T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GÜNCEL EN İYİLEME ALGORİTMALARININ PARALEL VE BİRLİKTE UYGULAMALARI VE PERFORMANS ANALİZLERİ

DOKTORA TEZİ

Hasan MAKAS

Enstitü Anabilim Dalı	**:**	**BİLGİSAYAR VE BİLİŞİM MÜHENDİSLİĞİ**
Tez Danışmanı	**:**	**Prof. Dr. Nejat YUMUŞAK**

Ocak 2015

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GÜNCEL EN İYİLEME ALGORİTMALARININ PARALEL VE BİRLİKTE UYGULAMALARI VE PERFORMANS ANALİZLERİ

DOKTORA TEZİ

Hasan MAKAS

Enstitü Anabilim Dalı : BİLGİSAYAR VE BİLİŞİM MÜHENDİSLİĞİ

Bu tez 12 / 01 / 2015 tarihinde aşağıdaki jüri tarafından Oybirliği ile kabul edilmiştir.

Prof. Dr.
Nejat YUMUŞAK
Jüri Başkanı

Prof. Dr.
İbrahim ÇİL
Üye

Doç. Dr.
Celal ÇEKEN
Üye

Doç. Dr.
Ayhan İSTANBULLU
Üye

Doç. Dr.
Rüştü GÜNTÜRKÜN
Üye

TEŞEKKÜR

Tez çalışmamın bütün aşamalarında destek ve yardımları ile sürekli yanımda olan, değerli görüş ve katkılarıyla beni yönlendiren, kıymetli tecrübelerinden faydalandığım danışman hocam Prof. Dr. Nejat Yumuşak'a ve tez izleme komitemde yer alan değerli hocalarım Prof. Dr. İbrahim Çil'e ve Doç. Dr. Celal Çeken'e teşekkürü bir borç bilirim.

Doktora çalışmalarım süresince gösterdiği anlayış ve verdiği destek ile beni her zaman teşvik eden eşim Funda'ya ve güler yüzü ile sağladığı motivasyon için kızım Gökçen'e teşekkür ederim.

İÇİNDEKİLER

SİMGELER VE KISALTMALAR LİSTESİ

ABC : Yapay arı kolonisi (Artificial Bee Colony)
ACO : Karınca koloni en iyilemesi (Ant Colony Optimization)
AIS : Yapay bağışıklık sistemi (Artificial Immune System)
ARMA : Oto-regresif hareketli ortalama (Auto Regressive Moving Average)
ARX : Oto-regresif dışsal girişli (Auto-Regressive with eXogenous inputs)
BA : Yarasa algoritması (Bat Algorithm)
BBO : Biyocoğrafya tabanlı en iyileme (Biogeography Based Optimization)
BCO : Arı koloni en iyilemesi (Bee Colony Optimization)
BFO : Bakteriyel beslenme en iyilemesi (Bacterial Foraging Optimization)
BH : Arı kovanı (Bee Hive)
BJ : Box-Jenkins
CA : Kültürel algoritmalar (Cultural Algorithms)
CART : Sınıflandırma ve regresyon ağacı (Classification And Regression Tree)
CHAID : CHi-kare otomatik etkileşim algılama (CHi-squared Automatic Interaction Detection)
CoEA : Birlikte evrimleşen algoritmalar (Co-evolutionary Algorithms)
CS : Guguk kuşu araması (Cuckoo Search)
CSO : Kedi sürüsü en iyilemesi (Cat Swarm Optimization)
DABC : Ayrık yapay arı kolonisi (Discrete Artificial Bee Colony)
DBD : Ayrık arı dansı (Discrete Bee Dance)
DC : Dendritik hücre (Dendritic Cell)

DE	: Diferansiyel gelişim (Differential Evolution)
DGA	: Ayrık genetik algoritmadır (Discrete Genetic Algorithm)
DMBO	: Ayrık göçmen kuşlar en iyileme (Discrete Migrating Birds Optimization)
EBP	: Hata geri yayılım (Error Back Propagation)
EP	: Gelişimsel programlama (Evolutionary Programming)
ES	: Gelişim stratejisi (Evolution Strategy)
FA	: Ateşböceği algoritması (Firefly Algorithm)
FSA	: Balık sürüsü algoritması (Fish Swarm Algorithm)
GA	: Genetik algoritma
GLS	: Güdümlü yerel arama (Guided Local Search)
GP	: Genetik programlama
GRASP	: Açgözlü rasgele adaptif arama prosedürü (Greedy Randomized Adaptive Search Procedure)
GS	: Yerçekimi araması (Gravitational Search)
HS	: Harmoni araması (Harmony Search)
ID3	: Iterative Dichotomiser 3
ILS	: Tekrarlı yerel arama (Iterated Local Search)
IWD	: Akıllı su damlası (Intelligent Water Drops)
LM	: Levenberg–Marquardt
MARS	: Çok değişkenli uyumlu regresyon uzanımları (Multivariate Adaptive Regression Splines)
MBABC-CO	: Göçmen kuşlar ve yapay arı koloni kooperatif en iyilemesi (Migrating Birds and Artificial Bee Colony Cooperative Optimization)
MBO	: Göçmen kuşlar en iyilemesi (Migrating Birds Optimization)
MBOGE	: Küresel araştırma ilaveli göçmen kuşlar en iyilemesi (Migrating Birds Optimization with Global Exploration)
MFMBO	: Çok sürülü göçmen göçmen kuşlar en iyilemesi (Multi-Flock Migrating Birds Optimization)
MS	: Maymun araması (Monkey Search)
MSE	: Ortalama karesel hata (Mean Squared Error)
OE	: Çıkış hatası (Output Error)

PORLA	: Saf derece tekrarlı merdiven algoritma (Pure Order Recursive Ladder Algorithm)
PSO	: Parçacık sürü en iyilemesi (Particle Swarm Optimization)
QBE	: Kraliçe arı gelişim (Queen-bee Evolution)
SA	: Isıl işlem (Simulated Annealing)
SABCMBO	: Sıralı yapay arı koloni ve göçmen kuşlar en iyilemesi (Sequential Artificial Bee Colony and Migrating Birds Optimization)
SCG	: Ölçeklendirilmiş gradyan eşlenik (Scaled Conjugate Gradient)
SLIQ	: Supervised Learning In Quest
SPRINT	: Scalable PaRallelizable INduction of decision Trees
TS	: Tabu araştırma (Tabu Search)
VBA	: Sanal arı algoritması (Virtual Bee Algorithm)
VNMBO	: Değişken komşuluklu göçmen kuşlar en iyilemesi (Varying Neighbourhood Migrating Birds Optimization)
VNS	: Değişken komşuluk arama (Variable Neighbourhood Search)
YSA	: Yapay sinir ağı

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Anahtar kelimeler: Göçmen Kuşlar En İyileme Algoritması, Yapay Arı Koloni Algoritması, Meta-sezgisel Sistem Kimliklendirme, Meta-sezgisel Yapay Sinir Ağı Eğitimi, Meta-sezgisel Karar Ağacı İnşası, Melez Meta-sezgisel Algoritmalar

En iyileme yöntemleri yapılan işin en iyi yapılmasını sağlamak için kullanılırlar. Bu tekniklerin kullanılmasındaki temel hedef her zaman için en iyi çözümleri yakalayabilmektir. Uygunluk veya hata değeri tanımlanabilen her sistemin en iyi çözümünün elde edilmesinde en iyileme algoritmaları kullanılabilir. Sadece ait oldukları problemlere özgü olmaları ve yüksek hesaplama maliyeti içermeleri gibi sebepler nedeniyle mevcut geleneksel en iyileme algoritmalarının kullanımı çok sayıda parametre içeren gerçek dünya problemlerinin çözümünde bazen yeterli olmayabilir. Bu gibi durumlarda daha az işlem ile daha kısa sürede en iyi çözüme yakınsayabilen meta-sezgisel yöntemlerin kullanımı daha makul çözümler olarak karşımıza çıkmaktadır. Son 20 yıl içerisinde doğadan ilham alınarak çok sayıda meta-sezgisel en iyileme algoritması geliştirilmiştir. Buna paralel olarak bazı araştırmacılar mevcut algoritmalar üzerinde birtakım iyileştirmeler yapmışlar, bazıları da birden fazla algoritmayı bir arada kullanarak performansı daha yüksek melez yöntemler elde etmişler ve daha sonra bu yöntemleri kullanarak gerçek dünya problemlerine en iyi çözümler üretmişlerdir.

Bu tez çalışmasında sistem kimliklendirme süreci, yapay sinir ağı eğitimi, sempozyum katılımcı listelerinin düzenlenmesi, slab kesme uzunluklarının planlanması gibi gerçek dünyaya ait problemlere birer en iyileme problemi olarak yaklaşılmış, seçilen güncel ve yaygın meta-sezgisel algoritmalar kullanılarak geleneksel yöntemlerin çözümleri ile rekabet edebilen çözümler üretilmiştir. Ayrıca, karar ağacı tasarım süreci hem kombinatoryal hem de nümerik en iyilemeleri içeren bir problem olarak ele alınmış, olası karar ağacı tasarımları arasında sistematik arama yapan yeni bir yöntem ile karar ağacı tasarımı gerçekleştirilmiştir. Önerilen yöntemle elde edilen test sonuçlarının aynı veri setinin kullanıldığı daha önceki karar ağacı çalışmaları ile elde edilen sonuçlardan daha iyi olduğu görülmüştür. Son olarak, yapay arı koloni ve göçmen kuşlar en iyileme algoritmaları kullanılarak yeni modifiye, melez ve paralel çalışma sistematikleri önerilmiştir. Önerilen yöntemlerin performans testlerinden elde edilen sonuçlar, onların daha iyi keşif ve yakınsama yeteneklerine sahip olduklarını ortaya koymuştur.

PARALLEL AND COLLABORATIVE APPLICATIONS OF THE RECENT OPTIMIZATION ALGORITHMS AND THEIR PERFORMANCE ANALYSES

SUMMARY

Keywords: Migrating Birds Optimization Algorithm, Artificial Bee Colony Algorithm, Metaheuristic System Identification, Metaheuristic Neural Network Training, Metaheuristic Decision Tree Construction, Hybrid Metaheuristics

Optimization methods are employed in order to make a job in an optimal way. The main aim of their usage is to get an optimal solution in every execution. Optimization algorithms can be applied to find optimal solutions for the systems whose fitness or error calculations can be defined. Sometimes, existing conventional optimization algorithms may be insufficient for the real world problems having many parameters because of the reason that they are problem specific and have higher calculation costs. Since metaheuristic algorithms can find near optimal solutions with less calculations requiring lower time, their usages seem more feasible for these cases. Within the past 20 years, so many metaheuristic algorithms which are inspired by the nature have been developed by researchers. In parallel to these studies, while some of the researchers were working on some enhancements for existing algorithms, some of them were working on their hybrid forms. Then, they tried to find more optimal solutions for real world problems by using these new enhanced and hybrid algorithms.

In this dissertation study, some real world problems such as system identification process, artificial neural network training, preparation of symposium attendee lists, scheduling slab cutting lengths etc. are thought to be optimization problems. Some competitive solutions with respect to solutions of the conventional methods are generated to these real world problems by using some recent and common metaheuristic algorithms. In addition, thinking the decision tree construction process as a problem including both numerical and combinatorial optimizations, a novel decision tree construction method which makes a systematic search among possible decision tree designs is proposed to get optimal decision tree. It is seen that the results obtained by proposed method are better than those of previous studies using same data set. Finally, some modified, hybrid and parallel running strategies using artificial bee colony and migrating birds optimization algorithms are proposed. It is observed from the performance test results that proposed strategies have better exploration and exploitation capabilities.

BÖLÜM 1. GİRİŞ

Son 30 yıl içerisinde bilgisayar teknolojisinde yaşanan büyük gelişime paralel olarak en iyileme (optimizasyon - optimization) uygulamaları fen, sosyal, eğitim ve sağlık bilimleri gibi farklı disiplinlere uygulanmış, bu farklı alanlarda çalışan araştırmacılar çalışmalarının büyük bir kısmını bilgisayarlar aracılığı ile yapmaya ve karşılaştıkları problemlere sayısal modeller kullanarak en iyi (optimum) çözümler aramaya başlamışlardır.

Optimum Latince bir sözcük olup nihai ideal anlamına gelir. En iyileme, mevcut koşullar altında bir problemin olası alternatif çözümleri içinden en iyi olanını seçme işlemidir. Kısaca en iyi olanı arama çalışmasıdır. En iyileme problemi ise belirli sınırlamaları sağlayacak şekilde, bilinmeyen parametre değerlerinin bulunmaya çalışıldığı herhangi bir problem olarak tanımlanabilir [1]. En iyileme problemleri birden fazla farklı çözüme sahip olabilen ve çözüm kalitelerinin açık ve net ifadelerle yazılabildiği problemlerdir. Yani farklı aday çözümlerin anlamlı ve tutarlı bir şekilde birbirleri ile mukayese edilebilmeleri mümkün ise burada bir en iyileme problemi var demektir [2].

Bir işin yapılması, o işin en iyi şekilde yapıldığı anlamına gelmez. En iyileme metotları yapılan işin en iyi şekilde yapılması için kullanılırlar. Bu tekniklerin kullanılmasındaki ana hedef, her zaman için en iyi çözümleri yakalayabilmektir. Bir makine parçasının imalatından bir web sitesinin tasarımına varıncaya kadar her alanda bu tekniklerin uygulandığı bir en iyileme işlemine ihtiyaç vardır. Kısacası, en iyileme her alanda kullanılmaktadır.

Çok sayıda parametre içeren gerçek dünya problemlerinin çözümü en iyileme bağlamında önemli ve kritik bir konudur. Karmaşık bir gerçek dünya probleminin bilimsel tanımlamaları ve sınırlamaları matematiksel ifadelerle verilmiş olsa dahi

problemin küresel en iyi noktasının uzmanlar tarafından bulunması oldukça zor, hatta bazen imkânsız olabilmektedir. Bazı durumlarda problemin karmaşıklığına ilave olarak bir ya da daha fazla parametrenin sabitlenmesi zorunluluğu eklenebilir. Bu gibi durumlara üretim sistemlerinde sıkça rastlanmaktadır. Örneğin; bazen üreticiler stoklarda bulunmaması veya yetersiz olması gibi nedenlerden dolayı bir sarf malzemenin tüketiminin azaltılmasını isteyebilirler. Bu gibi durumlarda ise kullanılan araştırma tekniği problemin gerçek en iyisinden ziyade problem kısıtları ile sınırlandırılan araştırma uzayındaki en iyiyi bulunmaya çalışır. Yani tanımlanan araştırma uzayında, küresel en iyiye mümkün olduğunca yakın alt en iyi çözümler için araştırma yapılır.

En iyileme problemlerinin çözümünde kullanılan çeşitli en iyileme algoritmaları vardır. Geleneksel çözüm yöntemleri olan kesin algoritmalar, hesaplama maliyetleri yüksek fakat en iyi sonucu veren yöntemlerdir. Problemin boyutundaki artışa paralel olarak bu yöntemlerin uygulanması üstel bir şekilde zorlaşır [3]. Bu sebeple, özellikle işlem zamanının önemli olduğu, çok hedefli ve yüksek boyutlu en iyileme problemlerinde daha az işlem ile daha kısa sürede en iyi çözümü olmasa da ona yakın olan sonuçları elde eden meta-sezgisel yöntemlerin kullanımı daha makul bir çözüm olarak karşımıza çıkar.

Araştırmacılar tarafından zor en iyileme problemlerini çözmek için çeşitli doğal veya insan yapımı süreçleri taklit eden pek çok en iyileme metodu önerilmiştir. Bunlar genellikle komşuluk arama teknikleri olup, gelişimsel algoritmalar içerisinde önemli bir yer teşkil ederler [4]. Bu algoritmalar daha iyi sonuçlar elde edebilmek için mevcut çözümlerin komşuluklarını araştırırlar. Dolayısıyla komşuluk yapısının tanımı ve komşu üretme yönteminin seçimi bu komşuluk araştırma algoritmalarının tasarımında oldukça önemlidir [5].

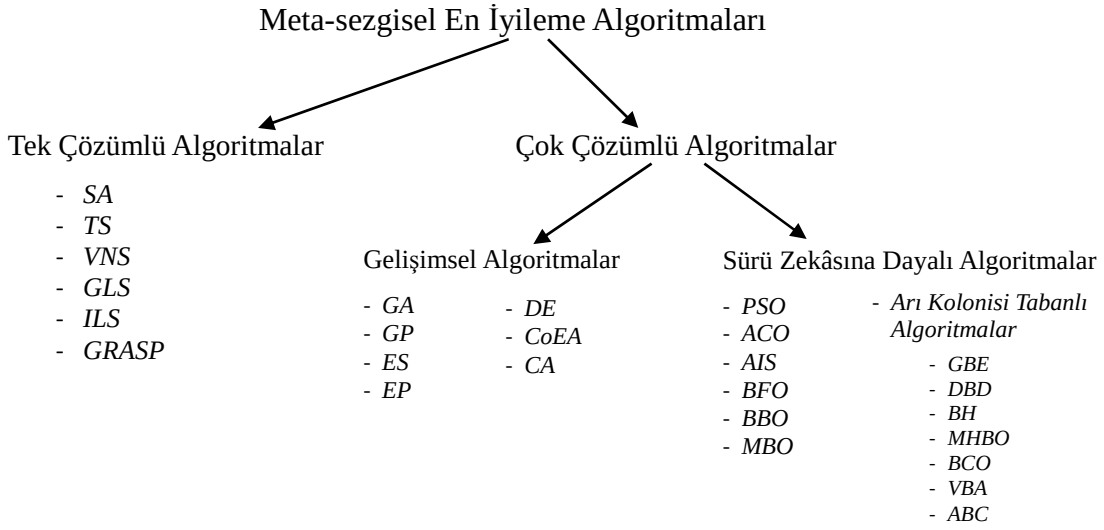
Meta-sezgisel algoritmalar geniş bir en iyileme problemi yelpazesine ilgili problemler ile alakalı detaylı bilgilere ihtiyaç duymaksızın çözümler üretebilmektedirler. Yunanca *meta* ön eki, probleme özel sezgisel algoritmaların aksine, bu algoritmaların daha yüksek seviyeli ve problemden bağımsız sezgisel algoritmalar olduğunu vurgulamaktadır. Özellikle endüstri, finans, mühendislik,

üretim planlama, üretim yönetimi vb. gibi alanlarda karşılaşılan, kompleks ve tatmin edici özel bir çözüm yöntemi bulunmayan problemlerin çözümünde tercih edilirler [6].

Meta-sezgisel algoritmaların birçoğu doğadan ilham almaktadır. Bu, yaratılmışların en akıllısı olmasına rağmen, insanların doğanın mükemmelliğinden hala alacağı çok ders olduğunun bir göstergesidir [4]. Meta-sezgisel algoritmalar çalışırken, mobil ajanlar lokal etkileşimlere girerler ve küresel yakınsamayı gerçekleştirebilmek için doğru koşullar altında çabucak kendi kendine organize olurlar. Ajanlar, bir taraftan araştırma uzayında tipik yerel araştırmalarını gerçekleştirirlerken, diğer taraftan da rasgele değişimleri kullanarak çözümlerin küresel çeşitliliğini artırmayı denerler. Dolayısıyla yoğun yerel arama ve küresel araştırma işlemleri arasında kurulmuş ince bir denge vardır [7].

Literatürde şu ana kadar tanıtılmış olan meta-sezgisel algoritmalar içerisinden popüler ve yaygın olarak kullanılanların bazıları: Genetik Algoritma (Genetic Algorithm – GA) [8], Isıl İşlem (Simulated Annealing – SA) algoritması [9], Tabu Araştırma (Tabu Search – TS) algoritması [10-12], Karınca Koloni En İyileme (Ant Colony Optimization – ACO) algoritması [13], Parçacık Sürü En İyileme (Particle Swarm Optimization – PSO) algoritması [14], Diferansiyel Gelişim (Differential Evolution – DE) algoritması [15], Harmoni Arama (Harmony Search – HS) algoritması [16], Maymun Arama (Monkey Search – MS) algoritması [17], Yapay Arı Koloni (Artificial Bee Colony – ABC) algoritması [18-19], Ateşböceği Algoritması (Firefly Algorithm – FA) [20], Akıllı Su Damlası (Intelligent Water Drops – IWD) algoritması [21], Guguk Kuşu Arama (Cuckoo Search – CS) algoritması [22-23], Yarasa Algoritması (Bat Algorithm – BA) [24-25], Balık Sürüsü Algoritması (Fish Swarm Algorithm – FSA) [26], Yapay Bağışıklık Sistemi (Artificial Immune System – AIS) algoritması [27-28], Bakteriyel Beslenme En İyileme (Bacterial Foraging Optimization – BFO) algoritması [29-30], Dendritik Hücre (Dendritic Cell – DC) algoritması [31-32], Kedi Sürüsü En İyileme (Cat Swarm Optimization – CSO) algoritması [33-34], Yerçekimi Arama (Gravitational Search - GS) algoritması [35-36], Göçmen Kuşlar En İyileme (Migrating Birds Optimization – MBO) algoritması [4] şeklinde sıralanabilir.

Hemen hemen bütün meta-sezgisel algoritmalar şu özelliklere sahiptirler: fizik, biyoloji ve etoloji gibi bazı bilimlerin ilkelerini esas alarak doğadan ilham alırlar; rasgele değişkenler içeren stokastik bileşenlerden faydalanırlar; amaç fonksiyonunun gradyan veya hessian matrisini kullanmazlar; eldeki probleme göre ayarlanması gereken bazı parametreleri vardır [6].



Şekil 1.1. Meta-sezgisel en iyileme algoritmalarının sınıflandırılması

Meta-sezgisel algoritmaları Şekil 1.1'de verildiği gibi kategorize etmek mümkündür. Tek çözümlü algoritmalar yörünge metotları olarak da bilinirler. Popülasyon tabanlı algoritmaların aksine en iyilemeye tek bir çözüm ile başlayıp problem uzayında bir yörünge tanımlayarak başlangıç çözümünü küresel minimuma doğru yakınsarlar. Bu gruptaki algoritmaların en yaygın olanlarından bazılarının ilham kaynakları ve yapılan literatür taramaları aşağıda listelenmiştir.

- İlk olarak Kirkpatrick ve arkadaşları tarafından önerilen daha sonra ise Cerny tarafından önerilen [9,37] SA algoritması, metalürjistlerin minimum enerjili ve iyi yapılanmış katı çelik elde ederken kullandıkları ısıl işlem tekniğinden ilham alır. Algoritmanın Laarhoven ve Aarts [38], Collins ve arkadaşları [39], Koulamas ve arkadaşları [40], Fleischer [41] ve Tan [42] tarafından hazırlanan detaylı literatür taramaları mevcuttur.

- Glover tarafından önerilen TS algoritması [10] bir yerel arama tekniği olarak tasarlanmıştır. Hem yerel minimumlardan kaçmak hem de bir araştırma stratejisi ortaya koymak için araştırma geçmişini kullanan algoritma bu yönüyle insan belleğinden ilham almaktadır. Algoritmanın kavramsal detayları Glover ve Laguna [43] tarafından, detaylı literatür taramaları ise Gendreau [44] ve Gendreau ve Potvin [45] tarafından sunulmuştur. TS algoritması başlangıçta kombinatoryal en iyileme problemleri için tasarlanmış olmasına rağmen daha sonra algoritmanın sürekli en iyileme problemleri için de adaptasyonu yapılmıştır [46].

- Hansen ve Mladenovic tarafından önerilen değişken komşuluk araştırma (Variable Neighbourhood Search – VNS) algoritması [47-49] verilen bir çözümün dinamik olarak değişen komşuluklarının araştırılması mantığı üzerine inşa edilmiştir. Algoritmanın güncel literatür taraması yine Hansen ve Mladenovic tarafından yapılmıştır [50-51].

- Voudouris tarafından önerilen güdümlü yerel arama (Guided Local Search – GLS) algoritması [52-54] TS algoritmasında olduğu gibi bir bellek kullanır ve bulduğu yerel en iyiye bağlı olarak yerel arama yolu ile optimize ettiği amaç fonksiyonunu dinamik olarak değiştirir. Algoritmanın güncel literatür taraması yine Voudouris tarafından yapılmıştır [55-56].

- Stützle tarafından doktora tezi olarak verilen tekrarlı yerel arama (Iterated Local Search – ILS) algoritmasında [57] rasgele üretilen başlangıç çözümlerine tekrarlayan yerel arama işlemleri yapmak yerine mevcut iterasyonda bulunan yerel en iyi kullanılarak bir sonraki iterasyon için başlangıç değeri üretilir. Lourenço ve arkadaşları tarafından algoritmanın güncel bir literatür taraması yapılmıştır [58].

- Feo ve Resende tarafından önerilen açgözlü randomize adaptif arama prosedürü (Greedy Randomized Adaptive Search Procedure – GRASP) [59-60] kombinatoryal problemler için geliştirilmiş bir meta-sezgisel algoritmadır. Algoritma her iterasyonunda sezgisel, açgözlü ve rasgele bir

prosedür kullanarak uygun bir çözüm üretir ve daha sonra bu çözümü yapacağı lokal araştırma prosedürünün başlangıç çözümü olarak kullanır. Algoritmaya ait detaylı kaynak taraması Resende ve Ribeiro [61] tarafından yapılmıştır.

Popülasyon tabanlı algoritmalar olarak bilinen çok çözümlü algoritmalar tek bir çözüm yerine bir çözüm seti üzerinde çalışırlar. Çok çözümlü algoritmalar gelişimsel algoritmalar ve sürü zekâsına dayalı algoritmalar olarak iki ana grup altında toplanırlar.

Darwin'in çevresine iyi adapte olan canlıların hayatta kalabileceği görüşünden ilham alınarak geliştirilen algoritmalar gelişimsel algoritmalar sınıfını oluştururlar. Bu algoritmaların her bir iterasyonu yeni bir nesli simüle eder. Her nesle sırasıyla, yeni bireyler oluşturmak için rekombinasyon, popülasyon çeşitliliğini korumak için mutasyon ve iyi bireyleri seçmek için seleksiyon operasyonları uygulanır. Böylece her bir nesilde yeni bireyler oluşur ve bunların içerisinden çevresine iyi adapte olabilenler (uygunluk değeri iyi olanlar) seçilir ve bir sonraki nesle taşınırlar. Bu işlemler belirlenen sonlandırma kriterleri sağlanana kadar devam eder. Gelişimsel algoritmaların en yaygın olanlarından bazılarının ilham kaynakları ve yapılan literatür taramaları aşağıda listelenmiştir.

- Holland tarafından önerilen GA [8] en çok bilinen ve en yaygın kullanılan gelişimsel hesaplama tekniğidir. Temel GA son derece genel bir felsefe ile sunulmuş olup, daha sonraları problem türlerine göre üzerinde yeni düzenlemeler yapılarak algoritmanın birçok yeni versiyonu türetilmiştir. Algoritmanın değişik versiyonları için Beasley ve arkadaşları [62-63], Alba ve Troya [64], Sinha ve Glodberg [65] ve Konak ve arkadaşları [66] tarafından hazırlanan detaylı literatür taramaları mevcuttur.

- Koza'nın çalışmaları ile popüler hale gelen Genetik Programlama (GP) [67], ilgili problemin çözümüne yönelik bilgisayar programı oluşturan otomatik bir sistematik sunmaktadır. Poli ve arkadaşları [68], William ve arkadaşları [69]

ve McKay ve arkadaşları [70] tarafından birçok GP gerçek dünya uygulamasını da içeren literatür taramaları yapılmıştır.

- Rechenberg tarafından önerilen gelişim stratejisi (Evolution Strategy – ES) [71-72] doğal gelişimi taklit etmek suretiyle en iyileme problemlerini çözmeye çalışır. Bäck ve arkadaşları [73], Bäck ve Schwefel [74], Beyer ve Schwefel [75] ve Kramer [76] tarafından hazırlanan literatür taramaları mevcuttur.

- İlk olarak L. J. Fogel tarafından sunulan gelişimsel programlama (Evolutionary Programming – EP) [77] daha sonra D. Fogel tarafından daha genel problemlerin çözümü için kullanıldı [78-79].

- Storn ve Price tarafından Chebyshev polinom uydurma probleminin çözümü için önerilen DE algoritması [15], zaman içerisinde sürekli küresel en iyileme problemlerinin çözümünde kullanılan en popüler algoritmalardan birisi haline gelmiştir. Algoritma halen sürekli araştırma uzaylarında tanımlı olan ve tek hedefli en iyileme problemleri için kullanılan en popüler algoritmalardan birisidir. Algoritmanın parametrelerinin iyileştirilmesi için çeşitli araştırmacıların üzerinde çeşitli çalışmalar yaptığı DE algoritmasının [80] değişik varyantlarının sunulduğu literatür taramaları Chakraborty [81], Neri ve Tirronen [82] ve Das ve Suganthan [83] tarafından sunulmuşlardır.

- Hillis tarafından önerilen birlikte evrimleşen algoritmalarda ise (Co-evolutionary Algorithms – CoEA) [84] çiçek ile böcek veya av ile avcı gibi farklı iki popülasyonun karşılıklı etkileşiminden ilham alınmıştır. Algoritma için genel bir çalışma yapısı Potter ve De Jong [85] tarafından oluşturulmuştur.

- Reynolds tarafından önerilen kültürel algoritmalar (Cultural Algorithms – CA) kültürel gelişim sürecini taklit eden bir hesaplama modeli sınıfını oluştururlar. Bu algoritmalar pek çok alanda başarı ile uygulanmışlardır [86-92].

Bazen tek başlarına hiçbir iş yapamayan varlıklar, toplu olarak hareket ettiklerinde çok zekice davranışlar sergileyebilmektedirler. Bir topluluğun bireyleri, en iyi bireyin davranışından ya da diğer bireylerin davranışlarından ve kendi deneyimlerinden yararlanarak yorum yaparlar. Elde ettikleri bu bilgileri ileride karşılaşacakları problemlerin çözümleri için bir araç olarak kullanırlar. Örneğin, bir canlı sürüsünü oluşturan bireylerden birisi bir tehlike sezdiğinde bu tehlikeye karşı tepki verir ve bu tepki sürü içinde ilerleyip tüm bireylerin tehlikeye karşı ortak bir davranış sergilemesini sağlar [93]. Canlıların sürü halinde sergiledikleri bu türden akıllı hareketlerden ilham alan sürü zekâsı tabanlı birçok en iyileme algoritması geliştirilmiştir [94-95]. Bu algoritmalar meta-sezgisel algoritmalar içerisinde önemli bir yer tutarlar. Sürü zekâsı tabanlı algoritmaların en yaygın olanlarından bazılarının ilham kaynakları ve yapılan literatür taramaları aşağıda listelenmiştir.

- Kuşların havada akınlar halindeki hareketlerinden ilham alan PSO algoritması Kennedy ve Eberhart tarafından bir küresel en iyileme tekniği olarak önerilmiştir [96]. Bugüne kadar pek çok alanda başarılı bir şekilde uygulanan algoritmanın farklı zamanlarda Kennedy ve Eberhart [97], Clerc [98], Banks ve arkadaşları [99], Thangaraj ve arkadaşları [100], Poli ve arkadaşları [101] ve Castillo ve Melin [102] tarafından literatür taramaları yapılmıştır.

- Karıncaların yiyecek arama davranışlarından ilham alan ACO algoritması Dorigo tarafından zor kombinatoryal en iyileme problemlerinin çözümü için önerilmiştir [13,103-104]. Yaygın olarak kullanılmış olan algoritmanın Dorigo ve Stützle [105], Blum [106], Dorigo ve Blum [107], Dorigo ve arkadaşları [108], Angus ve Woodward [109] ve Dorigo ve Thomas [110] tarafından yapılmış olan literatür taramaları mevcuttur.

