X_1X_2 + 9.2090X_1X_3 - 8.6491X_1X_4 + 27.5979X_2X_3 + \\ & 49.0494X_2X_4 - 22.4340X_3X_4 + 0.0274X_1^2 + 13.9565X_2^2 + \\ & 7.0607X_3^2 - 32.7737X_4^2 - 20594.6117 \end{aligned}$$

$$R^2 = \%99.77$$

$$\begin{aligned} CABCEE_{twopoint}^{quadratic} = & -155.1941X_1 + 25.1602X_2 - 828.9976X_3 - 524.4132X_4 + \\ & 5.6911X_1X_2 - 4.1521X_1X_3 + 5.9504X_1X_4 - 4.2434X_2X_3 + \\ & 77.6079X_2X_4 - 17.5840X_3X_4 - 0.4005X_1^2 + 6.6995X_2^2 + \\ & 34.5789X_3^2 - 85.0502X_4^2 + 307.9371 \end{aligned}$$

$$R^2 = \%99.78$$

Katsayıların belirlenmesi için 1979-2006 arası veriler kullanılmıştır ve önerilen yöntemler ile elde edilen modellerin tutarlılığı 1997-2006 arası gerçekleşen değerler ile

denklemlerden elde edilen değerlerin kıyaslaması ile kontrol edilmiştir. Bu kıyaslar Tablo 4.7, Tablo 4.8, Tablo 4.9 ve Tablo 4.10'da verilmiştir. Önerilen yöntemler ACO ve PSO ile birlikte değerlendirildiği için PSO ile elde edilen sonuçlar Tablo 4.11 ve ACO'nun sonuçları (Toksarı, 2009) Tablo 4.12'de verilmiştir. Ayrıca Şekil 4.3 R^2 değerlerinin bir kıyasını, Şekil 4.4 lineer model ile elde edilen sonuçların ve Şekil 4.5'te kuadratik model ile elde edilen sonuçların yöntemler arasındaki karşılaştırmasını sunmaktadır.

Tablo 4.7. *ABCEE* ile elde edilen sonuçların 1997-2006 yıllarına ait veriler ile doğrulanması

Yıllar	Gerçekleşen Elektrik Talebi(GWh)	Tahmin Edilen Elektrik Talebi (GWh)		Hata Değerleri(%)	
		Lineer Model	Kuadratik Model	Lineer Model	Kuadraik Model
1997	105.517	102.320	103.647	-3.02	-1.77
1998	114.023	101.414	110.332	-11.05	-3.23
1999	118.485	101.961	110.469	-13.94	-6.76
2000	128.276	126.182	130.538	-1.63	1.76
2001	126.871	121.062	115.637	-4.57	-8.85
2002	132.553	128.650	132.981	-2.94	0.32
2003	141.151	137.840	154.531	-2.34	9.47
2004	150.018	154.465	159.645	2.96	6.41
2005	160.794	162.043	168.939	0.77	5.06
2006	174.637	177.187	159.389	1.46	-8.73

Table 4.8. *BABCEE* ile elde edilen sonuçların 1997-2006 yıllarına ait veriler ile doğrulanması

Yıllar	Gerçekleşen Elektrik Talebi(GWh)	Tahmin Edilen Elektrik Talebi (GWh)		Hata Değerleri(%)	
		Lineer Model	Kuadratik Model	Lineer Model	Kuadratik Model
1997	105.517	102.435	103.668	-2.92	-1.75
1998	114.023	101.419	106.827	-11.05	-6.31
1999	118.485	101.957	106.064	-13.94	-10.48
2000	128.276	126.447	131.188	-1.42	2.27
2001	126.871	121.185	119.434	-4.48	-5.86
2002	132.553	128.753	131.993	-2.86	-0.42
2003	141.151	137.834	145.786	-2.34	3.28
2004	150.018	154.440	157.713	2.94	5.12
2005	160.794	161.930	160.555	0.70	-0.14
2006	174.637	177.116	171.046	1.41	-2.05

Tablo 4.9. $CABCEE_{onepoint}$ sonuçlarının 1997-2006 yıllarına ait veriler ile doğrulanması

Yıllar	Gerçekleşen Elektrik Talebi(GWh)	Tahmin Edilen Elektrik Talebi (GWh)		Hata Değerleri(%)	
		Lineer Model	Kuadratik Model	Lineer Model	Kuadratik Model
1997	105.517	102.435	103.546	-2.92	-1.86
1998	114.023	101.419	106.652	-11.05	-6.46
1999	118.485	101.957	103.826	-13.94	-12.37
2000	128.276	126.447	131.732	-1.42	2.69
2001	126.871	121.185	119.893	-4.48	-5.50
2002	132.553	128.753	130.580	-2.86	-1.48
2003	141.151	137.834	144.011	-2.34	2.02
2004	150.018	154.440	155.425	2.94	3.60
2005	160.794	161.930	165.750	0.70	3.08
2006	174.637	177.116	170.174	1.41	-2.55

Tablo 4.10. *CABCEE*$_{twopoint}$ sonuçlarının 1997-2006 yıllarına ait veriler ile doğrulanması

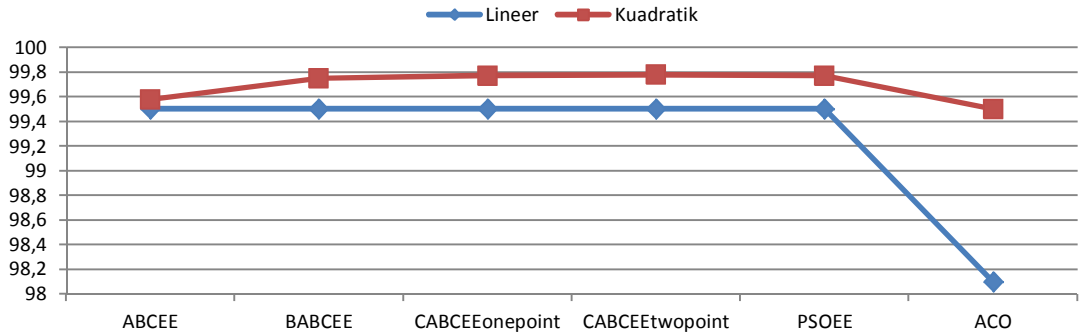
Yıllar	Gerçekleşen Elektrik Talebi(GWh)	Tahmin Edilen Elektrik Talebi (GWh)		Hata Değerleri(%)	
		Lineer Model	Kuadratik Model	Lineer Model	Kuadratik Model
1997	105.517	102.435	103.931	-2.92	-1.50
1998	114.023	101.419	105.631	-11.05	-7.35
1999	118.485	101.957	105.191	-13.94	-11.22
2000	128.276	126.447	130.567	-1.42	1.78
2001	126.871	121.185	118.532	-4.48	-6.57
2002	132.553	128.753	131.730	-2.86	-0.62
2003	141.151	137.834	145.184	-2.34	2.85
2004	150.018	154.440	157.386	2.94	4.91
2005	160.794	161.930	160.890	0.70	0.05
2006	174.637	177.116	171.733	1.41	-1.66

Tablo 4.11. ACO (Toksarı, 2009) sonuçların 1997-2006 yıllarına ait veriler ile doğrulanması

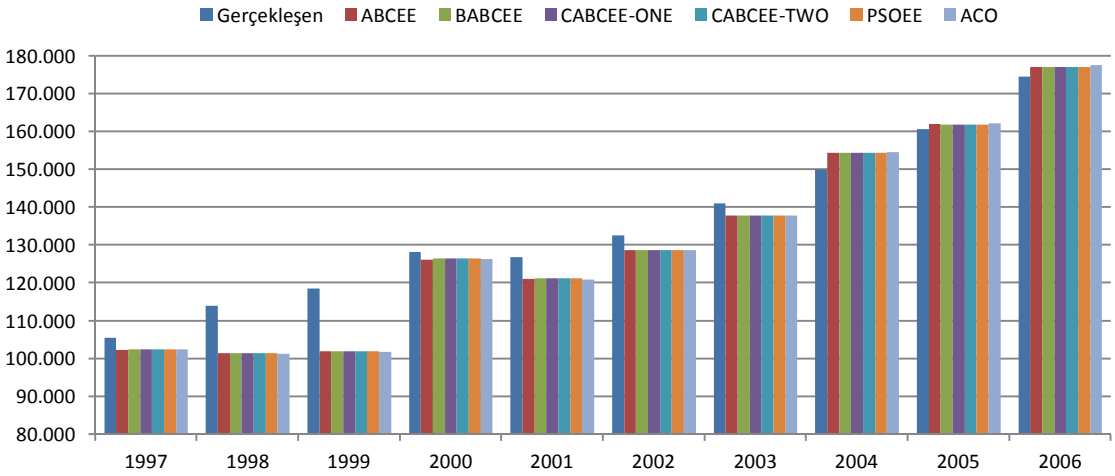
Yıllar	Gerçekleşen Elektrik Talebi(GWh)	Tahmin Edilen Elektrik Talebi (GWh)		Hata Değerleri(%)	
		Lineer Model	Kuadratik Model	Lineer Model	Kuadratik Model
1997	105.517	102.433	101.936	-2.92	-3.39
1998	114.023	101.365	110.643	-11.10	-2.96
1999	118.485	101.845	109.321	-14.04	-7.73
2000	128.276	126.397	129.396	-1.46	0.87
2001	126.871	121.010	123.629	-4.61	-2.55
2002	132.553	128.633	133.644	-2.95	0.82
2003	141.151	137.823	141.689	-2.35	0.38
2004	150.018	154.630	147.806	3.07	-1.47
2005	160.794	162.258	163.158	0.91	1.47
2006	174.637	177.612	172.819	1.70	-1.04

Table 4.12. *PSOEE* ile elde edilen sonuçların 1997-2006 yıllarına ait veriler ile doğrulanması

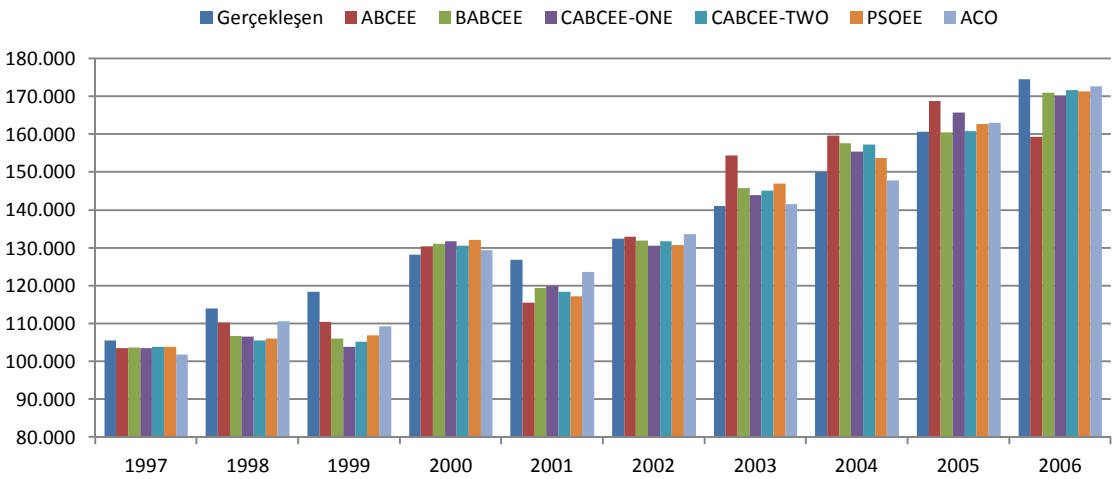
Yıllar	Gerçekleşen Elektrik Talebi(GWh)	Tahmin Edilen Elektrik Talebi (GWh)		Hata Değerleri(%)	
		Lineer Model	Kuadratik Model	Lineer Model	Kuadratik Model
1997	105.517	102.435	103.937	-2.92	-1.49
1998	114.023	101.419	106.043	-11.05	-6.99
1999	118.485	101.957	106.977	-13.94	-9.71
2000	128.276	126.447	132.200	-1.42	3.05
2001	126.871	121.185	117.191	-4.48	-7.62
2002	132.553	128.753	130.827	-2.86	-1.30
2003	141.151	137.834	147.050	-2.34	4.17
2004	150.018	154.440	153.819	2.94	2.53
2005	160.794	161.930	162.802	0.70	1.24
2006	174.637	177.116	171.425	1.41	-1.83



Şekil4.3. Sürü zekâsına dayanan tahmin metotlarının R^2 tabanlı karşılaştırılması



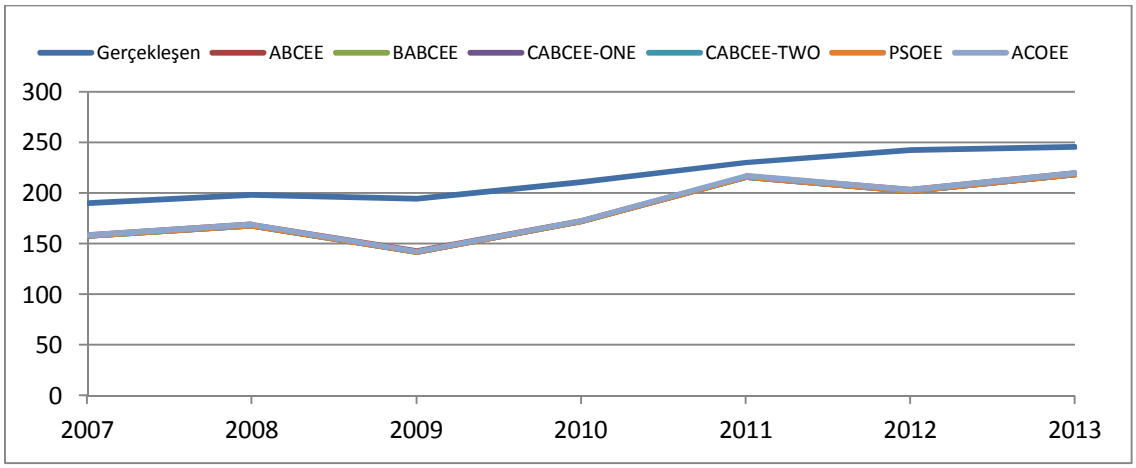
Şekil 4.4. Metotlar ile lineer model için elde edilen sonuçlarının karşılaştırılması



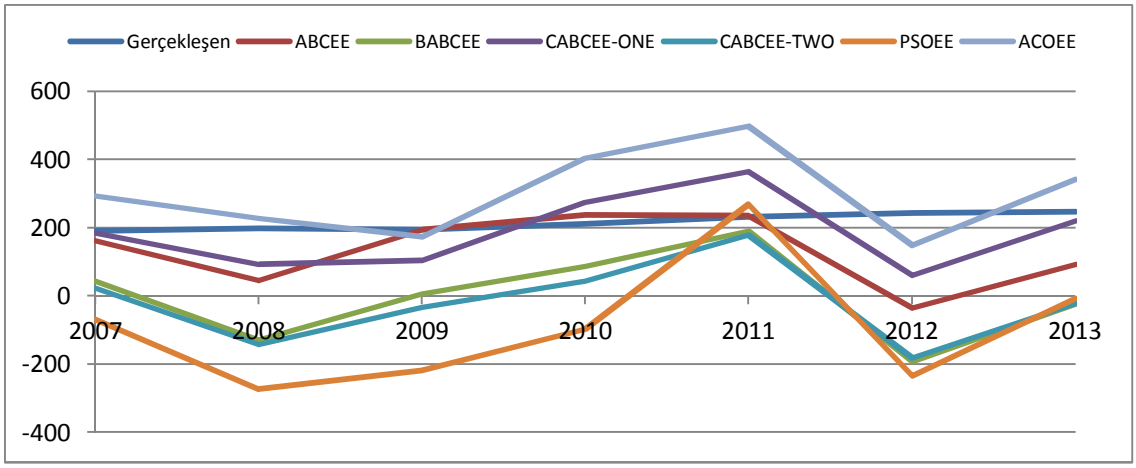
Şekil 4.5. Metotlar ile kuadratik model için elde edilen sonuçlarının karşılaştırılması

Bu çalışmada ele alınan yöntemler ile 1979-2006 yılları arası veriler kullanılarak modeller elde edilmiş ve 1997-2006 yılları arası veriler ile de yöntemlerin doğrulamaları yapılmıştır. 2007-2013 arası gerçekleşen enerji talebi için modellerin testleri Şekil 4.6 (Lineer Modeller) ve 4.7 (Kuadratik Modeller) ile sunulmuştur.

Şekil 4.6 ve 4.7'den görüleceğe üzere lineer modeller ile elde edilen sonuçlar kuadratik modelden daha tutarlıdır. Bunun temel nedeni sosyo-ekenomik göstergelerdeki doğrusal artışlar ve aralarındaki ilişkinin sıkı şekilde ifade edilememesi olarak gösterilebilir.



Şekil 4.6. 2007-2013 yılları arası gerçekleşen elektrik enerjisi talebi için yöntemler ile elde edilen lineer modellerin karşılaştırması

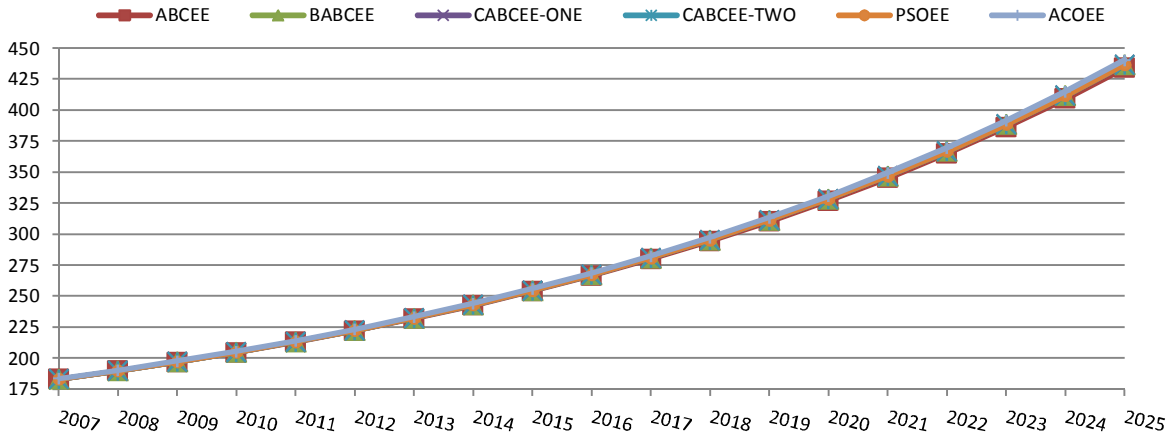


Şekil 4.7. 2007-2013 yılları arası gerçekleşen elektrik enerjisi talebi için yöntemler ile elde edilen kuadratik modellerin karşılaştırması

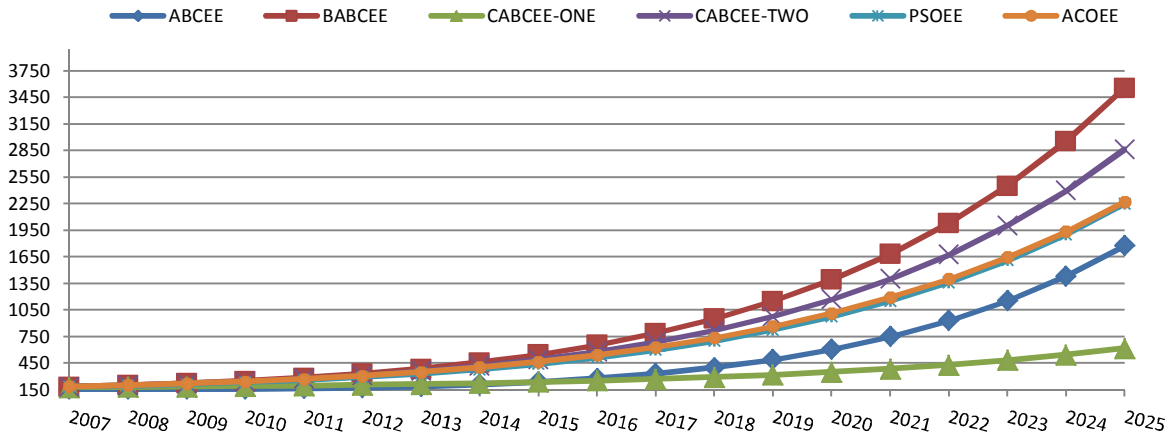
Bu çalışmada ABC ve ABC tabanlı yeni önerilen yöntemlerin elektrik enerjisi talep tahmini için uygulanabilirliğinin araştırılması temel amaç olsa da geleceğe yönelik tahmin değerlerinin ortaya konması da diğer bir amaçtır. Bundan dolayı Türkiye'nin elektrik enerjisi talebi için 3 senaryo hazırlanmış (Tablo 4.13) ve bu senaryolara göre 2025 yılına kadar tahmin edilen değerler Şekil 4.8, 4.9, 4.10, 4.11, 4.12 ve 4.13'de gösterilmiştir. Senaryolardaki değerler kullanılarak 2025 yılına kadar her göstergenin yıllık değeri hesaplanır. Bu değerler ile elde edilen lineer ve kuadratik formdaki denklemler ile çıkış yani tahmin edilen değerler hesaplanır. Tahminlerin gösterilmesi yöntemlerin elde ettiği sonuçları kıyaslama açısından şekil ile daha iyi ifade edileceği ve raporlanacak sonuçların tabloya sığdırılamadığından dolayı şekiller ile senaryolar için elde edilen değerler gösterilmiştir. Tahmin değerleri elde edilmek istenirse yukarıdaki süreç kullanılmalıdır.