- Bağışıklık sisteminin daha önce hakkında bilgi sahibi olmadığı patojenlere karşı vücudu korumak için kullandığı savunma mekanizmasını simüle eden AIS algoritmasının temellerini oluşturan çalışmalar ilk kez Farmer ve arkadaşları tarafından gerçekleştirilmiştir [111]. En iyileme ve makine

öğrenmesi alanlarında başarılı uygulamaları olan algoritmanın Hart ve Timmis [112], Zheng ve arkadaşları [113], Timmis ve arkadaşları [114] ve Hart ve arkadaşları [115] tarafından literatür taramaları vardır.

- Passino tarafından önerilen BFO algoritması [29,116] ise insan bağırsaklarında yaşayan Escherichia coli bakterisinin beslenme biçiminden esinlenmiştir. Das ve arkadaşları tarafından algoritmanın değişik versiyonları ve melez uygulamaları üzerinde literatür taraması yapılmıştır [117].

- Simon tarafından önerilen biyocoğrafya tabanlı en iyileme (Biogeography Based Optimization – BBO) algoritması [118] ada biyocoğrafyası denge teorisinden [119] esinlenilerek geliştirilmiştir. Dışarıdan adaya yapılan ve adadan dışarıya yapılan göçler arasındaki dengeye bağlı olarak bir adadaki toplam tür sayısının değişimi algoritmanın temel ilham kaynağıdır.

- Duman ve arkadaşlarının [4] önerdiği ve henüz daha yeni olan MBO algoritması göçmen kuşların enerji tüketimini minimize etmek için oluşturdukları V uçuş formasyonundan ilham almıştır. Algoritma çok yeni olmasına rağmen çeşitli araştırmacılar algoritma üzerinde değişik çalışmalar gerçekleştirmişlerdir [120-128].

- Arı kolonisi tabanlı en iyileme algoritmaları bal arılarının ortaya koyduğu değişik akıllı davranışları model alan algoritmalardır. Bunlardan önde gelenleri, kraliçe arı gelişim (Queen-bee Evolution – QBE) algoritması [129], ayrık arı dansı (Discrete Bee Dance – DBD) algoritması [130], arı kovanı (Bee Hive – BH) algoritması [131], bal arılarında çiftleşme en iyileme (Marriage in Honey Bees Optimization – MHBO) algoritması [132], arı koloni en iyileme (Bee Colony Optimization – BCO) algoritması [133], sanal arı algoritması (Virtual Bee Algorithm – VBA) [134] ve ABC algoritmasıdır [18]. Arıların davranışlarını konu alan algoritmaların ve bu algoritmaların uygulamalarını konu alan detaylı bir literatür taraması Karaboğa ve Akay tarafından yapılmıştır [135].

Bu tez çalışmasında, önce en iyileme problemlerinin çözümünde sıkça kullanılan ve oldukça popüler olan GA, DE, PSO, ABC ve MBO algoritmalarının tanıtımları yapılmış ve algoritmaların en iyi çözümü arama stratejileri verilmiş, daha sonra kullanılan test fonksiyonları, performans değerlendirme sistematiği, yapay sinir ağları (YSA) ve karar ağaçları gibi tez içerisinde kullanılan kavram ve yöntemler hakkında genel bilgiler verilmiştir. Ardından, tezin çalışma konusuna ait uygulamalar elde edilen mukayeseli sonuçlar ile birlikte sunulmuştur. İlk uygulamada, tezde kullanılan meta-sezgisel en iyileme algoritmalarının performans testleri yapılmış ve elde edilen sonuçlar birbiri ile mukayese edilmiştir. İkinci uygulamada, sistem kimliklendirme süreci bir en iyileme problemi olarak ele alınmış, belirlenen ve farklı derecelerden sistemlerin transfer fonksiyonları meta-sezgisel algoritmalar kullanılarak elde edilmiştir. Üçüncü uygulamada, YSA eğitimi bir en iyileme problemi olarak değerlendirilmiş, farklı veri setleri üzerinde yapılan çalışmalar ile verilerin sınıflandırılması sağlanmış ve meta-sezgisel YSA eğitiminin performansı değerlendirilmiştir. Dördüncü uygulamada, kombinatoryal en iyileme uygulaması olan bir vaka çalışmasına yer verilmiş, bir sempozyumun paralel oturumlarının katılımcı listelerinin en iyi dengelenmesi sağlanmıştır. Beşinci uygulamada, bir sınıflandırma sistematiği olan karar ağacı tasarımı hem kombinatoryal hem de nümerik en iyilemelerin birlikte uygulandığı bir en iyileme problemi olarak ele alınmış, seçilen örnek bir veri seti kullanılarak ABC ve MBO algoritmaları ile tasarlanan karar ağaçlarının performansları değerlendirilmiştir. Altıncı uygulamada, bir gerçek dünya problemi olan sürekli döküm tesisi kesme planlarının en iyilemesi ABC algoritması kullanılarak yapılmış ve elde edilen başarılı sonuçlar verilmiştir. Yedinci uygulamadan on birinci uygulamaya kadar olan uygulamalarda ABC ve MBO algoritmalarının modifiye ve birlikte kullanımlarından elde edilen 5 yeni sistematik performans testlerinden alınan mukayeseli sonuçlar ile birlikte sunulmuştur.

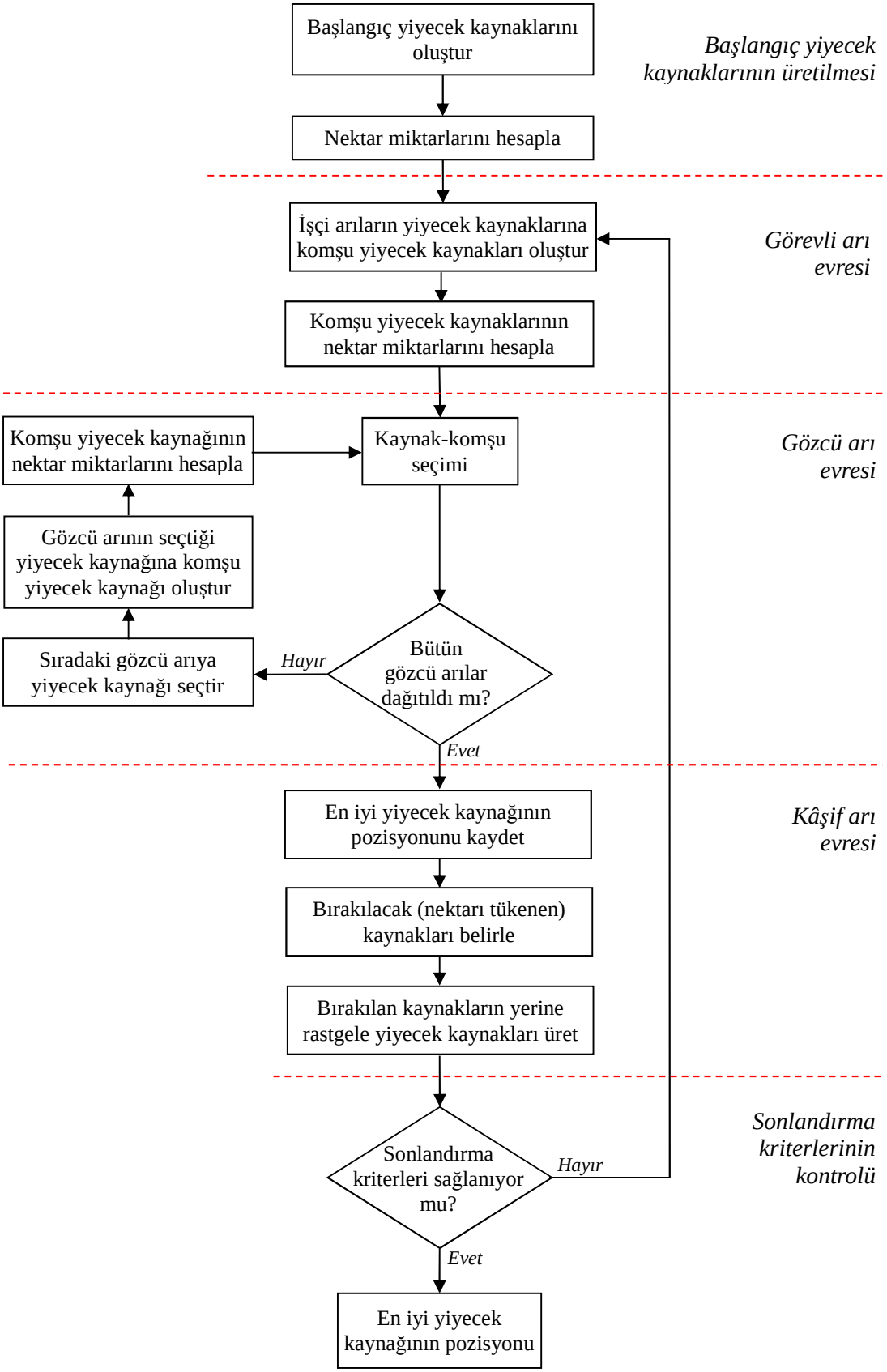
BÖLÜM 2. META-SEZGİSEL ALGORİTMALAR

Giriş kısmında kategorilere ayrılarak hakkında kısa literatür bilgisi sunulan çok sayıda meta-sezgisel algoritma içerisinden popüler, güncelliğini koruyan ve tez çalışması içerisindeki implementasyonlarda kullanılan sürü zekâsına dayalı olan ABC, MBO ve PSO algoritmaları ile gelişimsel temelli olan DE algoritması ve GA'nın çalışma sistematikleri bu bölümde verilmiştir.

2.1. Yapay Arı Koloni Algoritması

Arıların beslenme davranışlarını simüle eden yapay arı koloni (Artificial Bee Colony - ABC) algoritmasında her bir yiyecek kaynağı ilgili en iyileme probleminin olası bir çözümünü ve her bir yiyecek kaynağının ihtiva ettiği nektar miktarı ilgili çözümün uygunluk değerini temsil eder [18]. Algoritmada arı kolonisinin üç farklı sınıf arıdan oluştuğu kabul edilir: görevli arılar, gözcü arılar ve kâşif arılar. Koloninin yarısının görevli arı diğer yarısının da gözcü arı olduğu varsayılır [19,136]. Kâşif arı ise bir statüyü temsil eder ve yiyecek kaynağı tükenen görevli arının rastgele yiyecek kaynağı araması durumundaki geçici bir evre olarak tanımlanabilir. Algoritmanın her bir döngüsünde Şekil 2.1'de detayı görülen üç ayrı evre vardır [2]. Bunlar; görevli arı evresi, gözcü arı evresi ve kâşif arı evresidir.

Algoritma hesaplamaya başlarken görevli arıların tamamı kâşif arı olarak çalışmaya başlarlar ve rastgele yiyecek kaynağı ararlar. Bu durum sabah kovandan çıkan arıların durumuna benzer. Yani ilk aramaların tamamı rasgele yapılır. Diğer bir ifade ile ilk yiyecek kaynağı pozisyonlarının üretilmesi işlemi, pozisyon bilgisini temsil eden değişkenlere tanımlı oldukları değer aralığında rasgele değerler atanması işlemidir. Bu atama işlemi



Şekil 2.1. ABC algoritması akış diyagramı

$$x_{ij} = x_j^{min} + \text{rand} \cdot \left(x_j^{max} - x_j^{min} \right) \quad \textbf{(2.1)}$$

şeklinde gerçekleştirilir. Burada i [1, SN] aralığında bir tamsayı, SN yiyecek kaynağı sayısı, j [1, D] aralığında bir tamsayı, D problem boyutu veya optimize edilecek parametre sayısı, x_{ij} i çözümünün j boyutundaki pozisyon bilgisi, x_j^{min} ve x_j^{max} problem uzayının j boyutundaki minimum ve maksimum limit değerleri ve rand [0,1] aralığında düzgün dağılımlı rastgele bir sayıdır. Bu yöntemle, her biri kendisine farklı bir yiyecek kaynağı bulmuş olan kâşif arılar bu aşamadan sonra görevli arıya dönüşürler. Her bir görevli arının yiyecek kaynağının nektarını temsil eden uygunluk değeri

$$Fitness_i = \begin{cases} 1/(1+f_i) & , \quad f_i \geq 0 \\ 1+\text{abs}(f_i) & , \quad f_i < 0 \end{cases} \quad \textbf{(2.2)}$$

olarak hesaplanır. Burada f_i ve $Fitness_i$ sırası ile i çözümünün maliyet ve uygunluk değerleridir. Artık görevli arılar ile yiyecek kaynakları arasında bire bir eşleşme sağlanmıştır. Görevli arıların başlangıç konumları belirlendikten sonra algoritmanın sonlandırma kriterleri sağlanana kadar hesaplama evreleri birbiri ardınca işletilir. Her bir görevli arı için

$$\hat{x}_{ij} = x_{ij} + \phi \cdot \left(x_{ij} - x_{kj} \right) \quad \textbf{(2.3)}$$

şeklinde yeni komşuluklar üretirler. Burada x_{ij} i çözümünün j boyutundaki pozisyon bilgisi, x_{kj} k çözümünün j boyutundaki pozisyon bilgisi, k rastgele seçilmiş ve i'den farklı bir indeks, ϕ [-1,1] aralığında düzgün dağılımlı rastgele üretilmiş bir sayı ve $\hat{x}_{ij}$ i çözümü için üretilen komşuluğun j boyutundaki pozisyon bilgisidir. Eğer $\hat{x}_{ij}$ pozisyon bilgisi Denklem (2.1)'de kullanılan problem uzayının j boyutu için tanımlı olan minimum ve maksimum limit değerlerinden birini aşarsa, $\hat{x}_{ij}$ 'nin değeri aşılan limit değeri ile Denklem (2.4)'te tanımlandığı gibi değiştirilir.

$$\hat{x}_{ij} = \begin{cases} x_j^{min} & , \quad \hat{x}_{ij} \le x_j^{min} \\ \hat{x}_{ij} & , \quad x_j^{min} < \hat{x}_{ij} < x_j^{max} \\ x_j^{max} & , \quad \hat{x}_{ij} \ge x_j^{max} \end{cases} \tag{2.4}$$

Bu aşamada, üretilen her komşuluğun uygunluk değeri Denklem (2.2) kullanılarak hesaplanır. Görevli arı mevcut pozisyonu ile ürettiği yeni komşuluk arasında her iki pozisyonun uygunluk değerlerini mukayese ederek bir seçim yapar. Üretilen komşuluğun uygunluk değeri daha iyi ise komşu çözümün pozisyon bilgisi görevli arının pozisyonuna atanır. Aksi halde görevli arının mevcut pozisyonu korunur. Bu işlem görevli arının yerel arama yolu ile daha iyi yiyecek kaynaklarına yönelmesini simüle eder.

Daha sonra görevli arının kovana topladığı nektarı getirdiği ve kendi yiyecek kaynağı hakkındaki bilgiyi kovandaki gözcü arılarla paylaştığı varsayılır. İlgili yiyecek kaynağının uygunluk değeri, algoritmada gözcü arılarla paylaşılan bilgiyi temsil eder. Gözcü arıların uygunluk değeri yüksek yiyecek kaynaklarını tercih ihtimali daha yüksektir. Dolayısıyla, her bir görevli arının yiyecek kaynağının gözcü arılar tarafından seçilme ihtimali, Denklem (2.5)'te verildiği gibi ilgili kaynağın uygunluk değerinin tüm kaynakların uygunluk değerlerinin toplamına oranı şeklinde hesaplanır.

$$p_i = \frac{Fitness_i}{\sum_{j=1}^{SN} Fitness_j} \tag{2.5}$$

Burada p_i ve $Fitness_i$ sırası ile gözcü arıların i yiyecek kaynağını seçme ihtimali ve bu kaynağın uygunluk değeri ve SN toplam yiyecek kaynağı sayısıdır.

Hesaplamaların ikinci evresi olan gözcü arı evresinde gözcü arılar p_i olasılık değerlerini dikkate alarak seçimlerini yaparlar. Açıkça görülüyor ki; ilgili kaynağın uygunluk değeri ile o kaynağının gözcü arılar tarafından seçilme ihtimali doğru orantılıdır. Temel ABC algoritmasında gözcü arılar seçim işlemini rulet tekerleği yöntemi ile yaparlar. Şekil 2.2'de sözde kodu verilen rulet tekerleği yöntemi

uygunluk değeri daha yüksek olan yiyecek kaynaklarına daha fazla seçilebilme şansı verir.

```
RasgeleSayı ← rand(0,1)
KısmiToplam ← 0
i ← 0
repeat
        i ← i + 1
        KısmiToplam ← KısmiToplam + p_i
until (KısmiToplam ≥ RasgeleSayı)
return i
```

Şekil 2.2. Rulet tekerleği metodu sözde kodu

Gözcü arı rulet tekerleği yöntemini kullanarak seçimini yaptıktan sonra seçtiği yiyecek kaynağına Denklem (2.3)'ü kullanarak bir komşuluk üretir. Üretilen komşuluğun uygunluk değeri seçilen kaynağın uygunluk değerinden daha iyi ise komşu çözümün pozisyon bilgisi seçilen kaynağın pozisyonuna atanır. Aksi halde gözcü arının mevcut seçiminin pozisyon bilgisi korunur. Bu işlem, gözcü arıların nektarı daha çok olan yiyecek kaynaklarına yönelmelerini simüle eder. Algoritma bu sayede uygunluk değeri daha iyi olan çözümlerin daha fazla iyileştirilmesine imkân tanır.

İlerleyen iterasyonlarda çözümler birbirine benzemeye başladıkça Denklem (2.3)'te parantez içerisinde verilen terimler birbirine yaklaşacağından değişim miktarı giderek azalacaktır [137]. Yani komşuluk üretimi yolu ile çözümlerin iyileştirilmesi imkânsızlaşacaktır. Bu durumu engellemek için ABC algoritmasında her yiyecek kaynağına bir adet hata sayacı tahsis edilmiştir. Gerek görevli arılar tarafından gerekse gözcü arılar tarafından bir kaynağın uygunluğunu artırmak için komşuluk üretildiğinde, eğer deneme başarısızlıkla sonuçlanırsa ilgili kaynağın hata sayacı bir artırılır. Aksi halde sayaç değeri sıfırlanır. Kâşif arı evresine gelindiğinde yiyecek kaynaklarına ait hata sayaçları kontrol edilir. Değeri başlangıçta belirlenen bir limit değerini geçen hata sayaçlarına ait yiyecek kaynaklarının tükendikleri varsayılır. Bu kaynaklara ait görevli arılar kâşif arılara dönüştürülürler. Kâşif arılar Denklem (2.1)'i kullanılarak rasgele arama yaparlar. Algoritma bu davranışıyla, gerçek arıların

nektarı tükenen yiyecek kaynaklarını terk etmesini simüle ederken, ümit vadetmeyen problem çözümlerinin iyileştirilme denemelerini de sonlandırmış olur. Bu çözümlerin yerine küresel arama yaparak yeni çözüm adayları üretir. Kâşif arı rasgele yeni yiyecek kaynağı bulduktan sonra tekrar görevli arıya dönüşür.

Denklem (2.1) aracılığıyla bütün x_{ij} , $i \in [1, SN]$, $j \in [1, D]$ çözümlerine rastgele başlangıç değerlerini ata
Çözüm geliştirememe sayaçlarını sıfırla ($failure_i = 0$)
Denklem (2.2) aracılığıyla uygunluk değerlerini ($fitness_i$) hesapla
repeat
    for *i = 1* ***to*** *SN*
        Denklem (2.3) aracılığı ile x_i çözümünün görevli arısına $\hat{x}_i$ çözümünü üret
        Denklem (2.2) aracılığıyla $\hat{x}_i$ 'nin uygunluk değerini hesapla
        Uygunluk değerine göre x_i ile $\hat{x}_i$ arasında bir seçim yap
        x_i çözümü geliştirilememişse: $failure_i = failure_i + 1$, gelişmiş ise: $failure_i = 0$
    end for
    Denklem (2.5) aracılığıyla x_{ij}'lerin gözcü arılarca seçilme ihtimallerini hesapla
    for *i = 1* ***to*** *SN*
        Şekil 2.2'de verilen rulet tekerleği yöntemi ile r çözümünü seç
        Denklem (2.3) aracılığı ile x_r çözümünün gözcü arısına $\hat{x}_r$ çözümünü üret
        Denklem (2.2) aracılığıyla $\hat{x}_r$ 'nin uygunluk değerini hesapla
        Uygunluk değerine göre x_r ile $\hat{x}_r$ arasında bir seçim yap
        x_r çözümü geliştirilememişse: $failure_r = failure_r + 1$, gelişmiş ise: $failure_r = 0$
    end for
    if *max($failure_i$) > limit* ***then***
        Denklem (2.1) aracılığıyla x_i çözümünü rasgele bir çözümle değiştir
    end if
    En iyi çözümü sakla
until *Durdurma kriterleri*
return *Saklanan en iyi çözüm*

Şekil 2.3. ABC algoritması sözde kodu

Üzerinde yoğunlaşılması gereken diğer bir husus ise hata sayacı için kullanılacak limit değerinin tespitidir. Bu değer

$$FC_{Limit} = SN \cdot D \qquad \textbf{(2.6)}$$

olarak sadece problemin boyutuna ve koloni büyüklüğüne (toplam arı sayısı) bağlı olarak başlangıçta hesaplanabilir [138]. Burada *SN* toplam yiyecek kaynağı sayısı olup koloni büyüklüğünün yarısına eşittir ve *D* problem boyutudur. Sonuç olarak, ABC algoritması gerçekleştirilirken başlangıçta belirlenmesi gereken tek algoritma parametresi koloni büyüklüğüdür. Genel olarak tanıtımı yapılan ABC algoritmasının sözde kodu Şekil 2.3'te verildiği gibidir [2].

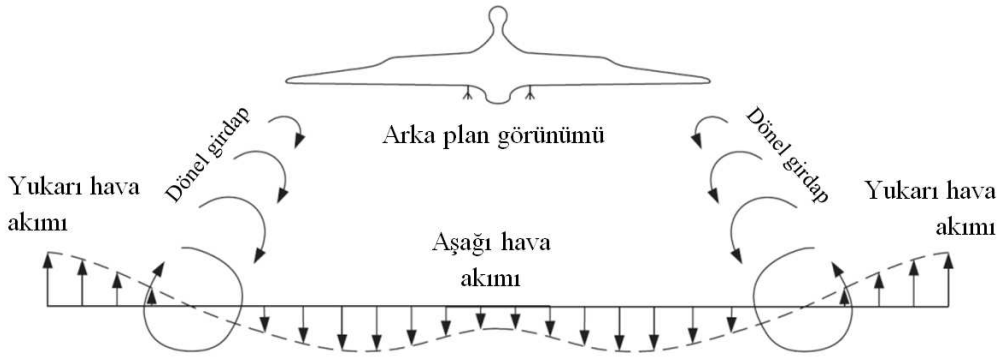
2.2. Göçmen Kuşlar En İyileme Algoritması

Göçmen kuşlar en iyileme (Migrating Birds Optimization - MBO) algoritması tabiattan ilham alınarak inşa edilen meta-sezgisel komşuluk arama sistematikleri arasında en yeni metotlardan birisidir [4]. Algoritma özet olarak göçmen kuşların uzun mesafe uçuşlarında oluşturdukları V biçimindeki uçuş formasyonunu simüle eder. Kuşların Şekil 2.4'te görülen V biçimli uçuş formasyonu onların maruz kaldıkları hava sürtünmesini azaltır. Dolayısıyla bu formasyon, onların enerji sarfiyatını minimize eden efektif bir diziliş biçimidir. Örneğin, 25 kuştan oluşan ve V biçiminde uçan bir kuş sürüsünde her bir kuşun maruz kaldığı hava sürtünmesi %65'lere varan oranda azalır. Bu durum sürünün uçuş menzilini %70'e varan oranlarda artırır [139].

Şekil 2.4. Kuşların V biçiminde uçuşu

Bu uçuş biçimindeki yarar mekanizması kısaca şu şekilde açıklanabilir. Uçuş halindeki bir kuşun kanat hareketleri nedeniyle Şekil 2.5'te görüldüğü gibi bir çift

dönel girdap oluşur. Şekle uçuş yönünde bakıldığında, girdaplardan sol taraftakinin saat yönünde ve diğerinin de saat yönünün aksi yönünde dönüş sergiledikleri görülür [4,139]. Bu girdaplar kuşun tam arkasında aşağı yönde hava akımı, sağ arka ve sol arka kısımlarında ise yukarı yönde hava akımları oluştururlar. Aşağı yönde oluşan hava akımlarına maruz kalma kanatlarda oluşan sürtünmeyi artırması nedeniyle istenmeyen bir durumdur. Diğer taraftan yukarı yönde oluşan hava akımları kanatlarda oluşan sürtünmeyi azaltacağından, bu bölgede uçan diğer kuşlar için fayda sağlarlar. Azalan hava sürtünmesi kuşun enerji sarfiyatını azaltıp, menzilini artıracağından, en önde bulunan lider kuş hariç sürüdeki diğer kuşlar bu yukarı yöndeki hava akımı bölgelerinde yer alacak şekilde konumlanarak V uçuş biçimini oluştururlar. V uçuş biçiminde lider kuş en fazla enerjiyi harcayan kuş olurken, diğer kuşlar önlerinde uçan kuşların girdaplarından yararlanarak daha az enerji sarf ederler [4].

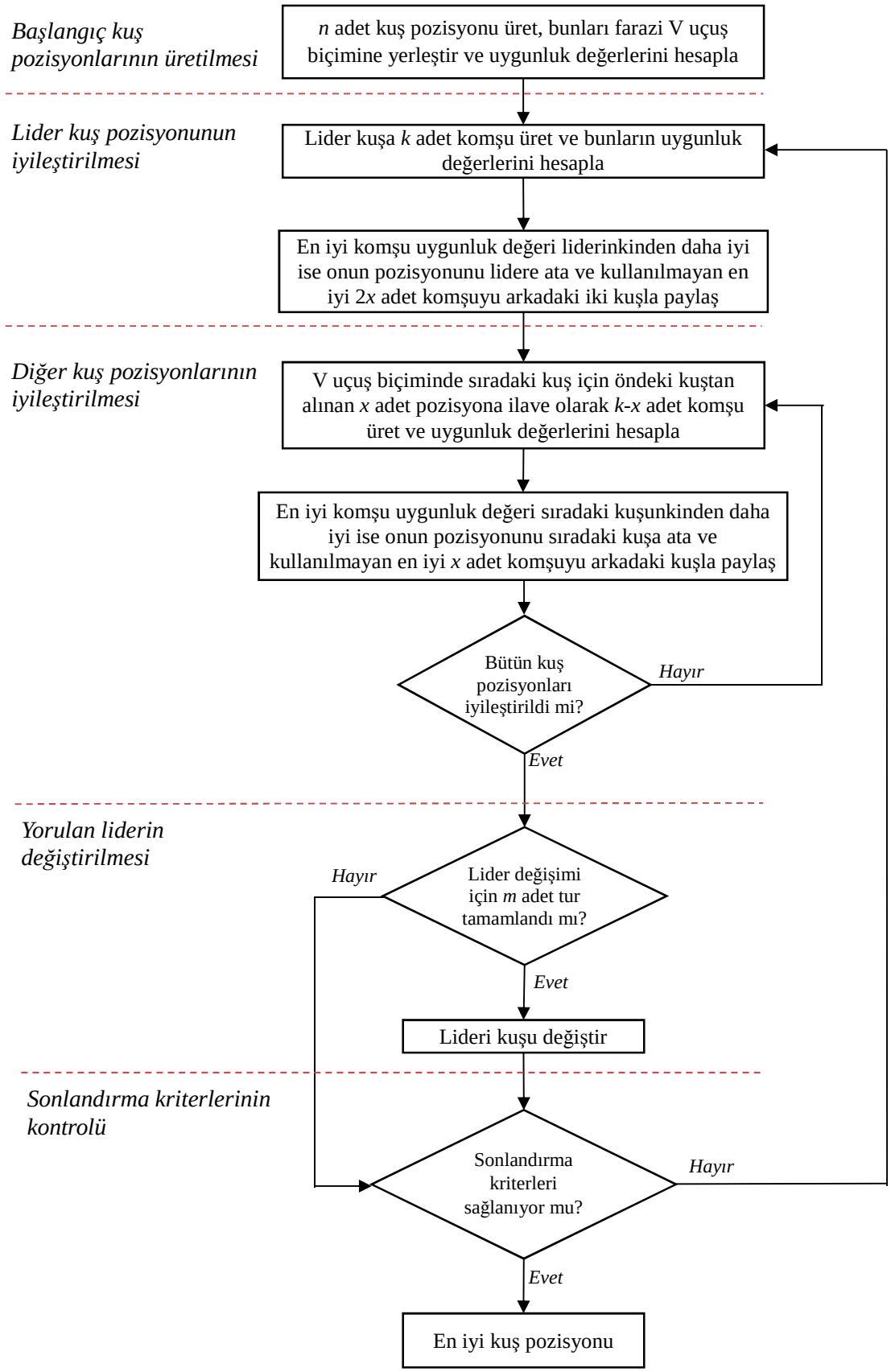


Şekil 2.5. Kanat hareketlerinden kaynaklanan dönel girdapların oluşturduğu aşağı ve yukarı yönlü hava akımı bölgeleri

MBO algoritmasının akış diyagramı Şekil 2.6'da ve algoritmanın kullandığı temel parametreler Tablo 2.1'de listelenmiştir [4].

Tablo 2.1. MBO algoritmasının temel parametreleri

Parametre	**Açıklama**
n	Toplam çözüm sayısı (sürü boyutu)
k	Her bir çözüm için işlem yapılacak olan toplam komşuluk sayısı ($^{1}/_{\text{uçuş hızı}}$)
x	Bir sonraki bireyle paylaşılacak çözüm sayısı (yukarı hava akımı)
m	Lider değişimi için gerekli toplam iterasyon sayısı
K	Maksimum iterasyon sayısı (uçuş süresi)



Şekil 2.6. MBO algoritması akış diyagramı

Algoritmanın çalışması şu şekildedir. Algoritmada her bir kuşun pozisyon bilgisi ilgili en iyileme probleminin olası çözümlerinden birini temsil eder. Algoritmanın ilk adımında kuşların pozisyonları rastgele üretilir. Bu işlem pozisyon bilgisini temsil eden değişkenlere tanımlı oldukları değer aralığında düzgün dağılımlı rasgele değerler atanması şeklinde gerçekleştirilir. Bu atama işlemi çoğu algoritmada olduğu gibi Denklem (2.1) kullanılarak yapılabilir [19,136]. Çözümler için pozisyon bilgileri üretildikten sonra bunlardan biri lider kuşu temsilen seçilir ve üretilen bütün çözümler farazi bir V formasyonu üzerine yerleştirilirler. Çözümlerin uygunluk değerleri hesaplanır. Daha sonraki algoritma adımlarında lider kuştan başlayıp V formasyonunun sağ ve sol kolları üzerinde en sondaki kuşlara varıncaya kadar tek tek ilerlenerek mevcut her bir çözüm ona komşuluklar üretilerek iyileştirilmeye çalışılır [4].

İkinci adımda lider kuşun pozisyonunun iyileştirilmesi hedeflenmiştir. Bu maksatla *k* adet komşu çözüm üretilir ve bunların uygunluk değerleri hesaplanır. Komşu çözümler üretilirken değişik komşu üretme yöntemleri kullanılabilir. Örneğin Denklem (2.3) bu işlem için uygundur. Eğer en iyi uygunluk değerine sahip komşu çözüm lider kuştan daha iyi uygunluk değerine sahip ise, bu durumda ilgili komşu çözümün pozisyonu lider kuşu temsil eden çözüme atanır. En iyi uygunluk değerlerine sahip olan 2*x* adet kullanılmayan komşu çözüm ikinci sıradaki iki adet kuşla paylaşılır. Bunlardan *x* adedi ikinci sıradaki sağ koldaki kuş için ve diğer *x* adedi de ikinci sıradaki sol koldaki kuş içindir.

Üçüncü adımda lider kuş haricindeki diğer kuşların pozisyonlarının iyileştirilmesi denenir. Bir döngü içerisinde sağ ve sol kolda sıradaki kuşların her biri için (*k*-*x*) adet komşu çözüm üretilir ve bunların uygunluk değerleri hesaplanır. Yine Denklem (2.3) komşu çözüm üretmek için kullanılabilir. Üretilen komşu çözümler ile öndeki kuşu temsil eden çözümün paylaştığı *x* adet komşu çözüm birleştirilerek lider kuşta olduğu gibi toplam yine *k* adet komşu çözüm elde edilir. Daha sonra bu yolla elde edilen toplam *k* adet komşu çözüm içerisindeki en iyi uygunluk değerine sahip çözüm alınır. Bu komşu çözümün uygunluk değeri ilgili kuşu temsil eden çözümünkinden daha iyi ise komşu çözümün pozisyon bilgisi ilgili kuşu temsil eden çözüme atanır. En iyi

uygunluğa sahip olan ve kullanılmayan *x* adet çözüm bir sonraki kuşu temsil eden çözüm ile paylaşılır.

Birinci ve ikinci adımlarda üretilen komşu çözümlerin pozisyon bilgileri problem uzayının ilgili boyutu için tanımlı olan minimum ve maksimum limit değerlerinden birini aşarsa, aşılan limit değeri Denklem (2.4)'te tanımlandığı gibi üretilen komşu çözümün ilgili boyutuna atanır.

Birinci ve ikinci adımlarda bahsedilen, üretilen komşu çözümlerinin bir kısmının bir sonraki çözüm ile paylaşılması işlemi, gerçek kuşların V biçiminde uçarken öndeki kuşların oluşturduğu yukarı yöndeki hava akımından istifade etmelerini simüle etmektedir.

Bütün kuşlar için iyileştirme denemeleri tamamlandığında algoritmanın bir döngüsü tamamlanmış olur. Lider kuşu temsil eden çözüm için *k* adet komşu çözüm üretilirken bu sayı diğer kuşları temsil eden çözümler için (*k*-*x*)'tir. Bu durum gerçek kuş sürüsündeki lider kuşun diğerlerinden daha çok enerji harcamasını ve bu yüzden daha çok yorulmasını simüle etmektedir.

Algoritmanın dördüncü adımında önceden belirlenen *m* adet döngü tamamlandığında lider kuşun yorulduğu varsayılır. Yorulan lider kuş V diziliminin sağ veya sol tarafındaki kolun en sonuna gider ve aynı kol üzerinde ikinci sırada bekleyen kuş lider konumuna geçer. Bu değişim her defasında V diziliminin farklı kollarında gerçekleşir. MBO algoritmasının farazi V dizilimi üzerinde kuşları temsil eden çözümlerin yerleşimleri de benzeri mantık kullanılarak değiştirilir.

İkinci adımdan dördüncü adıma kadar olan işlemler algoritmanın belirlenen sonlandırma kriterleri sağlanıncaya kadar tekrarlanır. Daha sonra algoritma durur ve mevcut çözümler (sürü) içerisinde en iyi uygunluk değerine sahip çözümü (kuşu) en iyileme probleminin elde edilen en iyi çözümü olarak verir.

MBO algoritmasının performansı *k* ve *x* parametrelerinin seçiminden direkt etkilenir. Dolayısıyla bu parametrelerin seçiminde dikkatli olunmalıdır. Gerçek kuşların uçuş

hızı ile *k* parametresi birbiriyle ters orantılıdır. Küçük *k* değerleri daha az komşu çözümler üretilmesine neden olur ve algoritmanın bir döngüsünün daha çabuk tamamlanmasını sağlar. Bu durum, *k* değerinin küçük seçilmesinin yüksek uçuş hızına neden olması şeklinde yorumlanır. Yüksek uçuş hızı MBO algoritmasının sonuca çabuk varmasını sağlar. Dolayısıyla az parametre içeren en iyileme problemlerinde *k* değerinin küçük seçilmesi bir avantaj olarak değerlendirilebilir. Öte yandan, *k* değerinin artırılması araştırma derinliğini artırır. Yüksek boyutlu problemlerde tatmin edici sonuçların yakalanabilmesi için *k* değerinin yüksek tutulması gerekebilir. Sonuç olarak, parametre sayısı fazla olan en iyileme problemlerinde daha iyi sonuçlar elde edebilmek için toplam işlem zamanından fedakârlık yapılarak *k* değeri artırılıp daha derin bir araştırma yapılması sağlanabilir.

```
Denklem (2.1) aracılığıyla bütün çözümlerine rastgele başlangıç değerlerini ata
Rasgele seçilen bir çözümü lider olarak seç
Çözümleri farazi bir V formasyonuna rasgele yerleştir
sol = true
repeat
    for i = 1 to m
        Lider çözüme k adet komşu çözüm üreterek onu iyileştirmeyi dene
        Kullanılmayan en iyi çözümlerden x adedini sol arkadaki x adedini de sağ
            arkadaki çözümlerle paylaş
        for lider haricindeki çözümlerin her biri
            Sıradaki çözüme (k-x) adet komşu çözüm üret ve öndeki çözümden alınan x
                adet çözümle birlikte sıradaki çözümü iyileştirmeyi dene
            Kullanılmayan en iyi çözümlerden x adedini bir arkadaki çözümle paylaş
        end for
    end for
    if sol then
        lider çözümü farazi V formasyonun sol kolunun sonuna kaydır ve sol koldaki ilk
            çözümü lider olarak ata
    else
        lider çözümü farazi V formasyonun sağ kolunun sonuna kaydır ve sağ koldaki
            ilk çözümü lider olarak ata
    end if
    sol = not (sol)
until Durdurma kriterleri
return Mevcut en iyi çözüm
```

Şekil 2.7. MBO algoritması sözde kodu

Gerçek kuşların meydana getirdiği dönel girdaplar neticesinde oluşan yukarı yöndeki hava akımlarından arkadaki kuşların faydalanması olayı MBO algoritmasında *x* parametresi ile temsil edilir. MBO algoritmasındaki fayda mekanizması bir önceki çözümden alınan iyi komşu çözümler olarak ifade edildiğinden, yüksek *x* değerlerinin kullanımı, algoritma döngüleri ilerledikçe çözümlerin birbirlerine daha çok benzemelerine neden olur. Bu durum ise erken yakınsama sonucunu doğurur. Genel olarak tanıtımı yapılan MBO algoritmasının sözde kodu Şekil 2.7'de verildiği gibidir [4].

2.3. Parçacık Sürü En İyileme Algoritması

Parçacık Sürü En İyileme (Particle Swarm Optimization - PSO) algoritması sürü zekâsına dayalı bir algoritma olup kuşların havada gerçekleştirdikleri akınlardan ilham alınarak inşa edilmiştir. Akış diyagramı Şekil 2.8'de verilen PSO algoritmasında çok boyutlu araştırma uzayında hareket halinde oldukları varsayılan parçacıklardan oluşan bir popülasyon (sürü) kullanılır. Popülasyon üyelerinin pozisyonları ilgili en iyileme probleminin olası çözümlerini temsil eder.

PSO algoritmasının başarılı bir en iyileme yapması büyük ölçüde başlangıç popülasyonunun çeşitliliğine bağlıdır. Arzu edilen kalitede bir başlangıç popülasyonu oluşturmak için Denklem (2.1) kullanılabilir. Popülasyon oluşturulduktan sonra üyelerin uygunluk değerleri hesaplanır. Popülasyon üyelerinin başlangıç pozisyonları aynı zamanda onların bireysel en iyi pozisyonları olarak kabullenilir ve parçacıkların başlangıç hızları sıfırlanır.

Algoritma ana döngüsü içerisinde, önce her bir parçacığın bir sonraki zaman adımında sahip olacağı hız vektörü hesaplanır. Bir sonraki zaman adımında *i* parçacığının *j* boyutundaki hızı

$$v_{ij}(t+1) = w \cdot v_{ij} + c_1 \cdot \text{rand} \cdot \left(p_{ij}(t) - x_{ij}(t)\right) + c_2 \cdot \text{rand} \cdot \left(p_{gj}(t) - x_{ij}(t)\right) \qquad \textbf{(2.7)}$$

olarak hesaplanır [94]. Burada *w* eylemsizlik ağırlığı, c_1 ve c_2 ivme katsayıları, rand [0,1] aralığında düzgün dağılımlı rasgele bir sayı, x_{ij} *i* parçacığının *j* boyutundaki

pozisyon değeri, p_{ij} *i* parçacığının *j* boyutunda o ana kadar elde ettiği en iyi değer ve p_{gj} popülasyonun *j* boyutunda o ana kadar elde ettiği en iyi değerdir.

Şekil 2.8. PSO Algoritması akış diyagramı

Denklem (2.7)'nin ikinci ve üçüncü terimleri sırasıyla parçacığın bilişsel ve sosyal bileşenlerini oluştururlar. Bir sonraki zaman adımı için hesaplanan hız değeri bileşenleri önceden belirlenen [$-V_{max}$,V_{max}] limitleri içerisinde tutulmalıdır. Hızın belirlenen limitler içerisinde tutulması küresel aramanın kontrolden çıkmasını engellemek için gereklidir. Hız *j* boyutunda tanımlı olan minimum ve maksimum limit değerlerinden birini aşarsa, aşılan limit değeri Denklem (2.8)'de tanımlandığı gibi hızın *j* boyutundaki değerine atanır.

$$v_{ij}(t+1)=\begin{cases} -V_{\max} & , \quad v_{ij}(\mathrm{t}+1)\le -V_{\max} \\ v_{ij}(\mathrm{t}+1) & , \quad -V_{\max}< v_{ij}(\mathrm{t}+1)< V_{\max} \\ V_{\max} & , \quad v_{ij}(\mathrm{t}+1)\ge V_{\max} \end{cases} \qquad \textbf{(2.8)}$$

V_{max} limit değeri çoğu test probleminde bütün boyutlar için aynı seçilmesine rağmen problemin özelliklerine bağlı olarak farklı boyutlarda farklı limit değerler de seçilebilir.

Denklem (2.7)'deki *w* eylemsizlik ağırlığı algoritmanın yakınsama karakteristiğinin kontrolü açısından önemli bir parametredir. Bu parametrenin birden büyük bir değer olarak seçilmesi hız bileşenlerinin limit değerlere yakınsamasına ve PSO algoritmasının küresel minimumdan uzaklaşmasına sebep olur. Çok küçük *w* değerleri ise algoritmanın küresel arama yeteneğini zayıflatır. Bu yüzden eylemsizlik ağırlığı *w* için [0,1] aralığındaki değerler tercih edilir. Bu aralıktaki büyük *w* değerleri PSO algoritmasının küresel arama yeteneğini güçlendirirken yerel arama yeteneğini zayıflatır. Yine bu aralıkta küçük *w* değerleri PSO algoritmasının küresel arama yeteneğini zayıflatıp yerel arama yeteneğini güçlendirir [137]. Eylemsizlik ağırlığı *w*'nun değeri PSO algoritmasının iterasyonları boyunca 0.9'dan başlayarak 0.4' e kadar doğrusal olarak azaltıldığında algoritma performansının önemli ölçüde arttığı deneysel olarak raporlanmıştır [140].

İvme katsayıları c_1 ve c_2'nin küçük değerleri parçacık yörüngelerinde yavaş değişimlere, büyük değerleri ise parçacık yörüngelerinde ani ve hızlı değişimlere neden olur. İvme katsayılarına farklı değerler atanabileceği gibi çoğu uygulamada bu katsayılara eşit değerler atanmıştır. Statik 1.494 değerinin uygulamalarda iyi sonuçlar verdiği deneysel olarak raporlanmıştır [141].

Ana döngünün ikinci adımında parçacıkların mevcut pozisyonlarına bir sonraki zaman adımı için hesaplanan hız değerleri ilave edilerek parçacıkların bir sonraki zaman adımındaki pozisyonları

$$x_i(t+1) = x_i(t) + v_i(t+1) \qquad \textbf{(2.9)}$$

şeklinde hesaplanır. Sonuç olarak pozisyon bilgisinin hesaplanmasında kullanılan hız vektörünün içerdiği bilişsel ve sosyal bileşenler sebebiyle, bir parçacığa ait pozisyon bilgisi hem o parçacığa ait tecrübi bilgiyi hem de parçacığın komşularının tecrübi bilgilerini ihtiva eder.

```
Denklem (2.1) aracılığıyla bütün parçacıklara rastgele başlangıç pozisyonlarını ata
Parçacıkların uygunluk değerlerini hesapla
Parçacıkların başlangıç pozisyonlarını en iyi parçacık pozisyonlarına ata
Sürüdeki en iyi parçacık pozisyonunu kaydet
Parçacıkların başlangıç hızlarını sıfırla
repeat
    for Parçacıkların her biri
        Denklem (2.7) aracılığı ile parçacıkların yeni hızlarını hesapla
        Denklem (2.9) aracılığı ile parçacıkların yeni pozisyonlarını hesapla
        Parçacığa ait en iyi pozisyonu güncelle
    end for
    Sürüdeki en iyi parçacık pozisyonunu güncelle
until Durdurma kriterleri
return Sürüdeki en iyi parçacık pozisyonu
```

Şekil 2.9. PSO algoritması sözde kodu

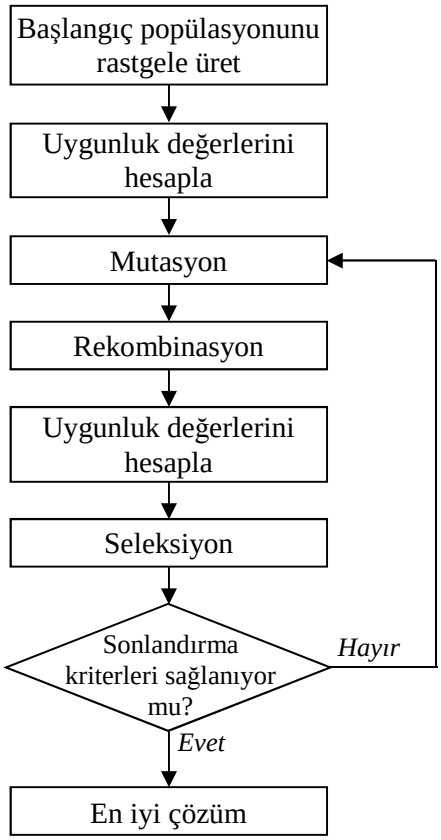
Bir sonraki adımda parçacıkların her birinin ve sürünün tamamının ulaştığı en iyi pozisyon değerleri takip eden iterasyonda kullanılacak olan hız vektörünü hesaplamak için güncellenir. Ana döngü, sonlandırma kriterleri sağlanıncaya kadar tekrarlanır. Genel olarak tanıtımı yapılan PSO algoritmasının sözde kodu Şekil 2.9'da verilmiştir.

2.4. Diferansiyel Gelişim Algoritması

Akış diyagramı Şekil 2.10'da verilen diferansiyel gelişim (Differential Evolution - DE) algoritması genetik algoritmanın (GA) kullandıklarına benzeyen operatörler kullanan bir en iyileme algoritmasıdır [15]. Bu işlem operatörleri: mutasyon, rekombinasyon ve seleksiyon operatörleridir. DE algoritması ile GA arasındaki ana farklılık: yeni nesiller üretilirken GA'nın rekombinasyon üzerinde odaklanması, DE algoritmasının ise mutasyon üzerinde odaklanmasıdır [142].

Algoritmada her bir popülasyon üyesinin pozisyon bilgisi ilgili en iyileme probleminin olası çözümlerinden birini temsil eder. Algoritmanın ilk adımında popülasyon üyelerinin pozisyonları, pozisyon bilgisini temsil eden değişkenlere her

birinin tanımlı olduğu değer aralığında düzgün dağılımlı rasgele bir değer atanarak belirlenir. Bu atama işlemi Denklem (2.1) kullanılarak yapılabilir. Başlangıç değerleri atandıktan sonra popülasyon üyelerinin uygunluk değerleri hesaplanır.



Şekil 2.10. DE Algoritması akış diyagramı

DE algoritması araştırma işlemini mutasyon operatörü aracılığı ile gerçekleştirir. Bu operatör Denklem (2.10)'u kullanarak mevcut popülasyon üyelerine mutant çözümler üretir.

$$v_{i,G+1} = x_{r_1,G} + F \cdot \left(x_{r_2,G} - x_{r_3,G}\right) \qquad \textbf{(2.10)}$$

Burada $v_{i,G+1}$ i çözümünün bir sonraki jenerasyonu için üretilen mutant çözüm; $x_{r_n,G}$ mevcut jenerasyondaki bir çözüm; r_1, r_2 ve r_3 rastgele seçilen, hem birbirinden hem de o an üzerinde işlem yapılan i indeksinden farklı indeks değerleri ve F [0,1] aralığında seçilen amplifikasyon sabitidir.

Rekombinasyon operatörü mutant ve ebeveyn çözümleri birbiri içerisinde karıştırarak deneme (trial) çözümlerini üretir. Deneme çözümleri

$$u_{ji,G+1} = \begin{cases} v_{ji,G+1} & , \quad \text{if } randb(j) \le CR \vee j = rnbr(i) \\ x_{ji,G} & , \quad \text{else} \end{cases} \qquad \textbf{(2.11)}$$

şeklinde ifade edilir. Burada $u_{ji,G+1}$ bir sonraki jenerasyon için üretilen i deneme çözümünün j boyutundaki değeri; $v_{ji,G+1}$ bir sonraki jenerasyon için üretilen i mutant çözümünün j boyutundaki değeri; $x_{ji,G}$ mevcut jenerasyondaki i çözümünün j boyutundaki değeri; $randb(j)$ j boyutu için [0,1] aralığında üretilen rasgele düzgün dağılımlı bir sayı; CR [0,1] aralığında seçilen rekombinasyon sabiti ve $rnbr(i)$ rasgele seçilen boyut indeksidir. Bu işlem ile elde edilen deneme çözümü, belirtilen şart sağlandığında rasgele seçilmiş olan üç adet vektörün doğrusal kombinasyonu olacaktır. Aksi halde bu çözüm ebeveyn (mevcut) çözümler arasından alınacaktır. Denklem (2.11)'deki "$\vee\ j = rnbr(i)$" şartı üretilen deneme çözümlerinde en az bir parametrenin ebeveyninkinden farklı olmasını garantilemek içindir [137].

$n \ge 4$, $F \in$ (0, 1) ve $CR \in$ [0, 1] kontrol parametrelerini belirle
Denklem (2.1) aracılığıyla n adet üyeden oluşan başlangıç popülasyonunu rasgele üret ve popülasyon üyelerinin uygunluk değerlerini hesapla
repeat
    for *i = 1* ***to*** *n*
        $r_1 \ne r_2 \ne r_3 \ne i$ şartını sağlayan $r_1, r_2, r_3 \in \{1, 2, \ldots, n\}$ indekslerini rasgele seç
        Mutasyon: Denklem (2.10) aracılığı ile mutant çözümleri üret
        Rekombinasyon: Denklem (2.11) aracılığı ile deneme çözümleri üret ve uygunluk değerlerini hesapla
        Seleksiyon: Denklem (2.12) aracılığı ile deneme ve ebeveynler arasında uygunluklarına göre seçim yap
    end for
until *Durdurma kriterleri*
return *En iyi çözüm*

Şekil 2.11. DE algoritması sözde kodu

Bir sonraki adımda rekombinasyon operatörü tarafından üretilen deneme çözümlerinin uygunluk değerleri hesaplanır. Seleksiyon operatörü uygunluk değerlerini esas alarak deneme çözümleri ile ebeveyn çözümler arasında bir seçim yapar. Yapılan işlemin bir minimizasyon problemi olduğu düşünülürse seleksiyon işlemi

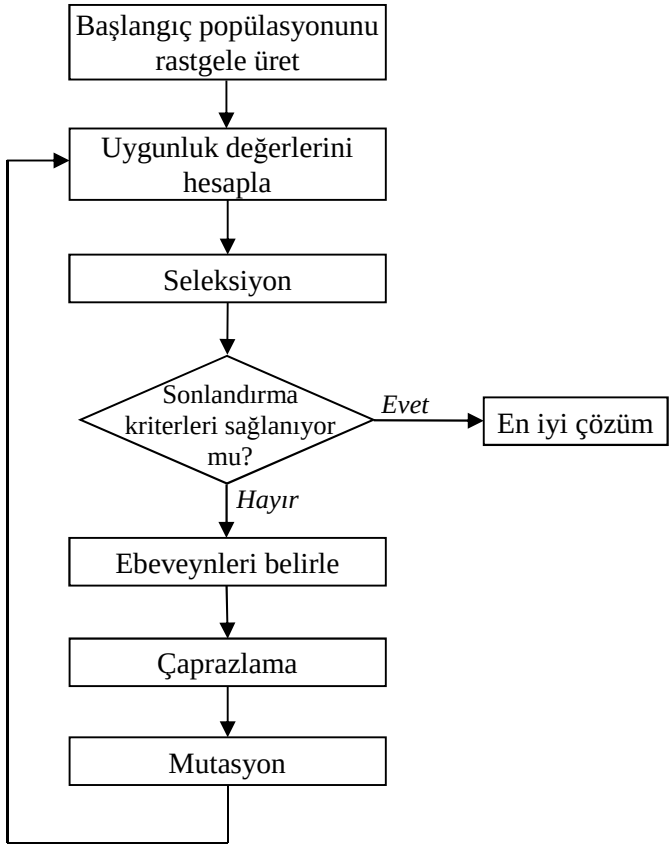
$$x_{i,G+1} = \begin{cases} u_{i,G+1} , & \text{if } f(u_{i,G+1}) \leq f(x_{i,G}) \\ x_{i,G} , & \text{else} \end{cases} \qquad \textbf{(2.12)}$$

şeklinde tanımlanabilir. Burada $x_{i,G}$, $x_{i,G+1}$ ve $u_{i,G+1}$ sırasıyla mevcut jenerasyonun i'inci çözümü, bir sonraki jenerasyonun i'inci çözümü ve bir sonraki jenerasyonun i'inci deneme çözümü ve f maliyet fonksiyonudur. Genel olarak tanıtımı yapılan DE algoritmasının sözde kodu Şekil 2.11'de verilmiştir.

2.5. Genetik Algoritma

Genetik algoritma (GA), tabiatta gözlemlenen evrimsel süreci simüle ederek çalışan bir arama ve en iyileme yöntemidir [8]. Karmaşık çok boyutlu problem uzayında en iyi çözümlerin hayatta kalması esasına göre problemlere bütünsel çözümler arar. GA'da fenotip olarak adlandırılan gerçek çözümler genotip olarak adlandırılan kromozomlar ile temsil edilirler. GA operatörleri uygulanmadan önce fenotipler genotiplere dönüştürülür. Genotiplere GA operatörleri uygulandıktan sonra elde edilen genotipler tekrar fenotiplere dönüştürülürler. Uygun bir kromozom gösteriminin belirlenmesi ve doğru bir uygunluk hesaplama yönteminin tanımlanması bir GA'nın başarılı çözümler üretmesinde gerekli olan iki başlangıç şartıdır.

Akış diyagramı Şekil 2.12'de verilen GA ilk olarak ilgili problemin çözümüne yönelik olarak tasarlanan gen yapısını kullanarak başlangıç popülasyonunu oluşturur. İkili kodlu GA'da rastgele seçilen sıfır ve birler ile başlangıç popülasyonunun kromozomları oluşturulabilirken gerçek kodlu GA'da kromozomlar sayılarla ifade edilen parametrelerden oluştuğu için Denklem (2.1) kromozomların parametrelerini oluşturmak için kullanılabilir.



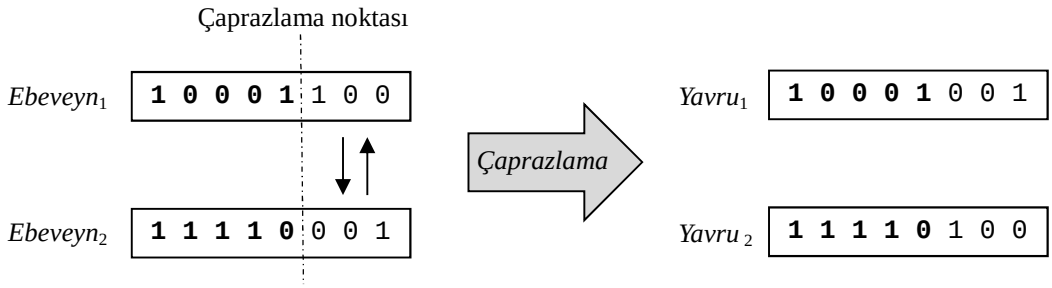
Şekil 2.12. GA akış diyagramı

Başlangıç popülasyonu oluşturulduktan sonra her bir popülasyon üyesinin uygunluk değeri hesaplanır. Eğer sonlandırma kriterlerini sağlayan kromozom mevcut değilse genetik operatörler bu kriterleri sağlayan çözüm üretilene kadar bir döngü içerisinde tekrarlı olarak kromozomlara uygulanır.

Ana döngü içerisinde ilk önce ebeveyn çiftler oluşturulur. Değişik ebeveyn seçim yöntemleri vardır. Örneğin, eşleştirme için üyelerin uygunluk değerlerinin esas alındığı bir rulet tekerleği yöntemi kullanarak popülasyon üye sayısının yarısı kadar ebeveyn çifti elde edilebilir.

Ebeveyn çiftleri oluşturulduktan sonra çaprazlama işlemine geçilir. Literatürde değişik çaprazlama yöntemleri mevcuttur. İkili kodlu GA'da yaygın olarak kullanılan çaprazlamalardan biri Şekil 2.13'te görülen tek noktalı çaprazlama işlemidir. Bu yöntemde her iki ebeveyn üzerinde aynı çaprazlama noktası rasgele seçilir. Çaprazlama noktasına kadar olan kısım her iki ebeveynde sabit tutulurken bu

noktadan sonraki kısımlar ebeveynler arasında karşılıklı değiştirilerek iki yeni birey (yavru) elde edilir.



Şekil 2.13. İkili kodlu GA için tek noktalı çaprazlama örneği

Gerçek kodlu GA'lar için literatürde değişik çaprazlama yöntemleri vardır. İkili kodlu GA'daki tek noktalı çaprazlama işlemini ustaca taklit eden ve bir ekstrapolasyon ile bir çaprazlama yönteminin kombinasyonu olan bir yöntem gerçek kodlu GA'larda sıklıkla kullanılır [143]. Bu yöntemde her bir kromozomun

$$parent = [\ p_1\ p_2 \ldots p_{N_{\text{var}}}\ ] \qquad \textbf{(2.13)}$$

formunda değişkenleri temsil ettiği varsayılır. Burada p problemin değişkenleri ve N_{var} problemdeki toplam değişken sayısıdır. Denklem (2.13) esas alınarak ebeveynler

$$parent_{\ 1} = [\ p_{m1}\ p_{m2} \ldots p_{m\alpha} \ldots p_{mN_{\text{var}}}\ ] \qquad \textbf{(2.14.a)}$$

$$parent_{\ 2} = [\ p_{d1}\ p_{d2} \ldots\ p_{d\alpha} \ldots\ p_{dN_{\text{var}}}\ ] \qquad \textbf{(2.14.b)}$$

şeklinde tanımlanabilir. Burada α [1, N_{var}] aralığında rasgele üretilmiş bir tamsayı, m ve d sırasıyla anne (mom) ve babayı (dad) temsil eden alt indislerdir. Yavru nesillerde ortaya çıkacak olan yeni değişkenler

$$p_{new1} = p_{m\alpha} - \beta\,[p_{m\alpha} - p_{d\alpha}] \qquad \textbf{(2.15.a)}$$

$$p_{new2} = p_{d\alpha} - \beta\,[p_{m\alpha} + p_{d\alpha}] \qquad \textbf{(2.15.b)}$$

şeklinde hesaplanır. Burada β [0, 1] aralığında rasgele üretilmiş düzgün dağılımlı bir sayıdır. Daha sonra yeni değişkenleri içeren yavrular üretilirken $p_{m\alpha}$ ve $p_{d\alpha}$ değişkenleri sırası ile p_{new1} ve p_{new2} değişkenleri ile değiştirilir ve bu seçim noktasının sağında kalan parametrelerin tamamı Denklem (2.16.a) ve Denklem (2.16.b)'de verildiği gibi karşılıklı olarak ebeveynler arasında yer değiştirilir.

$$offspring_{\ 1} = [\ p_{m1}\ p_{m2} \cdots p_{new1} \ \cdots \ p_{dN_{\text{var}}}\ ] \qquad \textbf{(2.16.a)}$$

$$offspring_{\ 2} = [\ p_{d1}\ p_{d2} \cdots \ p_{new2} \cdots \ p_{mN_{\text{var}}}\ ] \qquad \textbf{(2.16.b)}$$

Ana döngünün sıradaki işlemi mutasyondur. İkili kodlu GA'da önceden belirlenen bir oran kadar bit pozisyonu popülasyonun mevcut kromozomları içerisinden seçilirler ve seçilen bitlerin değerleri terslenirler (toggle). Gerçek kodlu GA'da ise yine önceden belirlenen bir oran kadar değişken pozisyonu popülasyonun mevcut kromozomları içerisinden seçilirler ve seçilen bu değişkenler Denklem (2.1) kullanılarak üretilen yeni değişkenler ile değiştirilirler. Mutasyon işlemi ile araştırma işleminin problem uzayının daha önce araştırılmamış yeni bölgelerine kaydırılması hedeflenir. Çok yüksek mutasyon oranları araştırma işlemindeki rastgeleliği artırır ve küresel minimumdan ıraksamaya neden olur. Diğer taraftan, çok düşük mutasyon oranları popülasyon içerisindeki çeşitliliğin azalmasına, popülasyon üyelerinin hızla birbirlerine benzemesine ve problem uzayının tamamının araştırılamamasına sebep olur. Genel olarak tanıtımı yapılan GA'nın sözde kodu Şekil 2.14'de verilmiştir.

Denklem (2.1) aracılığıyla n adet üyeden oluşan başlangıç popülasyonunu rasgele üret
ve popülasyon üyelerinin uygunluk değerlerini hesapla
repeat
Şekil 2.2' de verilen rulet tekerleği yöntemi ile ebeveynleri belirle
Belirlenen strateji ile çaprazlama işlemini gerçekleştir
Mutasyon işlemini gerçekleştir (son döngü hariç)
Yeni popülasyon üyelerinin uygunluk değerlerini hesapla
Seleksiyon: Uygunluk değerlerine göre yeni popülasyonun üyelerini seç
until *Durdurma kriterleri*
return *Popülasyonun en iyi üyesi*

Şekil 2.14. GA sözde kodu

BÖLÜM 3. KULLANILAN YÖNTEMLER

Bu bölümde, tez çalışması içerisinde kullanılan test fonksiyonları, yöntemler ve performans değerlendirme kriterleri gibi konulara yer verilmiştir.

3.1. Test Fonksiyonları

Test fonksiyonları bir en iyileme algoritmasının performansını ölçmek ve elde edilen performans değerini diğer algoritmaların performansları ile kıyaslamak için kullanılan önemli araçlardır. Literatürde çok sayıda test fonksiyonu vardır. Fakat algoritmaları test etmek için belirlenmiş standart bir test fonksiyonu listesi şu ana kadar henüz oluşturulmamıştır. Dolayısıyla, test çalışmaları için seçilen test fonksiyonları farklı farklı özelliklere sahip olurlar ise yapılan test işlemi çok yönlü, daha güvenilir ve tarafsız olacaktır [144].