Table 4.13. Senaryolar için girişlerin her yıl artış oranları

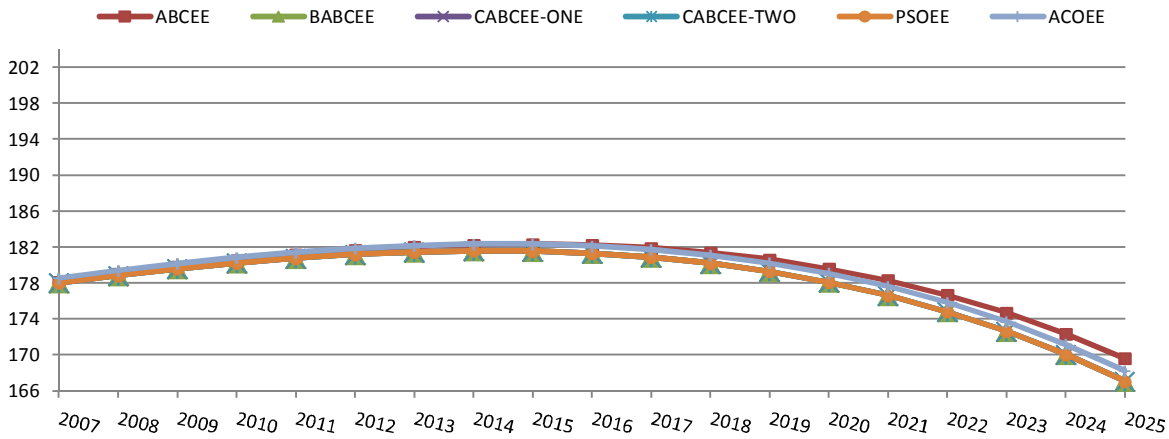
Senaryolar	Gayrisafi Milli Hâsıla Artış Oranı (%)	Nüfus Artış Oranı (%)	İthalat Artış Oranı (%)	İhracat Artış Oranı (%)
Senaryo 1	3.50	0.10	7.00	5.00
Senaryo 2	7.00	0.12	3.50	2.50
Senaryo 3	5.00	0.80	3.50	4.00



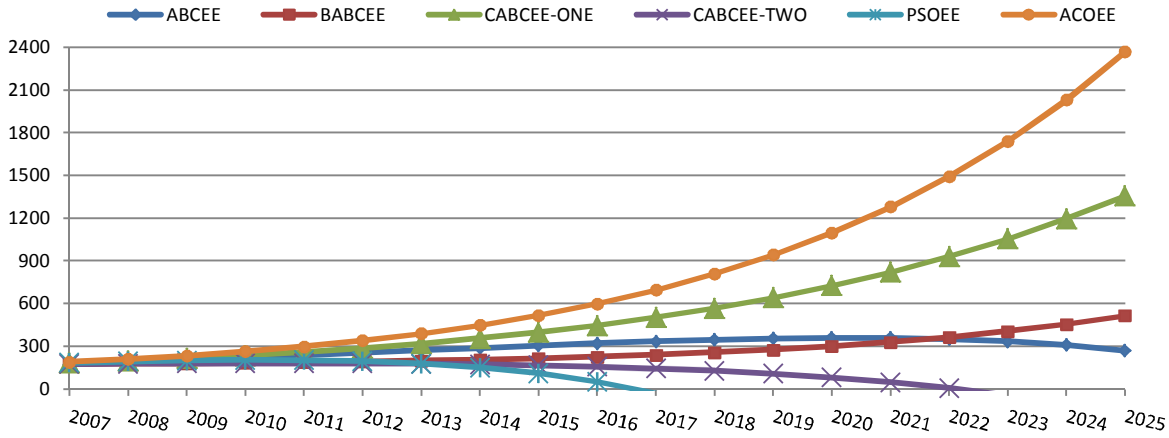
Şekil 4.8. Senaryo 1 ve lineer model ile elde edilen gelecek projeksiyonu



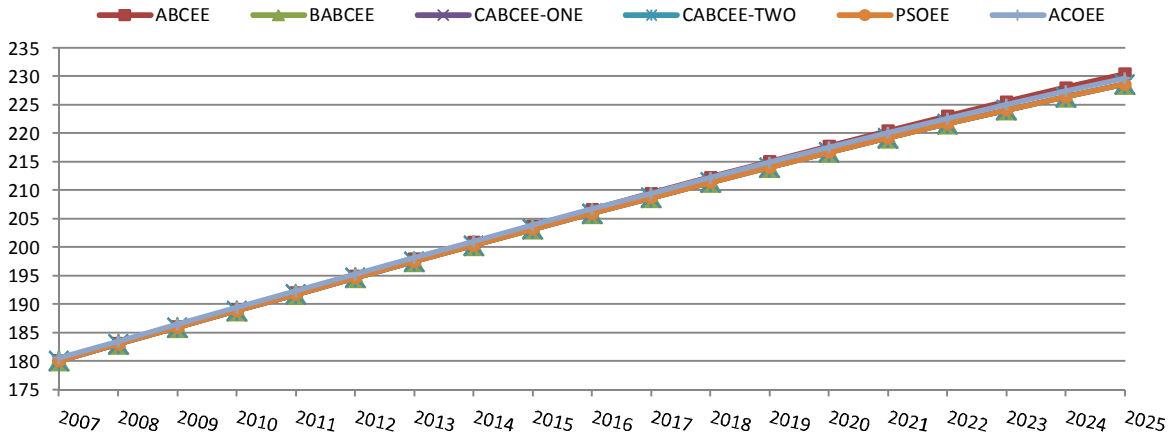
Şekil 4.9. Senaryo 1 ve kuadratik model ile elde edilen gelecek projeksiyonu



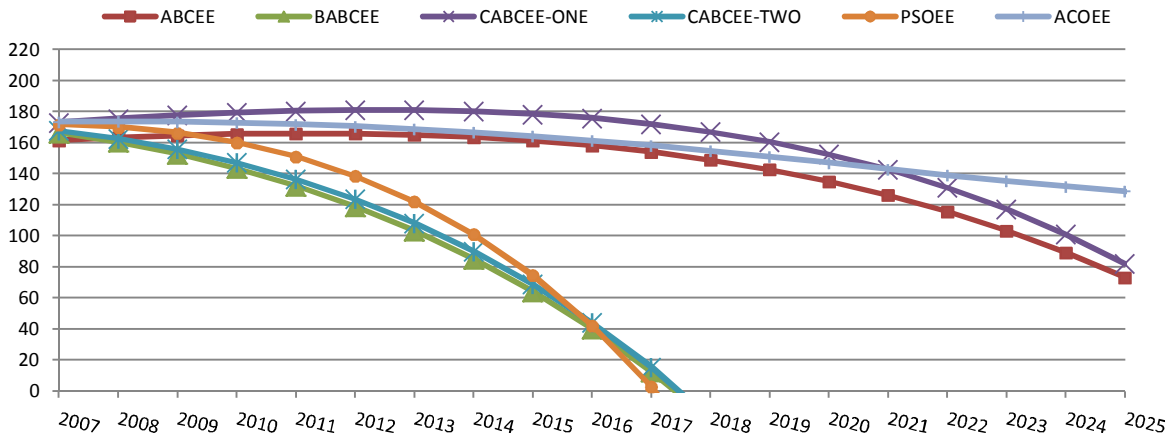
Şekil 4.10. Senaryo 2 ve lineer model ile elde edilen gelecek projeksiyonu



Şekil 4.11. Senaryo 2 ve kuadratik model ile elde edilen gelecek projeksiyonu



Şekil 4.12. Senaryo 3 ve lineer model ile elde edilen gelecek projeksiyonu



Şekil 4.13. Senaryo 3 ve kuadratik model ile elde edilen gelecek projeksiyonu

Tahmin, geçmiş verilerden faydalanarak geleceğe yönelik değerlerin elde edilmesidir. Bu noktada farklı modellerin kullanılması ve daha çeşitli yöntemler ile modellerin katsayılarının belirlenmesi sonuç zenginliğinin artması açısından önemlidir. Bir diğer gereksinim de geleceğe yönelik senaryoların doğru şekilde oluşturulmasıdır. Genel olarak farklı yöntemlerden ve farklı denklemlerden elde edilen sonuçların yıllar bazında ortalaması, üst ve alt sınırları belirlenerek ülkenin enerji ihtiyacının gelecekte karşılanması için tedbirlerin alınmasında karar vericiler ve politika üreticiler için faydalı olacağı düşünülmektedir.

4.1.3. ABC ve PSO algoritmalarının rekombinasyon tabanlı hibritleştirilmesi

ABC nümerik problemlerin optimizasyonunda oldukça başarılı olmasına rağmen literatürdeki karşılaştırmalı çalışmalar göstermektedir ki multimodal test fonksiyonların optimizasyonunda unimodal fonksiyonlara nazaran diğer yöntemlere göre daha başarılıdır. ABC bir sürü zekâsı yöntemidir ve sürekli problemlerin optimizasyonunda PSO ile aynı grupta yer almaktadır. PSO unimodal fonksiyonlarda daha iyi sonuç verirken ABC de multimodal fonksiyonlarda daha iyi sonuçlar üretmektedir. Bu açıdan bakıldığında hem unimodal hem de multimodal fonksiyonların optimizasyonda efektif sonuçlar üreten bir yönteme ihtiyaç bulunmaktadır fakat bireysel olarak düşünüldüğünde yerel araştırma ve keşif için yapılan araştırmanın bir yöntem içerisinde dengelenmesi oldukça zor olmaktadır. Bundan dolayı bu çalışma kapsamında ABC ve PSO yöntemleri hibritleştirilerek yeni bir yaklaşım ortaya konulmuş ve hem multimodal

hem de unimodal fonksiyonların optimizasyonu için bir alternatif optimizasyon metodu sunulmuştur.

4.1.3.1. Rekombinasyon ile yeni çözüm elde etme

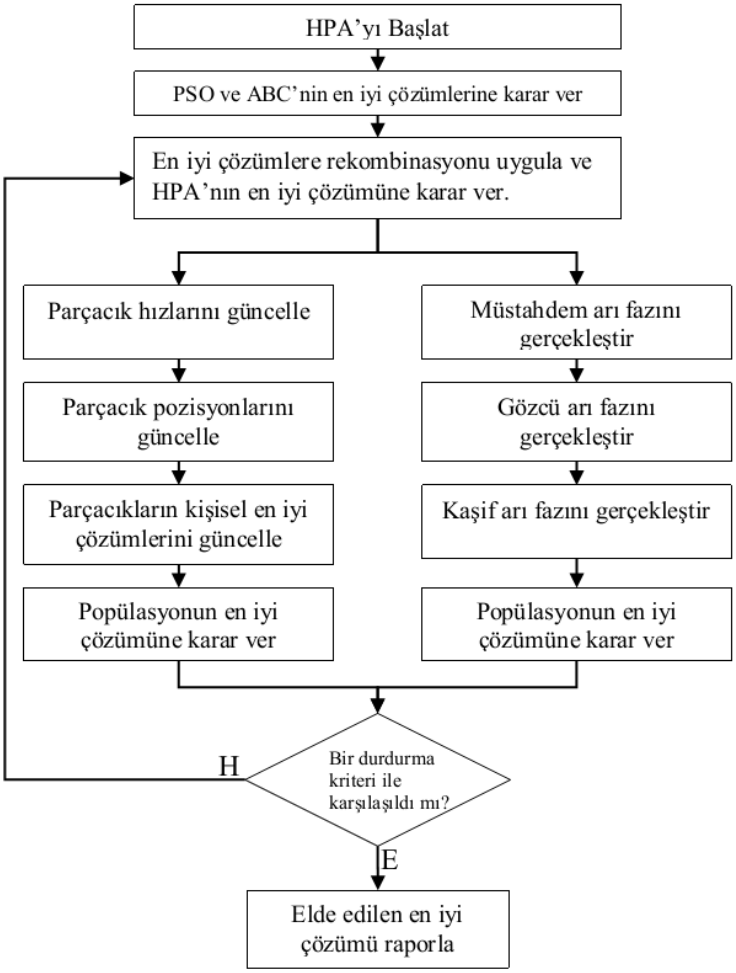
Rekombinasyon birden fazla bireyin çözümlerinin harmanlanmasıyla yeni bir çözüm elde etme süreci olarak tanımlanabilir. Rekombinasyon yani yeniden birleştirme farksal gelişim (Differential Evolution) (Storn ve Price, 1997) tekniğinde ebeveyn vektör ile mutasyona uğramış vektörün bir çaprazlama sabiti kullanılarak birleştirilmesi işlemidir (Karaboğa ve Ökdem, 2004). Bu çalışmada kullanılan rekombinasyon bu süreçten tamamen farklıdır ve bir ebeveyn vektörü, bir mutasyona uğramış vektör ve çaprazlama sabiti bulunmamaktadır. Uniform çaprazlama operatörüne biraz benzetilebilir fakat uniform çaprazlamada yavrular oluşturulurken uygunluk fonksiyonu tabanlı bir seçim bulunmamaktadır. Ayrıca bu çalışmada önerilen rekombinasyon ile sadece bir çözüm elde edilir.

PSO yönteminde bilindiği üzere sosyal bileşende popülasyonun en iyi çözümü kullanılmaktadır yani popülasyonun her bireyi popülasyonun global en iyi çözümünden doğrudan etkilenmektedir. ABC metodunda ise popülasyonun o ana kadar elde ettiği en iyi çözüm her çevrimde kaydedilmesine rağmen doğrudan bir araştırma sürecinde kullanılmamaktadır. Önerilen hibrit yöntemde ise ABC ve PSO tarafından elde edilen en iyi çözümler rekombine edilerek yeni bir çözüm oluşturulmaktadır. Oluşturulan bu çözüm parçacık popülasyonuna en iyi çözüm olarak verilmektedir. Aynı çözüm yapay arı popülasyonundaki gözcü arılar tarafından da komşu olarak kullanılmaktadır. Bu sayede popülasyonlar arasındaki bilgi akışı çift taraflı sağlanmaktadır. Rekombinasyon süreci bu işin yani bilgi akışının merkezinde bulunmaktadır ve kendisine gelen iki en iyi çözümü birleştirerek (harmanlayarak) çıkışına sunmaktadır. Rekombinasyonun eklenmesiyle hibrit yöntemin şematik gösterimi Şekil 4.14'de verilmiştir.

PSO_{best} parçacık popülasyonu tarafından elde edilen en iyi çözüm, ABC_{best} ise yapay arı kolonisi tarafından elde edilen en iyi çözüm olduğunu ve bu çözümlerin amaç fonksiyon değerlerinin F_{PSO} ve F_{ABC} olduğunu varsayalım. Öncelikle PSO_{best} için Denklem 4.11 kullanılarak uygunluk değeri yani çözümün kalitesi hesaplanır.

$$fit_{PSO} = \begin{cases} \dfrac{1}{1+F_{PSO}} & Eğer(F_{PSO} \geq 0) \\ 1+abs(F_{PSO}) & değilse \end{cases} \tag{4.11}$$

Denklem 4.11, Denklem 3.12'nin PSO'nun en iyi çözümün uygunluk değerini ölçmek amacıyla değiştirilmiş halidir. Yapay arı popülasyonun en iyi çözümünün uygunluk değeri zaten Denklem 3.12 kullanılarak hesaplanmıştır ve çevrimler boyunca tutulmaktadır.



Şekil 4.14. HPA algoritmasının akış diyagramı

Elde edilen uygunluk değerleri kullanılarak iki çözümünde seçimdeki baskısı aşağıda verilen denklemler kullanılarak hesaplanır.

$$B_{PSO} = \frac{fit_{PSO}}{fit_{PSO} + fit_{ABC}} \tag{4.12}$$

$$B_{ABC} = \frac{fit_{ABC}}{fit_{PSO} + fit_{ABC}} \tag{4.13}$$

Çözümlerin seçimdeki etkileri de hesaplandıktan sonra rekombinasyon ile en iyi çözüm Denklem 4.14 kullanılarak hesaplanır.

$$E_j = \begin{cases} PSO_{best}^j & Eğer(r_j \le B_{PSO}) \\ ABC_{best}^j & değilse \end{cases}, \quad j = 1,2,...,D \tag{4.14}$$

Denklemdeki E_j rekombinasyon ile elde edilen yeni çözümün j. boyutu, r_j her boyutun hangi çözümden seçileceğini belirlemek için kullanılan ve [0,1] aralığında üretilen rastgele sayı, D ise optimizasyon probleminin boyutudur.

Her çevrimde elde edilen E çözümü parçacık popülasyonuna en iyi çözüm olarak verilir. Aynı çözüm yapay arı popülasyonundaki gözcü arılara ise komşu olarak verilir. Yapay arı popülayonunda sadece gözcü arılara verilmesinin sebebi görevli arıların keşif için aramaya devam etmelerini sağlamak ve sadece en iyiden etkilenerek durağanlaşma göstermelerini engellemek içindir.

4.1.3.2. Deneysel Sonuçlar ve Karşılaştırmalar

Deneylerde kullanılan yöntemler için popülasyon boyutları 100 olarak alınmıştır. PSO 100 parçacıktan oluşmakta, ABC 50 görevli ve 50 gözcü arıdan oluşmakta iken önerilen HPA yönteminde 50 parçacık, 25 görevli arı ve 25 de gözcü arı bulunmaktadır. Durdurma kriteri olarak maksimum iterasyon sayısı (MIN) kullanılmıştır ve N birey sayısı, D optimizasyon probleminin boyutluluğu olmak üzere $MIN = (D \times 20000)/N$ olarak hesaplanmaktadır. HPA ve PSO için atalet ağırlığı iterasyon zamanına bağlı olarak azaltılmakta ve t çevrim indisi olmak üzere atalet ağırlığı $w = (MIN - t)/MIN$ olarak hesaplanır. ABC ve HPA için ortak olan *limit* parametresi ise *NE* popülasyondaki görevli arı sayısı olmak üzere $limit = (NE \times D)$ olarak hesaplanmıştır. Ayrıca belirtmekte fayda vardır ki her iterasyonda sadece bir tane kaşif arı oluşmasına izin verilmiştir.

Yukarıdaki kontrol parametreleri ile Tablo 4.14'te verilen temel fonksiyonların minimizasyonu için ABC, PSO ve HPA 30'ar defa rastgele başlatılarak çalıştırılmıştır. 30 çalıştırmanın ortalaması ve standart sapması tablolanmış ve 10^{-20}'den küçük değerler 0 kabul edilmiştir. HPA ile ABC'nin karşılaştırması Tablo 4.15'te, HPA ile PSO'nun karşılaştırması Tablo 4.16'da ve üç yöntemin ortalamalar bazında karşılaştırılması Tablo 4.17'da sunulmuştur.

Tablo 4.14. Deneylerde kullanılan kıyas fonksiyonları

F	D	C	Aralık	Min.	Formül
f_1	4	U,N	[-10,10]	0	$f(x)=100(x_1^2-x_2)^2+(x_1-1)^2+(x_3-1)^2+90(x_3^2-x_4)^2$ $+10.1((x_2-1)^2+(x_4-1)^2+19.8(x_2-1)(x_4-1)$
f_2	2	U,N	[-10,10]	0	$f(x)=0.26(x_1^2+x_2^2)-0.48x_1x_2$
f_3	2	M,N	[-100,100]	0	$f(x)=\frac{\sin^2(\sqrt{x_1^2+x_2^2}-0.5}{(1+0.001(x_1^2+x_2^2))^2}$
f_4	2	M,N	[-5,5]	-1.03163	$f(x)=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4$
f_5	6	U,N	[-36,36]	-50	$f(x)=\sum_{i=1}^{6}(x_i-1)^2-\sum_{i=2}^{6}x_ix_{i-1}$
f_6	10	U,N	[-100,100]	-210	$f(x)=\sum_{i=1}^{10}(x_i-1)^2-\sum_{i=2}^{10}x_ix_{i-1}$
f_7	30 60 90	U,S	[-100,100]	0	$f(x)=\sum_{i=1}^{D}x_i^2$
f_8	30 60 90	U,S	[-10,10]	0	$f(x)=\sum_{i=1}^{D}ix_i^2$
f_9	30 60 90	M,S	[-5.12,5.12]	0	$f(x)=\sum_{i=1}^{D}(x_i^2-10\cos(2\pi x_i)+10)$
f_{10}	30 60 90	M,N	[-600,600]	0	$f(x)=\frac{1}{4000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$
f_{11}	30 60 90	M,N	[-32,32]	0	$f(x)=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\right)-\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right)+20+e$
f_{12}	30 60 90	U,N	[-30,30]	0	$f(x)=\sum_{i=1}^{D}(100(x_{i+1}-x_i^2)^2+(x_i-1)^2)$

Tablo 4.15. ABC ve HPA için 30 çalıştırmadan elde edilen en iyi, en kötü, ortalama ve standart sapma sonuçları