Test fonksiyonları, yerel minimum ihtiva edip etmeme, geniş düzlüklere sahip olup olmama, vadiler içerip içermeme, ayrıştırılabilir olup olmama gibi kriterlere göre çeşitli kategorilere ayrılabilirler [145-147]. Bir en iyileme algoritması bu kategorilerden birinde oldukça başarılı sonuçlar verirken diğer bir kategorideki performansı vasatın altında kalabilir. Örneğin keşif (exploration) yeteneği iyi olan algoritmalar yerel minimumlar ihtiva eden (çok modlu) test fonksiyonlarının küresel minimumlarına kolayca yakınsarken, keşif yeteneği vasat olan algoritmalar bu test fonksiyonlarının yerel minimumlarına takılabilirler ve küresel minimuma yakınsayamayabilirler.

Bu bölümde daha önce araştırma stratejileri anlatılan GA, ABC, MBO, PSO ve DE algoritmalarının test fonksiyonları kullanılarak performans testleri yapılmış ve elde edilen sonuçlar mukayeseli olarak sunulmuştur. Yapılan performans testlerinde literatürde sıkça kullanılan on adet, değişik özelliklere sahip test fonksiyonu

kullanılmıştır. Bu test fonksiyonları ve özellikleri takip eden alt bölümlerde verilmiştir.

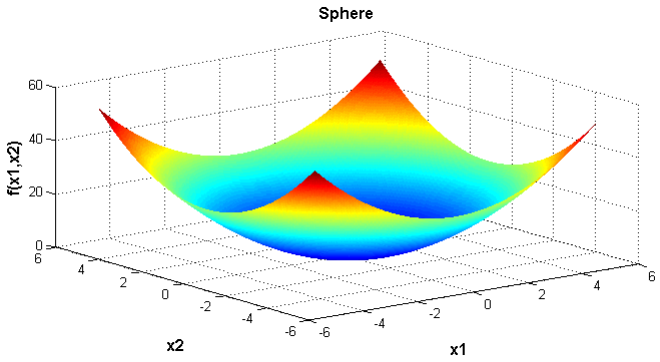
3.1.1. Sphere fonksiyonu

Sphere fonksiyonu en iyileme algoritmalarını test etmede kullanılan en basit test fonksiyonlarından biridir [148-149]. Fonksiyon sürekli, konveks ve tek modlu olup, $[-5.12, 5.12]^n$ aralığında tanımlı n boyutlu bir Sphere fonksiyonu ve bu fonksiyonun küresel minimumu sırasıyla

$$f_1(x) = \sum_{i=1}^{n} x_i^2 \qquad \textbf{(3.1.a)}$$

$$\min\left(f_1(x)\right) = 0 \ , \quad \left\{ x_i = 0 \mid i = 1,\ldots,n \right\} \qquad \textbf{(3.1.b)}$$

olarak tanımlanır.



Şekil 3.1. İki boyutlu Sphere fonksiyonu

Sphere fonksiyonunun $[-5.12, 5.12]^2$ aralığında elde edilen görüntüsü Şekil 3.1'de verilmiştir. Fonksiyonun küresel minimumu (0, 0) noktasında 0 olarak elde edilir.

3.1.2. Rastrigin fonksiyonu

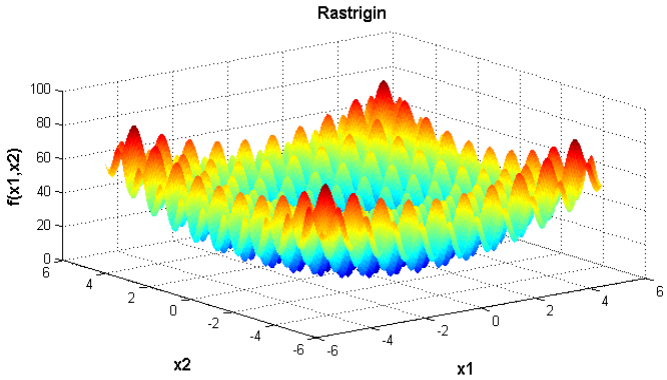
Rastrigin fonksiyonu, Sphere fonksiyonu üzerinde sık aralıklarla yerel minimumlar elde etmek maksadıyla Sphere fonksiyonuna kosinüs modülasyonu ilave edilerek

elde edilir [150]. Fonksiyon sürekli ve çok modlu olup, $[-5.12, 5.12]^n$ aralığında n boyutlu bir Rastrigin fonksiyonu ve bu fonksiyonun küresel minimumu sırasıyla

$$f_2(x) = 10n + \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) \right] \qquad \textbf{(3.2.a)}$$

$$\min\left(f_2(x)\right) = 0 \ , \quad \left\{ x_i = 0 \mid i = 1,\ldots,n \right\} \qquad \textbf{(3.2.b)}$$

olarak tanımlanır.



Şekil 3.2. İki boyutlu Rastrigin fonksiyonu

Rastrigin fonksiyonunun $[-5.12, 5.12]^2$ aralığında elde edilen görüntüsü Şekil 3.2'de verilmiştir. Fonksiyonun küresel minimumu (0, 0) noktasında 0 olarak elde edilir.

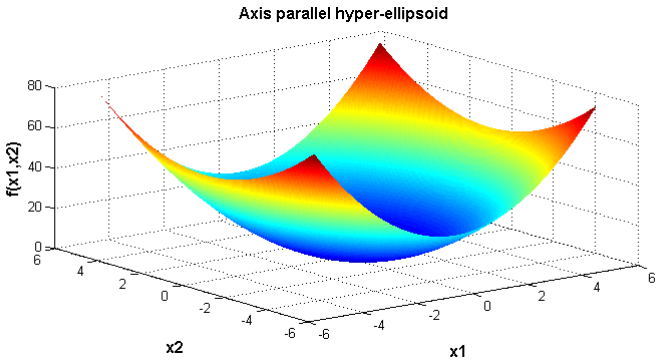
3.1.3. Axis parallel hyper ellipsoid fonksiyonu

Axis Parallel Hyper Ellipsoid (APHE) fonksiyonu görsel olarak Sphere fonksiyonuna benzer. Ağırlıklandırılmış Sphere fonksiyonu olarak da bilinir. Fonksiyon sürekli, konveks ve tek modlu olup, $[-5.12, 5.12]^n$ aralığında n boyutlu bir APHE fonksiyonu ve bu fonksiyonun küresel minimumu sırasıyla

$$f_3(x) = \sum_{i=1}^{n} (i \cdot x_i^2) \qquad \textbf{(3.3.a)}$$

$$\min(f_3(x)) = 0 \ , \quad \{ x_i = 0 \mid i = 1, \ldots, n \} \qquad \textbf{(3.3.b)}$$

olarak tanımlanır.



Şekil 3.3. İki boyutlu APHE fonksiyonu

APHE fonksiyonunun $[-5.12, 5.12]^2$ aralığında elde edilen görüntüsü Şekil 3.3'te verilmiştir. Fonksiyonun küresel minimumu (0, 0) noktasında 0 olarak elde edilir.

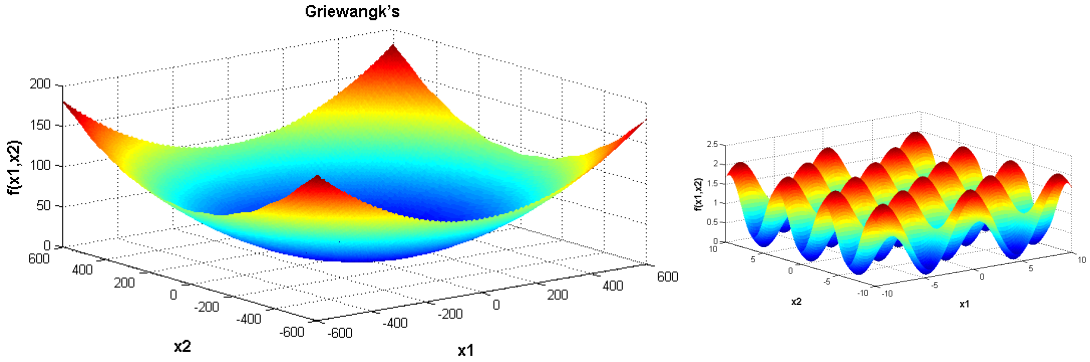
3.1.4. Griewangk fonksiyonu

Geniş yayılım alanı ve çok sayıda düzgün dağılımlı yerel minimumları ile Griewangk fonksiyonu Rastrigin fonksiyonuna benzemektedir. Fonksiyon sürekli ve çok modlu olup, $[-600, 600]^n$ aralığında n boyutlu bir Griewangk fonksiyonu ve bu fonksiyonun küresel minimumu sırasıyla

$$f_4(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1 \qquad \textbf{(3.4.a)}$$

$$\min(f_4(x)) = 0 \ , \quad \{ x_i = 0 \mid i = 1, \ldots, n \} \qquad \textbf{(3.4.b)}$$

olarak tanımlanır [151].



(a) (b)

Şekil 3.4. (a) $[-600, 600]^2$ aralığında Griewangk fonksiyonu ve (b) $[-10, 10]^2$ aralığında iki boyutlu (yakınlaştırılmış) Griewangk fonksiyonu

Griewangk fonksiyonunun $[-600, 600]^2$ aralığında elde edilen görüntüsü Şekil 3.4.a'da verilmiştir. Fonksiyonun küresel minimumu (0, 0) noktasında 0 olarak elde edilir. Fonksiyonun üzerinde ihtiva ettiği yerel minimumları görünür hale getirmek için $[-10, 10]^2$ aralığındaki parçası yakınlaştırılarak Şekil 3.4.b'de tekrar verilmiştir.

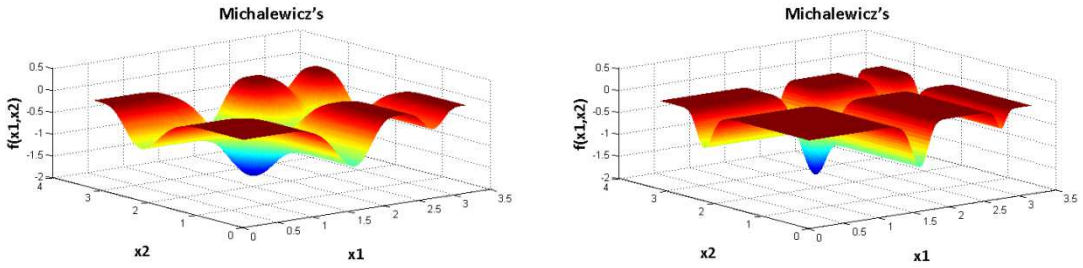
3.1.5. Michalewicz fonksiyonu

Sürekli ve çok modlu olan $[0, \pi]^n$ aralığındaki n boyutlu bir Michalewicz fonksiyonu ve bu fonksiyonun küresel minimumu sırasıyla

$$f_5(x) = -\sum_{i=1}^{n} \sin(x_i)\left[\sin\left(\frac{i \cdot x_i^2}{\pi}\right)\right]^{2m} \qquad \textbf{(3.5.a)}$$

$$\min\left(f_5(x)\right) = \begin{cases} -1.801\,,\ n=2 \ \text{ ve } \ x=(2.2029, 1.5708) \\ -4.687\,,\ n=5 \ \text{ ve } \ x=(2.2029, 1.5708, 1.2850, 1.9231, 1.7205) \\ \vdots \qquad\qquad\qquad\qquad \vdots \end{cases} \qquad \textbf{(3.5.b)}$$

olarak tanımlanır [152]. Fonksiyon verilen aralık içerisinde $n!$ adet yerel minimuma sahiptir ve görünüş itibarı ile vadiler içerir. Fonksiyondaki m parametresi vadi kenarlarının dikliğini ve keskinliğini belirler. Büyük m değerleri bu keskinliği artırdığından en iyileme algoritmalarının araştırmalarını oldukça zorlaştırır.



(a) (b)

Şekil 3.5. (a) m = 2 ve (b) m = 10 için iki boyutlu Michalewicz fonksiyonu

Michalewicz fonksiyonunun $[0, \pi]^2$ aralığında elde edilen görüntüleri Şekil 3.5'te verilmiştir. Fonksiyonun küresel minimumu (2.2029, 1.5708) noktasında -1.801 olarak elde edilir. Şekil 3.5.a'da m = 2 olarak kullanılmış, böylece oluşan vadilerin keskinliğinin daha yumuşak olması ve düzlük alanların problem uzayının küçük bir kısmını kaplaması sağlanmıştır. Şekil 3.5.b'de ise m = 10 olarak kullanılmış, bu sayede oluşan vadilerin keskinliğinin daha sert olması ve düzlük alanların problem uzayının daha büyük bir kısmını kaplaması sağlanmıştır.

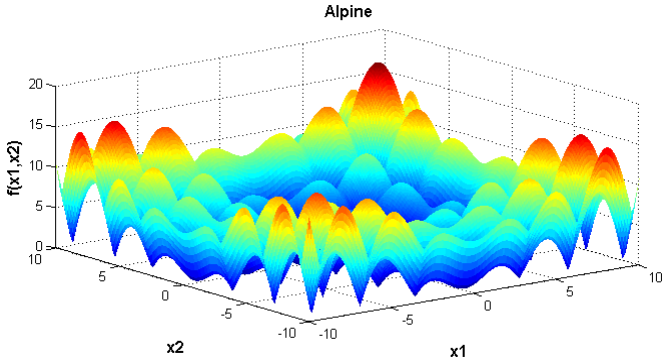
3.1.6. Alpine fonksiyonu

Sürekli ve çok modlu olan, $[-10, 10]^n$ aralığında n boyutlu bir Alpine fonksiyonu ve bu fonksiyonun küresel minimumu sırasıyla

$$f_6(x) = \sum_{i=1}^{n} \left| x_i \sin(x_i) + 0.1 x_i \right| \qquad \textbf{(3.6.a)}$$

$$\min(f_6(x)) = 0 \ , \quad \{ x_i = 0 \mid i = 1, \ldots, n \} \qquad \textbf{(3.6.b)}$$

olarak tanımlanır [153]. Fonksiyon Rastrigin fonksiyonunda olduğu gibi çok sayıda yerel minimum içerir. Rastrigin fonksiyonundaki yerel minimumlar simetrik olarak yerleşmiş olmasına rağmen Alpine fonksiyonunda içerdiği çarpanlar nedeni ile bu tarz bir simetri yoktur.



Şekil 3.6. İki boyutlu Alpine fonksiyonu

Rastrigin fonksiyonunun $[-10, 10]^2$ aralığında elde edilen görüntüsü Şekil 3.6'da verilmiştir. Fonksiyonun küresel minimumu (0, 0) noktasında 0 olarak elde edilir.

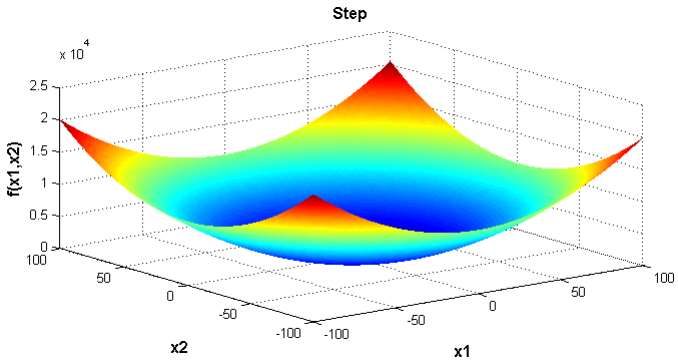
3.1.7. Step fonksiyonu

Süreksiz ve tek modlu olan, $[-100, 100]^n$ aralığında *n* boyutlu bir Step fonksiyonu ve bu fonksiyonun küresel minimumu sırasıyla

$$f_7(x) = \sum_{i=1}^{n} \left(\lfloor x_i + 0.5 \rfloor\right)^2 \qquad \textbf{(3.7.a)}$$

$$\min\left(f_7(x)\right) = 0 \ , \quad \left\{ x_i = -0.5 \mid i = 1,\ldots,n \right\} \qquad \textbf{(3.7.b)}$$

olarak tanımlanır [74].



Şekil 3.7. İki boyutlu Step fonksiyonu

Step fonksiyonunun $[-100, 100]^2$ aralığında elde edilen görüntüsü Şekil 3.7'de verilmiştir. Fonksiyonun küresel minimumu (-0.5, -0.5) noktasında 0 olarak elde edilir.

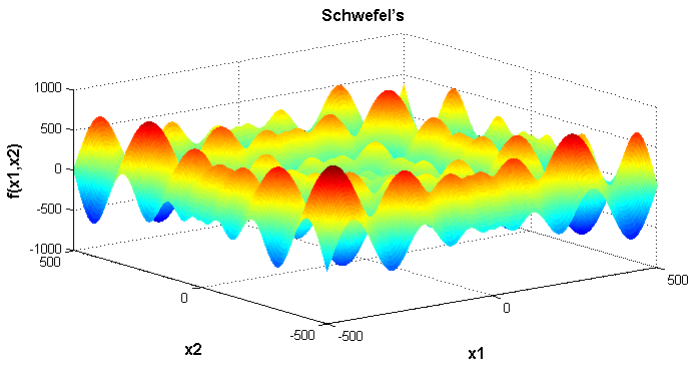
3.1.8. Schwefel fonksiyonu

Sürekli ve çok modlu olan, $[-500, 500]^n$ aralığında n boyutlu bir Schwefel fonksiyonu ve bu fonksiyonun küresel minimumu sırasıyla

$$f_8(x) = \sum_{i=1}^{n} \left[- x_i \sin\left(\sqrt{|x_i|}\right)\right] \qquad \textbf{(3.8.a)}$$

$$\min\left(f_8(x)\right) = -418.9829n \ , \quad \left\{ x_i = -420.9687 \mid i = 1,\ldots,n \right\} \qquad \textbf{(3.8.b)}$$

olarak tanımlanır [154]. Fonksiyonun küresel minimumu en iyi lokal minimumdan uzak bir koordinatta yer almaktadır. Bu aldatıcı özelliği ile Schwefel fonksiyonu en iyileme algoritmalarının sıklıkla yanılgıya düşmesine ve küresel minimum yerine lokal minimumlardan birisine yönelmesine sebep olur.



Şekil 3.8. İki boyutlu Schwefel fonksiyonu

Schwefel fonksiyonunun $[-500, 500]^2$ aralığında elde edilen görüntüsü Şekil 3.8'de verilmiştir. Fonksiyonun küresel minimumu (-420.9687, -420.9687) noktasında - 418.9829 olarak elde edilir.

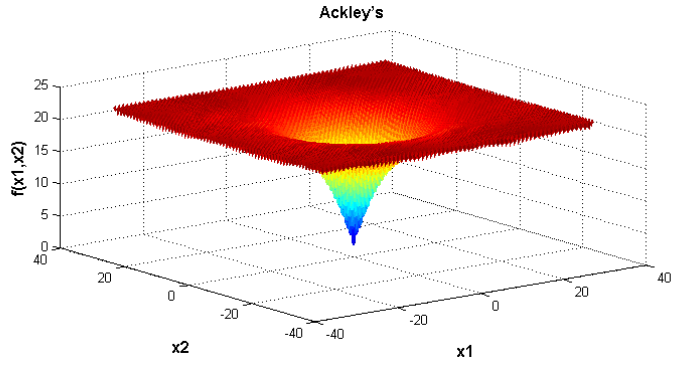
3.1.9. Ackley fonksiyonu

Sürekli ve çok modlu olan, $[-32.768, 32.768]^n$ aralığında n boyutlu bir Ackley fonksiyonu ve bu fonksiyonun küresel minimumu sırasıyla

$$f_9(x) = -a\exp\left(-b\cdot\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(cx_i)\right) + a + \exp(1) \qquad \textbf{(3.9.a)}$$

$$\min(f_9(x)) = 0\ ,\quad \{x_i = 0 \mid i = 1,\ldots,n\} \qquad \textbf{(3.9.b)}$$

olarak tanımlanır [155].



Şekil 3.9. İki boyutlu Ackley fonksiyonu

Bu fonksiyon için önerilen parametreler a = 20, b = 0.2 ve c = 2π şeklindedir. Bu parametrelerin kullanıldığı bir Ackley fonksiyonunun $[-32.768, 32.768]^2$ aralığında elde edilen görüntüsü Şekil 3.9'da verilmiştir. Fonksiyonun küresel minimumu (0, 0) noktasında 0 olarak elde edilir.

Fonksiyonda kullanılan a, b ve c parametreleri Ackley fonksiyonunun kontrol parametreleridir ve her biri oluşan fonksiyon görüntüsünün ayrı bir özelliğini kontrol eder. Şekil 3.9'daki iki değişkenli Ackley fonksiyonu görüntüsü esas alınırsa, kullanılan parametrelerden a parametresi oluşan yerel minimumların derinliklerini, b parametresi şeklin orta kısımda oluşan konik bölgenin genişliğini ve c parametresi fonksiyon yüzeyinde oluşan bölgesel dalgalanmaları kontrol eder.

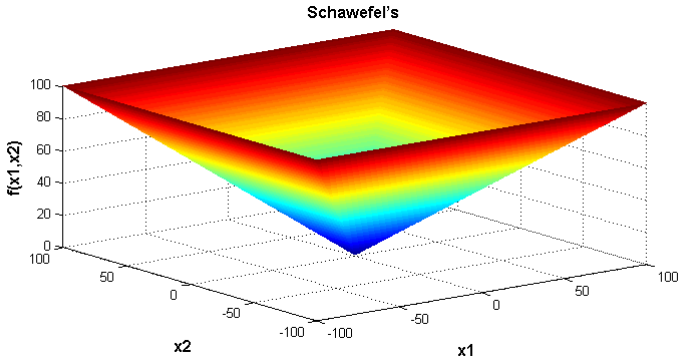
3.1.10. Schawefel fonksiyonu

Sürekli ve tek modlu olan, [-100, 100]^n aralığında n boyutlu bir Schawefel fonksiyonu ve bu fonksiyonun küresel minimumu sırasıyla

$$f_{10}(x) = \max\left\{ |x_i| \ , \ 1 \le i \le n \right\} \qquad \textbf{(3.10.a)}$$

$$\min\left(f_{10}(x)\right) = 0 \ , \quad \left\{ x_i = 0 \mid i = 1, \ldots, n \right\} \qquad \textbf{(3.10.b)}$$

olarak tanımlanır. Kare şeklindeki koniye benzeyen fonksiyon tek modlu olmasına rağmen, köşe ayrıtlarındaki fonksiyonun ani yön değişimleri en iyileme algoritmalarını zorlamakta, özellikle artan problem boyutlarında algoritmaların küresel minimuma yakınsamalarını güçleştirmektedir.



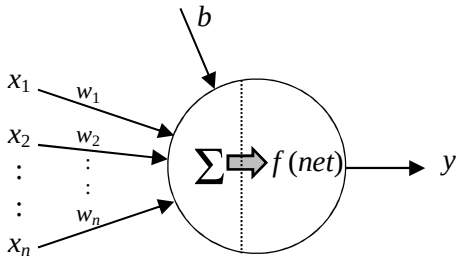
Şekil 3.10. İki boyutlu Schawefel fonksiyonu

Schawefel fonksiyonunun [-100, 100]^2 aralığında elde edilen görüntüsü Şekil 3.10'da verilmiştir. Fonksiyonun küresel minimumu (0, 0) noktasında 0 olarak elde edilir.

3.2. Yapay Sinir Ağları ve Yapay Sinir Ağı Eğitimi

Bir yapay sinir ağı (YSA) nöron adı verilen birbirine bağlı bir dizi işlem elemanından oluşur [156]. Bir YSA nöronunun şematik gösterimi Şekil 3.11'de verilmiştir. Bu nöronlar en genel gösterimi ile Şekil 3.12'dekine benzer bir topoloji ile n girişli m gizli katman nöronlu ve k çıkışlı bir YSA oluştururlar. Bu topolojide her bir nörona bir önceki katmanda bulunan nöronların çıkış sinyalleri ağırlık katsayıları ile çarpılarak gelir. Nöron bu ağırlıklandırılmış sinyalleri ve kendi eşik değerini toplar.

Toplam neticesini kullandığı transfer fonksiyonuna uygular. Transfer fonksiyonu çıkışını bir sonraki katmanın nöronlarına gönderir.

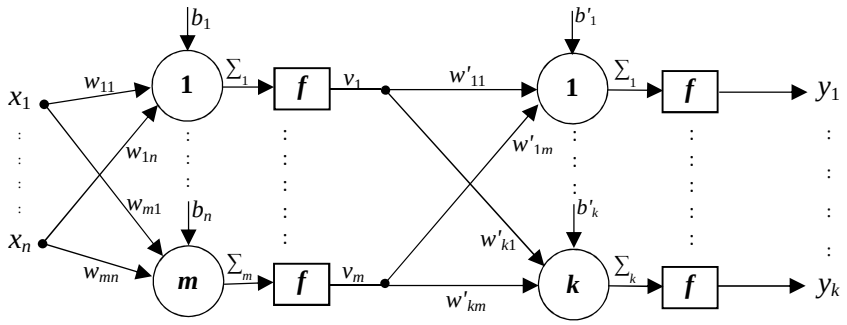


Şekil 3.11. Bir YSA nöronu

Bir YSA'nın giriş sinyalleri ile *i*'inci çıkış sinyali arasındaki ilişki

$$y_i = f_i\left(\sum_{j=1}^{n} w_{ij}x_j + b_i\right) \qquad \textbf{(3.11)}$$

olarak tanımlanır. Burada y_i *i*'inci çıkış nöronunun değeri, x_j *j*'inci giriş değeri, w_{ij} *i*'inci nöron ile *j*'inci girişi arasındaki bağlantının ağırlık değeri, b_i *i*'inci nöronunun eşik değeri ve f_i kullanılan transfer fonksiyonudur.



Şekil 3.12. *n* – *m* – *k* topolojisinde bir YSA

YSA kullanılarak doğrusal olmayan problemlere çözüm arandığından Denklem (3.12)' de verilen logaritmik sigmoid gibi transfer fonksiyonları kullanılır.

$$f(x) = \frac{1}{1+e^{-x}} \qquad \textbf{(3.12)}$$

Giriş sinyalleri seçilen transfer fonksiyonunun çalışma aralığı ile uyumlu olacak şekilde normalize edilirler. Hesaplanan çıkış sinyalleri üzerinde ise denormalizasyon işlemi yapılır.

Bir YSA eğitim algoritması eğitim işlemi süresince ağ çıkışında oluşan ortalama karesel hata (Mean Squared Error - MSE) değerini minimize etmeyi hedefler. Literatürde bu işlem için geliştirilen hata geri yayılım (Error Back Propagation - EBP) algoritması [157-158], Levenberg–Marquardt (LM) algoritması [159-160], ölçeklendirilmiş gradyan eşlenik (Scaled Conjugate Gradient - SCG) algoritması [161] gibi algoritmalar mevcuttur.

Eğitim seti esas alındığında, eğitim tamamlandığında erişilen MSE değeri ne kadar düşük olursa ağ çıkışından alınan çıkış değeri de o kadar yüksek doğrulukta olur. Eğitimi devam eden bir YSA'nın *t* iterasyonundaki MSE değeri

$$MSE(t) = \frac{1}{P}\sum_{j=1}^{P}\sum_{i=1}^{K}(d_i - y_i)^2 \qquad \textbf{(3.13)}$$

ile hesaplanır. Burada d_i ve y_i sırası ile arzulanan ve hesaplanan çıkışlar, P girişe eğitim işlemi sırasında uygulanan toplam patern sayısı ve K toplam YSA çıkış sayısıdır.

YSA ile çalışırken iki önemli aşama vardır. Bunlardan birincisi eğitim evresidir. Mevcut eğitim algoritmalarının ve ağ eğitimini bir en iyileme problemi olarak ele alan meta-sezgisel eğitim yöntemlerinin temel hedefleri eğitim süresince ağ çıkışında elde edilen MSE değerini minimize etmektir. Dolayısıyla YSA'nın eğitim aşamasının en önemli performans kriteri eğitim sonunda ulaşılan MSE değerinin mümkün olduğunca minimum seviyede gerçekleşmesidir.

İkinci evre test evresidir. Test evresinin en önemli performans kriteri ağın doğruluk oranıdır. Test aşamasında YSA'dan beklenen, doğruluk değerinin maksimum düzeyde gerçekleşmesidir. Eğer YSA bir sınıflandırma probleminin çözümü için

kullanılıyor ise ağın yaptığı sınıflandırmanın doğruluğu Denklem (3.14.a) ve Denklem (3.14.b) kullanılarak hesaplanabilir [162-163].

$$Accuracy = \frac{1}{|N|}\sum_{i=1}^{|N|} \text{assess}(n_i)\ , \quad n_i \in N \tag{3.14.a}$$

$$\text{assess}(n) = \begin{cases} 1 & \text{If classify}(n) = nc \\ 0 & \text{else} \end{cases} \tag{3.14.b}$$

Burada N test veri seti, n bu test veri setinin bir elemanı ($n \in N$), nc n elemanının gerçekte ait olduğu sınıf ve classify(n) YSA'nın n için döndürdüğü sınıf değerini hesaplayan bir fonksiyondur.

3.3. Sınıflandırmada Karar Ağacı Kullanımı

Sınıflandırma problemlerinde, karar ağaçları yaygın olarak kullanılan güçlü ve kullanımı kolay bir yöntem olarak yıllardır kullanılmaktadır [164-166]. Karar ağaçları örüntü tanıma, makine öğrenmesi, veri madenciliği vs. gibi farklı disiplinlerdeki birçok alana uygulanabilirler [167]. Karar ağaçları ile sınıflandırma işlemi, veri setleri üzerinde hızlı ve etkin bir kategorizasyon sağlar [168]. Bu yüzden tıp, ekonomi, mühendislik, hukuk vs. gibi farklı bilimlerden araştırmacılar, mevcut verileri kullanarak karar vermek için karar ağaçlarını kullanmaktadırlar [169-181].

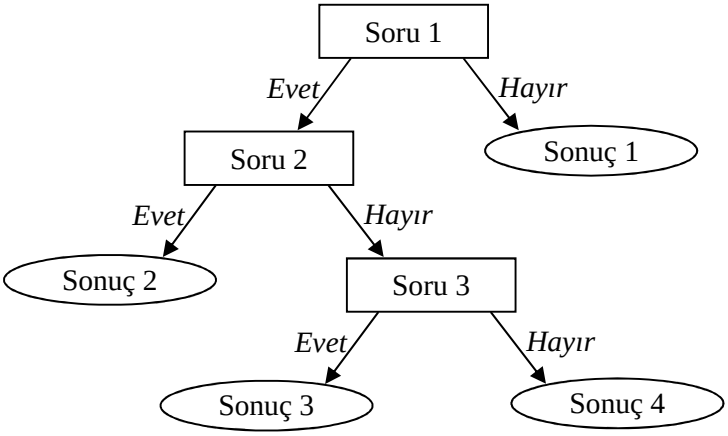
Literatürde çeşitli karar ağacı tasarım algoritmaları vardır. Bunlardan yaygın olarak bilinenleri ID3 (Iterative Dichotomiser 3) [166], C4.5 [182], sınıflandırma ve regresyon ağacı (Classification And Regression Tree – CART) [183], CHi-kare otomatik etkileşim algılama (CHi-squared Automatic Interaction Detection – CHAID) [184], çok değişkenli uyumlu regresyon uzanımları (Multivariate Adaptive Regression Splines – MARS) [185], SLIQ (Supervised Learning In Quest) [186] ve SPRINT (Scalable PaRallelizable INduction of decision Trees) [187] olarak verilebilir.

Karar ağacı, veri seti kayıtlarını art arda parçalara ayıran bir sınıflandırıcı olarak düşünülebilir. Karar ağacı tasarımında ana hedef eğitim aşamasında mevcut veriler üzerinde minimum sınıflandırma hatasını elde etmektir. Karar ağacı oluşturulurken eğitim setinin art arda parçalara ayrılması işleminde, seçilen özellikler ve bu özelliklerin değerleri elde edilen alt veri gruplarının ihtiva ettiği kayıtların tamamı aynı sınıfa ait olacak şekilde belirlenir. Bu ayırma işlemlerini gerçekleştirmek için değişik ölçütler geliştirilmiştir [188-189]. Yaygın olarak kullanılan ayırma ölçütleri entropi tabanlı kazanç oranı (Gain Ratio) [183], histogram tabanlı Gini indeks [184,187], istatistiksel anlamlılık tabanlı olabilirlik oranı (likelihood ratio) [190], saf olmama tabanlı DKM kriteri [191-192], normalize saf olmama tabanlı mesafe ölçütü [193], Twoing kriteri [184], ortogonal kriteri [194], Kolmogorov–Smirnov kriteri [195-196] ve AUC–ayırma kriteri [197] olarak verilebilir.

Bir sınıflandırma aracı olarak karar ağaçlarının çeşitli avantajları vardır. Bunlardan bazıları: kolay anlaşılır ve kolay takip edilebilir bir yapılarının olması, bir kural setine dönüştürülebilir olmaları, hem nominal hem de nümerik özelliklerle çalışabilir olmaları, parametrik olmamaları ve eksik değer içeren veri setleri ile de kullanılabilmeleridir. Diğer taraftan, karar ağaçlarının bazı dezavantajları da vardır. Bunlardan bazıları: ID3 ve C4.5 gibi çoğu karar ağacı yönteminin sadece ayrık hedef özelliklerine sahip veri setleri ile çalışabilmeleri, birbiri ile ilgili özellikleri içeren veri setlerinde iyi performans sergilemelerine rağmen özellikler arasında kompleks ilişkiler varsa performanslarının bundan kötü etkilenmesi, açgözlü (greedy) karakteristikleri nedeniyle eğitim setine, alakasız özelliklere ve gürültüye karşı aşırı duyarlılığa sebep olmalarıdır [182,198-199].

Bir karar ağacı, alt veri setleri tek bir sınıfa dahil olana kadar eğitim setini art arda daha küçük parçalara ayıran kurallar setidir [200]. Şekil 3.13'te görüldüğü gibi anlaşılır yapıda "eğer … ise o zaman …" (if then) kuralları oluşturması nedeni ile karar ağacının kullanımını anlamak oldukça kolaydır. Tipik olarak her bir düğüm noktasında bir özellik seçilir ve seçilen özelikle ilgili bir evet/hayır sorusu sorulur. Bu soru genellikle seçilen özellik için yöneltilen "…'dan büyük mü?" veya "…'dan küçük mü?" sorularıdır. Karar ağacının en üst noktasındaki giriş içermeyen düğüm kök düğüm olarak adlandırılır. Hem giriş hem de çıkışa sahip olan düğümler iç

düğümler olarak adlandırılırlar. Diğer düğümler ise yapraklar veya karar düğümleri olarak adlandırılırlar.



Şekil 3.13. Tipik bir karar ağacı

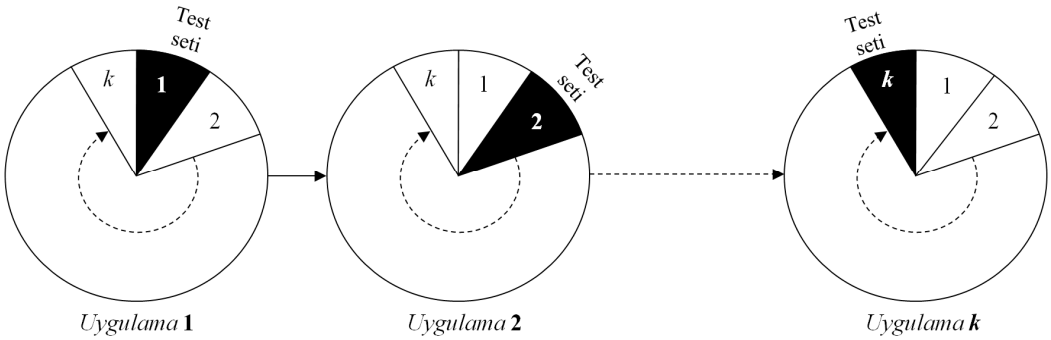
Karar ağaçlarının amacı evet/hayır sorularına alınan cevaplar doğrultusunda, kök düğümden başlayıp, iç düğümlerden geçip, yapraklara doğru ilerlerken veri setinin her bir kaydına uygun gelen sınıfı tespit etmektir. Karar ağacının inşası aşamasında ana hedef yapraklardaki sınıflandırma hatasını minimize etmektir.

3.4. K-Tekrarlı Çapraz Doğrulama Yöntemi

Bir sınıflandırıcının doğruluğu Denklem (3.14.a) ve Denklem (3.14.b) kullanılarak hesaplanabilir [162-163]. Bu doğruluk değeri tasarlanan sınıflandırıcılarda bir performans ölçütü olarak kullanılabilir.

Performans değerlendirme yapılırken eğitim seti verilerinin ve test seti verilerinin seçimi elde edilen performans değerini direkt olarak etkiler. Yapılan çalışmalarda eğitim ve test seti verilerinin içeriklerinde yapılan değişimler neticesinde performans değerlerinde olumlu veya olumsuz yönde değişimler meydana geldiği görülmüştür. Bu sebeple performans değerlendirmesini eğitim ve test setlerinin seçim şeklinden bağımsız hale getirecek bir yönteme ihtiyaç vardır. Bu yöntemlerden yaygın olarak kullanılanı *k*-tekrarlı çapraz doğrulama (*k*-fold cross validation) yöntemidir.

Şekil 3.14'te şematik olarak ifade edilmeye çalışılan *k*-tekrarlı çapraz doğrulama yöntemi test edilen bir modelin sınıflandırma işleminde ne kadar başarılı olduğunun ölçülmesinde faydalı bir tekniktir. Yaygın olarak kullanılan bu yöntemde orijinal veri setindeki örnekler eşit büyüklüklerdeki *k* adet gruba rasgele ayrılır. Her bir doğrulama uygulamasında oluşturulan *k* adet alt gruptan bir tanesi test (doğrulama) seti olarak seçilir. Geriye kalan (*k* - 1) adet alt veri grubu birleştirilerek eğitim seti olarak kullanılır. Eğitim seti kullanılarak ilgili modelin eğitimi tamamlanır ve seçilen test seti ile de modelin doğruluk oranı tespit edilir. Bu işlem her defasında *k* adet alt gruptan farklı bir tanesi test seti olarak seçilmek suretiyle *k* defa tekrarlanır. Gerçekleştirilen *k* adet uygulamadan elde edilen doğruluk değerlerinin ortalamaları alınarak modele ait tek bir doğruluk değeri hesaplanır.



Şekil 3.14. *k*-tekrarlı çapraz doğrulama yönteminin şematik gösterimi

Bu yöntemde her bir veri (*k* - 1) kez eğitim seti içerisinde yer alırken mutlaka bir kez de test seti içerisine dahil edilmiş olur. Dolayısıyla veri setini eğitim seti ve test seti olarak ikiye ayırırken kullanılan yöntemin, bu işlemi yaparken uyguladığı stratejiden kaynaklanan olumsuz etkiler minimize edilmiş olur. Yani bazı verilerin dengesiz oranlarda eğitim seti içerisinde yer alması veya test setine dahil edilmesi gibi olumsuzlukların önüne geçilmiş olur.

BÖLÜM 4. GÜNCEL EN İYİLEME ALGORİTMALARININ PERFORMANS ANALİZLERİ, UYGULAMALARI, PARALEL VE BİRLİKTE KULLANIMLARI

Bu bölümde güncel en iyileme algoritmaları arasından seçilen GA, PSO, DE, ABC ve MBO algoritmalarının performans testlerine, seçilen bu algoritmalar ile çözülen en iyileme problemlerine, algoritmaların paralel ve birlikte kullanımlarına ve bunların modifiye edilmiş versiyonlarına yer verilmiştir.

4.1. Meta-Sezgisel Algoritmalar Üzerinde Performans Testleri

Bu uygulamada, tez çalışması içerisindeki implementasyonlarda kullanılan GA, PSO, DE, ABC ve MBO algoritmalarının Bölüm 3.1'de tanıtılan test fonksiyonları ile performans testleri yapılmış ve elde edilen sonuçların mukayeseleri yapılmıştır.

4.1.1. Test prosedürü

Bölüm 3.1'in alt başlıklarında özelliklerinden bahsedilen 10 adet test fonksiyonunun her birinin 2, 5, 10, 30 ve 50 boyutlu versiyonları yapılan testlerde kullanılmıştır. İlk önce Denklem (2.1) kullanılarak 100 adet farklı popülasyon üretilmiş ve üretilen bu popülasyonlar testlerde kullanılmak üzere dosyalara kaydedilmiştir. Algoritmalar test edilirken dosyalara kaydedilen bu popülasyonlar kullanılarak algoritmaların aynı başlangıç popülasyonlarını kullanmaları, böylece algoritmaların performanslarının birbiri ile daha adil mukayese edilmesi hedeflenmiştir.

İşlemin olasılıksal yönü göz önünde tutularak algoritmaların sadece bir kez işletiminden elde edilen sonuçlar kullanılmamış, her bir algoritma her defasında farklı bir popülasyonu kaydedilen dosyalardan alarak toplamda 100'er defa çalıştırılmıştır. Elde edilen sonuçlardan en iyi uygunluk (en düşük hata) değerine

sahip olan 50 işletim sonucu algoritmaların ulaştığı en başarılı sonuçlar olarak kabul edilmiş, bunların ortalamasından elde edilen sonuçlar Tablo 4.1'den Tablo 4.5'e kadar olan tablolarda test sonuçları olarak verilmiştir.

Bu deneysel çalışmanın amacı algoritmaların seçilen test fonksiyonları üzerindeki ortalama performanslarını elde etmek ve birbirleri ile mukayese etmektir. Testlerde kullanılan kritik algoritma parametreleri test sonuçlarının yer aldığı tablolarda verilmiştir. MBO algoritması seçilen parametreler nedeni ile her iterasyonda diğer algoritmaların ürettiğinin iki katı komşu çözüm ürettiğinden diğer algoritmaların iki misli uygunluk hesaplaması yapmaktadır. En iyileme süresince toplamda eşit miktarda uygunluk hesaplamasının gerçekleştirilmesi ve böylece yapılan mukayeselerin daha tutarlı ve adil olması için MBO algoritmasının toplam iterasyon sayısı diğer algoritmalarınkinin yarısı kadar tutulmuştur.

4.1.2. Test Sonuçları

Fonksiyon testi sonuçları Tablo 4.1'den Tablo 4.5'e kadar olan tablolarda verilmiştir. Fonksiyonların 2, 5, 10, 30 ve 50 boyutlu test fonksiyonlarının en iyilenmesinde eriştikleri küresel minimum değerleri test prosedüründe anlatıldığı gibi deneysel sonuçlardan hesap yolu ile elde edilmiştir. Sonuçlar arasında öne çıkan bazıları şunlardır.

ABC algoritması, iyi keşif yeteneğini kullanarak genellikle makul en iyileme sonuçları vermektedir. Bunda, algoritmanın kâşif arı evresinde gerçekleşen küresel araştırmanın büyük payı vardır. Bu sayede, bütün test fonksiyonlarının her boyutunda tatmin edici sonuçlar sunmaktadır. Fakat küresel minimuma yakınsama miktarı yeterince iyi değildir. Örneğin MBO ve DE algoritmaları küresel minimuma mümkün olduğunca yakınsarlarken ABC algoritması aynı yakınsama performansını gösterememektedir. Dolayısıyla algoritmanın yerel araştırma yeteneğinin biraz daha iyileştirilmesi ile algoritmanın başarısının daha da artacağı açıkça görülmektedir.

PSO algoritması her boyutta tek modlu test fonksiyonları için makul sonuçlar üretmiştir. Güçlü yerel arama yeteneği sayesinde tek modlu test fonksiyonlarının

küresel minimumlarına mümkün olduğunca yakınsama gerçekleştirmiştir. Fakat algoritmanın çok modlu test fonksiyonlardaki başarısı tek modlu test fonksiyonlarında olduğu kadar iyi değildir. Özellikle yüksek dereceli çok modlu fonksiyonlardaki sonuçlar daha kötü olarak sonuçlanmıştır. Bu kötü sonuçlar algoritmanın keşif yeteneğinin zayıflığından ve başlangıç popülasyonunun kalitesine bağımlılığından kaynaklanmaktadır. Dolayısıyla algoritmaya keşif yeteneğini artıran ve onu başlangıç popülasyonunun kalitesine bağımlı kalmaktan kurtaran yeni stratejiler ilave edilmesi algoritmanın başarısını biraz daha artırabilecektir.

Tablo 4.1. İki boyutlu test fonksiyonları için algoritmaların eriştiği küresel minimum değerlerinin ortalamaları (*K*: toplam iterasyon, *n*: popülasyon büyüklüğü, *tnfc*: toplam uygunluk hesaplama sayısı)

F	**Gerçek Küresel minimum**	**MBO** *K* = 750 *n* = 15 *tnfc* = 23250	**ABC** *K* = 1500 *n* = 16 *tnfc* = 24000	**PSO** *K* = 1500 *n* = 15 *tnfc* = 22500	**DE** *K* = 1500 *n* = 15 *tnfc* = 22500	**GA** *K* = 1500 *n* = 16 *tnfc* = 24000
f_1	0	0	3.7719E-17	0	0	1.7836E-11
f_2	0	0	3.7067E-04	0	0	3.7503E-08
f_3	0	0	2.8856E-17	0	0	4.9702E-13
f_4	0	3.1104E-03	1.8944E-02	3.2543E-03	0	7.6190E-03
f_5	-1.8013	-1.8013	-1.8013	-1.8013	-1.8013	-1.8013
f_6	0	0	9.7922E-06	0	0	6.8008E-08
f_7	0	0	2.5502E-16	0	0	1.7879E-09
f_8	-8.3797E+02	-8.3797E+02	-8.3796E+02	-7.5032E+02	-8.3797E+02	-1.0440E+04
f_9	0	8.8818E-16	1.9223E-09	8.8818E-16	8.8818E-16	9.7299E-05
f_{10}	0	0	1.8925E-02	0	0	5.6174E-03

Tablo 4.2. Beş boyutlu test fonksiyonları için algoritmaların eriştiği küresel minimum değerlerinin ortalamaları (*K*: toplam iterasyon, *n*: popülasyon büyüklüğü, *tnfc*: toplam uygunluk hesaplama sayısı)

F	**Gerçek Küresel minimum**	**MBO** *K* = 1500 *n* = 31 *tnfc* = 94500	**ABC** *K* = 3000 *n* = 32 *tnfc* = 96000	**PSO** *K* = 3000 *n* = 31 *tnfc* = 93000	**DE** *K* = 3000 *n* = 31 *tnfc* = 93000	**GA** *K* = 3000 *n* = 32 *tnfc* = 96000
f_1	0	0	6.5052E-17	0	0	1.0206E-07
f_2	0	0	0	1.3133	0	1.2427E-05
f_3	0	0	6.9817E-17	0	0	1.2859E-07
f_4	0	1.2911E-02	1.7489E-07	7.7363E-02	0	1.8813E-02
f_5	-4.6877	-4.6877	-4.6877	-4.5642	-4.6877	-4.6877
f_6	0	0	1.6728E-16	1.1463E-16	0	6.5403E-05
f_7	0	0	6.2974E-17	0	0	3.3636E-05
f_8	-2.0949E+03	-2.0949E+03	-2.0949E+03	-1.5765E+03	-2.0949E+03	-1.3374E+04
f_9	0	3.4461E-15	4.2277E-15	4.2277E-15	8.8818E-16	4.3284E-03
f_{10}	0	0	1.3134E-08	0	4.5152E-84	5.0126E-02

Tablo 4.3. On boyutlu test fonksiyonları için algoritmaların eriştiği küresel minimum değerlerinin ortalamaları (*K*: toplam iterasyon, *n*: popülasyon büyüklüğü, *tnfc*: toplam uygunluk hesaplama sayısı)

F	**Gerçek Küresel minimum**	**MBO** *K* = 2500 *n* = 41 *tnfc* = 207500	**ABC** *K* = 5000 *n* = 42 *tnfc* = 210000	**PSO** *K* = 5000 *n* = 41 *tnfc* = 205000	**DE** *K* = 5000 *n* = 41 *tnfc* = 205000	**GA** *K* = 5000 *n* = 42 *tnfc* = 210000
f_1	0	0	8.2649E-17	0	0	7.2550E-07
f_2	0	0	0	5.6315	0	1.2293E-04
f_3	0	0	8.8269E-17	0	0	3.7363E-06
f_4	0	2.6046E-02	8.1595E-14	3.3098E-01	0	2.9947E-02
f_5	-9.6600	-9.6359	-9.6602	-8.5504	-9.6602	-9.6601
f_6	0	3.5527E-17	2.4721E-16	4.4720E-16	0	2.1913E-04
f_7	0	0	8.6227E-17	0	0	2.7967E-04
f_8	-4.1898E+03	-4.1849E+03	-4.1898E+03	-2.6354E+03	-4.1898E+03	-1.9546E+04
f_9	0	5.9330E-15	7.9226E-15	6.9988E-15	3.3040E-15	7.3117E-03
f_{10}	0	1.8978E-38	5.3854E-14	7.8663E-86	5.6296E-10	1.0803E-01

Tablo 4.4. Otuz boyutlu test fonksiyonları için algoritmaların eriştiği küresel minimum değerlerinin ortalamaları (*K*: toplam iterasyon, *n*: popülasyon büyüklüğü, *tnfc*: toplam uygunluk hesaplama sayısı)

F	**Gerçek Küresel minimum**	**MBO** *K* = 3500 *n* = 101 *tnfc* = 710500	**ABC** *K* = 7000 *n* = 102 *tnfc* = 714000	**PSO** *K* = 7000 *n* = 101 *tnfc* = 707000	**DE** *K* = 7000 *n* = 101 *tnfc* = 707000	**GA** *K* = 7000 *n* = 102 *tnfc* = 714000
f_1	0	0	4.9986E-16	0	0	6.8423E-06
f_2	0	0	0	3.0247E+01	0	1.2403E-03
f_3	0	0	4.3213E-16	0	0	9.0918E-05
f_4	0	0	7.5495E-17	2.4649E-03	0	8.6388E-03
f_5	-2.9583E+01	-2.9566E+01	-2.9631E+01	-2.3646E+01	-2.5930E+01	-2.9628E+01
f_6	0	7.9495E-16	5.7107E-16	1.1644E-12	5.1108E-60	1.0217E-03
f_7	0	0	4.5768E-16	7.3956E-34	0	2.4184E-03
f_8	-1.2570E+04	-1.2555E+04	-1.2569E+04	-6.5279E+03	-1.2569E+04	-6.0910E+04
f_9	0	2.1849E-14	3.0589E-14	3.2365E-14	4.4409E-15	1.1875E-02
f_{10}	0	2.9023E-10	5.5636E-02	1.1495E-06	1.7504E-02	4.0999E-01

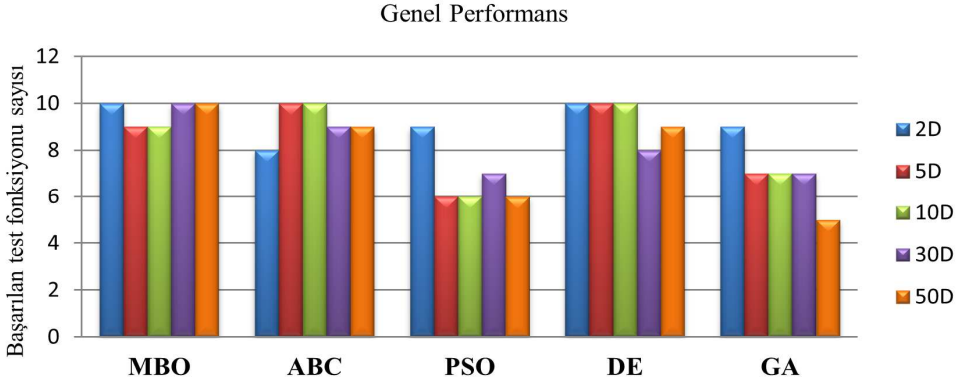
Problem boyutu arttıkça GA'nın küresel minimuma yakınsama miktarının azaldığı gözlemlenmektedir. Buradan GA performansı ile problem boyutunun ters orantılı olduğu gibi bir sonuç çıkarılmaktadır. GA'nın yerel araştırma yeteneğinin orta seviyelerde olması sebebi ile küresel minimuma arzu edilen hata oranı ile yakınsaması için daha fazla iterasyon gerekmektedir. Bu ise doğal olarak artan işlem maliyeti sonucunu doğuracaktır.

Tablo 4.5. Elli boyutlu test fonksiyonları için algoritmaların eriştiği küresel minimum değerlerinin ortalamaları (*K*: toplam iterasyon, *n*: popülasyon büyüklüğü, *tnfc*: toplam uygunluk hesaplama sayısı)

F	**Gerçek Küresel minimum**	**MBO** *K* = 4000 *n* = 151 *tnfc* = 1212000	**ABC** *K* = 8000 *n* = 152 *tnfc* = 1216000	**PSO** *K* = 8000 *n* = 151 *tnfc* = 1208000	**DE** *K* = 8000 *n* = 151 *tnfc* = 1208000	**GA** *K* = 8000 *n* = 152 *tnfc* = 1216000
f_1	0	0	9.0469E-16	3.5142E-38	0	9.0755E-05
f_2	0	0	0	5.9837E+01	0	1.5338E-02
f_3	0	0	8.2402E-16	1.3864E-36	0	1.9948E-03
f_4	0	0	9.1038E-17	1.6431E-14	0	4.2609E-02
f_5	-4.9513E+01	-4.9544E+01	-4.9624E+01	-3.8896E+01	-3.2365E+01	-4.9590E+01
f_6	0	1.9429E-15	2.1666E-15	1.0866E-07	6.9538E-03	4.0222E-03
f_7	0	0	8.2391E-16	4.1723E-32	0	3.3912E-02
f_8	-2.0949E+04	-2.0933E+04	-2.0949E+04	-9.9439E+03	-2.0949E+04	-1.0752E+05
f_9	0	3.7623E-14	5.5316E-14	9.9224E-13	7.7094E-15	3.4396E-02
f_{10}	0	2.8918E-04	4.2017E-01	4.2010E-02	3.3912E-05	1.5517

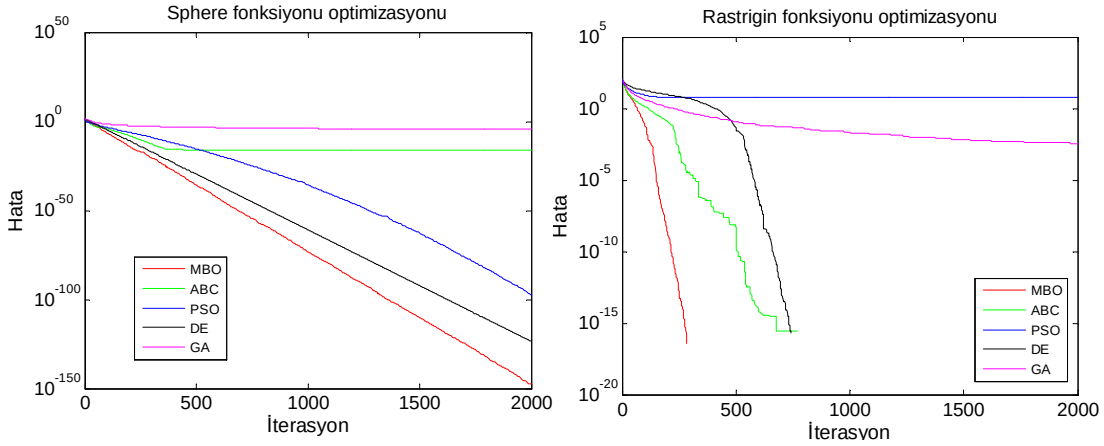
DE algoritması bütün problem boyutlarında makul sonuçlar üretmektedir. Bu başarının kaynağı, algoritmanın keşif ve yerel arama yetenekleri arasında kurulmuş olan güçlü dengedir.

MBO algoritması güçlü yerel arama yeteneği sayesinde küresel minimumlara yakınsamada iyi performans sergilemektedir. Algoritmanın her bir döngüde yapacağı toplam işlem sayısı *k* ve *x* parametrelerine bağlıdır. Örneğin, *k* parametresinin artırılması küresel minimuma yakınsama performansını artıracak, fakat bu durum işlem maliyetini de artıracaktır. Bu durumda algoritma sonuçlarının diğer algoritmalarınkilerle kıyaslanması ise adil olmayacağından, testlerdeki MBO algoritma parametreleri toplamda diğer algoritmalarınkine yakın miktarda işlem gerçekleştirecek şekilde ayarlanarak testler yapılmış ve sonuçlar öyle elde edilmiştir.



Şekil 4.1. Algoritmaların test fonksiyonu başarılarının problem boyutlarına göre dağılımları

Şekil 4.1 her bir algoritmanın toplam test fonksiyonu başarılarının problem boyutlarına göre dağılımlarını vermektedir. Bu kriter esas alındığında, seçilen parametreler ile MBO algoritmasının en iyi performansı sergilediği görülmektedir. MBO algoritması, 10 adet test fonksiyonu içerisinden 5 ve 10 boyutlu olanların 9'ar adedine ve 2, 30 ve 50 boyutlu olanların ise tamamına en iyi çözümler bulmuştur. Performans yönünden onu ABC ve DE algoritmaları takip etmektedir.



(a) (b)

Şekil 4.2. (a) 10 Boyutlu Sphere fonksiyonunun ve (b) 10 boyutlu Rastrigin fonksiyonunun küresel minimumlarını ararken algoritmaların ortalama hata-iterasyon ilerlemeleri

Bir algoritmanın küresel minimuma ne kadar hızlı ve kararlı bir şekilde yakınsadığı da üzerinde durulması gereken bir diğer önemli konudur. Şekil 4.2.a ve Şekil 4.2.b'de sırasıyla biri tek modlu diğeri ise çok modlu olan, 10 Boyutlu Sphere fonksiyonunun ve 10 boyutlu Rastrigin fonksiyonunun küresel minimumlarını ararken algoritmaların elde ettiği ortalama hata-iterasyon grafikleri verilmiştir. Sphere fonksiyonu testinde, GA ve ABC algoritmasının belirli bir hata değerine eriştikten sonra yakınsamayı bıraktıkları görülmektedir. Bu onların yerel arama yeteneklerinin yeterince güçlü olmamasından kaynaklanmaktadır. MBO, PSO ve DE algoritmaları ise güçlü yerel arama yetenekleri sayesinde duraksamadan küresel minimuma yakınsamaya devam etmektedirler. Rastrigin fonksiyonu testinde ise, GA ve PSO algoritmaları arzulanan çözümleri üretememişlerdir. Bu durum onların keşif yeteneklerinin yetersizliğinden kaynaklanmaktadır. MBO, ABC ve DE algoritmaları ise GA ve PSO algoritmalarına göre daha güçlü olan keşif yetenekleri sayesinde maksimum iterasyon limitine varmadan gerçek küresel minimuma ulaşmaktadırlar.

MBO algoritması ise her iki fonksiyon için makul hata değerlerine diğerlerinden daha önce erişmektedir.

4.2. Sistem Kimliklendirme Sürecine Meta-Sezgisel Yaklaşım

Sistem kimliklendirme işlemi bilinmeyen bir sistemin davranışlarını inceleyebilmek ve anlayabilmek için önemlidir. Bu işlem için geliştirilen yaklaşımlar gerçek sistem ile onun matematiksel modeli arasında bir arayüz kurmayı amaçlarlar. Geleneksel sistem kimliklendirme yaklaşımları modelleme yaparken bilinmeyen sistemin giriş ve çıkış sinyallerini kullanırlar. Bu sinyaller ilgili sistem çalışırken toplanan gerçek çalışma verileri olabileceği gibi deneysel olarak elde edilen sisteme ait giriş çıkış verileri de olabilir.

Literatürde, Oto-regresif dışsal girişli (Auto-Regressive with eXogenous inputs – ARX) model [201], çıkış hatası (Output Error – OE) modeli [202], Box-Jenkins (BJ) modeli [203] ve saf derece tekrarlı merdiven algoritma (Pure Order Recursive Ladder Algorithm – PORLA) [204-205] gibi değişik sistem kimliklendirme teknikleri vardır. Günümüzde ise bazı meta-sezgisel yöntemler sistem kimliklendirme maksadıyla kullanılmaya başlanmış, bunlardan ümit verici ve rekabetçi sonuçlar alınmıştır [206-210].

Sistem kimliklendirme metodolojisi gerçek deneysel verileri kullanarak bilinmeyen bir sistemin modelini inşa eder ve inşa ettiği modelin parametrelerini ayarlar. Kara-kutu modelleme olarak tabir edilen metotlar sistem girişlerini ve onlara ait çıkışları kullanırlar. Sistemin ne olduğu, nerede ve ne için kullanıldığı veya nasıl göründüğü gibi detaylarla ilgilenmezler [209]. Bu modellerin temel hedefleri, bilinmeyen sistem ile model çıkışları arasındaki hata arzulanan bir değere düşürülene kadar parametre ayarlaması yaparak tasarlanan transfer fonksiyonu katsayılarının nihai değerlerini elde etmektir. Sistem kimliklendirme sürecinde genellikle eğime (gradient) dayalı geleneksel yöntemler kullanılırlar. Bu yöntemler yumuşak geçişli ve türevi alınabilir araştırma uzayları üzerinde uygulanabildiklerinden türevi alınamayan çok modlu hata fonksiyonları üzerinde kullanılamazlar [210]. Bu matematiksel gerçek araştırmacıları yeni sistem kimliklendirme yöntemleri aramaya sevk etmiştir.

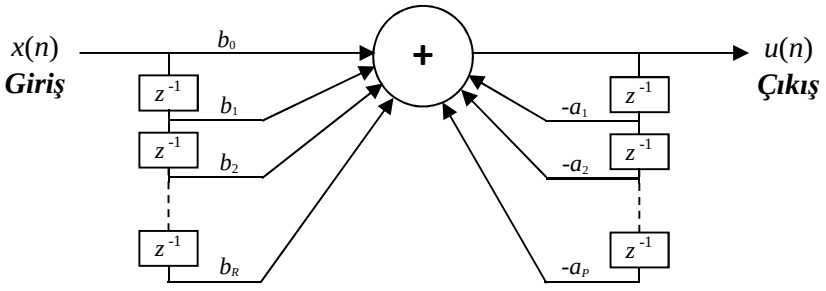
Meta-sezgisel algoritmalar problem karakteristiklerinden bağımsız olarak çalışan sistematik araştırma teknikleri olmaları nedeni ile sistem kimliklendirme problemlerine uygulanabilir niteliktedirler. Giriş ve çıkışı sırası ile $x(n)$ ve $u(n)$ ile ifade edilen bilinmeyen bir sistemin giriş-çıkış ilişkisi

$$u(n)+\sum_{i=1}^{P} a_i \cdot u(n-i)=\sum_{i=0}^{R} b_i \cdot x(n-i) \qquad \textbf{(4.1)}$$

şeklinde ifade edilir. Burada $x(n - i)$ ve $u(n - i)$ giriş ve çıkışa ait daha önceki gözlem değerleri, a_i ve b_i ise bu gözlem değerlerinin katsayılarıdır. Oto regresif hareketli ortalama (Auto Regressive Moving Average - ARMA) gösterimi Şekil 4.3'te verilen bu sistemin transfer fonksiyonu

$$H(z)=\frac{B(z)}{A(z)}=\frac{\sum_{i=0}^{R} b_i \cdot z^{-i}}{1+\sum_{i=1}^{P} a_i \cdot z^{-i}} \qquad \textbf{(4.2)}$$

olarak tanımlanır.