F	D	ABC				HPA			
		En İyi	En Kötü	Ortalama	Std.Sap.	En İyi	En Kötü	Ortalama	Std.Sap.
f_1	4	0.0129	0.6106	0.1157	0.1110	**2.03E-06**	**0.0456**	**0.009**	**0.0122**
f_2	2	1.25E-08	8.44E-06	1.90E-06	1.85E-06	**0**	**0**	**0**	**0**
f_3	2	0	4.86E-06	4.13E-07	1.23E-06	**0**	**0**	**0**	**0**
f_4	2	**-1.03163**	**-1.03163**	**-1.03163**	**0**	**-1.03163**	**-1.03163**	**-1.03163**	**0**
f_5	6	**-50**	**-50**	**-50**	**0**	**-50**	**-50**	**-50**	**0**
f_6	10	-209.9929	-209.8437	-209.9471	0.0440	**-210**	**-210**	**-210**	**0**
f_7	30	2.61E-16	5.54E-16	4.74E-16	9.30E-17	**0**	**0**	**0**	**0**
	60	9.47E-16	1.22E-15	1.12E-15	1.08E-16	**0**	**0**	**0**	**0**
	90	1.40E-15	2.10E-15	1.83E-15	1.45E-16	**0**	**0**	**0**	**0**
f_8	30	2.94E-16	5.55E-16	4.89E-16	9.04E-17	**0**	**0**	**0**	**0**
	60	7.73E-16	1.22E-15	1.11E-15	1.24E-16	**0**	**0**	**0**	**0**
	90	1.38E-15	2.10E-15	1.74E-15	1.430E-16	**0**	**0**	**0**	**0**
f_9	30	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	60	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	90	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_{10}	30	0	1.11E-16	9.25E-17	4.14E-17	**0**	**0**	**0**	**0**
	60	0	1.11E-16	8.51E-17	4.70E-17	**0**	**0**	**0**	**0**
	90	0	1.11E-16	8.88E-17	4.44E-17	**0**	**0**	**0**	**0**
f_{11}	30	2.93E-14	3.99E-14	3.27E-14	2.51E-15	**7.99E-15**	**1.51E-14**	**1.13E-14**	**3.55E-15**
	60	6.48E-14	8.62E-14	7.56E-14	5.31E-15	**1.51E-14**	**3.29E-14**	**2.39E-14**	**5.41E-15**
	90	1.11E-13	1.36E-13	1.21E-13	8.17E-15	**2.93E-14**	**4.35E-14**	**3.72E-14**	**4.56E-15**
f_{12}	30	6.80E-05	0.1601	0.0199	0.0337	**2.17E-05**	**0.0939**	**0.0189**	**0.0204**
	60	2.56E-04	0.4325	0.0364	0.0791	**4.53E-04**	**0.1782**	**0.0281**	**0.0409**
	90	0.0003	0.1115	0.0198	0.0272	**2.90E-04**	**0.1097**	**0.0180**	**0.0292**

Tablo 4.16. PSO ve HPA için 30 çalıştırmadan elde edilen en iyi, en kötü, ortalama ve standart sapma sonuçları

F	D	PSO				HPA			
		En İyi	En Kötü	Ortalama	Std.Sap.	En İyi	En Kötü	Ortalama	Std.Sap.
f_1	4	**6.90E-08**	**0.0045**	**0.0010**	**0.0013**	2.03E-06	0.0456	0.009	0.0122
f_2	2	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_3	2	0	3.57E-07	1.1911E-08	6.4142E-08	**0**	**0**	**0**	**0**
f_4	2	**-1.03163**	**-1.03163**	**-1.03163**	**0**	**-1.03163**	**-1.03163**	**-1.03163**	**0**
f_5	6	**-50**	**-50**	**-50**	**0**	**-50**	**-50**	**-50**	**0**
f_6	10	**-210**	**-210**	**-210**	**0**	**-210**	**-210**	**-210**	**0**
f_7	30	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	60	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	90	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_8	30	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	60	0	400	36.6667	111.0055	**0**	**0**	**0**	**0**
	90	0	2100	143.3333	395.5446	**0**	**0**	**0**	**0**
f_9	30	16.9143	67.7281	36.4842	11.5305	**0**	**0**	**0**	**0**
	60	78.6017	150.2385	114.8228	19.3891	**0**	**0**	**0**	**0**
	90	114.4202	273.6129	194.0995	33.4843	**0**	**0**	**0**	**0**
f_{10}	30	0	1.18E-01	2.0633E-02	2.3206E-02	**0**	**0**	**0**	**0**
	60	0	3.92E-02	5.98E-03	1.0451E-02	**0**	**0**	**0**	**0**
	90	0	2.46E-02	4.27E-03	6.1256E-03	**0**	**0**	**0**	**0**
f_{11}	30	**7.99E-15**	**1.51E-14**	**8.59E-15**	**1.8536E-15**	7.99E-15	1.51E-14	1.13E-14	3.55E-15
	60	1.51E-14	3.29E-14	2.42E-14	4.5574E-15	**1.51E-14**	**3.29E-14**	**2.39E-14**	**5.41E-15**
	90	2.93E-14	5.06E-14	4.02E-14	5.5747E-15	**2.93E-14**	**4.35E-14**	**3.72E-14**	**4.56E-15**
f_{12}	30	3.4567	81.1053	28.5674	23.6389	**2.17E-05**	**0.0939**	**0.0189**	**0.0204**
	60	30.649	90157.0859	9208.7223	26973.1499	**4.53E-04**	**0.1782**	**0.0281**	**0.0409**
	90	7.71E+01	9.02E+04	1.23E+04	3.06E+04	**2.90E-04**	**0.1097**	**0.0180**	**0.0292**

Tablo 4.17. PSO, ABC ve HPA yöntemlerinin ortalamalara göre kıyaslanması

F	Boyut	Ortalama Sonuçlar		
		PSO	ABC	HPA
f_1	4	**0.0010**	0.1157	0.0090
f_2	2	**0**	1.8978E-06	**0**
f_3	2	1.1911E-08	4.1307E-07	**0**
f_4	2	**-1.03163**	**-1.03163**	**-1.03163**
f_5	6	**-50.0000**	**-50.0000**	**-50.0000**
f_6	10	**-210.0000**	-209.9471	**-210.0000**
f_7	30	**0**	4.7403E-16	**0**
	60	**0**	1.1164E-15	**0**
	90	**0**	1.8294E-15	**0**
f_8	30	**0**	4.8909E-16	**0**
	60	36.6667	1.1088E-15	**0**
	90	143.3333	1.7360E-15	**0**
f_9	30	36.4842	**0**	**0**
	60	114.8228	**0**	**0**
	90	194.0995	**0**	**0**
f_{10}	30	2.0633E-02	9.2519E-17	**0**
	60	5.9831E-03	8.5117E-17	**0**
	90	4.2693E-03	8.8818E-17	**0**
f_{11}	30	**8.5857E-15**	3.2744E-14	1.1309E-14
	60	2.4218E-14	7.5614E-14	**2.3862E-14**
	90	4.0205E-14	1.2097E-13	**3.7244E-14**
f_{12}	30	28.5674	0.0199	**0.0189**
	60	9208.7223	0.0364	**0.0281**
	90	1.2258E+04	0.0198	**0.0180**

Tablolardan görüldüğü üzere çalışmanın başında vurgulanan farklı karakteristiklerdeki fonksiyonların minimizasyonu için önerilen yöntem hibritleştirildiği yöntemlerden genel olarak daha iyi sonuçar üretmektedir.

Önerilen yöntem ayrıca literatürde önerilen ABC-SPSO ve IABAP yöntemleri ile de karşılaştırılmıştır. Bu karşılaştırmalarda HPA, ABC-SPSO ve IABAP'ın uygulandığı fonksiyonların minimizasyonu için ABC-SPSO (El-Abd, 2011) ve IABAP (Shi ve ark, 2010) ile eşit şartlarda tekrar çalıştırılmıştır ve elde edilen sonuçlar ve karşılaştırmalar Tablo 4.18 ve 4.19'de sunulmuştur. Tablo 4.18'deki ABC-SPSO'nun sonuçları El-Abd (2011) tarafından sunulan çalışmadan, Tablo 4.19'daki IABAP'ın sonuçları ise Shi ve ark. (2010) tarafından tebliğ edilen çalışmadan doğrudan alınmıştır.

Yapılan testlerden ve karşılaştırmalardan optimizasyon problemleri için iyi sonuçların elde edilebilmesi için problemlerdeki farklı karakteristiklere ve zorluklara karşı yöntemlerdeki farklı yeteneklerin verimli çalışması gerektiği anlaşılmaktadır. Daha iyi sonuçların elde edilebilmesi için yöntemlerin hibritleştirilmesi veya farklı araştırma stratejilerinin bir arada kullanılması gerekmektedir.

Tablo 4.18. ABC-SPSO ve HPA ile CEC2005 fonksiyonları için elde edilen sonuçların kıyaslanması

Fonksiyon	**ABC-SPSO**		**HPA**	
	Ortalama	Std.Sapma	Ortalama	Std.Sapma
F1	**0**	**0**	**0**	**0**
F2	**0**	**0**	**0**	**0**
F3	**9.85E+04**	**6.47E+04**	1.06E+05	1.17E+05
F4	**0**	**0**	4.8E+1	8.2E+1
F5	**0**	**0**	**0**	**0**
F6	1.49E+1	5.59E+1	**1.62**	**2.74**
F7	**2.74E-02**	**1.41E-01**	6.81E-02	2.62E-02
F8	2.02E+01	5.06E-02	2.02E+01	4.81E-02
F9	**0**	**0**	**0**	**0**
F10	**6.11**	**2.07**	9.95	2.63
F11	4.67	7.58E-01	**2.55**	**1.65**
F12	210.31	117.38	**67.71**	**114.65**
F13	2.76E-01	5.72E-02	**2.51E-02**	**2.96E-02**
F14	2.76	3.7E-01	**2.48**	**6.1E-01**

Tablo 4.19. IABAP ve HPA ile bazı fonksiyonlar için elde edilen sonuçların karşılaştırılması

Fonksiyon	Yöntem-Sonuç		Boyut				
			10	30	50	100	200
Sphere	IABAP	Ortalama	4.80E-17	5.38E-16	1.34E-15	4.41E-15	2.23E-14
		Std.Sap.	1.11E-17	6.48E-17	1.79E-16	8.84E-16	9.04E-15
	HPA	Ortalama	**2.21E-42**	**3.95E-27**	**1.27E-23**	**2.06E-20**	**6.18E-18**
		Std.Sap.	**1.04E-41**	**6.84E-27**	**2.93E-23**	**2.95E-20**	**5.04E-18**
Rosenbrock	IABAP	Ortalama	8.00E-02	1.14E-01	1.17E-01	1.44E-01	**4.10E-01**
		Std.Sap.	7.82E-02	2.14E-01	1.99E-01	2.47E-01	**4.73E-01**
	HPA	Ortalama	**1.63E-02**	**2.27E-02**	**2.73E-02**	**7.12E-02**	5.85E-01
		Std.Sap.	**2.23E-02**	**3.28E-02**	**3.00E-02**	**8.65E-02**	1.51E+00
Griewank	IABAP	Ortalama	**3.11E-17**	4.73E-16	9.99E-16	2.22E-15	1.57E-14
		Std.Sap.	**5.04E-17**	4.92E-17	1.96E-16	3.17E-16	8.50E-15
	HPA	Ortalama	3.45E-03	**0**	**0**	**0**	**3.01E-16**
		Std.Sap.	4.44E-03	**0**	**0**	**0**	**1.79E-16**
Rastrigin	IABAP	Ortalama	**0**	1.42E-16	3.43E-14	3.90E-10	1.53E-04
		Std.Sap.	**0**	4.87E-16	4.66E-14	1.6E-09	1.04E-3
	HPA	Ortalama	**0**	**0**	**0**	**3.41E-14**	**4.77E-13**
		Std.Sap.	**0**	**0**	**0**	**5.21E-14**	**1.22E-13**

4.2. Ayrık Optimizasyon Problemlerinin Çözümü için Yapılan Çalışmalar

ABC algoritmasının performansının sürekli optimizasyon problemlerindeki özellikle kıyas fonksiyonlarında göstermiş olduğu başarı, araştırmacıları yöntemin ayrık problemler üzerindeki performansını araştırmaya yöneltmektedir. Bundan dolayı bu başlık altında sürekli problemlerinin çözümü için önerilmiş olan yöntemin ayrık problemlerin çözümü için yeniden yapılandırılması araştırılmış ve önerilen yöntemler ile elde edilen sonuçlar bildirilmiştir. Bu kapsamda ABC algoritması komşuluk operatörleri ile ayrıklaştırılmasının performansı gezgin satıcı problemi üzerinde araştırılmış ve elde edilen sonuçlar kullanılarak hiyerarşik bir yöntem önerilmiştir. Ayrıca ikili optimizasyon problemlerini çözmek için yöntem lojik operatörler kullanılarak modifiye edilmiş ve ikili çözüm uzayında da hareket edebilecek bir yapıya kavuşturulmuştur. Bu üç çalışmayı temel alan çalışmalar çeşitli dergilerde araştırmacıların ve uygulayıcıların dikkatine sunulmuştur.

4.2.1. Komşuluk operatörlü Ayrık ABC algoritmasının Gezgin Satıcı Problemi Üzerinde Analizi

Gerçek dünyada optimize edilmek istenen karar değişkenleri sürekli olabildiği gibi ayrık yapıda veya karar değişkenlerinin bir kısmı ayrık iken bir kısmı da sürekli yapıda olabilir. Ayrık problemlerinin çözümü için literatürde en çok kullanılan sürü zekâsına dayalı yöntem karınca kolonisi optimizasyonudur. Fakat karınca kolonisi optimizasyonunda kısa yollarda uzun yollara göre zaman içerisinde daha fazla feromon biriktiği için zamanla yöntem durağanlaşma davranışı göstermektedir. Bu çalışmanın temel amacı gezgin satıcı problemi üzerinde ABC algoritmasının performansını ölçmek olduğu kadar hangi ayrık komşuluk operatörünün ABC'de kullanılması gerektiğine karar vermektir. Bu sayede bir sonraki çalışmada hiyerarşik bir düzende kullanacağımız ayrık ABC yönteminde hangi komşuluk operatörünün kullanılacağına karar vermektir.

Bu çalışmada araç rotalama problemi için önerilmiş olan ayrık ABC yöntemi (Szeto ve ark., 2011) çok bilinen TSP üzerinde analiz edilerek farklı komşuluk operatörlerinin etkisi araştırılmıştır (Kıran ve ark., 2013).

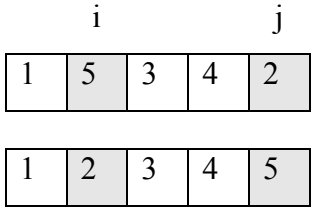
4.2.1.1. Ayrık ABC algoritması

Eğer yiyecek kaynakları sürekli uzaydaki bir çözümü temsil ederse bilindiği üzere tüm aritmetik operatörler ile elde edilecek yeni çözümler çözüm uzayı sınırları içerisindeyse sınırlamasız bir optimizasyon problemi için geçerli bir çözümdür. Eğer çözüm uzayı ayrıksa yani karar değişkenleri bir setin elemanları olacak şekilde sıralanacaksa veya seçilecekse yiyecek kaynakları pozisyonlarının aritmetik operatörler ile güncellenmesi ek tamir mekanizmalarını gerektirebilir. Bunun yerine komşuluk operatörleri kullanılarak geçerli bir çözüm üzerinde değişikler ortaya konulabilir. Bundan dolayı sürekli ABC algoritmasının çözüm güncelleme denklemi ve ilklendirme denkleminin düzenlenmesi gerekir. Yöntem gezgin satıcı problemi üzerinde analiz edildiği için popülasyonu ilklendirmek yani başlatmak için rastgele oluşturulmuş permütasyon prosedürü kullanılmıştır. 10 şehirli bir TSP için popülasyonda 10 görevli arı varsa 10x11 boyutunda bir matris başlangıç popülasyonudur. Her satır bir görevli arıya atanmış bir yiyecek kaynağı pozisyonunu göstermektedir ve matrisin 11. sütunu TSP'nin bir gereği olarak eklenmiştir.

Çözümün güncellenmesi ise komşuluk operatörleri kullanılarak yapılmaktadır. Herhangi bir görevli arı kendi çözümüne aday bir çözüm üretebilmek için komşuluk operatörlerinden birini kullanır. Gözcü arılar ise sürekli ABC'de olduğu gibi seçim mekanizmalarını kullanarak bir görevli arının çözümünü seçer ve yine komşuluk operatörlerinden birini kullanarak görevli arının çözümünü günceller. Elde edilen yeni aday çözümler eski çözümden iyiyse ikisi yer değiştirir ve görevli arının yiyecek kaynağını terk etme sayacı sıfırlanır, değilse görevli arıya atanan yiyecek kaynağının terk etme sayacı bir arttırılır.

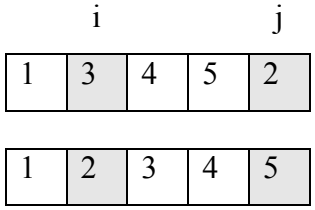
Çalışmada kullanılan komşuluk operatörleri aşağıda verilmiştir.

1. Rastgele Yer Değiştirme (Swap-RS): Rastgele seçilen iki noktadaki elemanlar yer değiştirir. Bu durum Şekil 4.15'te gösterilmiştir.



Şekil 4.15. Rastgele yer değiştirme operatörünü çalışması

2. Rastgele Ekleme (Insertion-RI): Rastgele seçilen bir noktaya rastgele seçilen bir pozisyondaki eleman eklenir. Ekleme noktasından itibaren kalan elemanlar kaydırılır. Rastgele Ekleme operatörünün çalışması Şekil 4.16'te gösterilmiştir.

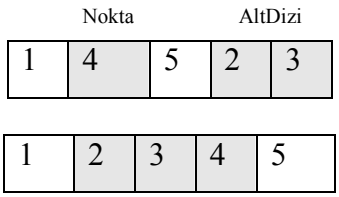


Şekil 4.16. Rastgele ekleme operatörünün fonksiyonu

3. Rastgele Seçilen Alt Dizilerin Yer Değiştirmesi (RSS): Bu operatör yer değiştirme operatörünün noktalar yerine alt diziler kullanılarak uygulanmasıdır. Şekil 4.17'de RSS operatörünün işlevi gösterilmiştir.

Şekil 4.17. Alt dizilerin yer değiştirmesi

4. Rastgele Seçilen Alt Dizinin Eklenmesi (RIS): Rastgele seçilen bir noktaya rastgele seçilen bir alt dizinin eklenmesidir. Geri kalan elemanlar dizi eklenmeden önce kaydırılır. Bu durum Şekil 4.18'da gösterilmiştir.

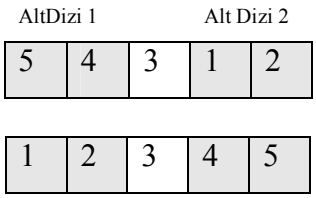


Şekil 4.18. Dizi ekleme işlevi

5. Rastgele Tersleme (Reverse- RRS): Diziden rastgele seçilen bir alt dizinin terslenmesidir. Operatörün çalışması Şekil 4.19'de verilmiştir.

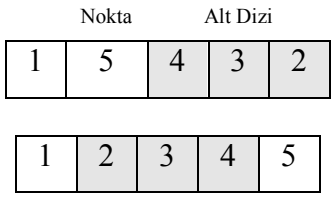
Şekil 4.19. Rastgele tersleme operatörünün işlevi

6. Alt Dizilerin Rastgele Terslenerek Yer Değiştirmesi (RRSS): Birinci ve beşinci operatörlerin bir kombinasyonudur. Diziden seçilen iki alt dizi yer değiştirilirken %50 ihtimalle terslenebilir. İhtimal her alt dizi için ayrı ayrı uygulanır. Yani birinci dizi terslenerek yeri değiştirilirken ikinci dizi terslenmeden yeri değiştirilebilir veya tersi de olabilir. Operatörün çalışması Şekil 4.20'de verilmiştir.



Şekil 4.20. RRSS operatörünün fonksiyonu

7. Alt Dizinin Rastgele Terslenerek Eklenmesi (RRIS): Rastgele seçilen bir alt dizinin rastgele seçilen bir noktaya %50 ihtimalle terslenerek eklenmesidir. Dizinin geri kalanı sağa doğru kaydırılmaktadır. Operatörün çalışması Şekil 4.21'de sunulmuştur.



Şekil 4.21. RRIS operatörünün fonksiyonu

8. Kombine 1 ve 2: Yukarıda verilen operatörlere ek olarak yer değiştirme ve ekleme operatörleri iki grupta birleştirilmiştir. Birinci grup yer değiştirme (RS), alt dizilerin yer değiştirmesi (RSS) ve alt dizilerin terslenerek yer değiştirmesi (RRSS) operatörleri yer almaktadır. Aynı şekilde ikinci grupta ise ekleme (RI), alt dizinin eklenmesi (RIS) ve alt dizinin terslenerek eklenmesi (RRIS) operatörleri yer almaktadır. Her kombinasyondaki operatörler arılar çözümlerini güncellerken eşit ihtimale sahip olarak seçilir. Bu sayede bir operatöre bağlı kalmadan daha fazla araştırma operatörü bir arada kullanılmıştır.

Yukarıda operatörler ve yönteme değinildikten sonra ayrık ABC algoritması Şekil 4.22'de adım adım verilmiştir.

Başlat
Her görevli arı arı için
Bir çözüm üret (rastgele permütasyon)
Problem için spesifik olan amaç fonksiyon değerini hesapla
Çözümün uygunluk değerini (fit) Denklem 3.12 ile hesapla
Deneme sayacını sıfırla.
En iyi çözüme karar ver.
Limit parametresinin değerini ata.
Until (Durdurma kriteri){
Görevli Arı Fazı
Her işçi arı için
Bir komşuluk operatörünü kullanarak çözümü güncelle
Problem için spesifik olan amaç fonksiyon değerini hesapla
Çözümün uygunluk değerini (fit) Denklem 3.12 ile hesapla
Yeni aday çözüm önceki çözümden iyiyse, aday çözüm ile önceki çözümü yer değiştir ve deneme sayacını sıfırla; değilse deneme sayacını 1 arttır.
Gözcü Arı Fazı
Her gözcü arı için
Denklem 3.15'i kullanarak bir işçi arının çözümünü seç.
Seçilen çözümü komşuluk operatörlerinden birini kullanarak güncelle
Problem için spesifik olan amaç fonksiyon değerini hesapla
Çözümün uygunluk değerini (fit) Denklem 3.12 ile hesapla
Gözcü arı tarafından bulunan çözüm görevli arının çözümünden iyiyse, yeni çözümü ile eski çözümü yer değiştir ve deneme sayacını sıfırla; değilse deneme sayacını 1 arttır.
Kaşif Arı Fazı
En büyük içeriğe sahip deneme sayacını bul.
IF sayacın değeri limitten büyükse **THEN** sayacı sıfırla ve görevli için yeni bir çözüm üret.
En iyi çözümü bul
Popülasyonun en iyi çözümünü bul.
IF popülasyonun en iyi çözümü global en iyi çözümden daha iyiyse **THEN** popülasyonun en iyi çözümünü global best olarak tayin et.
} End Until
En iyi çözümü raporla.