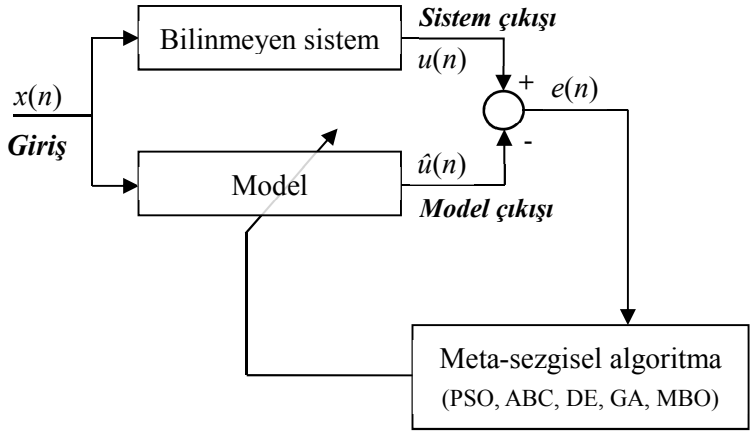


Şekil 4.3. Bilinmeyen bir sistemin ARMA model gösterimi

Denklem (4.2) ile ifade edilen, bilinmeyen bir sistemin transfer fonksiyonunun katsayıları

$$[b_0, b_1, \ldots, b_R, a_1, a_2, \ldots, a_P] \qquad \textbf{(4.3)}$$

olarak vektör formunda ifade edilebilir. Sistem kimliklendirme işlemi popülasyon tabanlı bir algoritmanın kullanıldığı bir en iyileme problemi olarak düşünülürse, popülasyonun her bir üyesi Denklem (4.3)'te verilen formda transfer fonksiyonu katsayılarını ihtiva eder. Meta-sezgisel en iyileme algoritmaları sistematik teknikler kullanarak Şekil 4.4'te gösterilen $e(n)$ hata miktarını minimize etmeye çalışırlar. Kullanılan meta-sezgisel algoritmalar her iterasyonda, transfer fonksiyonu katsayılarını içeren yeni popülasyon üyeleri üretirler. Bu üyelerin temsil ettiği transfer fonksiyonları için hesaplanan MSE değerleri ise ilgili üyelerin maliyet değerleri olarak kullanılırlar. Problemin bir minimizasyon problemi olduğu göz önünde bulundurulursa MSE değeri daha düşük olan popülasyon üyesinin uygunluğunun daha yüksek olduğu kabul edilir.

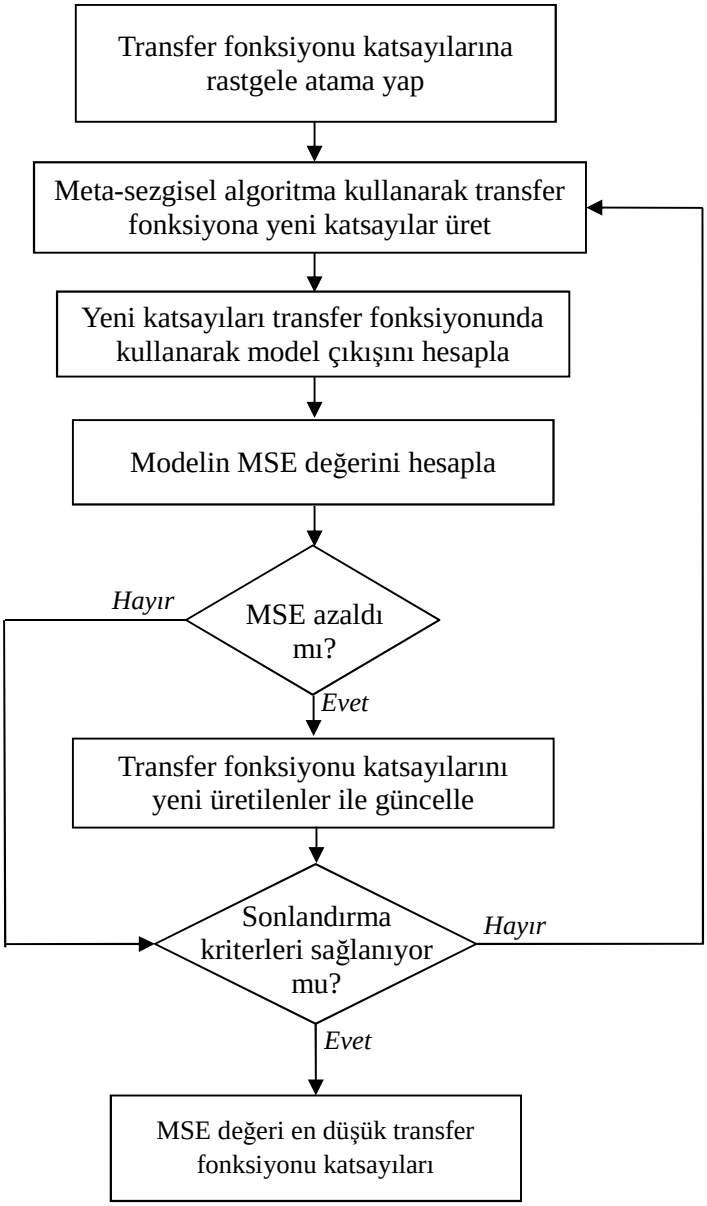


Şekil 4.4. Meta-sezgisel algoritma kullanılan bir sistem kimliklendirme işlemi

Modellenen sisteme ait MSE değeri

$$E(t) = \frac{1}{N}\sum_{i=1}^{N}\left(u(i) - \hat{u}(i)\right)^2 \qquad \textbf{(4.4)}$$

olarak tanımlanır. Burada $u(i)$ ve $\hat{u}(i)$ sırasıyla i'inci giriş için sistemin arzulanan gerçek çıkışı ve modellenen sistemin çıkışı, N sistem kimliklendirme işleminde kullanılan toplam giriş paterni sayısı ve $E(t)$ t'inci iterasyonda sistemin ulaştığı MSE değeridir. Bir meta-sezgisel en iyileme algoritmasının kullanıldığı sistem kimliklendirme süreci Şekil 4.5'te gösterildiği gibi önceden tanımlanan sonlandırma kriterleri sağlanana kadar devam eder.



Şekil 4.5. Meta-sezgisel en iyileme algoritması kullanılan bir sistem kimliklendirme yönteminin akış diyagramı

Özet olarak, meta-sezgisel en iyileme algoritması her iterasyonda sistematik olarak vektör formundaki mevcut transfer fonksiyonu katsayılarını kullanarak yeni transfer fonksiyonu katsayılarını yine vektör formunda üretir. Daha sonra MSE değerlerine bakarak mevcut ve üretilen katsayı vektörleri arasında uygunluğu yüksek olanı bir sonraki nesilde kullanmak üzere seçer. İterasyonlar ilerledikçe bu yolla üretilen transfer fonksiyonu katsayılarının uygunlukları artar. Sonunda arzulanan hata değerine ulaşıldığında elde edilen transfer fonksiyonu sistemin tahmin edilen modeli olur.

4.2.1. Testlerde kullanılan sistem kimliklendirme problemleri

Bu deneysel çalışmada meta-sezgisel en iyileme algoritmalarının sistem kimliklendirme performansları farklı derecelerdeki 5 adet sistem kullanılarak test edilmiştir. Sistemlere önce $x(n)$ giriş sinyali olarak beyaz gürültü uygulanıp sistemlerin $u(n)$ gerçek çıkışları hesap yolu ile elde edilmiş, elde edilen bu $x(n)$ ve $u(n)$ verileri kullanılarak seçilen sistemler üzerinde sistem kimliklendirme denemeleri yapılmıştır. Deneysel çalışmada kullanılan 2, 3, 4, 5 ve 8'inci dereceden sistemlere ait transfer fonksiyonları Denklem (4.5.a)'dan Denklem (4.5.e)'ye kadar olan denklemlerde verilmiştir [211].

$$H_1(z) = \frac{2 + 3z^{-1} + 4z^{-2}}{1 - 0.8z^{-1} + 0.15z^{-2}} \quad \textbf{(4.5.a)}$$

$$H_2(z) = \frac{1 - 1.4z^{-1} - 1.71z^{-2} + 2.34z^{-3}}{1 - 0.5z^{-1} - 0.29z^{-2} + 0.105z^{-3}} \quad \textbf{(4.5.b)}$$

$$H_3(z) = \frac{2 - 1.5z^{-1} + 4.1z^{-2} + 6.8z^{-3} - 2z^{-4}}{1 - 0.2z^{-1} - 0.55z^{-2} + 0.116z^{-3} + 0.042z^{-4}} \quad \textbf{(4.5.c)}$$

$$H_4(z) = \frac{1.41 - 2.0404z^{-1} - 0.1739z^{-2} + 1.0973z^{-3} - 0.2595z^{-4} - 0.0339z^{-5}}{1 - 0.443z^{-1} - 1.067z^{-2} + 0.43z^{-3} + 0.187z^{-4} - 0.045z^{-5}} \quad \textbf{(4.5.d)}$$

$$H_5(z) = \frac{0.01 - 0.041z^{-2} + 0.061z^{-4} - 0.041z^{-6} + 0.01z^{-8}}{\begin{pmatrix} 1 - 2.472z^{-1} + 4.309z^{-2} - 4.886z^{-3} + 4.477z^{-4} \\ - 2.914z^{-5} + 1.519z^{-6} - 0.5z^{-7} + 0.12z^{-8} \end{pmatrix}} \quad \textbf{(4.5.e)}$$

4.2.2. Meta-sezgisel sistem kimliklendirmede izlenen yöntem

Bilinmeyen bir sistemin giriş ve çıkışlarının $x(n)$ ve $u(n)$ oldukları ve bu sistemin transfer fonksiyonu katsayılarını temsil eden değerlerin Denklem (4.3) formatında oluşturduğu üyelerden meydana gelen bir popülasyonun kullanıldığı kabullenilirse, meta-sezgisel en iyileme algoritmaları bu transfer fonksiyonu katsayılarını optimize

etmek için kullanılabilirler. Temel felsefesi MSE'yi minimize etme üzerine kurulu olan meta-sezgisel en iyileme algoritmaları ile sistem kimliklendirme işleminin olasılıksal yönü göz önünde tutularak algoritmaların sadece bir kez işletiminden elde edilen sonuçları kullanmak yerine, yapılan çoklu denemelerin sonuçlarının ortalamalarının kullanıldığı bir yöntem tercih edilmiştir. Bu maksatla ilk önce Denklem (2.1) kullanılarak her bir transfer fonksiyonu için 100'er adet farklı popülasyon üretilmiş ve üretilen bu popülasyonlar testlerde kullanılmak üzere dosyalara kaydedilmişlerdir. Bütün algoritmalar test edilirken dosyalara kaydedilen bu popülasyonlar kullanılarak algoritmaların aynı başlangıç popülasyonu üzerindeki performansı izlenerek daha adil mukayeseler yapılması hedeflenmiştir. Her bir algoritma, her bir transfer fonksiyonu için her işletimde farklı bir popülasyonu kaydedilen dosyalardan alarak toplamda 100'er defa çalıştırılmıştır. Elde edilen sonuçlardan en küçük MSE değerine sahip olan 50 işletim sonucu algoritmaların ulaştığı en başarılı sonuçlar olarak kabul edilmiş, bunların ortalamasından elde edilen sonuçlar test sonuçları olarak Tablo 4.6 ve Tablo 4.7'de verilmiştir. Algoritmalara birbirlerine yakın sayıda uygunluk hesaplaması yaptırarak daha anlamlı karşılaştırmalar yapılması hedeflenmiş ve bunu sağlayacak şekilde ayarlanan temel algoritma parametreleri Tablo 4.6 ve Tablo 4.7'de belirtilmiştir.

4.2.3. Meta-sezgisel sistem kimliklendirme sonuçları

Önceki bölümde bahsedilen prosedüre uygun olarak gerçekleştirilen sistem kimliklendirme deneysel çalışmasının H_1, H_2 ve H_3 sistemleri için ortalama sonuçları Tablo 4.6'da, H_4 ve H_5 sistemleri için ortalama sonuçları ise Tablo 4.7'de verilmiştir. Sonuçlar arasından öne çıkan bulgular şunlardır.

GA sistem kimliklendirme yönteminin sonuçları genel olarak en yüksek MSE değerlerine sahiptir. Yöntem, karakteristiği gereği MSE'yi daha fazla düşürebilmek için daha fazla iterasyona ihtiyaç duymaktadır. Bu durum toplam işlem süresini artıran bir dezavantajdır.

ABC sistem kimliklendirme yönteminin sonuçları genellikle makul seviyelerdedir. Düşük dereceli sistemlerde elde edilen transfer fonksiyonu katsayıları gerçeklerine

oldukça yakın, yüksek dereceli sistemlerde ise modellere ait MSE değerleri yeterince düşüktür. Buna rağmen, sonuçlar yöntemler içerisindeki en iyi sonuçlar değildir.

Tablo 4.6. Düşük dereceli H_1, H_2 ve H_3 sistemlerinin kimliklendirme sonuçları (K: maksimum iterasyon, n: popülasyon büyüklüğü, *tnfc*: toplam uygunluk hesaplama sayısı)

	Parametreler	**Bilinmeyen sistem**	**MBO**	**ABC**	**PSO**	**DE**	**GA**
			K = 1000 n = 31 *ntfc* = 63000	K = 2000 n = 30 *ntfc* = 60000			
H_1	b_0	2.00000	2.00000	2.00170	2.00000	2.00000	2.00440
	b_1	3.00000	3.00050	2.99410	3.00000	3.00000	3.05200
	b_2	4.00000	4.00070	3.98600	4.00000	4.00000	4.07440
	a_1	-0.80000	-0.79982	-0.80235	-0.80000	-0.80000	-0.78049
	a_2	0.15000	0.14985	0.15169	0.15000	0.15000	0.13414
	MSE		4.682E-05	1.344E-03	3.752E-31	0	2.229E-02
			K = 2000 n = 39 *ntfc* = 158000	K = 4000 n = 38 *ntfc* = 152000			
H_2	b_0	1.00000	1.00000	1.00000	1.04660	0.99987	0.99691
	b_1	-1.40000	-1.40000	-1.40000	-1.35940	-1.39960	-1.39370
	b_2	-1.71000	-1.71000	-1.71000	-1.88400	-1.71000	-1.71060
	b_3	2.34000	2.34000	2.34000	2.35870	2.33930	2.32860
	a_1	-0.50000	-0.50000	-0.50000	-0.47877	-0.49975	-0.49537
	a_2	-0.29000	-0.29000	-0.29000	-0.30218	-0.28988	-0.28749
	a_3	0.10500	0.10500	0.10500	0.10968	0.10499	0.10506
	MSE		3.209E-32	8.779E-12	1.238E-02	3.821E-07	4.510E-05
			K = 3000 n = 51 *ntfc* = 309000	K = 6000 n = 50 *ntfc* = 300000			
H_3	b_0	2.00000	2.00120	2.00070	2.00160	2.00320	2.02550
	b_1	-1.50000	-1.43210	-1.49450	-1.41220	-1.41490	-1.26280
	b_2	4.10000	4.06680	4.09970	4.05700	4.06280	4.02140
	b_3	6.80000	6.93280	6.79720	6.97180	6.94220	7.10720
	b_4	-2.00000	-1.72960	-1.98340	-1.65040	-1.66270	-0.99449
	a_1	-0.20000	-0.16594	-0.19833	-0.15596	-0.15881	-0.08655
	a_2	-0.55000	-0.54791	-0.54948	-0.54731	-0.54633	-0.52747
	a_3	0.11600	0.09825	0.11446	0.09305	0.09264	0.03892
	a_4	0.04200	0.04217	0.04217	0.04223	0.04303	0.04672
	MSE		7.113E-05	2.825E-04	7.978E-05	2.862E-04	1.954E-02

Tablo 4.7. Yüksek dereceli H4 ve H5 sistemlerinin kimliklendirme sonuçları (*K*: maksimum iterasyon, *n*: popülasyon büyüklüğü, *tnfc*: toplam uygunluk hesaplama sayısı)

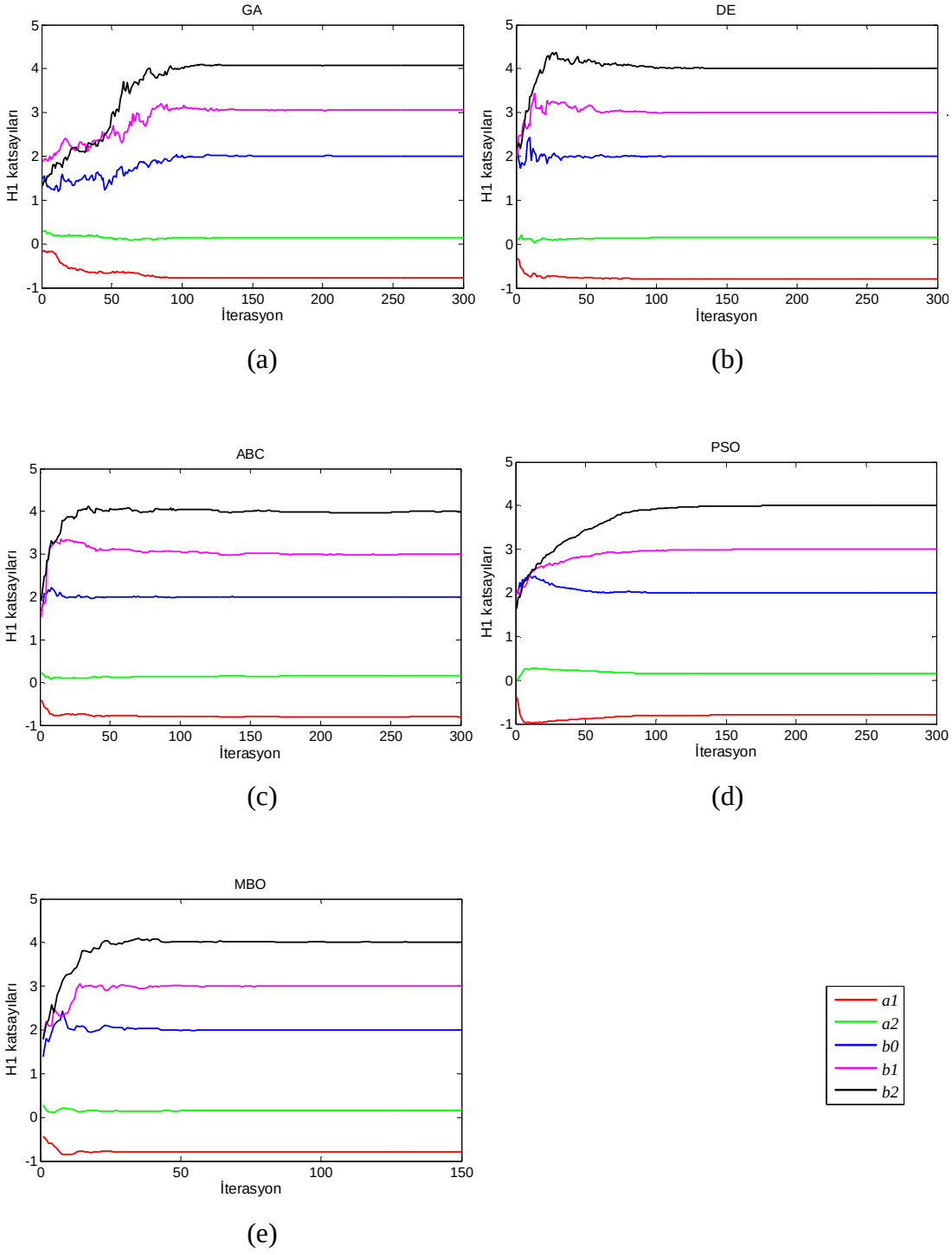
	Parametreler	**Bilinmeyen sistem**	**MBO**	**ABC**	**PSO**	**DE**	**GA**
			K = 2500 *n* = 59 *ntfc* = 297500	*K* = 5000 *n* = 58 *ntfc* = 290000			
H_4	b_0	1.41400	1.41240	1.41300	1.46880	1.41040	0.01260
	b_1	-2.04040	-1.24930	-0.98289	-0.84852	-1.19470	0.00831
	b_2	-0.17390	0.22646	0.04390	-0.22938	0.11894	-0.00263
	b_3	1.09730	0.09728	0.02168	0.10685	-0.00642	-0.07510
	b_4	-0.25590	-0.49338	-0.43686	-0.34856	-0.37424	0.00845
	b_5	-0.03390	0.15051	0.12206	0.07243	0.18511	0.18332
	a_1	-0.44300	0.11707	0.30221	0.38575	0.15586	-0.63028
	a_2	-1.06700	-0.22301	-0.16957	-0.26717	-0.25801	0.85377
	a_3	0.43000	0.28252	0.19105	0.11044	0.15311	0.32330
	a_4	0.18700	-0.26819	-0.25548	-0.17571	-0.27800	-0.16278
	a_5	-0.04500	-0.08654	-0.08784	-0.07993	-0.04360	0.47587
	MSE		2.988E-04	5.509E-04	1.784E-03	6.859E-04	3.943E-03
			K = 4000 *n* = 91 *ntfc* = 732000	*K* = 8000 *n* = 90 *ntfc* = 720000			
H_5	b_0	0.01000	0.00958	0.01273	0.00982	0.00771	0.00728
	b_1	0	0.02042	0.04974	0.01985	0.01862	0.01765
	b_2	-0.04100	-0.02847	0.06309	-0.02836	-0.02197	-0.02492
	b_3	0	-0.07883	-0.03790	-0.07672	-0.09756	-0.09398
	b_4	0.06100	-0.01118	0.01724	0.00038	-0.03538	-0.02399
	b_5	0	0.10633	0.07659	0.10181	0.15224	0.14779
	b_6	-0.04100	0.06153	0.05958	0.04461	0.13248	0.10576
	b_7	0	-0.06171	0.01078	-0.05995	-0.08742	-0.09479
	b_8	0.01000	-0.06805	-0.01882	-0.04875	-0.17069	-0.13733
	a_1	-2.47200	-0.42854	-0.16695	-0.52595	-0.15186	-0.27205
	a_2	4.30900	0.82151	0.88426	0.88877	0.31134	0.39061
	a_3	-4.88600	0.36758	-0.17745	0.38345	0.33339	0.31193
	a_4	4.47700	0.20769	0.43342	0.06808	0.13762	0.09285
	a_5	-2.91400	0.28702	-0.07887	0.46050	-0.05005	-0.01935
	a_6	1.51900	0.42684	0.22607	0.31384	0.09981	0.11957
	a_7	-0.50000	-0.12332	-0.00746	-0.08003	-0.00285	-0.02962
	a_8	0.12000	0.19351	0.14485	0.20715	-0.18008	-0.15381
	MSE		1.514E-04	4.453E-02	3.017E-05	2.306E-03	1.825E-03

DE sistem kimliklendirme yöntemi bütün test sistemleri için iyi sonuçlar üretmiştir. Özellikle düşük dereceli sistemlerin kimliklendirilmesi işlemlerinde ulaşılan MSE değerleri oldukça düşüktür. Örneğin H_1 sistemi için kısa sürede hata fonksiyonunun küresel minimumuna erişmekte ve gerçek transfer fonksiyonu katsayılarını elde etmektedir.

PSO sistem kimliklendirme yöntemi bütün test sistemleri için makul sonuçlar üretmiştir. Özellikle DE algoritmasında olduğu gibi H_1 sisteminin kimliklendirilmesinde çok iyi sonuçlar elde etmiştir. Her defasında gerçek transfer fonksiyonu katsayılarını elde edemese de bunlara çok yakın sonuçlar üretmiştir.

MBO sistem kimliklendirme yöntemi bütün test sistemleri için iyi sonuçlar elde etmiştir. Yöntem MSE'yi arzu edildiği gibi minimize etmeyi başarmış, özellikle H_1, H_2 ve H_3 sistemleri için gerçek transfer fonksiyonu katsayılarına çok yakın değerler elde etmiştir. H_4 ve H_5 sistemleri için de arzulanan düşük MSE değerlerine ulaşmıştır. Erişilen bu düşük MSE değerleri, elde edilen transfer fonksiyonu katsayılarının gerçek sisteminkinden farklı olmasına rağmen, gerçek sisteme yakın özelliklerde başka bir eşdeğer sistemin transfer fonksiyonunun katsayıları olduğunu göstermektedir.

Sistem transfer fonksiyonu derecesinin dördün üzerinde olduğu durumlarda orijinal transfer fonksiyonu katsayılarının elde edilmesi güçleşir. Bu sistemlerde transfer fonksiyonunun herhangi bir katsayısının küçük bir değişimi sistem cevabında büyük dalgalanmalara sebep olacağından, belirtilen algoritmaların geçek transfer fonksiyonu katsayılarını komşu çözümler arayarak elde etmesi oldukça güçtür. Algoritmalar bu sistemler için bire bir aynı transfer fonksiyonu katsayılarını elde edememelerine rağmen, elde ettikleri sistemler bilinmeyen orijinal sistemlerin yakın en iyi eşdeğerleridir. Gerçek sistemler ile modellenenler arasındaki eşdeğerliğin ölçüsü erişilen MSE değerinin küçüklüğüdür. Yani sistem kimliklendirme işlemi ne kadar küçük MSE değeri ile sonuçlanırsa elde edilen transfer fonksiyonunun temsil ettiği sistem gerçeğine o kadar yakın olur. Bu yaklaşımla sonuçlar değerlendirilirse, modellenen sistemler gerçeğine oldukça yakın sistemlerdir.



Şekil 4.6. H_1 sisteminin kimliklendirilmesinde (a) GA, (b) DE, (c) ABC, (d) PSO, (e) MBO algoritmalarının ortalama trendleri

Şekil 4.6'da verilen grafiklerde H_1 sisteminin kimliklendirilmesi işleminde algoritmaların ortalama trendleri verilmektedir. Bu trendler yapılan denemelerde elde edilen trendlerin ortalamaları alınarak elde edilmiştir. MBO sistem kimliklendirme yönteminin 25. iterasyon civarlarında en iyi çözüme yaklaştığı görülmektedir.

Toplam uygunluk hesaplama adedi esas alındığında bu, diğer algoritmaların yaklaşık 50. iterasyonlarına denk gelmektedir. ABC sistem kimliklendirme yönteminin haricindeki diğer yöntemler 50. iterasyonda gerçek transfer fonksiyonu katsayılarına arzulanan ölçüde yaklaşamamışlardır. Dolayısıyla MBO ve ABC algoritmaları sistem kimliklendirmede sonuca diğerlerinden daha hızlı yaklaşmışlardır.

4.3. Meta-Sezgisel YSA Eğitimi

Literatürdeki klasik YSA eğitim algoritmaları türevi alınabilir hata fonksiyonlarına ihtiyaç duyarlar. Bu yüzden türevi alınamayan hata fonksiyonlarına sahip YSA'ların eğitiminde klasik eğitim algoritmaları yerel minimumlara takılabilir. Böylesi bir durumda da makul çözümler üretemeyebilirler. Diğer taraftan popülasyon tabanlı meta-sezgisel algoritmalar araştırma uzayının karakteristiklerinden bağımsız olarak çalışabildiklerinden, YSA eğitiminde daha iyi performanslar sergilerler [212-217]. Bu algoritmalar eğitim prosesine bir en iyileme problemi olarak yaklaşırlar ve gerçek sistem hakkında detaylı bilgiye ihtiyaç duymaksızın, en iyileme işlemi yaparak YSA eğitimini gerçekleştirirler. Ayrıca bu tekniklere ilave olarak, küresel ve geleneksel araştırma tekniklerinin avantajlarını bir arada toplayan yeni, melez veya modifiye modeller de YSA eğitimi için geliştirilmiştir [7,218-221].

Şekil 3.12'dekine benzer bir YSA çok parametreli bir en iyileme algoritması olarak düşünülebilir. Bu parametreler ağırlık katsayıları ve nöron eşik değerlerinden oluşur. Bu bağlamda, tek gizli katmanı bulunan bir YSA için optimize edilmesi gereken toplam parametre sayısı

$$N_{par} = n \cdot m + m \cdot k + m + k \qquad \textbf{(4.6)}$$

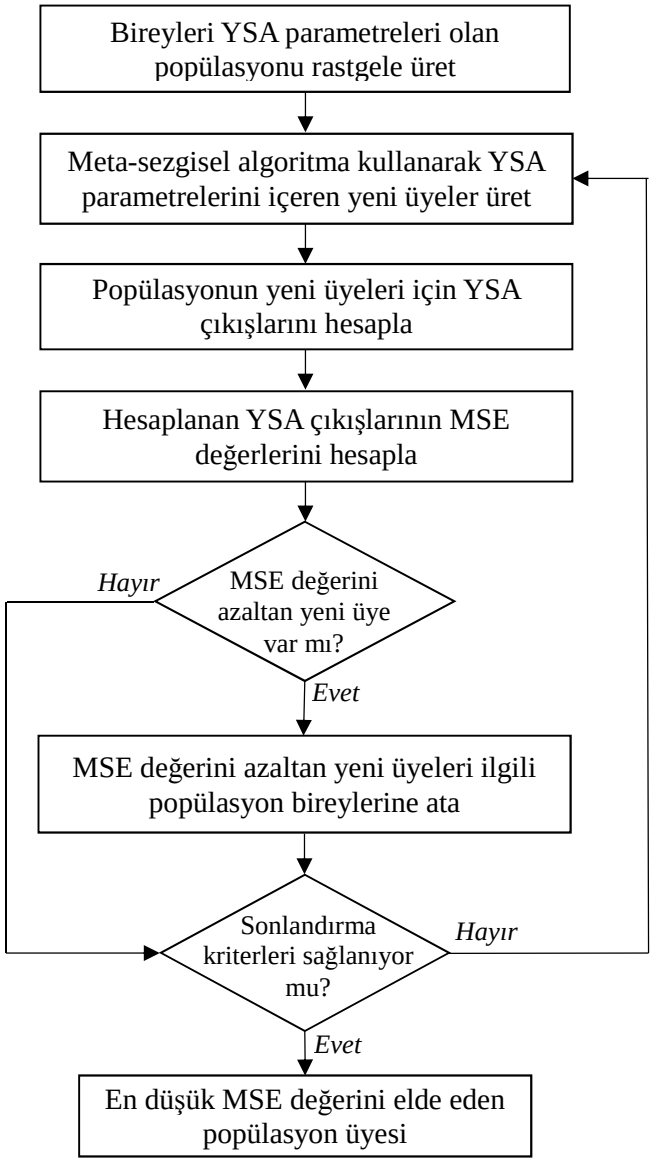
olarak hesaplanır. Burada N_{par} toplam parametre sayısı, n YSA'daki toplam giriş sayısı, m gizli katmandaki toplam nöron sayısı ve k çıkış katmanındaki toplam nöron sayısıdır. Benzer şekilde iki gizli katmanı bulunan bir YSA için optimize edilmesi gereken toplam parametre sayısı ise

$$N_{par} = n \cdot m_1 + m_1 \cdot m_2 + m_2 \cdot k + m_1 + m_2 + k \qquad \textbf{(4.7)}$$

şeklinde hesaplanır. Burada m_1 ve m_2 sırası ile birinci ve ikinci gizli katmanlardaki toplam nöron sayılarıdır. Tek gizli katmanlı bir YSA ele alındığında, bu YSA'nın toplam N_{par} adet parametresi

$$P = [w_{11}, w_{12}, \ldots, w_{1n}, b_1, \ldots\ldots, w_{m1}, w_{m2}, \ldots, w_{mn}, b_m] \quad \textbf{(4.8)}$$

vektör formunda ifade edilebilir. Meta-sezgisel YSA eğitiminde, kullanılan meta-sezgisel algoritma N_{par} adet YSA parametresini Denklem (4.8) formatında içeren popülasyon üyelerini kullanır ve N_{par} adet YSA parametresini ağa ait MSE değerini minimum yapacak şekilde optimize etmeye çalışır.



Şekil 4.7. Meta-sezgisel YSA eğitimi akış diyagramı

Meta-sezgisel en iyileme algoritması her iterasyonda yeni popülasyon üyelerini, dolaylı olarak yeni YSA parametrelerini, kendi sistematiğini kullanarak üretir. Üretilen ağ parametreleri kullanılarak YSA çıkışları hesaplanır. Hesaplanan çıkışlar için Denklem (3.13) aracılığı ile ağın MSE değeri hesaplanır. Meta-sezgisel algoritma bu MSE değerini bir geri besleme ölçütü olarak kullanır ve mevcut popülasyon üyelerinin yeni üretilenlerle güncellenip güncellenmeyeceğine karar verir. Eğer üretilen bir popülasyon üyesi MSE değerinde düşüş sağlamış ise bu mevcut popülasyon üyesi üretilen ile değiştirilir. Aksi halde mevcut popülasyon üyesi korunur. Bu işlemler sonlandırma kriterleri sağlanıncaya kadar tekrarlanır. Anlatılan bu meta-sezgisel ağ eğitiminin akış diyagramı Şekil 4.7'de verilmiştir.

4.3.1. Kullanılan veri setleri, YSA topolojileri ve test yöntemi

UCI [222] ve KEEL [223] web sitelerinden indirilen ve Tablo 4.8'de listelenen 20 adet veri seti bu bölümde sunulan sınıflandırma çalışmasında kullanılmıştır.

Tablo 4.8. Veri seti özellikleri, YSA topolojileri ve ilgili en iyileme probleminin boyutları

No	Veri Seti	Özellik (*Giriş* + *Çıkış*)	Kayıt Sayısı	YSA Topolojisi	Problem Boyutu
1	Akut inflamasyon [222,224]	6 + 2	120	6 - 10 - 2	92
2	Kan bağışı [222,225]	4 + 1	748	4 - 10 - 1	61
3	Göğüs Kanseri [222,226-227]	9 + 1	683	9 - 25 - 1	276
4	Doğurganlık [222,228]	9 + 1	100	9 - 21 - 1	232
5	Hint karaciğer hastalığı [222, 229-230]	10 + 1	579	10 - 25 - 1	301
6	Lens [222,231-232]	4 + 1	24	4 - 25 - 3	203
7	Karaciğer hastalıkları [222,233]	6 + 1	345	6 - 25 - 1	201
8	Pima yerlileri diyabeti [222,234]	8 + 1	768	8 - 25 - 1	251
9	EEG planlama/rahatlama [222,235-236]	12 + 1	182	12 - 36 - 1	505
10	Kalp SPECT [222,237-238]	22 + 1	267	22 - 30 - 1	721
11	Tiroit [163-222,239]	5 + 1	215	5 - 15 - 20 - 3	473
12	Omurga [222,240-241]	6 + 1	310	6 - 21 - 1	169
13	Apandisit [223,242]	7 + 1	106	7 - 25 - 1	226
14	Titanik [223,243]	3 + 1	2201	3 - 15 - 1	76
15	Fonem [223]	5 + 1	5404	5 – 20 – 1	141
16	Süsen çiçeği [222,244]	4 + 1	150	4 - 25 – 3	203
17	Mamografik kütle [222,245]	5 + 1	830	5 – 25 – 1	176
18	Banknot doğrulama [222]	4 + 1	1372	4 – 10 – 1	61
19	Denge ölçeği [222,246]	4 + 1	625	4 – 25 – 3	203
20	Haberman hayatta kalma [222,247-248]	3 + 1	360	3 – 20 – 1	101

Önce veri setleri yarısı eğitim seti ve diğer yarısı test seti olacak şekilde iki eşit parçaya ayrılmıştır. Daha sonra GA, DE, PSO, ABC ve MBO algoritmaları kullanılarak, topolojileri Tablo 4.8'de verilen ağlar üzerinde meta-sezgisel ağ eğitimi gerçekleştirilmiştir. Meta-sezgisel ağ eğitim teknikleri yine Tablo 4.8'de problem boyutu olarak verilen sayıda YSA parametresinin en iyilemesi için kullanılmıştır. Algoritmalarda popülasyon büyüklüğü 150 üye olarak ve maksimum iterasyon sayısı 500 olarak ayarlanmıştır. Her bir ağ eğitimi 20'şer kez tekrarlanıp en yüksek doğruluğa sahip 10 eğitimin sonuçlarının ortalamaları alınarak ağ doğruluk değerleri, MSE değerleri ve hata - iterasyon ilerleme grafikleri elde edilmiştir.

4.3.2. Deneysel sonuçlar

Deneylerden elde edilen nümerik sonuçlar Tablo 4.9'da verilmiştir. Elde edilen en yüksek doğruluk değerleri ile en düşük MSE değerleri kalın puntolarla verilmiştir. Sonuçlar aşağıdaki gibi yorumlanabilir.

LM algoritması ağ eğitim performansı oldukça iyi olan geleneksel ağ eğitim algoritmalarından birisidir. Nümerik sonuçlar analiz edildiğinde meta-sezgisel YSA eğitimlerinden elde edilen doğruluk değerlerinin eğim hesaplama tabanlı eğitim algoritması olan LM algoritması ile rekabet eder seviyede olduğu görülmektedir. Doğruluk oranlarında fark az olmasına rağmen meta-sezgisel YSA eğitimlerindeki erişilen MSE değerleri LM algoritmasına kıyasla oldukça düşük seviyelerdedir.

Özellikle MBO algoritması bu çalışmadaki bütün algoritmalar içerisinde en düşük MSE değerlerine erişmiştir. Algoritma kendine özgü çözüm paylaşım mekanizması sayesinde iyi bir yakınsama gerçekleştirmektedir. Normalde, MBO algoritması haricindeki algoritmalar her bir iterasyonda mevcut popülasyon üyelerine kendi sistematik araştırma yöntemlerini kullanarak komşuluklar üretirler ve bunların içerisinde ilgili mevcut üyelerden daha iyi olanlar mevcut üyelerin yerine kullanılırlar. Yani ilgili popülasyon üyesinin iyileşebilme şansı tamamen onun için üretilen komşu çözümün kalitesine bağlıdır. Diğer taraftan MBO algoritması mevcut bir popülasyon üyesine komşu çözümler üretirken ayrıca önceki çözümlerin paylaştığı kullanılmamış en iyi komşu çözümleri de kullanır. Yani bir popülasyon

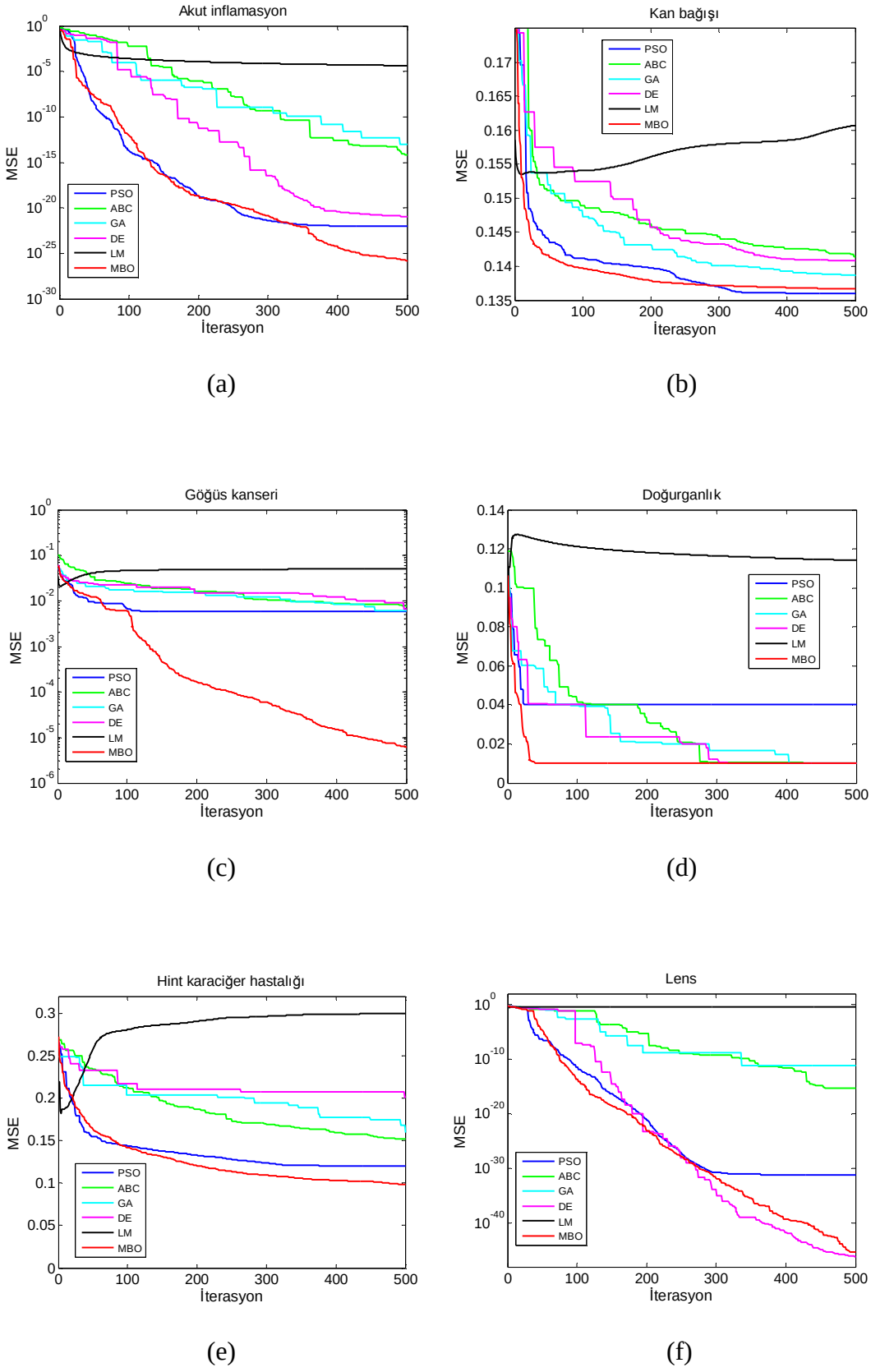
üyesinin iyileşebilme şansı sadece üretilen komşu çözümün kalitesine bağlı değildir. Kısmen üretilen komşu çözümün kalitesine kısmen de önceki çözümün paylaştığı komşu çözümün kalitesine bağlıdır. Kısacası, eğer üretilen komşuluk ilgili çözümü iyileştiremezse, paylaşılan çözüm sayesinde ilgili çözümün ilave bir iyileşebilme şansı daha vardır. Aynı döngü içerisinde hem komşu üretme yolu ile hem de komşu paylaşma yolu ile ilgili popülasyon üyesinin iyileştirilmesinin denenmesi MBO algoritmasının literatüre getirdiği yenilik olup, algoritmayı birçok uygulamada diğerlerinden daha başarılı kılmaktadır.

Şekil 4.8'den Şekil 4.11'e kadar olan grafiklerde algoritmaların eğitim sürecindeki MSE trendleri gösterilmektedir. MBO algoritmasının uygulamaların çoğunda küresel minimuma diğerlerinden daha çabuk yakınsadığı, kendine özgü paylaşım mekanizmasının avantajını kullanarak doyuma (saturation) girmeye karşı direndiği ve daha minimumu bulmak için çabaladığı açıkça görülmektedir. MBO algoritmasının eriştiği doyum değerleri genellikle diğerlerininkinden daha düşüktür. Dört veri seti hariç diğer bütün veri setlerinde en düşük MSE'yi elde eden veya elde eden algoritmalardan birisi olmuştur. Diğer taraftan, bu dört veri setindeki elde edilen doğruluk oranları ise yeterince iyi seviyelerdedir.

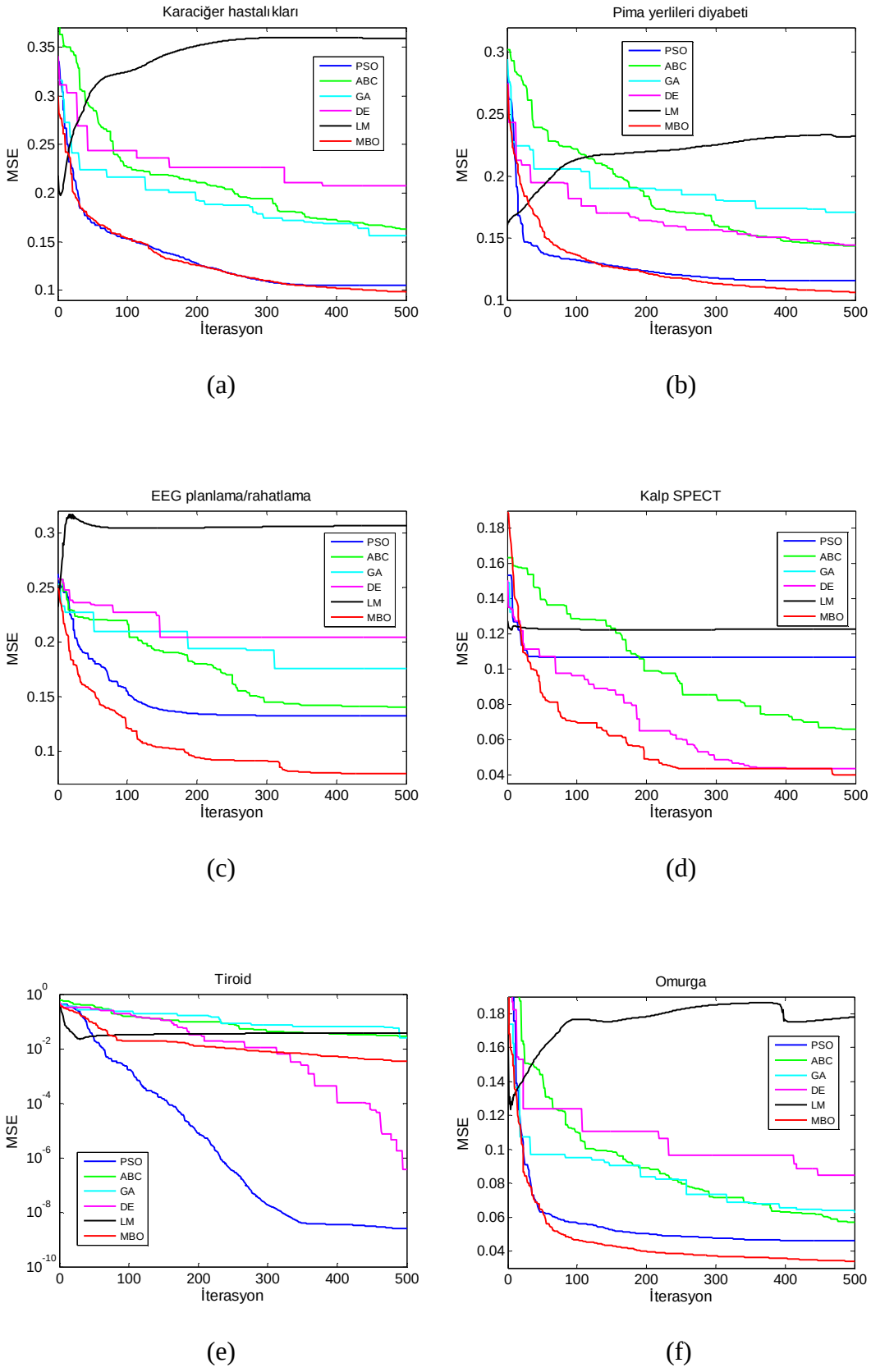
Diğer algoritmaların performansları deneyden deneye değişimler göstermektedir. PSO algoritmasının keşif yeteneğinin yeterince iyi olmayışı nedeniyle başlangıç popülasyonunun kalitesinden oldukça etkilendiği bilinmektedir. Bu yüzden özellikle parametre sayısının yüksek olduğu *Kalp SPECT* gibi problemler bu algoritmanın yerel minimumlara takılma riskinin yüksek olduğu problemlerdir. GA algoritması keşif yeteneği iyi olmasına rağmen küresel minimuma yeterince yakınsama hususunda fazla iyi değildir. Bu yüzden hiçbir eğitimde en düşük MSE değerine erişememiştir. ABC algoritması yerel minimumlardan ustaca kurtulabilmesine rağmen küresel minimuma çok iyi yakınsamalar gerçekleştirememiştir. DE algoritması küresel ve yerel arama yetenekleri arasında başarılı bir dengeye sahip olduğundan genellikle iyi en iyileme sonuçları vermektedir. Bu durum DE algoritmasının YSA eğitimindeki performansına yansısa da MBO algoritmasının genel performansını geçememiştir.

Tablo 4.9. YSA'ların eğitimlerinde erişilen MSE ve testlerinde elde edilen doğruluk değerleri (Popülasyon üye sayısı: MBO için 75 ve diğerleri için 150, toplam uygunluk hesaplama sayısı: MBO için 75500 ve diğerleri için 75000 ve toplam iterasyon sayısı: 500)

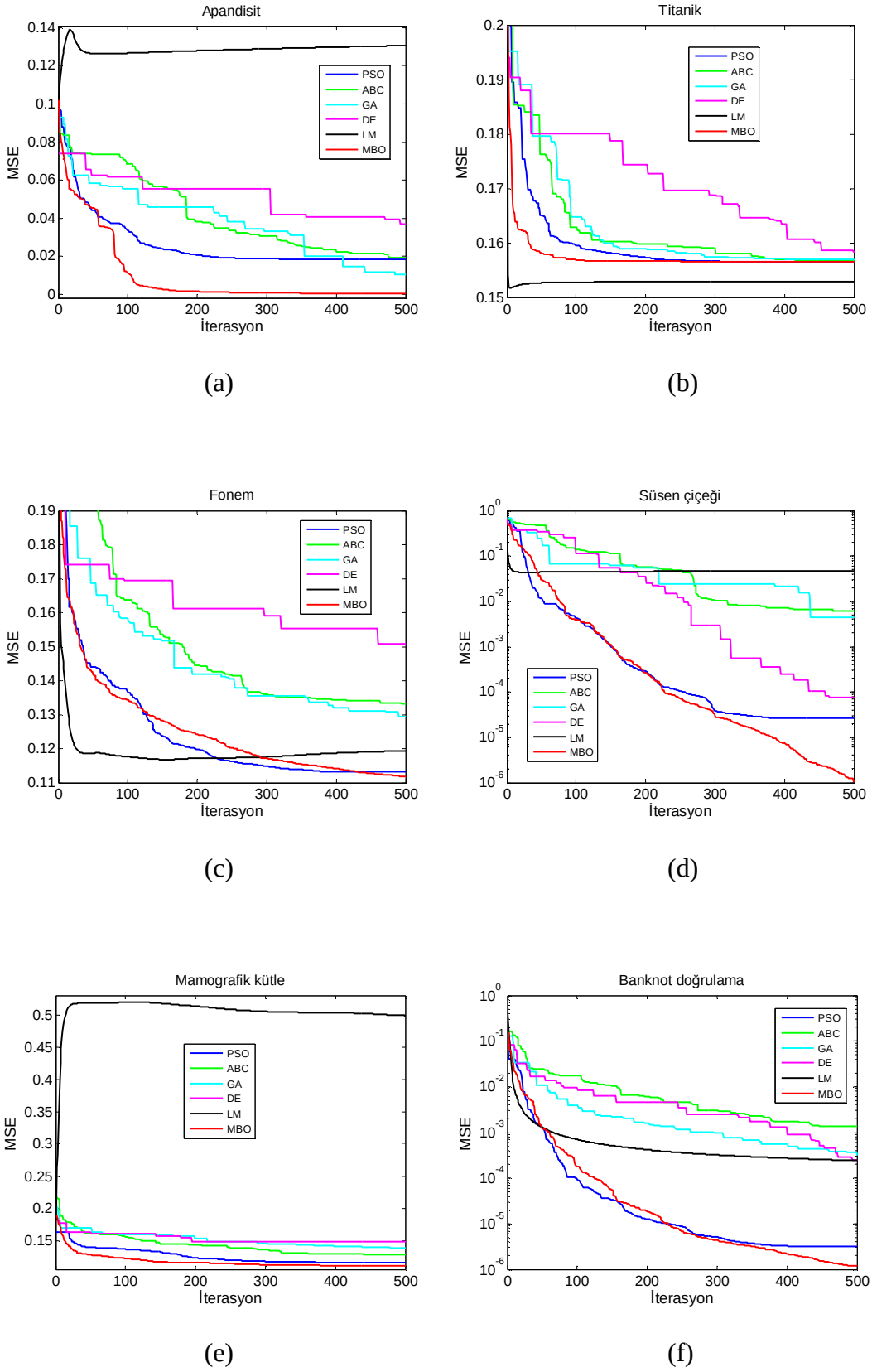
Veri Setleri		**GA**	**DE**	**PSO**	**ABC**	**MBO**	**LM**
Akut inflamasyon	***MSE***	1.06E-13	1.06E-21	1.05E-22	6.98E-15	**1.38E-26**	2.56E-05
	Doğruluk	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
Kan bağışı	***MSE***	0.1386	0.1409	**0.1361**	0.1414	0.1367	0.1584
	Doğruluk	79.41	79.14	79.41	**79.68**	79.41	79.41
Göğüs Kanseri	***MSE***	0.0061	0.0066	0.0058	0.0078	**6.09E-06**	0.0327
	Doğruluk	96.19	96.48	**97.07**	96.48	**97.07**	96.19
Doğurganlık	***MSE***	0.0101	**0.0100**	0.0400	0.0101	**0.0100**	0.0847
	Doğruluk	90.00	90.00	90.00	90.00	**92.00**	**92.00**
Hint karaciğer hastalığı	***MSE***	0.1596	0.2005	0.1201	0.1524	**0.0984**	0.2772
	Doğruluk	70.93	71.63	72.66	70.93	71.97	71.97
Lens	***MSE***	7.47E-12	**5.82E-47**	7.03E-32	4.71E-16	5.28E-46	0.3568
	Doğruluk	**81.82**	**81.82**	**81.82**	**81.82**	**81.82**	**81.82**
Karaciğer hastalıkları	***MSE***	0.1563	0.2075	0.1049	0.1628	**0.0982**	0.2900
	Doğruluk	**71.51**	**71.51**	70.93	68.60	**71.51**	69.77
Pima yerlileri diyabeti	***MSE***	0.1710	0.1442	0.1160	0.1434	**0.1062**	0.2325
	Doğruluk	76.30	**77.08**	76.56	74.22	76.56	73.96
EEG planlama/rahatlama	***MSE***	0.1758	0.2044	0.1322	0.1403	**0.0790**	0.3066
	Doğruluk	72.53	72.53	72.53	**74.73**	72.53	69.23
Kalp SPECT	***MSE***	0.0590	0.0434	0.1068	0.0659	**0.0396**	0.1227
	Doğruluk	84.21	84.21	84.21	84.96	**85.71**	84.21
Tiroit	***MSE***	0.0242	3.70E-07	**2.49E-09**	0.0284	0.0034	0.0381
	Doğruluk	96.26	**97.20**	**97.20**	96.26	**97.20**	**97.20**
Omurga	***MSE***	0.0626	0.0849	0.0463	0.0569	**0.0341**	0.1780
	Doğruluk	**83.23**	**83.23**	82.58	**83.23**	**83.23**	81.29
Apandisit	***MSE***	0.0103	0.0368	0.0185	0.0191	**1.90E-04**	0.1307
	Doğruluk	84.62	86.54	**90.38**	**90.38**	**90.38**	86.54
Titanik	***MSE***	0.1570	0.1584	0.1567	0.1569	0.1567	**0.1530**
	Doğruluk	**79.64**	**79.64**	**79.64**	**79.64**	**79.64**	**79.64**
Fonem	***MSE***	0.1295	0.1509	0.1131	0.1333	**0.1117**	0.1193
	Doğruluk	81.64	80.76	83.64	81.35	**84.04**	83.64
Süsen çiçeği	***MSE***	0.0044	7.71E-05	2.58E-05	0.0060	**1.05E-06**	0.0466
	Doğruluk	**97.33**	**97.33**	**97.33**	96.00	**97.33**	**97.33**
Mamografik kütle	***MSE***	0.1383	0.1485	0.1159	0.1285	**0.1100**	0.4987
	Doğruluk	**80.72**	80.00	80.00	78.31	**80.72**	49.64
Banknot doğrulama	***MSE***	3.16E-04	2.50E-04	3.17E-06	0.0013	**1.18E-06**	2.41E-04
	Doğruluk	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
Denge ölçeği	***MSE***	0.1380	0.0782	0.0812	0.1180	**0.0690**	0.1511
	Doğruluk	86.86	87.50	88.46	87.82	**89.74**	87.82
Haberman hayatta kalma	***MSE***	0.1434	0.1501	0.1290	0.1378	**0.1228**	0.2764
	Doğruluk	74.03	**74.68**	74.03	**74.68**	**74.68**	70.78



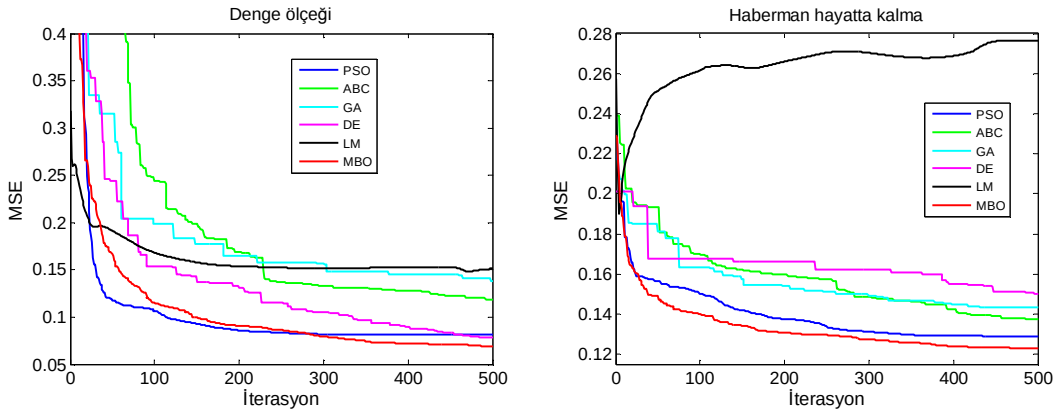
Şekil 4.8. (a) Akut inflamasyon, (b) Kan bağışı, (c) Göğüs kanseri, (d) Doğurganlık, (e) Hint karaciğer hastalığı ve (f) Lens veri setleri için tasarlanan YSA'ların eğitim sürecinde algoritmaların MSE ilerlemeleri



Şekil 4.9. (a) Karaciğer hastalıkları, (b) Pima yerlileri diyabeti, (c) EEG planlama/rahatlama, (d) Kalp SPECT, (e) Tiroit ve (f) Omurga veri setleri için tasarlanan YSA'ların eğitim sürecinde algoritmaların MSE ilerlemeleri



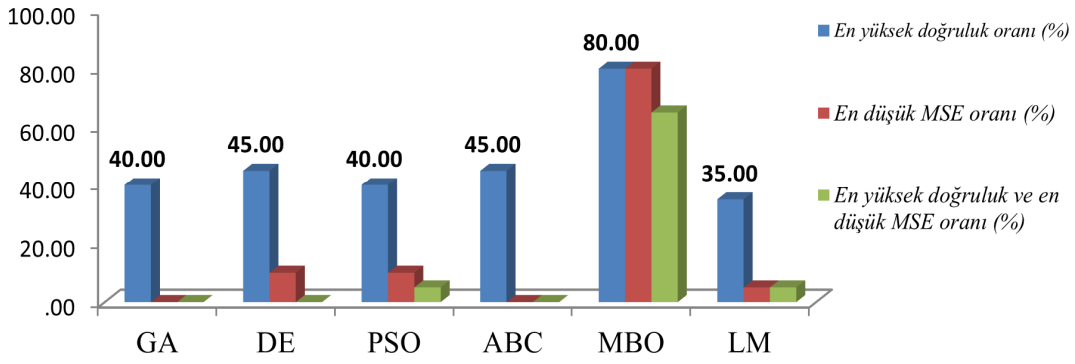
Şekil 4.10. (a) Apandisit, (b) Titanik, (c) Fonem, (d) Süsen çiçeği, (e) Mamografik kütle ve (f) Banknot doğrulama veri setleri için tasarlanan YSA'ların eğitim sürecinde algoritmaların MSE ilerlemeleri



(a) (b)

Şekil 4.11. (a) Denge ölçeği ve (b) Haberman hayatta kalma veri setleri için tasarlanan YSA'ların eğitim sürecinde algoritmaların MSE ilerlemeleri

Tablo 4.9'daki kalın puntolara bakıldığında MBO algoritmasının veri setlerinin %80'inde en yüksek doğruluğu, yine veri setlerinin %80'inde en düşük MSE değerini ve veri setlerinin %65'inde hem en yüksek doğruluğu hem de en düşük MSE değerini elde ettiği görülmektedir. Şekil 4.12'deki bar grafiği, algoritmaların bu çalışmada kullanılan yirmi veri seti üzerinde elde ettikleri başarı oranlarını göstermektedir.



Şekil 4.12. Algoritmaların, çalışmadaki 20 veri seti üzerinde elde ettikleri başarı oranları

Özetle, meta-sezgisel algoritmalardan elde edilen sonuçların geleneksel YSA eğitim algoritmaları ile rekabet eder durumda olduğu ve özellikle MBO algoritmasının elde ettiği sonuçlarla diğer meta-sezgisel algoritmaları da geride bıraktığı deneysel olarak gözlemlenmiştir.

4.4. Sempozyum Katılımcı Listelerinin En İyilenmesi

Kombinatoryal en iyileme problemleri giderek artan ilgi odağı haline gelmiş ve endüstriyel uygulamalardan bilimsel uygulamalara varıncaya kadar pek çok alanda hızla gelişen bir araştırma alanını oluşturmuştur. Kombinatoryal en iyilemenin çalışma alanı uygulamalı matematik, bilgisayar bilimi ve yöneylem araştırmalarının kesişim noktasıdır. Bir tamsayı, bir alt küme, bir permütasyon veya bir graf yapısı kombinatoryal en iyileme problemlerinin kullandığı objeler olarak düşünülür ve bu objeler sonlu kümeler içerisinden araştırılırlar [249]. Genellikle bir değişkenler kümesi ile kısıtlamaları sağlayan bir karar arasındaki ilişkileri temsil eden kombinatoryal en iyileme problemlerinin çözümünde kullanılan probleme özgü metotlar en iyi sonuçlar üretmelerine rağmen problemin en kötü durumu için üstel hesaplama sürelerine ihtiyaç duyabilirler. Bu yüzden yaklaşık yöntemler olarak meta-sezgiseller son 20 yıl içerisinde kombinatoryal en iyileme problemlerinin çözümünde giderek artan oranlarda kullanılmaya başlamışlardır [250]. Bu yöntemler, model alınan ajanların keşif ve yerel araştırma davranışlarını taklit eden iteratif metotlar olarak tarif edilebilirler. Yaygın olarak bilinen kombinatoryal en iyileme problemlerinden bazıları: gezgin satıcı problemi, kuadratik atama problemleri, zaman çizelgeleme problemleri, planlama problemleri, ağ akış problemleri ve araç yönlendirme problemleridir.

4.4.1. Problem tanıtımı

TED Kdz. Ereğli Koleji her yıl öğrencilere meslekleri tanıtmak amacıyla “Meslekler Sempozyumu” düzenlemektedir. Çeşitli mesleklerden davet edilen konuklar sempozyumda meslekleri ve bu mesleğin iş imkânları, yetkinlikleri vs. gibi konularda öğrencilere bilgi vermektedirler. Sempozyumda, meslek sayısı kadar paralel oturum düzenlenmekte ve her bir meslek sunumu üç kez tekrarlanmaktadır. Dolayısıyla her bir öğrencinin en fazla üç adet mesleğin oturumlarına katılabilme imkânı vardır. Üç kez tekrarlanan oturumlarda, salonlarda oluşabilecek dengesiz miktarlardaki dinleyici dağılımlarını engellemek ve mevcut salon kapasitelerinin aşılmaması için sempozyum öncesinde öğrencilere hangi meslek oturumlarına katılacaklarına dair bir anket yapılmaktadır. Ankette, öğrencilerden dinleyici olarak

katılmak istedikleri üç tane mesleği seçmeleri istenmektedir. Öğrencilerin yaptıkları tercihlere göre her bir meslek sunumunun oturumlarına hangi öğrencilerin katılacağına dair listeler oluşturulmaktadır. İşin bu aşaması problemin en önemli ve çözüm bekleyen kısmını oluşturmaktadır.