Şekil 4.22. Detaylı Ayrık ABC algoritması

4.2.1.2. Deneysel Sonuçlar

Yöntemin performansı literatürde çok kullanılan 9 gezgin satıcı problemi üzerinde analiz edilmiştir. Bu problemlerin isimleri, şehir sayıları ve en kısa tur uzunlukları Tablo 4.20'de verilmiştir.

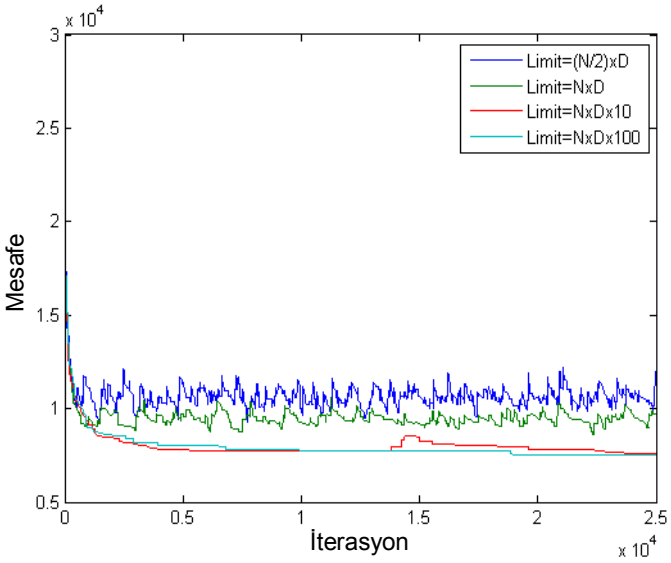
Tablo 4.20. Deneylerde kullanılan gezgin satıcı problemleri için düğüm sayıları ve optimum tur uzunlukları

Problem	Düğüm Sayısı	Optimum Tur Uzunluğu
Oliver	30	423.74
Eil	51	428.87
Berlin	52	7544.37
St	70	677.11
Pr	76	108,159.44
Kroa	100	21,285.44
Eil	101	642.31
TSP	225	3,859.00
A280	280	2,586.77

Literatürde bu tür problemlerin çözümünde genellikle popülasyon boyutu şehir sayısına eşit olarak seçilmekle birlikte ABC algoritmasındaki arıların yarısı görevli ve yarısı da gözcü olduğu için popülasyon boyutu oluşabilecek bir kaosu ortadan kaldırmak için Denklem 4.15 kullanılarak hesaplanmıştır.

$$N = \left\lceil \frac{D}{2} \right\rceil \times 2 \tag{4.15}$$

Burada, N popülasyon boyutu, D ise TSP'deki düğüm veya şehir sayısıdır. Algoritmanın kendine özgü parametresi yani limit değeri için ise büyük değerlerin kullanılması önerilir. Çünkü küçük limit değerlerinde popülasyon doyuma ulaşmadan popülasyonun iyi çözümleri kaşif arı olmakta ve popülasyon doyuma ulaşamamaktadır. Berlin52 problemi üzerinde popülasyonun en iyi çözümünün iterasyonlardaki durumu Şekil 4.23'te gösterilmiştir. Şekil 4.23'den görüldüğü üzere literatürde sürekli ABC için önerilen ve $Limit = (N/2) \times D$ ile elde edilen değerde popülasyonun doyuma ulaşması mümkün olmamaktadır. Bundan dolayı limit değeri $Limit = (N \times D \times 100)$ olarak alınmıştır. Koşturmalarda algoritmanın durdurma kriteri olarak maksimum çevrim sayısı kullanılmıştır ve 100000 olarak alınmıştır.



Şekil 4.23. Limit parametresinin etkisi

Yukarıdaki parametreler ile 9 komşuluk operatörü ve 8 problem için de 20 defa algoritma koşturulmuş ve elde edilen en iyi, en kötü, ortalama değerler raporlanmıştır. Tablo 4.21'de Oliver30, Tablo 4.22'de Eil51, Tablo 4.23'te Berlin52, Tablo 4.24'te St70, Tablo 4.25'te Pr76, Tablo 4.26'da Kroa100, Tablo 4.27'de Eil101, Tablo 4.28'de Tsp225 ve Tablo 4.29'da A280 TSP'leri için sonuçlar sunulmuştur. Ayrıca yöntemin yakınsama grafiği en küçük boyutlu problem olan Oliver30 üzerinde elde edilen sonuçlara göre tüm komşuluk operatörleri için Şekil 4.24'de verilmiştir.

Tablo 4.21. Ayrık ABC ile Oliver30 Problemi için Elde Edilen Sonuçlar

Operatör	**En İyi**			**En Kötü**			**Ortalama**			***Hata Değeri* (%)**		
	ABC	**2-OPT**	**3-OPT**	**ABC**	**2-OPT**	**3-OPT**	**ABC**	**2-OPT**	**3-OPT**	**ABC**	**2-OPT**	**3-OPT**
RS	439.83	439.83	439.83	510.94	510.94	503.59	477.86	477.86	476.37	12.77	12.77	12.42
RSS	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**0.00**	**0.00**	**0.00**
RI	424.69	424.69	424.57	480.69	480.69	480.69	447.36	444.43	444.58	5.57	4.88	4.92
RIS	423.74	423.74	423.74	424.69	424.69	424.69	423.88	423.88	423.88	0.03	0.03	0.03
RR	423.74	423.74	423.74	429.83	429.83	429.38	425.16	425.16	424.91	0.33	0.33	0.28
RRIS	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**0.00**	**0.00**	**0.00**
RRSS	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**0.00**	**0.00**	**0.00**
Komb. 1	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**0.00**	**0.00**	**0.00**
Komb. 2	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**423.74**	**0.00**	**0.00**	**0.00**

Tablo 4.22. Ayrık ABC ile Eil51 problemi için elde edilen sonuçlar

Operatör	En İyi			En Kötü			Ortalama			*Hata Değeri* (%)		
	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT
RS	478.13	478.13	478.13	536.55	536.55	530.27	506.32	506.32	504.45	18.06	18.06	17.62
RSS	428.87	428.87	428.87	434.62	434.62	434.62	431.11	431.03	431.11	0.52	0.50	0.52
RI	452.21	449.71	449.64	486.17	486.17	486.01	467.62	463.28	465.03	9.04	8.02	8.43
RIS	431.17	431.17	431.17	439.33	439.33	439.33	435.78	435.77	435.78	1.61	1.61	1.61
RR	432.57	432.57	432.57	447.34	447.34	447.34	440.00	440.00	439.92	2.59	2.59	2.58
RRIS	428.87	428.87	428.87	433.93	433.93	433.93	430.41	430.37	430.41	0.36	0.35	0.36
RRSS	428.98	428.98	428.98	433.49	433.49	433.49	430.25	430.25	430.25	0.32	0.32	0.32
Komb. 1	**428.87**	**428.87**	**428.87**	**433.60**	**433.60**	**433.60**	**430.08**	**430.08**	**430.08**	**0.28**	**0.28**	**0.28**
Komb. 2	428.87	428.87	428.87	433.71	433.71	433.71	430.55	430.55	430.55	0.39	0.39	0.39

Tablo 4.23. Ayrık ABC ile Berlin52 problemi için elde edilen sonuçlar

Operatör	En İyi			En Kötü			Ortalama			*Hata Değeri* (%)		
	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT
RS	8194.67	8194.67	8194.67	9648.29	9648.29	9550.59	9177.02	9177.02	9117.92	21.64	21.64	20.86
RSS	7544.37	7544.37	7544.37	7716.69	7716.69	7716.69	7562.80	7562.80	7562.80	0.24	0.24	0.24
RI	7898.21	7898.21	7898.21	8850.17	8850.17	8850.17	8470.14	8395.98	8401.40	12.27	11.29	11.36
RIS	7544.37	7544.37	7544.37	7746.86	7746.86	7746.86	7591.43	7591.43	7591.43	0.62	0.62	0.62
RR	7544.37	7544.37	7544.37	7972.17	7972.17	7966.49	7774.77	7774.77	7763.02	3.05	3.05	2.90
RRIS	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**0.00**	**0.00**	**0.00**
RRSS	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**0.00**	**0.00**	**0.00**
Komb. 1	7544.37	7544.37	7544.37	7598.44	7598.44	7598.44	7548.47	7547.07	7548.47	0.05	0.04	0.05
Komb. 2	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**0.00**	**0.00**	**0.00**

Tablo 4.24. Ayrık ABC ile St70 problemi için elde edilen sonuçlar

Operatör	En İyi			En Kötü			Ortalama			*Hata Değeri* (%)		
	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT
RS	860.67	860.67	860.67	979.69	979.69	979.69	927.69	927.69	922.84	37.01	37.01	36.29
RSS	678.51	678.51	677.91	698.97	698.97	698.97	688.59	687.15	688.46	1.69	1.48	1.68
RI	739.29	728.66	736.17	831.12	820.76	826.70	785.14	773.34	777.03	15.95	14.21	14.76
RIS	**681.87**	**677.11**	**681.87**	**700.37**	**697.10**	**700.37**	**690.04**	**687.87**	**690.03**	**1.91**	**1.59**	**1.91**
RR	684.54	684.54	684.54	705.71	705.71	705.07	695.17	695.17	693.84	2.67	2.67	2.47
RRIS	**677.11**	**677.11**	**677.11**	**688.70**	**688.70**	**688.70**	**681.52**	**680.85**	**681.32**	**0.65**	**0.55**	**0.62**
RRSS	**677.11**	**677.11**	**677.11**	**688.47**	**688.47**	**688.47**	**683.18**	**682.55**	**683.18**	**0.90**	**0.80**	**0.90**
Komb. 1	678.51	678.51	678.51	691.33	691.33	691.33	684.17	684.17	684.05	1.04	1.04	1.02
Komb. 2	**677.11**	**677.11**	**677.11**	**687.62**	**687.62**	**687.62**	**681.60**	**680.88**	**681.56**	**0.66**	**0.56**	**0.66**

Tablo 4.25. Ayrık ABC ile Pr76 problemi için elde edilen sonuçlar

Operatör	En İyi			En Kötü			Ortalama			*Hata Değeri* (%)		
	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT
RS	137607.01	137607.01	137504.39	155064.55	155064.55	154988.63	147206.79	147206.79	146809.12	36.10	36.10	35.73
RSS	108501.34	108304.51	108501.34	111889.32	111005.89	111889.32	110106.91	109871.01	110106.91	1.80	1.58	1.80
RI	121188.99	118774.03	119910.98	133682.89	131665.70	131665.70	126341.76	124429.93	124738.89	16.81	15.04	15.33
RIS	108428.00	108425.53	108428.00	112039.72	111969.42	112039.72	110397.12	110005.95	110356.60	2.07	1.71	2.03
RR	108633.72	108633.72	108633.72	110872.90	110872.90	110872.90	109817.41	109817.41	109787.82	1.53	1.53	1.51
RRIS	**108159.44**	**108159.44**	**108159.44**	**110050.90**	**109759.39**	**110050.90**	**109005.00**	**108830.04**	**108965.31**	**0.78**	**0.62**	**0.75**
RRSS	**108183.42**	**108159.44**	**108183.42**	**109590.17**	**109590.17**	**109590.17**	**108911.08**	**108848.04**	**108895.81**	**0.69**	**0.64**	**0.68**
Komb. 1	**108159.44**	**108159.44**	**108159.44**	**109782.95**	**109782.95**	**109777.01**	**109164.47**	**109164.47**	**109164.13**	**0.93**	**0.93**	**0.93**
Komb. 2	**108159.44**	**108159.44**	**108159.44**	**109556.93**	**109556.93**	**109556.93**	**108668.29**	**108642.94**	**108624.77**	**0.47**	**0.45**	**0.43**

Tablo 4.26. Ayrık ABC ile Kroa100 problemi için elde edilen sonuçlar

Operatör	En İyi			En Kötü			Ortalama			*Hata Değeri* (%)		
	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT
RS	29849.71	29849.71	29287.85	36084.80	36084.80	36058.15	33761.06	33761.06	33437.72	58.61	58.61	57.09
RSS	21866.36	21450.50	21864.04	23076.54	22519.65	22989.99	22663.10	22081.42	22560.99	6.47	3.74	5.99
RI	23410.06	23055.07	23212.28	27475.19	27379.34	27400.60	26204.50	25856.45	25959.69	23.11	21.47	21.96
RIS	21724.60	21479.20	21595.15	23069.65	22742.56	22871.99	22490.43	21997.55	22331.84	5.66	3.35	4.92
RR	21474.52	21474.52	21474.52	22340.88	22340.88	22294.82	21845.07	21845.07	21834.45	2.63	2.63	2.58
RRIS	21730.11	21425.26	21729.52	22417.75	22202.61	22323.63	22080.16	21688.40	22013.77	3.73	1.89	3.42
RRSS	21549.84	21285.44	21549.84	22196.76	21957.07	22150.71	21825.11	21687.13	21800.82	2.54	1.89	2.42
Komb. 1	21419.13	21419.13	21419.13	22042.91	22042.91	22042.91	21759.41	21747.45	21749.99	2.23	2.17	2.18
Komb. 2	**21285.44**	**21285.44**	**21285.44**	**21728.76**	**21728.76**	**21728.76**	**21521.00**	**21506.96**	**21493.58**	**1.11**	**1.04**	**0.98**

Tablo 4.27. Ayrık ABC ile Eil101 problemi için elde edilen sonuçlar

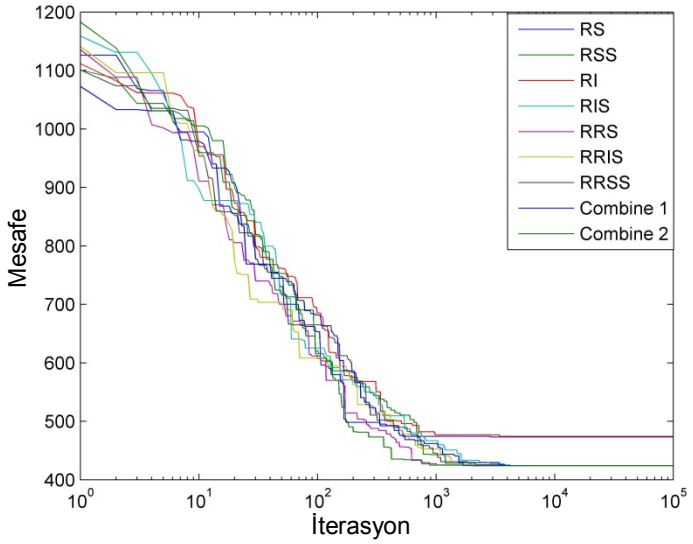
Operatör	En İyi			En Kötü			Ortalama			*Hata Değeri* (%)		
	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT	ABC	2-OPT	3-OPT
RS	792.71	792.71	789.28	867.90	867.90	855.29	837.37	837.37	829.74	30.37	30.37	29.18
RSS	673.49	660.64	668.41	690.45	688.72	690.45	683.64	675.86	681.99	6.44	5.22	6.18
RI	710.79	706.00	710.79	759.81	753.16	753.67	730.02	724.38	726.53	13.66	12.78	13.11
RIS	675.26	665.05	672.80	686.47	682.30	686.15	681.49	674.82	679.90	6.10	5.06	5.85
RR	665.36	665.36	665.36	685.03	685.03	684.21	676.34	676.34	674.95	5.30	5.30	5.08
RRIS	660.60	652.62	657.54	677.46	672.59	676.57	670.82	664.86	669.15	4.44	3.51	4.18
RRSS	659.22	657.48	658.40	675.25	673.06	675.25	668.05	665.35	666.96	4.01	3.59	3.84
Komb. 1	658.60	658.60	657.53	675.56	675.56	675.56	667.80	667.58	667.27	3.97	3.93	3.89
Komb. 2	**653.30**	**651.94**	**653.30**	**667.35**	**667.35**	**667.35**	**661.25**	**660.96**	**660.92**	**2.95**	**2.90**	**2.90**

Tablo 4.28. Ayrık ABC ile Tsp225 problemi için elde edilen sonuçlar

Operatör	En İyi			En Kötü			Ortalama			*Hata Değeri* (%)		
	ABC	**2-OPT**	**3-OPT**	**ABC**	**2-OPT**	**3-OPT**	**ABC**	**2-OPT**	**3-OPT**	**ABC**	**2-OPT**	**3-OPT**
RS	6732.46	6728.73	6690.53	7642.61	7577.63	7520.06	7264.12	7219.30	7142.35	88.24	87.08	85.08
RSS	5370.21	4656.61	4934.29	5785.40	5060.11	5287.12	5578.45	4866.89	5069.09	44.56	26.12	31.36
RI	4785.27	4726.63	4697.31	5222.84	5160.86	5180.19	5030.64	4916.51	4952.39	30.36	27.40	28.33
RIS	5117.24	4517.09	4791.78	5456.12	4893.74	5065.92	5273.92	4720.48	4903.77	36.67	22.32	27.07
RR	**4150.86**	**4122.50**	**4130.17**	**4231.84**	**4210.13**	**4231.73**	**4183.45**	**4170.36**	**4177.70**	**8.41**	**8.07**	**8.26**
RRIS	5114.12	4448.90	4735.50	5391.70	4868.10	5019.40	5242.72	4693.40	4875.97	35.86	21.62	26.35
RRSS	5031.31	4470.80	4713.42	5348.51	4790.32	5047.78	5165.84	4666.68	4835.13	33.86	20.93	25.30
Komb. 1	4616.82	4491.62	4521.25	4877.50	4774.55	4817.68	4741.70	4624.32	4672.41	22.87	19.83	21.08
Komb. 2	4217.11	4187.58	4216.11	4503.33	4402.15	4489.95	4439.00	4337.75	4404.18	15.03	12.41	14.13

Tablo 4.29. Ayrık ABC ile A280 problemi için elde edilen sonuçlar

Operatör	En İyi			En Kötü			Ortalama			*Hata Değeri* (%)		
	ABC	**2-OPT**	**3-OPT**	**ABC**	**2-OPT**	**3-OPT**	**ABC**	**2-OPT**	**3-OPT**	**ABC**	**2-OPT**	**3-OPT**
RS	5387,17	5330,41	5256,68	5993,44	5910,20	5934,68	5759,04	5642,34	5611,08	122,63	118,12	116,91
RSS	4321,42	3531,69	3689,53	4485,69	3863,26	4009,45	4416,36	3685,15	3849,63	70,73	42,46	48,82
RI	3450,76	3417,87	3411,70	3772,11	3714,14	3727,61	3674,23	3570,72	3574,11	42,04	38,04	38,17
RIS	4003,09	3411,22	3601,74	4212,11	3684,28	3826,07	4136,35	3538,74	3714,04	59,90	36,80	43,58
RR	**2818,38**	**2812,68**	**2810,32**	**2985,13**	**2953,46**	**2960,68**	**2912,31**	**2878,30**	**2894,80**	**12,58**	**11,27**	**11,91**
RRIS	3973,27	3305,97	3514,56	4171,97	3567,81	3734,22	4082,35	3452,81	3642,13	57,82	33,48	40,80
RRSS	4011,30	3354,50	3583,15	4286,36	3681,99	3877,89	4156,83	3570,00	3707,45	60,70	38,01	43,32
Komb. 1	3476,62	3365,82	3426,27	3670,07	3467,54	3576,73	3584,49	3407,18	3497,76	38,57	31,72	35,22
Komb. 2	3250,56	3151,34	3210,17	3396,22	3275,21	3328,07	3322,47	3205,48	3267,81	28,44	23,92	26,33



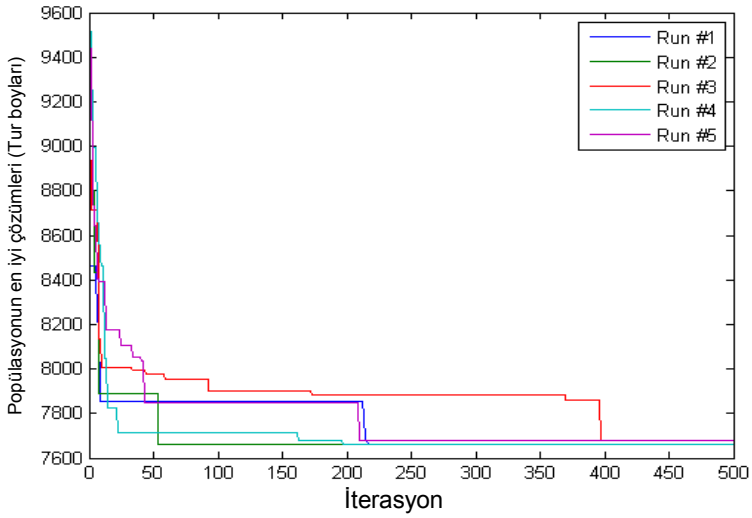
Şekil 4.24. Ayrık ABC algoritmasının Oliver30 problemi üzerinde yakınsama grafiği

Sonuçlara ait tablolar dikkate alındığında Oliver30, Eil51, Berlin52, St70, Pr76 ve Kroa100 problemleri için en az biz defa optimum değerler elde edilmiştir ve ortalama sonuçların göreceli hataları kabul edilebilir düzeydedir. Eil101, Tsp225 ve A280 problemleri içinse umut verici sonuçlar üretilmiştir. Bu problemlerde başarının düşüş sebebi düğüm sayısının artışı olarak gösterilebilir. Ayrıca yöntemin her güncellemede sadece bir değişiklik yapması bir diğer sebeptir. Operatörler düzeyinde konuşulursa kombinasyon 2'nin genel olarak en başarılı seçenek olduğu ve ileriki çalışmalarda kullanılabileceği söylenebilir fakat RR operatörü en büyük boyutlu iki problemde diğerlerine nispeten başarılı sonuçlar üretmiştir. Sonuçlardan görüldüğü üzere ayrık ABC yöntemi küçük ve orta boyutlu problemlerde alternatif bir optimizasyon aracı olmasına rağmen büyük boyutlu problemlerde geliştirilmesine ihtiyaç vardır.

4.2.2. ABC ve ACO tabanlı yeni bir yaklaşım

Bir önceki bölümde ayrık ABC algoritmasının performansı gezgin satıcı problemi üzerinde analiz edilmişti ve ayrıca yöntemin sonuçlarına komşuluk operatörlerinin etkileri de araştırılmıştı. Hiyerarşik yaklaşımda kullanılan yöntemlerden birisinin ayrık ABC olması açısından önceki bölümde diğer operatörlere göre iyi sonuçların üretilmesine katkı sağlayan komşuluk operatörü bu bölümdeki ABC algoritmasında kullanılmaktadır.

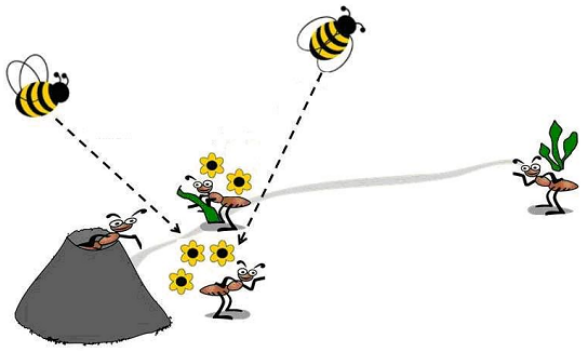
Bilindiği üzere karınca kolonisi (ACO) algoritması gezgin satıcı problemi için çözümleri adım adım oluşturmaktadır ve bir sonraki iterasyonda oluşturulacak çözümlere mevcut iterasyondaki çözümler yol göstermektedir (feromon mekanizması). Ayrık ABC yönteminde ise durum adım adım çözüm oluşturmadan biraz farklıdır. Ayrık ABC'de çözümler rastgele üretilmekte ve çözümler iterasyonlar boyunca iyileştirilmeye çalışılmaktadır. Yani arılar karıncalar gibi çözümlerini terk etmemektedirler. Feromon mekanizması ACO algoritmasının ilerleyen iterasyonlarında yöntem için durağanlaşma davranışına sebep olmaktadır. Durağanlaşmanın sebebi ise birbirine yakın düğümler arasındaki feromon miktarının aşırı artması sonucu yöntemin yeni çözümler elde edememesidir. Bunun sonucu olarak yöntemde araştırma faaliyeti devam etmesine rağmen bu yöntemde daha önceki iterasyonlarda elde edilen en iyi çözümden daha iyi bir çözüm elde edilememektedir. Bu davranış Şekil 4.25'de açıkça görülmektedir (Gündüz ve ark., 2012).



Şekil 4.25. ACO için Berlin52 problemi üzerinde durağanlaşma davranışı

Şekil 4.25'den görüldüğü üzere toplam iterasyon zamanının yarısında karıncalar tarafından elde edilen en iyi çözüm güncellenmemektedir. Bu açıklanan davranışa bir çözüm olması açısından ayrık ABC algoritması hiyerarşik düzende ACO yönteminden sonra kullanılmakla yöntemin durağanlaşmasının önüne geçilmeye çalışılmıştır. Ayrıca önerilen hiyerarşik yaklaşımın bir diğer avantajı da daha kaliteli çözümlerin ACO algoritmasından daha kısa sürede elde edilmesidir. Çünkü ABC yöntemi tüm çözümü adım adım yeniden oluşturmak yerine sadece bir komşuluk operatörü ile mevcut çözümü güncellemektedir. Bu işlemin zamansal açısından farkı da oldukça önemli avantajlar sağlamaktadır. Önerilen hiyerarşik yaklaşım ABC'den yavaş olmasına

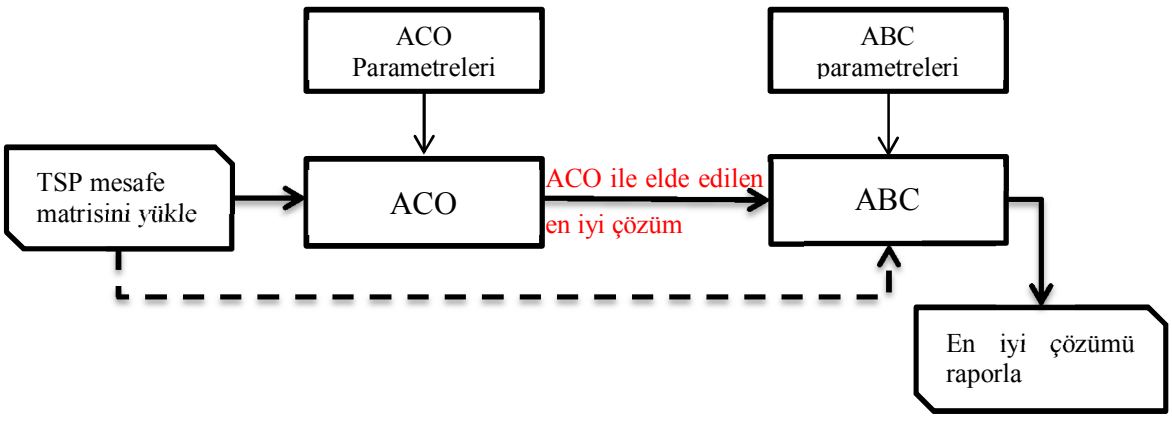
rağmen çok daha kaliteli çözümler elde etmektedir. ACO ile karşılaştırıldığında ise daha hızlı (yaklaşık yarı yarıya çalışma zamanı avantajı) ve daha kaliteli çözümlere ulaşılmaktadır. Yöntem için önerilen yaklaşımda karıncalar ve arılar arasındaki etkileşim Şekil 4.26 ile ifade edilmeye çalışılmıştır.



Şekil 4.26. Hiyerarşik yaklaşımda karıncalar ile arılar arasındaki etkileşim (Arılar, karıncalar tarafından elde edilen en iyi çözümü iyileştirmeye çalışır)

4.2.2.1. Hiyerarşik Yaklaşım

ACO ve ABC algoritmalarının sözde kodları ve/veya akış diyagramları daha önceki bölümlerde sunulduğu için bu bölümde sadece önerilen yöntemin çalışma diyagramı verilmekle yetinilmiştir (Şekil 4.27). Hiyerarşik yöntemin parçası olan ayrık ABC için Kombinasyon 2 olarak adlandırılan komşuluk operatörleri grubu kullanılmıştır. Yukarıda ifade edildiği üzere bu grup ekleme (RI), alt dizinin eklenmesi (RIS) ve alt dizinin terslenerek eklenmesi (RRIS) operatörlerini kapsamaktadır.



Şekil 4.27. Hiyerarşik yaklaşımın çalışma diyagramı

Akış diyagramından görüldüğü üzere önerilen yaklaşımda metotlar ardışıl olarak çalışmaktadır ve toplam iterasyon zamanının yarısı süresince ACO algoritması koşturulmaktadır. ACO algoritmasından elde edilen en iyi çözüm ABC algoritmasındaki tüm işçi arılara yiyecek kaynağı olarak atanmaktadır. İşçi arılar kendi çözümlerini Kombinasyon 2 komşuluk operatör grubu ile iyileştirmeye çalışmaktadır. İşçi arıların çözümleri kalitelerine göre gözcü arılar tarafından seçilmekte ve aynı operatör gurubu ile gözcüler tarafından iyileştirilmeye çalışılmaktadır. Bu işlem iterasyon zamanının sonuna kadar devam etmektedir.

4.2.2.2. Deneysel Sonuçlar

Önerilen yaklaşımın performansı ve doğruluğu Tablo 4.30 ile verilen test problemleri üzerinde araştırılmıştır. Karşılaştırılan yöntemler aynı PC (Intel i5 3.1 Ghz Cpu, 4GB Ram, Windows 7, Matlab Platformu) üzerinde 20 defa çalıştırılmış ve elde edilen sonuçlar en iyi, en kötü, ortalama, standart sapma ve çalışma süresi olarak her yöntem için ayrı ayrı raporlanmıştır. Bu çalışma kapsamında başarının yanı sıra çözüm süresi de iyileştirilmeye çalışıldığı için çözüm süresi iki farklı açıdan ele alınmıştır. Birincisinde işlemci zamanı (Cpu Süre), ikincisinde ise gerçek zaman (süre) tabloda raporlanmıştır. İşemci zamanı Process Explorer v16.02 yazılımı ile gerçek çalışma zamanı ise Matlab platformunun tic-toc fonksiyonları ile elde edilmiştir. Her yöntem için karar verilen parametreler Table 4.31'den bulunabilir. Verilen şartlar altında elde edilen sonuçlar ise Tablo 4.32'de sunulmuştur.

Tablo 4.31. ABC, ACO ve Hiyerarşik Yaklaşım için parametre değerleri

Parametre	**ABC**	**ACO**	**Hiyerarşik Yaklaşım**	
			ABC	**ACO**
Popülasyon Boyutu (P)	D*	D*	D*	D*
Çevrim Sayısı	500	500	250	250
Alpha (α)	N / A	1.0	N / A	1
Beta (β)	N / A	5.0	N / A	5.0
Rho (ρ)	N /A	0.65	N /A	0.65
Q	N/ A	100	N/ A	100
Limit	$(P/2)\times D\times 1000$	N/A	$(P/2)\times D\times 1000$	N /A

* D gezgin satıcı problemindeki düğüm sayısını gösterir.

Tablo 4.32. Test problemleri için ACO, ABC ve Hiyerarşik Yaklaşım ile elde edilen sonuçlar

Problem	Metot	En iyi	En Kötü	Ortalama	Std.Sap.	*REM (%)	Cpu Süre (s)	Süre (s)
	ACO	423.74	429.36	424.68	1.41	0.22	29.19	35.20
Oliver30	ABC	439.49	484.83	462.55	12.47	9.16	2.04	1.26
	HA**	**423.74**	**423.74**	**423.74**	**0.0**	**0.00**	**17.07**	**19.63**
	ACO	450.59	463.55	457.86	4.07	6.76	91.38	112.11
Eil51	ABC	563.75	619.44	590.49	15.79	37.68	2.98	2.16
	HA	**431.74**	**454.97**	**443.39**	**5.25**	**3.39**	**48.27**	**58.33**
	ACO	7548.99	7681.75	7659.31	38.7	1.52	94.46	116.67
Berlin52	ABC	9479.11	11021.99	10390.26	439.69	37.72	3.14	2.17
	HA	**7544.37**	**7544.37**	**7544.37**	**0.0**	**0.00**	**50.01**	**60.64**
	ACO	696.05	725.26	709.16	8.27	4.73	185.57	226.06
St70	ABC	1162.12	1339.24	1230.49	41.79	81.73	3.99	3.15
	HA	**687.24**	**716.52**	**700.58**	**7.51**	**3.47**	**95.79**	**115.65**
	ACO	554.46	568.62	561.98	3.5	3.04	224.17	271.98
Eil76	ABC	877.28	971.36	931.44	24.86	70.78	4.46	3.49
	HA	**551.07**	**565.51**	**557.98**	**4.1**	**2.31**	**115.5**	**138.82**
	ACO	115166.66	118227.41	116321.22	885.79	7.55	225.66	272.41
Pr76	ABC	195198.9	219173.64	205119.61	7379.16	89.65	4.31	3.50
	HA	**113798.56**	**116353.01**	**115072.29**	**742.9**	**6.39**	**115.14**	**138.92**
	ACO	22455.89	23365.46	22880.12	235.18	7.49	427.6	615.06
Kroa100	ABC	49519.51	57566.05	53840.03	2198.36	152.94	5.66	5.17
	HA	**22122.75**	**23050.81**	**22435.31**	**231.34**	**5.40**	**218.87**	**311.12**
	ACO	678.04	705.65	693.42	6.8	7.96	440.47	527.42
Eil101	ABC	1237.31	1392.64	1315.95	35.28	104.88	5.73	5.17
	HA	**672.71**	**696.04**	**683.39**	**6.56**	**6.39**	**224.06**	**267.08**
	ACO	6648.51	6726.27	6702.87	20.73	2.61	1187.87	1387.65
Ch150	ABC	20908.89	22574.99	21617.48	453.71	230.93	8.63	8.95
	HA	**6641.69**	**6707.86**	**6677.12**	**19.3**	**2.21**	**595.04**	**698.61**
	ACO	4112.35	4236.85	4176.08	28.34	8.22	4844.54	4038.75
Tsp225	ABC	16998.41	18682.56	17955.12	387.35	365.28	14.27	16.68
	HA	**4090.54**	**4212.08**	**4157.85**	**26.27**	**7.74**	**2443.63**	**2037.33**

* 20 çalıştırmadan elde edilen ortalama sonuç için göreceli hata değeridir.
** Hiyerarşik Yaklaşım tabloda HA olarak isimlendirilmiştir.

Tablo 4.32'den görüldüğü üzere ele alınan tüm problemlerde hiyerarşik yaklaşım diğer yöntemlere nazaran üstün performans sergilemiştir. ABC kısa çalışma zamanı gerektirmesine rağmen optimum veya optimuma yakın bir sonuca ulaşabilmek için çok fazla iterasyona dolayısıyla çok fazla zamana ihtiyaç vardır. ACO ise iyi sonuçlar elde etmesine rağmen belirli iterasyonlardan sonra en iyi çözümü güncelleyememekte ve popülasyon durağanlaşmaya başlamaktadır. ABC yönteminin hızlı çalışması ve ACO'nun iyi sonuçlar üretmesi gibi avantajları bir araya getirerek oluşturduğumuz hiyerarşik yaklaşımda ise daha iyi sonuçlar daha makul zamanlarda elde edilmektedir. Çünkü önerilen yöntemde ACO toplam iterasyon zamanının yarısı süresince çalışmaktadır ve elde edilen en iyi çözüm ABC yöntemi ile iyileştirilmeye çalışılmaktadır. Bu da hem süre hem de doğruluk açısından yöntemi bireysel yaklaşımlara nazaran öne geçirmiştir.

4.2.3. İkili Optimizasyon Problemlerinin Çözümü için Yapay Arı Kolonisi Yaklaşımı

Eğer optimizasyon probleminin karar seti sadece ikili değerler (0 veya 1) alabiliyorsa bu tip optimizasyon problemlerine ikili optimizasyon problemi adı verilir. İkili optimizasyon problemi ayrık bir problemdir fakat çözüm ya da araştırma uzayı aynı sayıda karar değişkenine sahip bir ayrık problemden daha küçüktür. Yine de optimum çözümü bulmak için uygulanacak yöntem kesin sonuç verecek bir yöntem ise uzayın tamamının taranması gereklidir. Bu da karar setinin büyük olması durumunda hesaplama zamanı açısından pahalı olabilmektedir. Bundan dolayı ikili optimizasyon problemlerinin çözümünde sezgisel metotlar rağbet görmektedir. Bu çalışma kapsamında yapay arı kolonisi algoritması ikili optimizasyon problemlerini çözecek şekilde biçimlendirilmiştir. Yöntem ile elde edilen sonuçlar ikili parçacık sürü optimizasyonu (BPSO) (Kennedy ve Eberhart, 1997), iyileştirilmiş ikili parçacık sürü optimizasyonu (IBPSO) (Yuan ve ark., 2009) ve ikili problemlerin çözümü için önerilmiş ayrık ABC (DisABC) algoritması (Kashan ve ark., 2011) ile kıyaslanmıştır.

4.2.2.1. İkili ABC (Binary ABC-binABC) Algoritması

Temel yapay arı kolonisi algoritması, araştırma denklemindeki aritmetik operatörlerden dolayı karar değişkenlerinin herhangi bir aralıkta sürekli olduğu durumlarda çalışabilmektedir. Eğer karar değişkenleri farklı bir çözüm uzayında ise yöntemin ajanlarının bu uzayda hareket edebilecek şekilde yapılandırılması gerekmektedir. Bu çalışmada ABC algoritmasının araştırma denklemi (Denklem 3.13) ikili uzayda işletilmek için yapılandırılmıştır. Yapılandırma aritmetik operatörler yerine mantık operatörlerinin (“özel veya” ve “tersleme”) üzerine kuruludur. Bilindiği üzere mantık operatörleri ikili değerler üzerinde işlemleri yapar ve çıkışları yine bir ikili değerdir. Bundan dolayı yapılandırma için herhangi bir tamir mekanizmasına ve ek ayarlamalara ihtiyaç yoktur. Denklem 3.13, mantık operatörleri kullanılarak Denklem 4.16’da verildiği şekilde değiştirilmiştir.

$$\begin{aligned} &\overrightarrow{V_i} = \overrightarrow{X_i} && (a) \\ &V_i^j = X_i^j \oplus \left[\varphi\left(X_i^j \oplus X_k^j\right)\right] && (b) \\ &\quad i = 1,2,...,N\,,\ k \in \{1,2,...,N\} && (c) \\ &\quad j \in \{1,2,...,D\} && (d) \\ &\quad i \neq k && (e) \end{aligned} \qquad (4.16)$$

Denklem 4.16(b)'de $\oplus$ "özel veya" operatörüdür. φ ise tasarımcı tarafından belirlenen bir ihtimal ile $\left(X_i^j \oplus X_k^j\right)$ işlem sonucunu tersler. Algoritmanın akışında herhangi bir değişiklik yapılmamıştır bundan dolayı bu bölümde tekrar aynı algoritma verilmemiştir.

binABC metodu diğer yöntemlerle kıyaslanmadan önce ABC'nin kontrol parametrelerinin etkisi araştırılmıştır. Bilindiği üzere temel ABC algoritmasının iki kontrol parametresi vardır. Birincisi tüm popülasyon tabanlı yaklaşımlar için ortak olan popülasyon boyutu, ikincisi ise ABC algoritmasının kendisine özgü olan limit değeridir. Popülasyon boyutunun etkisi 10, 20, 30, 40 ve 50 olmak üzere 5 farklı değer için ve limit değeri ise Denklem 4.17'den 4.21'e kadar önerilen denklemlerden elde edilen değerler kullanılarak araştırılmıştır.

$$lb4 = D \times N / 4 \qquad (4.17)$$

$$lb2 = D \times N / 2 \qquad (4.18)$$

$$lb = D \times N \qquad (4.19)$$

$$lc2 = D \times N \times 2 \qquad (4.20)$$

$$lc4 = D \times N \times 4 \qquad (4.21)$$

Denklemlerdeki D, optimizasyon probleminin boyutu (karar seti boyutu), N ise görevli arı sayısını gösterir. Denklem 4.19, Karaboğa ve Akay (2009) tarafından temel ABC algoritması için önerilmiş limit değeridir. Bu değer popülasyonun büyüklüğü ile problem boyutuna göre düzgün bir şekilde limit değerinin ayarlanması için kullanılmaktadır. Bu çalışmada önerilen problemin ve yöntemin yapısı farklı olduğu için 5 değişik limit değeri altında analizler yapılmıştır.

Limit ve popülasyon boyutunun etkisi yukarıda verilen değerler ile kombinasyonel olarak araştırılmıştır. Yani her limit değeri için tüm popülasyon boyutları tüm problemler için ele alınmıştır. Bu şartlar altında elde edilen sonuçlar Tablo 4.33, 4.34, 4.35, 4.36 ve 4.37'de ayrı ayrı raporlanmıştır.

Tablo 4.33. Farklı popülasyon boyutları ve *lb4* limit değeri ile elde edilen sonuçlar

Problem	Pop=10		Pop=20		Pop=30		Pop=40		Pop=50	
	Std.Sap.	GAP	Std.Sap.	GAP	Std.Sap.	GAP	Std.Sap.	GAP	Std.Sap.	GAP
Cap71	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap72	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap73	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap74	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap101	0.00	0.00	428.66	0.04	0.00	0.00	0.00	0.00	0.00	0.11
Cap102	0.00	0.15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap103	435.75	0.04	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap104	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap131	405.56	0.29	861.40	0.10	219.63	0.24	0.00	0.00	261.98	0.25
Cap132	439.16	0.06	421.27	0.04	119.82	0.02	0.00	0.00	166.50	0.03
Cap133	511.01	0.09	372.28	0.14	298.61	0.04	0.00	0.12	311.18	0.04
Cap134	97.83	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Avg(GAP)		0.05		0.03		0.03		0.01		0.04

Table 4.34. Farklı popülasyon boyutları ve *lb2* limit değeri ile elde edilen sonuçlar

Problem	Pop=10		Pop =20		Pop =30		Pop=40		Pop =50	
	Std.Sap.	GAP	Std.Sap.	GAP	Std.Sap.	GAP	Std.Sap.	GAP	Std.Sap.	GAP
Cap71	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap72	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap73	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap74	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap101	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap102	0.00	0.15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap103	415.27	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap104	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap131	383.02	0.30	855.17	0.11	323.74	0.26	397.92	0.02	337.28	0.26
Cap132	436.75	0.05	632.39	0.06	291.02	0.04	133.53	0.01	327.21	0.04
Cap133	422.17	0.08	505.42	0.13	345.44	0.06	273.86	0.13	348.64	0.06
Cap134	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
AVG(GAP)		0.05		0.02		0.03		0.01		0.03

Tablo 4.35. Farklı popülasyon boyutları ve *lb* limit değeri ile elde edilen sonuçlar

Problem	Pop=10		Pop =20		Pop =30		Pop=40		Pop =50	
	Std.Sap.	GAP	Std.Sap.	GAP	Std.Sap.		Std.Sap.	GAP	Std.Sap.	GAP
Cap71	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap72	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap73	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap74	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap101	0.00	0.00	350.00	0.02	0.00	0.00	0.00	0.00	0.00	0.00
Cap102	0.00	0.15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap103	453.28	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap104	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap131	396.65	0.30	1030.47	0.13	392.69	0.28	727.24	0.06	261.98	0.25
Cap132	475.12	0.06	479.43	0.05	620.86	0.05	272.23	0.02	200.24	0.03
Cap133	444.47	0.07	584.87	0.15	477.98	0.07	252.61	0.13	503.58	0.07
Cap134	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
AVG(GAP)		0.05		0.03		0.03		0.02		0.03

Tablo 4.36. Farklı popülasyon boyutları ve *lc2* limit değeri ile elde edilen sonuçlar

Problem	Pop=10		Pop=20		Pop=30		Pop=40		Pop=50	
	Std.Sap.	GAP	Std.Sap.	GAP	Std.Sap.		Std.Sap.	GAP	Std.Sap.	GAP
Cap71	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap72	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap73	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap74	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap101	0.00	0.00	428.66	0.04	0.00	0.00	0.00	0.00	0.00	0.11
Cap102	0.00	0.15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap103	496.70	0.06	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap104	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap131	527.26	0.32	1517.80	0.27	357.95	0.26	1075.24	0.10	360.10	0.26
Cap132	684.52	0.09	658.90	0.06	419.29	0.06	423.63	0.03	414.31	0.05
Cap133	433.39	0.09	1300.91	0.19	480.09	0.09	368.93	0.14	387.44	0.09
Cap134	163.49	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
AVG(GAP)		0.06		0.05		0.03		0.02		0.04

Tablo 4.37. Farklı popülasyon boyutları ve *lc4* limit değeri ile elde edilen sonuçlar

Problem	Pop=10		Pop=20		Pop=30		Pop=40		Pop=50	
	Std.Sap.	GAP	Std.Sap.	GAP	Std.Sap.		Std.Sap.	GAP	Std.Sap.	GAP
Cap71	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap72	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap73	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap74	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap101	0.00	0.00	386.94	0.03	218.26	0.01	0.00	0.00	0.00	0.11
Cap102	0.00	0.15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap103	466.38	0.07	0.00	0.00	105.35	0.01	91.96	0.01	78.15	0.00
Cap104	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cap131	817.66	0.38	1321.66	0.22	516.17	0.30	860.40	0.07	371.85	0.27
Cap132	1793.61	0.19	946.09	0.10	757.16	0.07	464.00	0.05	549.11	4.98
Cap133	1609.71	0.19	1505.94	0.26	810.38	0.13	342.29	0.15	549.11	0.09
Cap134	494.87	0.02	0.00	0.00	135.94	0.00	0.00	0.00	945.59	0.02
AVG(GAP)		0.08		0.05		0.04		0.02		0.46

Tablolardaki GAP değeri, elde edilen ortalama çözümün maliyetinin problemin optimum çözümünün maliyetinden oransal fazlalığını (göreceli hatasını) belirten değerdir ve Denklem 4.22 ile hesaplanır.

$$GAP = \frac{f_{mean} - f_{opt}}{f_{opt}} \times 100 \tag{4.22}$$

f_{mean} algoritmanın 30 defa koşturulmasından elde edilen sonuçların ortalaması, f_{opt} ise problemin optimum çözümünün maliyetidir ve Tablo 4.38'de problemler ve optimum çözüm maliyetleri verilmiştir. Bu problemler ORLIB'den (operations research library- yöneylem araştırma kütüphanesi) alınmıştır (Beasley, 1990).

Tablo 4.38. Kapasitesiz tesis yerleşim problemleri test kümesi

Problem	Problem Boyutu	Optimum Çözüm Maliyeti
Cap71	16×50	932615.75
Cap72	16×50	977799.40
Cap73	16×50	1010641.45
Cap74	16×50	1034976.98
Cap101	25×50	796648.44
Cap102	25×50	854704.20
Cap103	25×50	893782.11
Cap104	25×50	928941.75
Cap131	50×50	793439.56
Cap132	50×50	851495.33
Cap133	50×50	893076.71
Cap134	50×50	928941.75
CapA	100×1000	17156454.48
CapB	100×1000	12979071.58
CapC	100×1000	11505594.33

Analizlerin raporlandığı tablolardan görüldüğü üzere küçük limit değerlerinde sonuçlar nispeten daha iyidir. Çünkü önerilen denklem ile sadece bir boyut değiştirilmekte bu da yöntem için yerel araştırmaya karşılık gelmektedir. Global araştırma ise kaşif arılar tarafından yapılmakta dolayısıyla küçük limit değerlerinde daha fazla kaşif arı oluşmakta ve yöntemin global araştırma yeteneği de güçlendirilmektedir. Küçük popülasyon boyutlarında yöntem daha az noktadan araştırmaya başladığı için uzayı verimli bir şekilde tarayamamaktadır. Bundan dolayı popülasyon boyutu için 40 civarı bir değer makul görülmektedir.

4.2.2.2. Deneysel Sonuçlar ve Karşılaştırmalar

15 test problemi üzerinde analiz edilen parametrelerin en uygun değerleri kullanılarak yöntem ile elde edilen sonuçlar BPSO, IBPSO ve DisABC metotlarıyla doğruluk, sağlamlık ve çalıştırma zamanı dikkate alınarak karşılaştırılmıştır. Tüm yöntemler aynı bilgisayar (Intel i5 3.1 Ghz Cpu, 4GB Ram, Windows 7, Matlab Platformu) üzerinde koşturulmuş ve tablo boyutlarından dolayı elde edilen sonuçlar Tablo 4.39, Tablo 4.40 ve Tablo 4.41'de ayrı ayrı sunulmuştur. Çözüm kaliteleri ve yöntemin standart sapma tabanında sağlamlığı sonuçların raporlandığı tabloda sunulmakla birlikte çalışma zamanı açısından karşılaştırma Şekil 4.28'da verilmiştir. Çalışma zamanları ilgili platformun tic-toc fonksiyonları ile elde edilmiştir.

Tablo 4.39. BPSO ve binABC yöntemlerinin çözüm kalitesi ve sağlamlık açısından karşılaştırılması

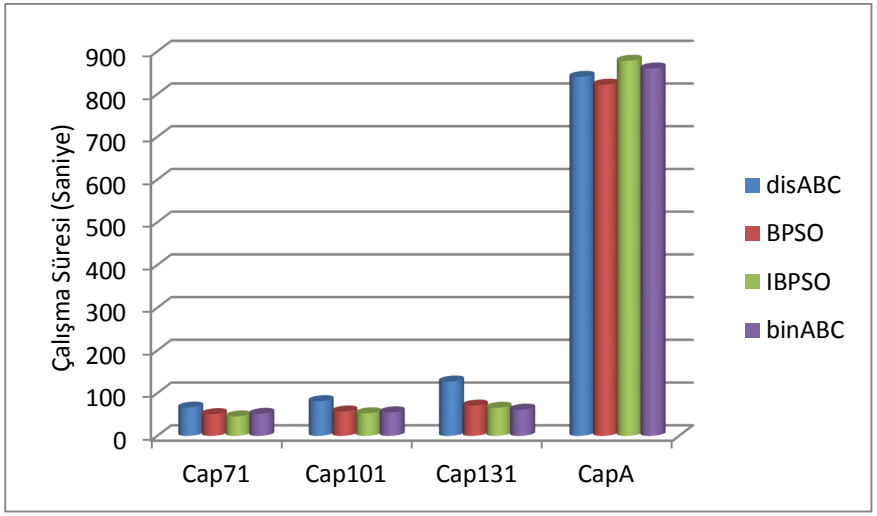
Problem	BPSO					binABC				
	En İyi	En Kötü	Ortalama	GAP	Std.Sap.	En İyi	En Kötü	Ortalama	GAP	Std.Sap.
Cap71	932615.75	932615.75	932615.75	0.0000	0.00	932615.75	932615.75	932615.75	0.0000	0.00
Cap72	977799.4	977799.4	977799.4	0.0000	0.00	977799.4	977799.4	977799.4	0.0000	0.00
Cap73	1010886.187	1012476.975	1010886.187	0.0242	634.62	1010641.45	1010641.45	1010641.45	0.0000	0.00
Cap74	1035068.312	1037717.075	1035068.312	0.0088	500.27	1034976.975	1034976.975	1034976.975	0.0000	0.00
Cap101	797016.6558	799092.1125	797016.6558	0.0462	566.44	796648.4375	796648.4375	796648.4375	0.0000	0.00
Cap102	854830.955	855971.75	854830.955	0.0148	386.76	854704.2	854704.2	854704.2	0.0000	0.00
Cap103	894159.4667	895027.1875	894159.4667	0.0422	485.26	893782.1125	893782.1125	893782.1125	0.0000	0.00
Cap104	929694.4467	934586.975	929694.4467	0.0810	1951.81	928941.75	928941.75	928941.75	0.0000	0.00
Cap131	794484.4496	797735.5375	794484.4496	0.1317	1207.63	793439.5625	793439.5625	793439.5625	0.0000	0.00
Cap132	852273.2071	855328.675	852273.2071	0.0914	1196.19	851495.325	851495.325	851495.325	0.0000	0.00
Cap133	894072.9054	896661.5625	894072.9054	0.1115	821.28	894095.7625	894752.025	894161.3888	0.1215	200.24
Cap134	930192.0425	934586.975	930192.0425	0.1346	2285.42	928941.75	928941.75	928941.75	0.0000	0.00
CapA	17530210.57	18682895.54	17530210.57	2.1785	374302.81	17180539.56	18030263.31	17664663.43	2.9622	236833.5
CapB	13232039.15	13633079.9	13232039.15	1.9490	176206.07	13100041.02	13476652.7	13304594.27	2.5081	91430.13
CapC	11676684.07	11871643.26	11676684.07	1.4870	92977.85	11535255.5102	11867887.0012	11802532.8641	2.5800	82312.70

Tablo 4.40. DisABC ve binABC yöntemlerinin çözüm kalitesi ve sağlamlık açısından karşılaştırılması

Problem	DisABC					binABC				
	En İyi	En Kötü	Ortalama	GAP	Std.Sap.	En İyi	En Kötü	Ortalama	GAP	Std.Sap.
Cap71	932615.75	932615.7500	932615.75	0.0000	0.00	932615.75	932615.75	932615.75	0.0000	0.00
Cap72	977799.4	977799.4	977799.4	0.0000	0.00	977799.4	977799.4	977799.4	0.0000	0.00
Cap73	1010641.45	1010641.4500	1010641.45	0.0000	0.00	1010641.45	1010641.45	1010641.45	0.0000	0.00
Cap74	1034976.975	1034976.9750	1034976.975	0.0000	0.00	1034976.975	1034976.975	1034976.975	0.0000	0.00
Cap101	796648.4375	796648.4375	796648.4375	0.0000	0.00	796648.4375	796648.4375	796648.4375	0.0000	0.00
Cap102	854704.2	854704.2000	854704.2	0.0000	0.00	854704.2	854704.2	854704.2	0.0000	0.00
Cap103	893782.1125	893782.1125	893782.1125	0.0000	0.00	893782.1125	893782.1125	893782.1125	0.0000	0.00
Cap104	928941.75	928941.75	928941.75	0.0000	0.00	928941.75	928941.75	928941.75	0.0000	0.00
Cap131	794299.85	802709.225	798355.4917	0.6196	2337.64	793439.5625	793439.5625	793439.5625	0.0000	0.00
Cap132	851495.325	854704.2	852300.2575	0.0945	813.37	851495.325	851495.325	851495.325	0.0000	0.00
Cap133	893076.7125	894095.7625	893352.4167	0.0309	359.03	894095.7625	894752.025	894161.3888	0.1215	200.24
Cap134	928941.75	928941.75	928941.75	0.0000	0.00	928941.75	928941.75	928941.75	0.0000	0.00
CapA	17156454.48	17420032.38	17182558.16	0.1522	74782.61	17180539.56	18030263.31	17664663.43	2.9622	236833.5
CapB	13205522.04	13683628.78	13407728.05	3.3027	109738.5	13100041.02	13476652.7	13304594.27	2.5081	91430.13
CapC	11834640.02	12203264.48	12045991.08	4.6968	95778.78	11535255.5102	11867887.0012	11802532.8641	2.5800	82312.70

Tablo 4.41. IBPSO ve binABC yöntemlerinin çözüm kalitesi ve sağlamlık açısından karşılaştırılması

Problem	IBPSO					binABC				
	En İyi	En Kötü	Ortalama	GAP	Std.Sap.	En İyi	En Kötü	Ortalama	GAP	Std.Sap.
Cap71	932615.75	934199.1375	932964.7188	0.0374	587.49	932615.75	932615.75	932615.75	0.0000	0.00
Cap72	977799.4	983122.2375	980486.9238	0.2749	1844.64	977799.4	977799.4	977799.4	0.0000	0.00
Cap73	1010641.45	1014917.6	1012642.103	0.1980	1513.78	1010641.45	1010641.45	1010641.45	0.0000	0.00
Cap74	1034976.975	1045383.788	1039149.436	0.4031	4426.67	1034976.975	1034976.975	1034976.975	0.0000	0.00
Cap101	796648.4375	809077.4875	801403.0875	0.5968	3799.52	796648.4375	796648.4375	796648.4375	0.0000	0.00
Cap102	856660.0125	865438.2	860957.6625	0.7317	3249.38	854704.2	854704.2	854704.2	0.0000	0.00
Cap103	894008.1375	909765.25	899511.3063	0.6410	4978.98	893782.1125	893782.1125	893782.1125	0.0000	0.00
Cap104	928941.75	964540.85	938197.2625	0.9964	10845.26	928941.75	928941.75	928941.75	0.0000	0.00
Cap131	806761.2875	821236.3625	812669.325	2.4236	4244.29	793439.5625	793439.5625	793439.5625	0.0000	0.00
Cap132	865407.5125	901297.475	882160.8613	3.6014	11569.02	851495.325	851495.325	851495.325	0.0000	0.00
Cap133	918298.525	966072.7125	940076.1188	5.2626	14905.27	894095.7625	894752.025	894161.3888	0.1215	200.24
Cap134	973744.7875	1027632.238	999855.6225	7.6338	15788.86	928941.75	928941.75	928941.75	0.0000	0.00
CapA	35704093	45511134.48	40812839.67	137.8862	3357138.19	17180539.56	18030263.31	17664663.43	2.9622	236833.5
CapB	17971425.35	22154126.28	20152622.54	55.2701	1406575.7	13100041.02	13476652.7	13304594.27	2.5081	91430.13
CapC	14564601.36	18555781.02	16747099.16	45.5561	1245252.2	11535255.5102	11867887.0012	11802532.8641	2.5800	82312.7



Şekil 4.28. Yöntemlerin çalışma sürelerinin karşılaştırılması

Karşılaştırma doğruluk açısından incelendiğinde görülür ki yöntem benzerlerine göre daha iyi sonuçlar üretmektedir. DisABC, PSO ve IBPSO yöntemleri her çözüm güncelleştirmesinde birden fazla boyutu değiştirdiği için yöntemin pertürbasyonu fazla olmakta bu da yerel araştırmayı zayıflatmaktadır. Önerilen yöntemde yerel araştırma denklemdeki bir boyutun değiştirilmesi üzerine kurulduğundan ve global araştırmanın kaşif arılar tarafından yapılmasından dolayı önerilen yaklaşım daha dengeli ve güçlü bir araştırma kapasitesine sahiptir. Çözüm kalitelerinin her çalıştırmada iyi olması nedeniyle genellikle standart sapma tabanlı bir sağlamlık karşılaştırmasında önerilen yöntem sağlam olarak görülmektedir. Zamansal karşılaştırmada ise yöntemlerin çalışma süreleri arasında ciddi farklar gözlenmemiştir. Basitlik açısından bakıldığında önerilen yöntem BPSO yöntemi kadar basittir ve ürettiği sonuçlar açısından bu yöntemden daha başarılı olarak nitelendirilebilir. Aynı metodu temel alan DisABC yöntemi ise uyarlaması ve implementasyonu zor bir yöntem olarak göze çarpmaktadır. Bundan dolayı bu çalışmada önerilen yöntemin diğer ikili optimizasyon problemlerinin çözümünde diğerlerine nispeten araştırmacılar ve uygulayıcılar tarafından göz ardı edilmeyeceği düşünülmektedir.

5.SONUÇLAR VE ÖNERİLER

Optimizasyon problemlerinin çözümünde önerilen metotlardan gürbüz olmaları, yüksek çözüm kalitesi ve kısa çalışma zamanı beklenmektedir. Özellikle büyük çözüm uzayına sahip gezgin satıcı gibi problemler için bilinen kesin çözüm metotları optimum çözüme ulaşmak için uzun çalışma zamanı gerektirmektedir. Çok boyutlu sürekli optimizasyon problemlerinin çözümünde ise problemin sürekli ve/veya türevlenebilir olmaması nedeniyle ya bilinen kesin çözüm yöntemleri hiç uygulanamamakta ya da ağır matematik işlemler gerektirmektedir. Ayrıca farklı problemler için düşük adaptasyon seviyeleri de kesin yöntemlerin tercih edilmeme sebeplerinden biridir. Bu noktada düşük maliyetli ve birden fazla problem için az değişiklik ile uygulanabilecek genel yaklaşık optimizasyon yöntemlerine ihyiyaç duyulmaktadır. Bu yöntemler kesin çözümü (optimum) garanti edememelerine rağmen optimal veya yakın optimal çözümü elde etmede oldukça baarılıdırlar. Ayrıca bu yöntemler düşük hesaplama maliyeti ve probleme kolay adaptasyon nedeniyle farklı disiplinlerde oldukça popüler hale gelmişlerdir.

Bu tez çalışması kapsamında sürü zekâsı metotları incelenmiş ve sürekli ve ayrık optimizasyon problemlerinin çözümü için ABC tabanlı yeni yaklaşımlar önerilmiştir. Önerilen yaklaşımlar literatürde var olan kıyas problemleri için uygulanmış ve elde edilen sonuçlar problemlerin optimum çözümleri ile karşılaştırılmıştır. Ayrıca önerilen metotları doğrulamak amacıyla bilinen diğer sürü zekâsı yöntemleri ile kıyaslamalar yapılmıştır. Önerilen çalışmalar üç başlık altında değerlendirilebilir. Birinci başlıkta sürekli optimizasyon problemlerinin çözümü yeni yaklaşımlar geliştirilmesi için zaman ve çaba sarfedilmiştir. İlk çalışmada çaprazlama operatörleri yapay arı kovanındaki görevli arılar ile gözcü arılar arasındaki bilgi paylaşımını güçlendirerek yöntem yerel arama kabiliyeti güçlendirilmeye çalışılmıştır. Önerilen yeni yaklaşım literatürde sıklıkla kullanılan nümerik kıyas fonksiyonlarına uygulanmış ve elde edilen sonuçlar ABC algoritmasının temel versiyonu ile kıyaslanmıştır. İkinci çalışmada ise sürü zekâsı yöntemleri elektrik enerjisi tahmini için uygulanmıştır. Bu çalışmada ABC yönteminin elektrik enerjisi tahmini probleminin çözümünde yakınsama hızını arttırabilmek ve yerel aramayı güçlendiribilmek amacıyla hem birinci çalışmadaki metot hem de en iyi çözüm tabanlı bir yaklaşım önerilmiştir. Son yıllarda elektrik talebine yöntemlerden elde edilen sonuçların benzerliği noktasında PSO ve ACO yöntemleri ile kıyaslamalar yapılmış ve elektirik talebi için oluşturulan senaryolar üzerinde yöntemlerden elde

edilen tahmin sonuçları raporlanmıştır. Sürekli optimizasyon problemlerinin çözümü için yapılan son çalışmada ABC ve PSO yöntemleri hibritleştirilerek yeni bir hibrit yöntem ortaya konulmuştur. Hibritleştirmenin temelinde rekombinasyon süreci bulunmaktadır. Bu süreçte farklı sürüler arasındaki etkileşim arttırılarak sıkı bir işbirliği sağlanması amaçlanmıştır. Önerilen yöntem çeşitli problemlerin çözümü için kullanılmış ve elde edilen sonuçlar bildirilmiştir. İkinci başlıkta ABC algoritmasının ayrık problemlerin çözümü için analiz edilmesi, uyarlanması ve yeni yaklaşımlar geliştirilmesi üzerinde durulmuştur. Öncelikle dokuz komşuluk operatörü ile ABC algoritmasının gezgin satıcı problemi üzerinde başarısı test edilmiştir. Elde edilen sonuçlara göre en başarılı sonuçlara sahip komşuluk operatörü belirlenmiş ve hiyerarşik bir yakılaşım geliştirilmiştir. Bu hiyerarşik yaklaşımda önce karınca kolonisi optimizasyon algoritması ile ABC için başlangıç çözümü oluşturulmuş ve daha sonra bu başlangıç çözümün ayrık ABC algoritmasının iyileştirilmesi sağlanmıştır. Üçüncü başlıkta ise ayrık problem olmasına rağmen kendine has karakteristikleri bulunan ikili optimizasyon problemlerinin çözümü için ABC algoritması modifiye edilmiştir. Bu yaklaşımda ABC'nin yapay arılarının ikili olarak yapılandırılmış çözüm uzayında hareket edebilmesi sağlanmıştır.

Tez kapsamında ayrık, ikili veya sürekli olsun ele alınan optimasyon problemlerinin çözümü için ABC algoritmasının aşağıda maddelenen yeteneklerinin iyileştirilmesi veya dengelenmesi üzerinde durulmuştur.

i) güncelleme kuralı- pozisyon güncelleme denklemi- araştırma stratejisi

ii) popülasyonun bireyleri arasındaki bilgi paylaşımı

iii) yerel arama ve global arama arasındaki denge

Temel çalıştırma ve analiz sonuçlar göstermiştir ki ABC veya diğer sürü zekâsı yöntemlerinde iyileştirilmesi gereken temel aksaklıklar popülasyonun durağanlaşması, yoğunlaşamaması, araştırma uzayını verimli tarayamaması olarak göze çarpmaktadır. Bundan dolayıdır ki yukarıda anılan ABC tabanlı iyileştirmelerin tümü bu olumsuzlukları ortadan kaldırmak amacıyla ortaya konmuştur.

Sürü zekâsı yöntemleri temel versiyonları ile kabul edilebilir sonuçlar üretebiliyorsa literatüre kazandırılmalıdır. İyileştirmeler çeşitli araştırmacılar tarafından yapılmaktadır ve yöntemin varabileceği en iyi noktaya götürülmesi zamanla sağlanmaktadır. Yeni çalışmalarda aşağıdaki hususlara özellikle dikkat edilmelidir;

i) Global ve yerel arama arasında iyi bir denge kurulmalıdır ki popülasyonun hem çözüm uzayını verimli bir şekilde taraması sağlansın hem de bulunan çözümler üzerinde yoğunlaşma sağlanabilsin.

ii) Yapay arı kolonisi algoritmasındaki gibi kaşif arı benzeri mekanizmalar bu yöntemlerde olmalı ki popülasyonun durağanlaşması engellenerek popülasyonun sürekli bir devinim içerisinde olması sağlanabilsin.

iii) Popülasyonun bireyleri arasındaki işbirliği sıkı bir şekilde işletilmeli ki popülasyonun bulduğu çözümler üzerinde yerel araştırma artabilsin.

Sonuçlar ve öneriler sunulduktan sonra son olarak gelecek çalışmalara ait bir planlama verilmesi yerinde olacaktır. ABC algoritması 2005 yılından bu yana güncellenmeye, iyileştirilmeye ve farklı problemlere uygulanmaya devam edilmektedir. Özellikle son yıllarda sürü zekâsının bu güçlü yöntemi üzerine literatürde çok sayıda makale ortaya çıkmaktadır. ABC'nin gelişimi dikkate alındığında ortaya çıkan sonuç şudur ki ABC algoritmasının çözüm uzayına bağlı olarak araştırma stratejisini güncelleyebilmesi ve öğrenebilir bir yapıya kavuşması gerekmektedir. Bundan dolayı gelecek çalışmalarımızda birden fazla araştırma stratejisine sahip ABC tabanlı yeni yaklaşımlar üzerine araştırmalarımız devam edecektir.

KAYNAKLAR

Abbass H.A., 2001, Marriage in Honey Bees Optimisation: A Haplometrosis Polygynous Swarming Approach, Proceedings of the IEEE congress on Evolutionary Computation, vol. 1, s. 207–214.

Aderhold A., Diwold K., Scheidler A., Middendorf M., 2010, Artificial bee colony optimization: a new selection scheme and its performance, Ed: Gonzlez J., Pelta D., Cruz C., Terrazas G., Krasnogor N., Nature-inspired Cooperative Strategies for Optimization, Studies in Computational Intelligence, vol. 284.s. 283–294

Agrafiotis, D. K., & Cedeño, W. (2002). Feature selection for structure-activity correlation using binary particle swarms. Journal of Medicinal Chemistry, 45(5), 1098–1107

Akay B., 2009, Nümerik Optimizasyon Problemlerinde Yapay Arı Kolonisi (Artificial Bee Colony) Algoritmasının Performans Analizi, Doktora Tezi, *Erciyes Üniversitesi Fen Bilimleri Enstitüsü*, Kayseri.

Akay B., Karaboğa D., 2009, Solving integer programming problems by using artificial bee colony algorithm, Ed.: Serra R., Cucchiara R., Proceedings of the XIth International Conference of the Italian Association for Artificial Intelligence Reggio Emilia on Emergent Perspectives in Artificial Intelligence, University Modena Reggio Emilia, Lecture notes in Artificial Intelligence, vol 5883, s. 355–364

Akay B., Karaboğa D., 2009a, Parameter tuning for the artificial bee colony algorithm, Ed: Nguyen N.T., Kowalczyk R., Chen S.M., Computational Collective Intelligence: Semantic Web, Social Networks and Multiagent Systems, Lecture Notes in Artificial Intelligence, vol. 5796, s. 608–619

Akay B., Karaboğa D., 2010, A modified artificial bee colony algorithm for real-parameter optimization, Information Sciences, vol. 192, s. 120-142.

Al-kazemi B., Mohan C.K., 2002, Multi-phase Discrete Particle Swarm Optimization. Proceedings of Fourth International Workshop on Frontiers in Evolutionary Algorithms, s. 622-625, Kinsale, Ireland.

Angus D., 2006, Niching for population-based ant colony optimization, Second IEEE International Conference on e-Science and Grid Computing, s. 115-122, Amsterdam, Netherlands

Angus, D., 2007, Population-based Ant Colony Optimisation for Multiobjective Function Optimisation, Progress in Artificial Life, Ed: Randall M., Abbass H.A., Wiles J., Lecture Notes in Computer Science, vol. 4828, s. 232-244.

Aydın D., 2011, Sürü Zekâsı Yaklaşımlarının Renkli Görüntü Kesimlemeye Uyarlanması ve Tanıma Sistemleri Üzerinde Gerçekleştirimi, Ege Üniversitesi Fen Bilimleri Enstitüsü, Doktora Tezi, Bornova-İzmir.

Bao L., Zeng J.C., 2009, Comparison and analysis of the selection mechanism in the artificial bee colony algorithm, Proceedings of the Ninth International Conference on Hybrid Intelligent Systems, s. 411–416, 12-14 August, Shenyang, China.

Beasley J.E., 1990, OR-Library: Distributing Test Problems by Electronic Mail, The Journal of the Operational Research Society, vol. 41, ss. 1069-1072.

Blum C., 2005, Beam-ACO—hybridizing ant colony optimization with beam search: an application to open shop scheduling, Computers and Operations Research, vol. 32, s.1565-1591

Blum C., Aguilera M. J. B., Roli A., Sampels M. (Editörler), 2008, Hybrid Metaheuristics: An Amerging Approach to Optimization, Studies in Computational Intelligence, vol. 114, Springer-Verlag Heidelberg, Berlin.

Blum C., Dorigo, M., 2004, The hyper-cube framework for ant colony optimization, IEEE Transactions on Systems, Man, and Cybernetics–Part B vol. 34(2), s. 1161-1172

Blum C., Roli A., Dorigo M., 2001, HC–ACO: The hyper-cube framework for Ant Colony Opti-mization, Proceedings of Metaheuristics International Conference, vol. 2, s. 399–403.

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

Boyer, D.O., Martinez, C.H., Pedrajas, N.G., 2005, Crossover operator for evolutionary algorithms based on population features, *Journal of Artificial Intelligence Research*, 24, 1-48

Bullnheimer B., Hartl R. F., Strauss C., 1997, A new rank based version of the Ant System-A computational study. Technical Report, Institute of Management Science, University of Vienna, Austria.

Bullnheimer B., Hartl R. F., Strauss C., 1999, A new rank-based version of the Ant System: A computational study, Central European Journal for Operations Research and Economics, vol. 7(1), s. 25–38.

Caia X., Cuib Z., Zenga J., Tana Y., 2008, Dispersed particle swarm optimization, Information Processing Letters, vol. 105(6), s. 231–235

Carlisle A., Dozier G., 2000, Adapting Particle Swarm Optimization to Dynamic Environments, Proceedings of the International Conference on Artificial Intelligence, s. 429-434.

Cedeño W, Agrafiotis DK, 2003, Application of niching particle swarms to QSAR and QSPR, Ed: Ford M., EuroQSAR 2002 Designing Drugs and Crop Protectants: Processes, Problems and Solutions, Blackwell Publishing, s. 255–259.

Chen D., Zhao C., 2009, Particle Swarm Optimization with Adaptive Population Size and its Applications, Applied Soft Computing, vol. 9(1), s. 39-48.

Chu S. C., Tsai P. W., 2007, Computational Intelligence based on the Behavior of Cats, International Journal of Innovative Computing, Information and Control, 3, s. 163-173.

Chu, S. C., P. W. Tsai, J. S. Pan, Cat Swarm Optimization, Proceedings of 9th Pacific Rim International Conference on Artificial Intelligence, vol. 4099, s. 854-858, 2006

Cordon, O., Viana, I.F. and Moreno, L., 2000, New ACO Model Integrating Evolutionary Computation Concepts: The Best-Worst Ant System, In Proceeding of ANTS 2000, s. 22-29, Brussels, Belgium.

de Oca M.A.M., Stutzle T., Van den Enden K., Dorigo M., 2011, Incremental Social Learning in Particle Swarms, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 41 (2), s. 368-384.

Dorigo M., 1992, Optimization, Learning and Natural Algorithms, Ph.D. Thesis, Dipartimento di Elettronica, Politecnico di Milano, Milan, Italy, 1992.

Dorigo M., Gambardella L.M., 1996, A study of some properties of Ant-Q, Ed: Voigt H., Ebeling W., Rechenberg I., Schwefel H., Proceedings Fourth Conference on Parallel Problem Solving from Nature, Lecture Notes in Computer Science, vol. 1141, s.656-665.

Dorigo M., Maniezzo, V., Colorni A., 1991, Positive Feedback as a Search Strategy, *Technical Report 91-016*, Politecnico di Milano, Milan Italy.

Dorigo M., Maniezzo V., Colorni A., 1996, Ant system: optimization by a colony of cooperating agents, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 26, ss. 29-41.

Dorigo M., Stützle T., 2004, Ant Colony Optimization, MIT Press, Massachusetts, USA.

Dorigo M., Stützle T., 2010, Ant Colony Optimization: Overview and Recent Advances, Ed: Gendreau M., Potvin J.-Y., Handbook of Metaheuristics, International Series in Operations Research & Management Science, vol. 146, s. 227-263

Dorigo, M., Gambardella, L. M., (1997b), Ant Colony System: A cooperative learning approach to the traveling salesman problem, IEEE Transactions on Evolutionary Computation, 1(1), s. 53–66.

Dorigo, M., Gambardella, L. M., 1997a, Ant colonies for the traveling salesman problem, BioSystems, 43(2), s. 73–81.

Eberhart R. C., Kennedy J., 1995, A new optimizer using particle swarm theory. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, s. 39–43, Nagoya, Japan.

Eberhart R. C., Shi Y., 2000, Comparing inertia weights and constriction factors in particle swarm optimization, Proceedings of the IEEE Congress on Evolutionary Computation, s. 84–88, San Diego, CA, USA.

Eberhart R. C., Shi Y., 2001, Tracking and optimizing dynamic systems with particle swarms, Proceedings of the IEEE Congress on Evolutionary Computation s. 94–100, Seoul, Korea.

Eberhart, R.C., Kennedy, J.A., 1995, A new optimizer using particle swarm theory, *Proc. of the Sixth International Sysmposium on Micromachine and Human Science*, Nagoyaa, Japan, 39-43.

El-Abd M., 2011, A hybrid ABC-SPSO algorithm for continuous function optimization, Proc. Of IEEE Symposium on Swarm Intelligence, s..1-6, 11-15 April, Paris, France.

El-Abd M., 2012, Performance assessment of foraging algorithms vs. evolutionary algorithms, Information Sciences, vol. 182(1), s. 243–263.

Engelbrecht A.P. ,2005, Fundamentals of Computational Swarm Intelligence, John Willey, West Sussex, England.

Fogel D.B., 1997, The Advantages of Evolutionary Computation, Biocomputing and Emergent Computation: Proceedings of BCEC97, pp.1-11.

Fogel, D.B., 1995, Evolutionary Computation: Toward a New Philosophy of Machine Intelligence, IEEE Press, New York.

Fourie P.C., Groenwold A.A, 2002, The particle swarm optimization algorithm in size and shape optimization, Structural and Multidisciplinary Optimization, 23, s. 259–267.

Gambardella L. M., Dorigo M., 1995, Ant-Q: A reinforcement learning approach to the traveling salesman problem, Proceedings of the Twelfth International Conference on Machine Learning, s. 252–260, Palo Alto, California, USA.

Gao W.-f, Liu S.-y, 2012, A modified artificial bee colony algorithm, Computers&Operations Research, vol. 39 (3), s. 687-697.

Gao W.-f, Liu S.-y, Lingling H., 2012, A global best artificial bee colony algorithm for global optimization, Journal of Computational and Applied Mathematics, vol. 236 (11), s. 2741-2753

Gazi V., Passino K.M., 2011, Swarm Stability and Optimization, Springer, e-ISBN: 978-3-642-18041-5.

Glover, F., 1989, Tabu Search – Part I, *ORSA Journal on Computing*, 1(3), 190-206.

Glover, F., 1990, Tabu Search – Part II, *ORSA Journal on Computing*, 2(1), 4-32.

Grassé, P.P., 1959, La Reconstruction dun id et les Coordinations Interindividuelles Chez Bellicosimetes Natalensis et Cubitemes sp. la Théorie de la Stigmerie: Essai d'Inrprétation du Comportement des Termites Constructeurs. Insectes Sociaux, 6, 41-81

Guntsch M., Middendorf M., 2002a, Applying population-based ACO to dynamic optimization problems, Ed: Dorigo M., Di Caro G.A., Sampels, M., ANTS2002, Lecture Notes in Computer Sciences, vol. 2463, s. 97–104

Guntsch, M., Middendorf, M., 2002b, A population based approach for ACO, EvoWorkshops2002, Ed: Cagnoni S., Gottlieb J., Hart E., Middendorf M., Raidl G.R, Lecture Notes in Computer Sciences, vol. 2279, s. 72–81.

Guo P., Cheng W., Liang J., 2011, Global artificial bee colony search algorithm for numerical function optimization, Proceedings of Seventh International Conference on Natural Computation, s. 1280–1283, 26-28 July, Shanghai, China.

Gündüz M., Kıran M.S., Özceylan E., 2013, A hierarchic approach based on swarm intelligence to solve traveling salesman problem, Turkish Journal of Electrical Engineering and Computer Sciences, doi:10.3906/elk-1210-147.

Hiroyasu T., Miki M., Ono Y., Minami Y., 2000, Ant colony for continuous functions, The Science and Engineering, vol. XX (Y), Doshisha University, Japan.

Holland, J.H., 1975, Adaptation in Natural and Artificial Systems, *University of Michigan Press*, Ann Arbor - Michigan,.

Janson S., Middendorf M., 2005, A hierarchical particle swarm optimizer and its adaptive variant, IEEE Transactions on System Man and Cybernetics Part –B: Cybernetics, vol. 35(6), s. 1272–1282

Jiang M., Yuan D., Cheng Y., 2009, Improved Artificial Fish Swarm Algorithm, Proceedings of Fifth International Conference on Natural Computation, s. 281-285, 14-16 Ağustos, Tianjin, Çin.

Karaboğa D, 2012, Yapay Arı Kolonisi Algoritması Matlab Kodu, Artificial Bee Colony (ABC) Algorithm Web Page, http://mf.erciyes.edu.tr/abc/pub/MATLABABCv2.rar [Erişim Zamanı: 15.11.2012 Saat: 23:20]

Karaboğa D., 2005, An Idea based on Honey Bee Swarm for Numerical Optimizatıon, Technical Report-Tr06, Erciyes University, Engineering Faculty, Computer Engineering Department, http://mf.erciyes.edu.tr/abc/pub/tr06_2005.pdf

Karaboğa D., Akay B., 2009, A comparative study of Artificial Bee Colony algorithm, Applied Mathematics and Computation, vol. 214, 108-132

Karaboğa D., Akay B., 2009a, Artificial bee colony (abc), harmony search and bees algorithms on numerical optimization, Innovative Production Machines and Systems Virtual Conference, <http://conference.iproms.org/conference/download/4153/76>

Karaboğa D., Baştürk B., 2007, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm, Journal of Global Optimization, vol. 39, s. 459–471.

Karaboğa D., Baştürk B., 2008, On the performance of artificial bee colony (abc) algorithm, Applied Soft Computing, vol. 8(1), s. 687–697.

Karaboğa D., Görkemli B., 2011, A combinatorial artificial bee colony algorithm for traveling salesman problem. In: 2011 international symposium on innovations in intelligent systems and applications (INISTA), s. 50–53.

Karaboğa D., Görkemli B., Öztürk C., Karaboğa N., 2012, A comprehensive survey: artificial bee colony (ABC) algorithm and applications, Artificial Intelligence Review, doi:10.1007/s10462-012-9328-0.

Karaboğa D., Ökdem S., 2004, A Simple and Global Optimization Algorithm for Engineering Problems: Differential Evolution, Turkish Journal of Electrical Engineering and Computer Sciences, vol. 12(1), s. 53-60.

Karaboğa D., Öztürk C., Karaboğa N., Görkemli B., 2012a, Artificial bee colony programming, Information Sciences, vol. 20, s. 1-15.

Karaboğa, D., 2011, Yapay Zekâ Optimizasyon Algoritmaları, *Nobel Yayın Dağıtım*, Genişletilmiş 2. Basım, Ankara.

Kashan M.H., Nahavandi N., Kashan A.H., 2011, Disabc: A new artificial bee colony algorithm for binary optimization, Applied Soft Computing, vol. 12(1), s.342-352.

Kashan M.H., Nahavandi N., Kashan A.H., 2011, DisABC: A new artificial bee colony algorithm for binary optimization, Applied Soft Computing, vol. 12, s. 342-352.

Kaveh A., Talatahari S., 2009, A particle swarm ant colony optimization for truss structures with discrete variables, Journal of Constructional Steel Research, vol. 65(8–9), s. 1558–1568

Kennedy J., 1997, The Particle Swarm: Social Adaptation of Knowledge, Proceedings of the IEEE International Conference on Evolutionary Computation, s. 303-308.

Kennedy J., 1999, Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance, Proceedings of the IEEE Congress on Evolutionary Computation, s. 1931–1938, Washington, USA.

Kennedy J., Eberhart R. C., 1997, A discrete binary version of the particle swarm algorithm, Proceedings of IEEE International Conference on Systems, Man, and Cybernetics,: Computational Cybernetics and Simulation, s. 4104–4109, Florida, USA.

Kennedy J., Eberhart R., 1997, A discrete binary version of the particle swarm algorithm, Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics, s. 4101-4109, Piscataway, NJ, USA.

Kennedy J., Eberhart, R. C., 1995, Particle swarm optimization, Proceedings of the IEEE International Conference on Neural Networks, vol. 4, s. 1942–1948, Piscataway, USA.

Kennedy J., Mendes R., 2002, Population structure and particle swarm performance, Proceedings of the IEEE Congress on Evolutionary Computation, s. 1671–1676, Honolulu, USA.

Kennedy, J., Eberhart, R. C, Particle Swarm Optimization. IEEE International Conference on Neural Networks, vol. IV, 1942-1948, Piscataway, NJ, 1995

Kıran M.S., Baykan Ö.K., Gündüz M., 2012, A novel hybrid algorithm based on particle swarm and ant colony optimization for finding the global minimum, Applied Mathematics and Computation, vol. 219(4), s. 1515-1521

Kıran M.S., Gündüz M., 2013, XOR-based Artificial Bee Colony Algorithm for Binary Optimization, Turkish Journal of Electrical Engineering and Computer Sciences, vol. 21, s.2307-2328.

Kıran M.S., Gündüz M., 2013b, A Recombination-based Hybridization of Particle Swarm Optimization and Artificial Bee Colony Algorithm for Continuous Optimization Problems, Applied Soft Computing, vol. 13, s. 2188-2203.

Kıran M.S., Gündüz M., 2012,A novel artificial bee colony-based algorithm for solving the numerical optimization problems, International Journal of Innovative Computing, Information and Control, vol. 8(9), 6107-6121.

Kıran M.S., İşcan H., Gündüz M., 2013, The analysis of discrete artificial bee colony algorithm with neighborhood operator on traveling salesman problem, Neural Computing and Applications, vol. 23(1), s. 9-21.

Kıran M.S., Özceylan E., Gündüz M., Paksoy T., 2012a, A novel hybrid approach based on Particle Swarm Optimization and Ant Colony Algorithm to forecast energy demand of Turkey, Energy Conversion and Management, vol. 53(1), s. 75-83.

Kıran M.S., Özceylan E., Gündüz M., Paksoy T., 2012b, Swarm intelligence approaches to estimate electricity energy demand in Turkey, Knowledge-based Systems, vol. 36, s. 93-103.

Kirkpatrick, S., Gelatt C.D., Jr., Vecchi M.P., 1953, Optimization by Simulated Annealing, *Science*, 220(4598), 671-680.

Krink T., 2012, Swarm Intelligence-Introduction, Department of Computer Science, University of Aarhus, faculty.washington.edu/paymana/swarm/krink_01.pdf,

Krishnanand K.N., Ghose D, 2005, Detection of Multiple Source Locations Using a Glowworm Metaphor with Applications to Collective Robotics, Proceedings of IEEE Swarm Intelligence Symposium, s. 84-91, 8-10 Haziran, Passenada-California, ABD.

Li C., Yang S., Nguyen T.T., 2012, A Self-Learning Particle Swarm Optimizer for Global Optimization Problems, IEEE Transactions on System Man and Cybernetics Part-B: Cybernetics, vol. 42(3), s. 627-646.

Li G., Niu P., Xiao X., Development and investigation of efficient artificial bee colony algorithm for numerical function optimization, Applied Soft Computing, vol. 12(1), 320-332.

Li W.H., Li W.J., Yang Y., Liao H.Q., Li J.L., Zheng X.P., 2011, Artificial bee colony algorithm for traveling salesman problem, Advanced Materials Research, vol. (314-316), s. 2191–2196

Li X., Yin M, 2012, A discrete artificial bee colony algorithm with composite mutation strategies for permutation flow shop scheduling problem, Scientia Iranica, doi:10.1016/j.scient.2012.10.034.

Liang J. J., Qin A. K., Suganthan P.N., Baska S., 2006, Comprehensive Learning Particle Swarm Optimizer for Global Optimization of Multimodal Functions, IEEE Transactions on Evolutionary Computation, vol. 10(3), s. 281-295.

Liang J. J., Suganthan P. N., 2005, Dynamic multiswarm particle swarm optimizer (DMS-PSO), Proceedings of the IEEE Swarm Intelligence Symposium, s. 124–129, California, USA.

Liu C., Yan X., Liu C., Wu H., 2011, The Wolf Colony Algorithm and Its Application, Chinese Journal of Electronics,20(2), s.212-216.

Liu Y., Qin Z., Shi Z., Lu J., 2007, Center particle swarm optimization, Neurocomputing, vol. 70(4–6), s. 672-679.

Machado L., Schirru R., 2002, The Ant-Q Algorithm applied to the nuclear reload problem, Annals of Nuclear Energy, vol. 29(12), 1455-1470.

Maniezzo V., 1999, Exact and approximate nondeterministic tree-search procedures for the quadratic assignment problem, INFORMS Journal on Computing, vol. 11(4), s. 358–369

Mendes R., 2004, Population topologies and their influence in particle swarm performance. PhD thesis, Departamento de Informatica, Escola de Engenharia, Universidade do Minho.

Mendes R., Cortes P., Rocha M., Neves J., 2002, Particle swarms for feedforward neural net training, Proceedings of the International Joint Conference on Neural Networks, s. 1895–1899, Honolulu, USA.

Mendes R., Kennedy J., Neves J., 2003, Watch thy neighbor or how the swarm can learn from its environment, Proceedings of the IEEE Swarm Intelligence Symposium, s. 88–94, Indiana, USA.

Michelakos I, Mallios N., Papageorgiou E., , Vassilakopoulos M, 2011, Ant Colony Optimization and Data Mining, Ed.: Bessis N., Xhafa F., Next Generation Data Technologies for Collective Computational Intelligence, Studies in Computational Intelligence, vol. 352, s. 31-60.

Millonas M.M., 1994, Swarms, phase transitions, and collective intelligence, Ed. Christopher G. Langton, Artificial Life III, Proc. Volume XVII Santa Fe Institute Studies in the Sciences of Complexity, Addison-Wesley, New York, s. 417-445.

Millonas M.M., 1994, Swarms, phase transitions, and collective intelligence, Ed. Christopher G. Langton, Artificial Life III, Proc. Volume XVII Santa Fe Institute Studies in the Sciences of Complexity, Addison-Wesley, New York, s. 417-445.

Parsopoulos K. E., Vrahatis M.N., UPSO–A unified particle swarm optimization scheme, Lecture Series on Computational Sciences, s. 868–873

Passino K.M., 2002, Biomimicry of bacterial foraging for distributed optimization and control, IEEE Control Systems Magazine, 22(3), s. 52–67.

Peram T., Veeramachaneni K., Mohan C., 2003, Fitness-distance-ratio based particle swarm optimization, Proceedings of the IEEE Swarm Intelligence Symposium s. 174–181, Indiana, USA.

Pan W.-T., 2012, A new Fruit Fly Optimization Algorithm: Taking the financial distress model as an example, Knowlegde-based Systems, vol. 26, s. 69-74.

Pham D.T., Ghanbarzadeh A., Koc E., Otri S., Rahim S., Zaidi M., 2005, The Bees Algorithm, Technical Report, Manufacturing Engineering Centre, Cardiff University, UK.

Poli R., Kennedy J., Blackwell T., Particle Swarm Optimization: An overview, Swarm Intelligence, 1, 33-57.

Rajabioun R., 2011, Cuckoo Optimization Algorithm, Applied Soft Computing, 11(8), 5508-5518.

Rajasekhar A., Abraham A., Pant M., 2011, Levy mutated artificial bee colony algorithm for global optimiza-tion, Proceedings of IEEE International Conference on Systems, Man and Cybernetics, s. 665–662, 9-12 Ekim, Alaska, USA.

Schutte J.F., Groenwold A.A., 2003, Sizing Design of Truss Structures using Particle Swarms, Structural and Multidisciplinary Optimization, vol. 25(4), s. 261--269.

Schutte J.F., Groenwold A.A., 2005, A Study of Global Optimization Using Particle Swarms, Journal of Global Optimization, vol. 31(1), s. 93-108.

Shelokar P.S., Siarry P., Jayaraman V.K., Kulkarni B.D., 2007, Particle swarm and ant colony algorithms hybridized for improved continuous optimization, Applied Mathematics and Computation, vol. 188, s. 129–142.

Shi X., Li Y., Li H., Guan R., Wang L., Liang Y., 2010, An integrated algorithm based on artificial bee colony and particle swarm optimization, Proc. of Sixth International Conference on Natural Computation, s. 2586-2590, 10-12 August, Yantai, China.

Shi Y., Eberhart R. C., 1998, A modified particle swarm optimizer, Proceedings of the IEEE Conference on Evolutionary Computation, s. 69–73, Piscataway, USA.

Shi Y., Eberhart R.C., 1998, A modified particle swarm optimizer, In: Proceedings of the IEEE International Conference on Evolutionary Computation, s. 69-73, Anchorage, AK , USA

Socha K., Dorigo M, 2008, Ant colony optimization for continuous domains, European Journal of Operational Research, vol. 185, s. 1155–1173

Storn R., Price K., 1997, Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization, vol. 11, s. 341–359.

Stützle T., Hoos H. H., 1996, Improving the Ant System: A detailed report on the MAX-MIN Ant System, Technical Report AIDA-96-12, FG Intellektik, FB Informatik, TU Darmstadt, Germany.

Stützle T., López-Ibáñez M., Pellegrini P., Maur M., de Oca M.M., Birattari M., Dorigo M., 2012, Parameter Adaptation in Ant Colony Optimization, Autonomous Search, s. 191-215.

Stützle T., ve Hoos H. H., 2000, MAX-MIN Ant System, Future Generation Computer Systems, vol. 16(8), s. 889–914.

Suganthan P. N., 1999, Particle swarm optimiser with neighbourhood operatör, Proceedings of the IEEE Congress on Evolutionary Computation, s. 1958–1962, Washington, USA.

Szeto W.Y., Wu Y., Ho S.C., 2011, An artificial bee colony algorithm for the capacitated vehicle routing problem, European Journal of Operational Research, vol. 215, 126-135.

Tao G., Michalewicz Z., 1998, Inver-over operator for TSP, Proceedings of the 5th Parallel Problem Solving from Nature, Ed: Baeck T., Eiben A.E., Schoenauer M.,

Schwefel H.-P., Amsterdam, September 27-30, Lecture Notes in Computer Science, s. 803-812.

TDK, 2012, Türk Dil Kurumu Güncel Türkçe Sözlüğü, http://tdk.gov.tr/index.php?option=com_gts&view=gts [Erişim Tarihi: 17.11.2012]

Teodorovic D., Dell'orco M., 2005, Bee colony optimization - a cooperative learning approach to complex transportation problems, Proceedings of the 16th mini-EURO Conference on Advanced OR and AI Methods in Transportation, s. 51–60.

Thangaraj R., Pant M., Abraham A., Bouvry P., 2011, Particle swarm optimization: Hybridization perspectives and experimental illustrations, Applied Mathematics and Computation, vol. 217(12), s. 5208-5226

Theraulaz G., Deneubourg J.-L., Swarm Intelligence in Social Insects and The Emergence of Cultural Swarm Patterns, Santa Fe Institute Working Papers, No: 92-09-046.

Toksarı M. D., 2006, Ant colony optimization for finding the global minimum, Applied Mathematics and Computation, vol. 176(1), s. 308–316

van den Bergh F., Engelbrecht A.P., 2004, A Cooperative Approach to Particle Swarm Optimization, IEEE Transactions on Evolutionary Computation, vol. 8(3), s. 225-239.

Wang J., Li T., Ren R. 2010, A real time IDSs based on artificial bee colony-support vector machine algorithm, Proceedings of Third International Workshop on Advanced Computational Intelligence, s. 91–96, 27-27 August, Suzhou, China

Wu X.J., Hao D., Xu C., 2011d, An improved method of artificial bee colony algorithm. Journal of Applied Mechanics and Materials, vol. 101-102, s. 315–319.

Xie X., Zhang W., Yang Z., 2002a, Adaptive Particle Swarm Optimization on Individual Level, Proceedings of the Sixth International Conference on Signal Processing, vol. 2, s. 1215-1218.

Xie X., Zhang W., Yang Z., 2002b, A Dissipative Particle Swarm Optimization, Proceedings of the IEEE Congress on Evolutionary Computation, vol. 2, s. 1456-1461.

Xinchao Z., 2010, A perturbed particle swarm algorithm for numerical optimization, Applied Soft Computing, vol. 10(1), s. 119-124.

Yang X. S., 2009, Firefly Algorithms for Multimodal Optimization, Proceedings of the Stochastic Algorithms: Foundations and Applications, Lecture Notes in Computing Sciences, vol. 5792, s. 178-178.

Yang X.S., 2005, Engineering optimizations via nature-inspired virtual bee algorithms, Ed.: Mira J., lvarez J.R., Artificial intelligence and knowledge engineering applications: a bioinspired approach, Springer, Lecture Notes in Computer Ccience, vol. 3562, s. 317–323.

Yang X.S., Deb S, 2009, Cuckoo search via Lévy flights, Proceedings of the World Congress on Nature & Biologically Inspired Computing, Coimbatore, India, pp. 210‑214.

Yörükoğlu A., 2004, Zekâ Nedir?, Çocuk Ruh Sağlığı, Özgür Yayınları, İstanbul.

Yuan X., Nie H., Su A., Wang L., Yuan Y., 2009, An improved binary particle swarm optimization for unit commitment problem, Expert System with Applications, vol. 39, s. 8049-8055.

Zhan Z.-H., Zhang J., Li Y., Shi Y.-H., 2011, Orthogonal Learning Particle Swarm Optimization, IEEE Transactions on Evolutionary Computation, vol. 15(6), s. 832-847

Zheng Y.-L., Ma L.-H., Zhang L.-Y., Qian J.-X., 2003, On the convergence analysis and parameter selection in particle swarm optimization, Proceedings of the IEEE International Conference on Machine Learning and Cybernetics, s. 1802–1807, Piscataway

Zhu G., Kwong S., Gbest-guided artificial bee colony algorithm for numerical function algorithm, Applied Mathematics and Computation, Vol. 217, s. 3166-3173.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı	**:**	Mustafa Servet Kıran
Uyruğu	**:**	T.C.
Doğum Yeri ve Tarihi	**:**	Osmaniye, 18.07.1983
Telefon	**:**	+90 332 223 19 92
Faks	**:**	+90 332 241 06 35
E-mail	**:**	mskiran@selcuk.edu.tr
Web	**:**	www.rdbilisim.com/mskiran

EĞİTİM

Derece		**Adı, İlçe, İl**	**Bitirme Yılı**
Lise	:	İmam-Hatip Lisesi, Merkez, Erzurum	2000
Ön Lisans		Çukurova Üniversitesi Osmaniye Meslek Yüksekokulu Bilgisayar Programcılığı Programı, Merkez, Osmaniye	2003
Lisans	:	Selçuk Üniversitesi Mühendislik-Mimarlık Fakültesi Bilgisayar Mühendisliği Bölümü, Selçuklu, Konya	2007
Yüksek Lisans	:	Selçuk Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı, Selçuklu, Konya	2010
Doktora	:	Selçuk Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı, Selçuklu, Konya	2014

İŞ DENEYİMLERİ

Yıl	**Kurum**	**Görevi**
2005-2007	Selçuk Üniversitesi Bilgi-İşlem Daire Başkanlığı	Bilgisayar İşletmeni
2007-2010	Selçuk Üniversitesi Güneysınır Meslek Yüksekokulu	Öğretim Görevlisi
2010-…	Selçuk Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı	Araştırma Görevlisi

UZMANLIK ALANI

Sürü Zekâsı, Yapay Zekâ Optimizasyon Yöntemleri, Parçacık Sürü Optimizasyonu, Yapay Arı Kolonisi Optimizasyon Algoritması, Karınca Kolonisi Optimizasyonu, Tahmin Problemlerinin Modellenmesi ve Çözümü, Özellik seçme ve sınıflandırma

YABANCI DİLLER

İngilizce (Orta),
Arapça (Başlangıç)

YAYINLAR

ULUSAL/ULUSLARASI HAKEMLİ DERGİ ÇALIŞMALARI

Kıran M.S., Özceylan E., Gündüz M., Paksoy T., 2012, A Novel Hybrid Approach based on Particle Swarm Optimization and Ant Colony Algorithm to Forecast Energy Demand of Turkey, Energy Conversion and Management, 53/1, 75-83. (SCI-E)

***Kıran M.S.,** İşcan H., Gündüz M., 2012, The Analysis of Discrete Artificial Bee Colony Algorithm with Neighborhood Operator on Traveling Salesman Problem, Neural Computing & Applications, doi:10.1007/s00521-011-0794-0.(SCI-E)

***Kıran M.S.,** Gündüz M., 2012, A Novel Artificial Bee Colony-based Algorithm for Solving the Numerical Optimization Problems, International Journal of Innovative Computing-Information&Control, 8/9, 6107-6122.(SCI-E)

***Kıran M.S.,** Özceylan E., Gündüz M., Paksoy T., 2012, Swarm Intelligence Approaches to Estimate Electricity Energy Demand of Turkey, Knowledge-Based Systems, 36, 93-103.(SCI)

***Kıran M.S.,** Baykan Ö.K.., Gündüz M., 2012, .A Novel Hybrid Algorithm based on Particle Swarm and Ant Colony Optimization for Finding the Global Minimum, Applied Mathematics and Computation, 219/4, 1515-1521. (SCI-E)

***Kıran M.S.,** Gündüz M., 2012, XOR-based Artificial Bee Colony Algorithm for Binary Optimization, Turkish Journal of Electrical Engineering & Computer Sciences, 21, 2307-2328. (SCI-E)

***Kıran M.S.,** Gündüz M., 2012, A Recombination-based Hybridization of Particle Swarm Optimization and Artificial Bee Colony Algorithm for Continuous Optimization Problems, Applied Soft Computing, 13/4, 2188-2203. (SCI-E)

*Gündüz M., **Kıran M.S.,** Özceylan E., 2012, A hierarchic approach based on swarm intelligence to solve traveling salesman problem, Turkish Journal of Electrical Engineering & Computer Sciences, doi:10.3906/elk-1210-147. (SCI-E)

Babaoglu İ., **Kıran M.S.,** Ülker E., Gündüz M., 2013, Diagnosis of Coronary Artery Disease using Artificial Bee Colony and K-nearest Neighbor Algorithms, International Journal of Computer and Communication Engineering, 2/1, 56-59.

Kıran M.S., Gündüz M., 2014, The analysis of peculiar control parameters artificial bee colony algorithm on the numerical optimization problems, Journal of Computer and Communications, 2/4, 127-136.

Fındık O., **Kıran M.S.**, Babaoglu İ., 2014, Investigation effects of selection mechanims for gravitational search algorithm, Journal of Computer and Communications, 2/4, 117-126.

Kıran M.S., Babalik A., 2014, Improved artificial bee colony algorithm for continuous optimization problems, Journal of Computer and Communications, 2/4, 108-116.

****Kıran M.S.,** Gündüz M., 2013, A Bee Colony Optimization-based Approach for Binary Optimization, International Journal of Intelligent Systems and Applications in Engineering, vol. 1(3), pp.47-51.

****Kıran M.S.,** Gündüz M., 2012, Arı Kolonisi Optimizasyon Algoritması Kullanarak Şoför-Hat-Zaman Çizelgeleme, Selçuk Teknik-Online Dergi, Cilt:11, Sayı: 2-2012, sayfa: 71-81.

ULUSAL/ULUSLARARASI SEMPOZYUM BİLDİRİLERİ

Iscan H., **Kıran M.S.**, Gündüz M., 2011, Supply Chain Optimization using Ant System, The International Conference on Computing and Information Technology, pp.1-5, 11-12 May, Bangkok, Thailand.

Kıran M.S., Turanoğlu E., Özceylan E., 2011, Artificial Bee Colony Approach to Estimate CO2 Emission of Turkey, Proceedings of the 41st International Conference on Computers & Industrial Engineering, pp. 536-541, October 23-26, Los Angeles-USA.

Özceylan E., **Kıran M.S.**, Atasagun Y., 2011, A New Hybrid Heuristic Approach for Solving Green Traveling Salesman Problem, Proceedings of the 41st International Conference on Computers & Industrial Engineering, pp. 720-725, October 23-26, Los Angeles-USA.

Turanoğlu E., Özceylan E., **Kıran M.S.**, 2011, Particle Swarm Optimization and Artificial Bee Colony Approaches to Optimize of Single Input-Output Fuzzy Membership Functions, Proceedings of the 41st International Conference on

Computers & Industrial Engineering, pp. 542-547, October 23-26, Los Angeles, USA.

Babaoglu İ., **Kıran M.S.**, Ülker E., Gündüz M., 2012, Diagnosis of Coronary Artery Disease using Artificial Bee Colony and K-nearest Neighbor Algorithms, Presented in International Conference on Software and Computer Engineering, October 6-7, Singapore, "Published in International Journal of Computer and Communication Engineering, vol. 2(1), pp.56-59, 2013".

Babaoglu İ., **Kıran M.S.**, An artificial bee colony algorithm-based feature selection model, 6th International Conference on Advanced Computer Systems and Networks: Design and Application, pp. 239-242, 16-18 September, Lviv, Ukraine.

Kıran M.S., Özceylan E., Paksoy T., 2012, Artificial Bee Colony Algorithm for Solving Uncapacitated Facility Location Problems, 25th European Conference on Operational Research, pp. 165 (abstract), July 8-11, Vilnius, Lithuania.

Kıran M.S., Gündüz M., Haklı H., Sahman M.A., 2013, A hybridization of artificial bee colony and gravitational search algorithms for nonlinear global optimization, Euro|Informs 26th European Conference on Operational Research, pp. 222, 1-4 July, Rome, Italy.

Dundar A.O., Sahman M.A., Tekin M., **Kıran M.S.**, An application showing the impact of the travelling salesman problem based on logistic costs through heuristic methods, Euro|Informs 26th European Conference on Operational Research, pp. 104, 1-4 July, Rome, Italy

Sahman M.A., Dundar A.O., **Kıran M.S.**, Altun A.A., 2013, Comparison of linear programming and heruistic hybrid methods in preparing flour blend, Euro|Informs 26th European Conference on Operational Research, pp. 229, 1-4 July, Rome, Italy.

Kıran M.S., Gündüz M., 2014, The analysis of peculiar control parameters artificial bee colony algorithm on the numerical optimization problems, Presented in 2th Conference on Artificial Intelligence and Data Mining, March 10-12, Suzhou, China, "Published in Journal of Computer and Communications, 2/4, 127-136".

Fındık O., **Kiran M.S.**, Babaoglu İ., 2014, Investigation effects of selection mechanims for gravitational search algorithm, Presented in 2th Conference on Artificial Intelligence and Data Mining, Suzhou, China, "Published in Journal of Computer and Communications, 2/4, 117-126".

Kiran M.S., Babalik A., 2014, Improved artificial bee colony algorithm for continuous optimization problems, Presented in 2th Conference on Artificial Intelligence and Data Mining, Suzhou, China, "Published in Journal of Computer and Communications, 2/4, 108-116".

****Kıran M.S.,** Şahman M.A., Gündüz M., 2009, Arı Kolonisi Optimizasyon Algoritması Kullanarak En Kısa Yol Bulma, IV. İletişim Teknolojileri Ulusal Sempozyumu Bildiriler Kitabı, sayfa 13-16, 15-16 Ekim, Adana.

*: Doktora tezinden üretilmiş yayınlar.

**: Yüksek Lisans tezinden üretilmiş yayınlar.

PROJE DENEYİMLERI

1. Tubitak 1001 Araştırma Destekleme Programı

Proje No: 111M040

Proje Adı: Demontaj Hattı Planlaması Problemi İçeren Kapalı Çevrim Tedarik Zincirlerinin Bulanık Ortamda Modellenmesi ve Optimizasyonu

Yürütücü: Doç.Dr. Turan Paksoy

Görev: Bursiyer, Uygulama Geliştirici

Süre: 01.01.2012-01.08.2012

2. Avrupa Birliği 7. Çerçeve Programı

Proje No: 286161

Proje Adı: Development and demostration of a dynamic, web-based, renewable energy rating platform (Codename: EAGLE)

Proje Yürütücüsü: Chartered Institute of Plumbing & Heating Engineering, Mr. Kevin Wellman

Proje Türkiye Ortakları: Solimpeks, Selcuk Üniversitesi (Prof.Dr.Ahmet Arslan)

Görev: Uygulama Geliştirici

Süre: 01.08.2012-...

3. T.C. Bilim, Sanayi ve Teknoloji Bakalığı Teknogirişim 2013

Proje No: 118.TGSD.2013

Proje Adı: Yazılım tabanlı radyo ile konumsal yoğunluk tespiti

Proje Yürütücüsü ve Girişimcisi: Mustafa Servet Kıran

4. T.C. Bilim, Sanayi ve Teknoloji Bakalığı Teknogirişim 2013

Proje No: 252.TGSD.2013

Proje Adı: Lineer ve zeki teknikler ile buğday paçalı hazırlama yazılımı

Proje Yürütücüsü ve Girişimcisi: Mehmet Akif Şahman

Görevi: Yazılım Mühendisi

5. Tübitak 1507 Sanayi Ar-ge Destekleme Programı

Proje Adı: Yeraltı suları ölçümü için debimetre geliştirilmesi (AquaDb)

Yürütücü: Enelsis Endüstriyel Elektronik Ltd.Şti.

Görev: Yazılım Mühendisi / Teknik Danışman

Hakemlikler

Computer Sciences Student Workshop, Program Committee Member, 2012.

Expert Systems with Applications

Knowledge-Based Systems

IEEE Sensors Journal

International Journal of Production Economics

Applied Soft Computing (Outstanding Reviewer -2013/2014)

Turkish Journal of Electrical Engineering & Computer Sciences

Journal of Computer Science and Computational Mathematics

International Journal of Computer Systems Science and Engineering

International Journal of Production Research

Neural Computing and Applications

International Journal of Physical Sciences