Okul öğretmenleri oturum listelerini hazırlarken her bir öğrencinin tercihlerini oturum listelerine tek tek ve dengeli bir şekilde dağıtmaya çalışmaktadırlar. Bunun için bazı listeler üzerinde defalarca revizyonlar yapıp, öğrencilerin tercihlerini karşılayan değişik kombinasyonlar denemektedirler. Bu işlem uzun, zahmetli ve dikkat isteyen bir çalışma gerektirmektedir. Her şeye rağmen, listelerin hazırlanması tamamlandığında yine de arzulanan tam dengeli bir dağılım elde edilememektedir. Örneğin, hukuk mesleğinin ilk oturumunda çok fazla öğrenci bulunurken, ikinci ve üçüncü oturumlarında öğrenci sayısı daha az olmakta ve bu durumda hukuk mesleği sunumları için birinci oturumdan dolayı büyük salon tahsisi gerekmektedir. Büyük salon sayısının sınırlı olması nedeni ile bu gibi durumlarda listeler üzerinde yeniden revizyonlar yapılmakta ve her şey biraz deneme sınama yöntemi ile biraz da kişisel sezgilerle tekrarlanmaktadır. Sunum yapacak olan bazı meslek temsilcilerinin çeşitli sebeplerden dolayı sadece iki oturuma katılabileceklerini belirtmeleri ise ayrı bir problem teşkil etmekte ve mevcut problem içerisinde ilave kısıtlar oluşturmaktadır. Bu durumda bu meslekleri tercih eden dinleyiciler ilk iki oturumda bu mesleklerin sunumlarına yönlendirilirken diğer meslek sunumlarının üçüncü oturumlarındaki dinleyici sayısı kaçınılmaz bir şekilde artmaktadır.

Bu problem bir kombinatoryal en iyileme problemi olup, değişik meta-sezgisel algoritmaların ayrık versiyonlarının üzerinde denenebileceği bir vaka çalışmasıdır.

4.4.2. Problemin matematiksel ifadesi

Bu bölümde, önceki bölümde tanıtımı yapılan problem matematiksel ifadelerle tanımlanmıştır. Toplam meslek sunumu sayısı n_p olmak üzere meslek sunumlarına 1 den n_p' ye kadar kimlik numarası verilmiştir. Her bir meslek sunumunun oturumlarına dinleyici olarak katılacak olan toplam katılımcı sayısı bir matris ile

$$S = \begin{bmatrix} s_{11} & s_{12} & \dots & s_{1n_p} \\ s_{21} & s_{22} & \dots & s_{2n_p} \\ s_{31} & s_{32} & \dots & s_{3n_p} \end{bmatrix} \quad \textbf{(4.9)}$$

şeklinde ifade edilebilir. Burada S oturum dağılım matrisi, s_{ij} j meslek sunumunun i'inci oturumundaki toplam katılımcı sayısı ve n_p toplam meslek sunumu sayısıdır. S matrisinin her bir elemanı öğrencilerin anketteki tercihlerine göre sunum oturumlarına dağıtılma işlemi tamamlandıktan sonra hesaplanır. Öğrencilerin her bir oturumda kaç numaralı sunuma katılacakları da bir matris aracılığı ile

$$C = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1n_s} \\ c_{21} & c_{22} & \dots & c_{2n_s} \\ c_{31} & c_{32} & \dots & c_{3n_s} \end{bmatrix} \quad \textbf{(4.10)}$$

olarak ifade edilebilir. Burada C öğrenci sunum sıralama matrisi, c_{ij} j öğrencisinin i'inci oturumunda kaç numaralı meslek sunumuna katılacağını gösteren tamsayı değeri ve n_s sempozyuma katılacak olan toplam öğrenci sayısıdır. Yani C matrisinin her bir sütunu bir öğrenciyi temsil eder ve o öğrencinin sırası ile hangi meslek sunumlarına katılacağını gösterir. Bu matristeki her bir j öğrencisinin ilk sunum sıralaması öğrencinin tercihleri dikkate alınarak ve oturum sınırlamaları göz önünde bulundurularak rasgele yapılır. Oturum sınırlamaları da bir matris formunda

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n_p} \\ a_{21} & a_{22} & \dots & a_{2n_p} \\ a_{31} & a_{32} & \dots & a_{3n_p} \end{bmatrix} \quad \textbf{(4.11)}$$

olarak tanımlanabilir. Burada A oturum kısıt matrisi, a_{ij} j sunumunun i'inci oturumunun yapılıp yapılmayacağını temsil eden mantıksal değişken ve n_p toplam meslek sunumu sayısıdır. Kısacası A matrisi 0 ve 1'lerden oluşur. A matrisindeki 1'ler ilgili oturumların yapılacağını, 0'lar ise ilgili oturumların olmayacağını ifade eder. Her bir sunumun oturumları için arzulanan ortalama katılımcı sayısı

$$Mean = \lfloor mean_1 \quad mean_2 \quad \cdots \quad mean_{np} \rfloor \tag{4.12}$$

olarak matris formunda ifade edilir. *Mean* ortalama katılımcı matrisi, $mean_i$ *i* sunumunun her bir oturumunda arzulanan ortalama katılımcı sayısı ve n_p toplam meslek sunumu sayısıdır.

Daha önce de bahsedildiği gibi bu kombinatoryal problemde ana hedef dengeli bir *S* matrisi elde etmektir. Yani *A* oturum kısıt matrisi ve öğrenci tercihleri göz önünde bulundurularak, *C* matrisinin değişik kombinasyonları denenmek suretiyle, *S* matrisinin her bir s_{ij} elemanının *Mean* matrisinin $mean_j$ elemanına mümkün olduğunca yakınsaması sağlanacaktır. Sonuç olarak, *C* öğrenci sunum sıralama matrisinin her yeni kombinasyonu için yeni bir *S* oturum dağılım matrisi elde edilir. *S* oturum dağılım matrisi için MSE değeri

$$MSE(S) = \frac{1}{n_p} \sum_{j=1}^{n_p} \sum_{i=1}^{3} \left(s_{ij} - mean_j \right)^2 \tag{4.13}$$

kullanılarak hesaplanabilir. Burada s_{ij} *j* meslek sunumunun *i*'inci oturumundaki toplam katılımcı sayısı, $mean_j$ *j* sunumunun her bir oturumunda arzulanan ortalama katılımcı sayısı ve n_p toplam meslek sunumu sayısıdır. Hesaplanan MSE değeri ilgili *C* kombinasyonunun maliyet değeri olarak kullanılabilir.

Sonuç olarak eğer hesaplanabilen bir maliyet değeri varsa, bir kombinatoryal en iyileme algoritması kullanılarak bu maliyeti minimize eden en iyi kombinasyon bulunabilir. Kullanılacak olan bu algoritma, *S* matrisinde en iyi dağılımı elde etmek için *A* matrisini dikkate alarak *C* matrisi üzerinde en iyi kombinasyonu elde etmeye çalışacaktır.

4.4.3. Önerilen kombinatoryal algoritmalar

Kombinatoryal problemlerin çözümü için kombinatoryal algoritmalar veya mevcut nümerik algoritmaların ayrık versiyonları kullanılabilir. Bu bölümde üç ayrı meta-

sezgisel algoritmanın ayrık versiyonları bu kombinatoryal problemin çözümünde kullanılmak üzere önerilmiştir. Bunlar ayrık yapay arı koloni (Discrete Artificial Bee Colony - DABC) algoritması, ayrık göçmen kuşlar en iyileme (Discrete Migrating Birds Optimization - DMBO) algoritması ve ayrık genetik algoritmadır (Discrete Genetic Algorithm - DGA).

4.4.3.1. DABC algoritması

Bölüm 2.1'de anlatılan ABC algoritması ile aynı temel felsefe üzerine oturan DABC algoritmasının ABC algoritmasından farkları aşağıda sıralanmıştır.

- ABC algoritması parametrelere atanmış değerleri içeren bireylerden oluşan bir popülasyon kullanır. Bu problem için önerilen DABC algoritmasında ise popülasyon, C matrisinin A oturum kısıt matrisini sağlayan kombinasyonlarından oluşan üyelerden meydana gelir.

- ABC algoritması başlangıç popülasyonunu oluştururken Denklem (2.1)'i kullanarak rasgele çözümler üretir. Önerilen DABC algoritması ise C öğrenci sunum sıralama matrisindeki öğrenci tercihlerini A matrisi ile verilen kısıtlamalara uymak şartı ile rasgele sıralayarak başlangıç kombinasyonlarını oluşturur.

- ABC algoritması komşu çözümleri oluştururken Denklem (2.3)'ü kullanır. Önerilen DABC algoritması komşu çözümleri oluştururken, her biri bir C matrisi kombinasyonu olan popülasyon üyeleri için, rasgele seçilen bir öğrencinin tercihlerini temsil eden rasgele seçilen bir sütundaki tercih sıralamasını A matrisi ile verilen kısıtlamalara uymak şartı ile rasgele değiştiren bir yöntem kullanır.

- DABC algoritması popülasyon üyelerinin maliyet değerlerini hesaplarken Denklem (4.13)'ü kullanır. Elde ettiği maliyet değerlerini Denklem (2.2)'de f_i yerine koyarak her bir popülasyon üyesinin ve üretilen komşu çözümlerin uygunluk değerlerini hesaplar.

4.4.3.2. DMBO algoritması

Bölüm 2.2'de anlatılan MBO algoritması ile aynı temel felsefe üzerine oturan DMBO algoritmasının MBO algoritmasından temel farkları kullandığı popülasyonun yapısı, başlangıç popülasyonunun üretimi, komşu çözümlerin oluşturulması ve üye uygunluklarının hesapları şeklinde sıralanabilir. Bu hesaplamaların yapılmasında DABC algoritmasında uygulanan yöntemler kullanılır.

4.4.3.3. DGA

Bölüm 2.5'de anlatılan GA ile aynı temel felsefe üzerine oturan DGA, hesaplamaların yapılması aşamasında bazı farklılıklar içirir. Önerilen DGA'da başlangıç popülasyonunun oluşturulması ve uygunluk hesaplamaları DABC algoritmasında olduğu gibi yapılır. Kombinatoryal bir yaklaşım ile ikili GA'daki çaprazlama işlemini başarılı bir şekilde taklit eden bir yöntem kullanılır. Bu yöntemde popülasyon üyeleri kromozom olarak kabul edilir ve GA'da Denklem (2.14.a) ve Denklem (2.14.b) ile tanımlanan ebeveyn kromozomlarının önerilen DGA'da

$$Parent_1 = \begin{bmatrix} c_{11}^{mom} & c_{12}^{mom} & \cdots & c_{1\alpha}^{mom} & \cdots & c_{1n_s}^{mom} \\ c_{21}^{mom} & c_{22}^{mom} & \cdots & c_{2\alpha}^{mom} & \cdots & c_{2n_s}^{mom} \\ c_{31}^{mom} & c_{32}^{mom} & \cdots & c_{3\alpha}^{mom} & \cdots & c_{3n_s}^{mom} \end{bmatrix} \quad \textbf{(4.14.a)}$$

$$Parent_2 = \begin{bmatrix} c_{11}^{dad} & c_{12}^{dad} & \cdots & c_{1\alpha}^{dad} & \cdots & c_{1n_s}^{dad} \\ c_{21}^{dad} & c_{22}^{dad} & \cdots & c_{2\alpha}^{dad} & \cdots & c_{2n_s}^{dad} \\ c_{31}^{dad} & c_{32}^{dad} & \cdots & c_{3\alpha}^{dad} & \cdots & c_{3n_s}^{dad} \end{bmatrix} \quad \textbf{(4.14.b)}$$

formatında olduğu düşünülür. Burada her bir kromozom bir *C* matrisi kombinasyonu olup, bu matrisin her bir sütunu ilgili öğrencinin katılacağı sunumların numaralarını oturum sırası ile temsil eder, α 1 ile n_s arasında rasgele üretilmiş bir tamsayı olup çaprazlama için tespit edilen seçim noktasını temsil eder ve n_s sempozyuma katılacak olan toplam öğrenci sayısıdır. GA'da Denklem (2.15.a) ve Denklem (2.15.b) ile hesaplanan yavru nesillerde ortaya çıkacak olan yeni değişkenler önerilen DGA'da

$$P_{new\_mom} = \begin{bmatrix} p_{new\_mom1} \\ p_{new\_mom2} \\ p_{new\_mom3} \end{bmatrix} = \text{randc}\left(\begin{bmatrix} c_{1\alpha}^{mom} \\ c_{2\alpha}^{mom} \\ c_{3\alpha}^{mom} \end{bmatrix} \right) \quad \textbf{(4.15.a)}$$

$$P_{new\_dad} = \begin{bmatrix} p_{new\_dad1} \\ p_{new\_dad2} \\ p_{new\_dad3} \end{bmatrix} = \text{randc}\left(\begin{bmatrix} c_{1\alpha}^{dad} \\ c_{2\alpha}^{dad} \\ c_{3\alpha}^{dad} \end{bmatrix} \right) \quad \textbf{(4.15.b)}$$

şeklinde hesaplanır. Burada randc bir 3×1 matrisi *A* matrisi ile verilen kısıtlamalara uymak şartı ile yeniden rasgele sıralayan bir fonksiyondur. Daha sonra yeni değişkenleri içeren yavrular üretilirken $c_{*\alpha}^{mom}$ ve $c_{*\alpha}^{dad}$ sırası ile P_{new_mom} ve P_{new_dad} ile değiştirilir ve seçim noktasının sağında kalan parametrelerin tamamı Denklem (4.16.a) ve Denklem (4.16.b)'de verildiği gibi karşılıklı olarak ebeveynler arasında yer değiştirilir.

$$Offspring_1 = \begin{bmatrix} c_{11}^{mom} & c_{12}^{mom} & \cdots & p_{new\_mom1} & \cdots & c_{1n_s}^{dad} \\ c_{21}^{mom} & c_{22}^{mom} & \cdots & p_{new\_mom2} & \cdots & c_{2n_s}^{dad} \\ c_{31}^{mom} & c_{32}^{mom} & \cdots & p_{new\_mom3} & \cdots & c_{3n_s}^{dad} \end{bmatrix} \quad \textbf{(4.16.a)}$$

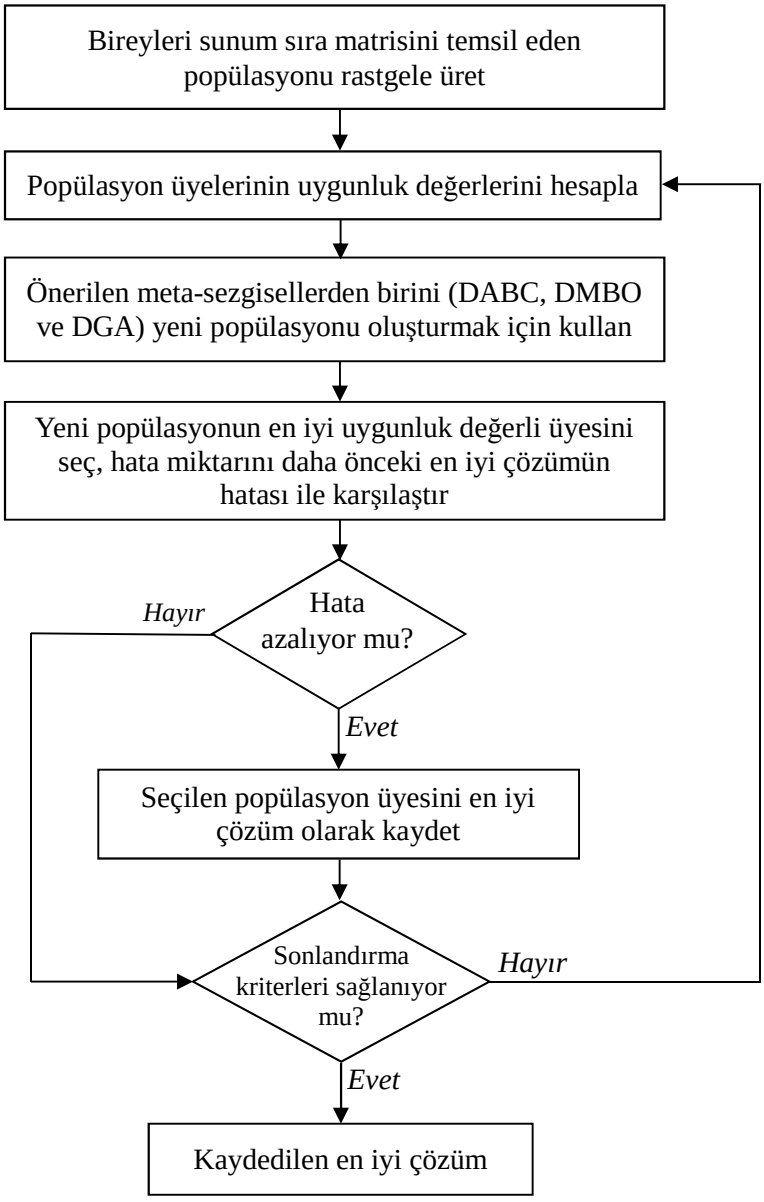
$$Offspring_2 = \begin{bmatrix} c_{11}^{dad} & c_{12}^{dad} & \cdots & p_{new\_dad1} & \cdots & c_{1n_s}^{mom} \\ c_{21}^{dad} & c_{22}^{dad} & \cdots & p_{new\_dad2} & \cdots & c_{2n_s}^{mom} \\ c_{31}^{dad} & c_{32}^{dad} & \cdots & p_{new\_dad3} & \cdots & c_{3n_s}^{mom} \end{bmatrix} \quad \textbf{(4.16.b)}$$

Önerilen DGA'nın mutasyon adımında kombinatoryal bir mutasyon işlemi uygulanır. Bu yöntemde önceden belirlenen bir oran kadar sütun pozisyonu popülasyonu oluşturan kromozom matrisleri içerisinden seçilirler. Rasgele seçilen genleri temsil eden bu matris sütunlarına randc olarak tanımlanan fonksiyon uygulanarak genlerin temsil ettiği kombinasyon değiştirilir.

Daha sonra seleksiyon işlemi uygunluk değeri daha iyi olan popülasyon üyesinin seçilmesi şeklinde normal GA'da olduğu gibi icra edilir.

4.4.4. Önerilen kombinatoryal algoritmaların probleme tatbiki

Şekil 4.13'te akış diyagramı verilen prosedür sempozyum katılımcı listelerinin oluşturulması probleminin çözümünde kullanılır. Prosedürün çalışması kısaca şöyledir. Önce, üyeleri olası *C* matrislerinden oluşan bir başlangıç popülasyonu üretilir. Daha sonra prosedürün ana döngüsü başlar.



Şekil 4.13. Önerilen sempozyum katılımcı listesi oluşturma prosedürü akış diyagramı

Bu döngü içerisinde ilk olarak, her bir popülasyon üyesinin uygunluk değerleri hesaplanır. İkinci olarak, kullanılacak olan ayrık meta-sezgisel algoritma (DABC,

DMBO veya DGA) popülasyona uygulanarak yeni popülasyon üyeleri üretilir. Yeni popülasyondaki en yüksek uygunluk değerine sahip olan üye prosedürün o ana kadar tespit ettiği en yüksek uygunluklu çözümden daha iyi ise bu üye prosedürün eriştiği en yüksek uygunluklu çözüm olarak saklanır. Prosedürün ana döngüsündeki bu işlemler sonlandırma kriterleri sağlanana kadar devam eder. Prosedür durduğunda en yüksek uygunluklu çözüm olarak saklanan çözümün temsil ettiği *C* matrisi problemin en iyi çözümüdür.

4.4.5. Temsili vaka çalışması ve sonuçları

Mayıs 2013'te organizasyon komitesi tarafından TED Kdz. Ereğli Koleji Geleneksel Meslekler Sempozyumu'na 20 konuşmacı Tablo 4.10'da listelenen mesleklerin tanıtımlarını yapmaları için davet edildiler. Çalışmada meslekler P1'den P20'ye kadar sembollerle ifade edildi.

Tablo 4.10. Meslekler

Sıra	Meslek adı	Kısaltma
1	Tıp	P1
2	Diş hekimliği	P2
3	Tıbbi dokümantasyon	P3
4	Biyomedikal mühendisliği	P4
5	Eczacılık	P5
6	Elektronik mühendisliği	P6
7	Bilgisayar mühendisliği	P7
8	Metalürji mühendisliği	P8
9	Makine mühendisliği	P9
10	İnşaat mühendisliği	P10
11	Çevre mühendisliği	P11
12	Endüstri mühendisliği	P12
13	Ekonomi	P13
14	Gastronomi	P14
15	Turizm ve otel yönetimi	P15
16	Lojistik	P16
17	Arkeoloji	P17
18	Hukuk	P18
19	Muhasebe	P19
20	Psikoloji	P20

Önce, test amacı ile toplam 347 öğrenci için temsili öğrenci tercihleri rasgele oluşturuldu. Bu temsili tercih verisi Tablo 4.11'de kısmi olarak listelenmiştir.

Temsili tercih verisinin tamamı EK-F'de verilmiştir. Tablolardaki "●" işareti öğrencinin katılacağı meslek sunumlarını göstermektedir. Daha sonra, bu tercihler doğrultusunda oluşturulan *C* matrislerinden oluşan popülasyonu optimize etmek için DABC, DMBO ve DGA meta-sezgisel algoritmalarını kullanan prosedürler kullanılmıştır. Bu çalışma gerçek vaka çalışmasının taklit edildiği bir temsili vaka çalışmasıdır. Deneysel çalışmalar önce bütün meslek sunumlarında 3 oturumun yapılacağı kabullenilerek oturum sınırlamasız olarak yapılmış, daha sonra ise bazı oturumlara sınırlamalar getirilerek deneyler tekrarlanmıştır.

Tablo 4.11. Kısmi öğrenci tercih listesi (Listenin tamamı EK-F'de verilmiştir)

Öğrenci Adı	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20
St-001	●		●		●															
St-002		●	●	●																
St-003	●			●			●													
St-004		●											●			●				
St-005														●	●	●				
St-006															●		●	●		
St-007																●	●	●		
St-008				●	●	●														
:																				
:																				
:																				
St-346	●			●														●		
St-347					●	●												●		

Meta-sezgisel algoritmalarda her komşuluk üretimi sonrasında üretilen komşu çözümün kalitesini değerlendirmek için uygunluk hesaplaması yapılır. DMBO algoritmasının her iterasyonunda lider kuşu temsil eden çözüm için *k* adet ve diğer çözümler için (*k*-*x*) adet uygunluk hesaplaması yapılır. DABC ve DGA'da ise her bir iterasyonda *n* adet uygunluk hesaplaması yapılır. Dolayısıyla algoritmalarda gerçekleşen toplam uygunluk hesaplama sayıları

$$N_{DMBO} = N * [(n-1)(k-x)+k] \quad \textbf{(4.16.a)}$$

$$N_{DABC} = N_{DGA} = n * N \quad \textbf{(4.16.b)}$$

olarak ifade edilir. Burada N_{DMBO}, N_{DABC} ve N_{DGA} sırası ile DMBO, DABC ve DGA'daki toplam uygunluk hesaplama sayıları, *n* popülasyon büyüklüğü, *N* toplam

iterasyon sayısı, *k* DMBO algoritmasında her bir çözüm için değerlendirilecek olan toplam komşu sayısı ve *x* DMBO algoritmasında bir sonraki çözüm ile paylaşılacak toplam komşu sayısıdır.

Algoritma parametrelerinin seçimindeki temel motivasyon algoritmaların toplamda eşit sayıda uygunluk hesabı gerçekleştirmelerini sağlayarak yaklaşık olarak eşit işletim sürelerini elde etmek ve bu sayede algoritmalar arasında daha adil performans mukayesesi yapabilmektir. Bu sebeple, algoritmalarda Tablo 4.12'deki parametreler kullanılarak her üç algoritmada da gerçekleşecek olan toplam uygunluk hesabı sayısının 5500 olması sağlanmıştır. Her bir deneysel çalışma 20'şer defa gerçekleştirilmiş ve her defasında 0.3 MSE değeri elde edilmiştir.

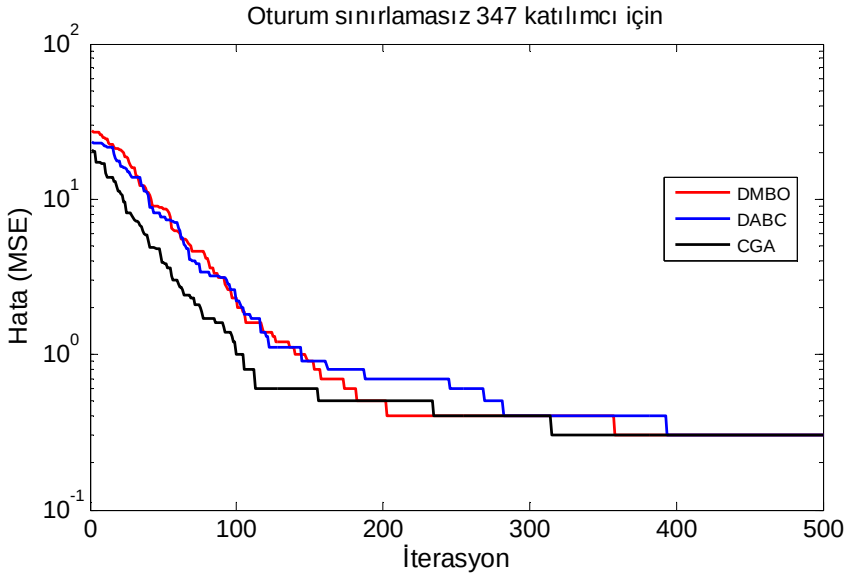
Tablo 4.12. Temsili vaka çalışmasında kullanılan temel algoritma parametreleri ve hesaplama maliyetleri

		DMBO	**DABC**	**DGA**
Algoritma parametreleri	İterasyon (*N*)	500	550	550
	Popülasyon büyüklüğü (*n*)	5	10	10
	Değerlendirilen komşuluk (*k*)	3	-	-
	Paylaşılan komşuluk (*x*)	1	-	-
	Lider değişim periyodu (*m*)	10	-	-
	Mutasyon oranı (%)	-	-	0.1
Hesaplama maliyetleri	Toplam uygunluk hesabı sayısı	**5500**	**5500**	**5500**
	MSE	**0.3000**	**0.3000**	**0.3000**

Oturum sınırlaması olmayan durum için her bir algoritmanın başarılı bir örnek işletimi sonrasında elde ettiği oturum dağılımları Tablo 4.13'te verilmiştir. Burada her bir sunumun bütün oturumlarındaki katılımcı sayılarının eşitlendiği görülmektedir. Dolayısıyla algoritmalar hedeflenen dengeli oturum dağılımlarını elde etmişlerdir. Algoritmaların en iyileme süresince gerçekleştirdikleri MSE ilerlemeleri Şekil 4.14'teki grafiklerde verilmiştir. Grafikler, belirlenen maksimum iterasyon sayısının bu vaka çalışması için yeterli olduğunu göstermektedir. DMBO, DABC ve DGA'nın oturum sınırlamasız durum için elde ettiği örnek katılımcı listeleri sırasıyla EK-G, EK-H ve EK-I'da verilmiştir. Bu listelerden hem algoritmaların katılımcıları oturumlara dengeli bir şekilde dağıttığı hem de her bir katılımcının hangi meslek sunumunun hangi oturumuna katılacağı açıkça görülmektedir.

Tablo 4.13. Algoritmaların oturum sınırlamasız durum için başarılı bir örnek işletim sonrasında elde ettiği oturum dağılımları (P1, P2, …, P20: Sunumlar; S1, S2 ve S3: Oturumlar)

	DMBO			DABC			DGA		
	S1	**S2**	**S3**	**S1**	**S2**	**S3**	**S1**	**S2**	**S3**
P1	28	27	27	27	28	27	27	28	27
P2	20	20	21	20	20	21	21	20	20
P3	20	20	20	20	20	20	20	20	20
P4	22	22	22	22	22	22	22	22	22
P5	19	19	19	19	19	19	19	19	19
P6	12	11	11	11	12	11	12	11	11
P7	19	20	20	20	20	19	20	19	20
P8	14	15	14	14	15	14	15	14	14
P9	20	20	20	20	20	20	20	20	20
P10	19	19	19	19	19	19	19	19	19
P11	17	17	17	17	17	17	17	17	17
P12	14	14	14	14	14	14	14	14	14
P13	18	19	18	18	18	19	18	18	19
P14	14	13	14	14	13	14	13	14	14
P15	16	16	15	16	15	16	15	16	16
P16	15	15	15	15	15	15	15	15	15
P17	12	12	12	12	12	12	12	12	12
P18	15	15	15	15	15	15	15	15	15
P19	17	17	18	18	17	17	17	18	17
P20	16	16	16	16	16	16	16	16	16



Şekil 4.14. Algoritmaların oturum sınırlamasız durum için en iyileme süresince gerçekleştirdikleri MSE ilerlemeleri

Bir ve iki oturum sınırlamalı durumlar için de aynı hesaplamalar yapılmıştır. Bir oturum sınırlamalı durum için P1 sunumuna toplam 2 oturum sınırlaması uygulanmış, iki oturum sınırlamalı durum için ise P1 ve P2 sunumlarına 2 oturum sınırlaması uygulanmıştır.

Yine Tablo 4.12'de verilen parametrelerle algoritmalar tekrar çalıştırılmış, algoritmaların MSE değerleri bir oturum sınırlamalı durum için 3.4667 olarak ve iki oturum sınırlamalı durum için 9.9583 olarak gerçekleşmiştir. Bir ve iki oturum sınırlamalı durumlar için her bir algoritmanın başarılı bir örnek işletimi sonrasında elde edilen oturum dağılımları sırası ile Tablo 4.14 ve Tablo 6.15'te verilmiştir.

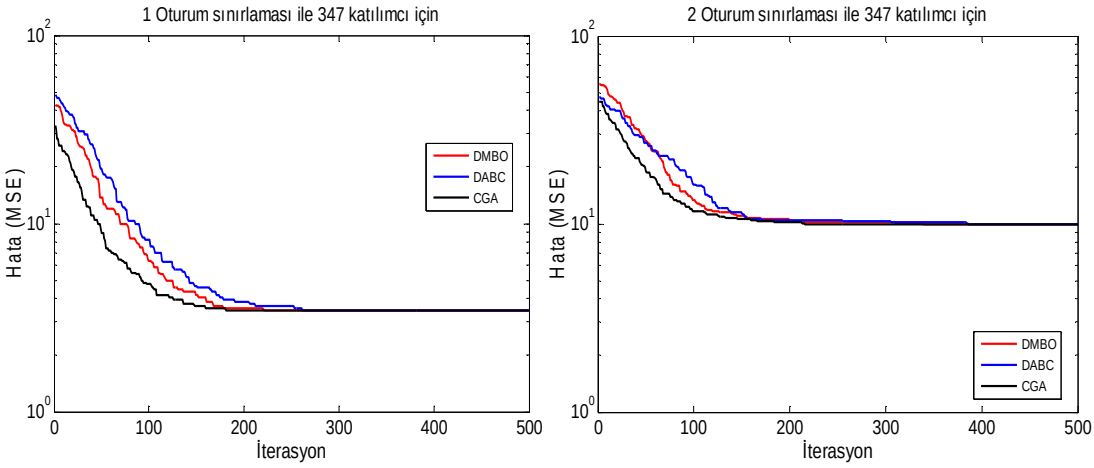
Bu tablolarda oturum kısıtlamasından kaynaklanan küçük dengesizlikler görülmektedir. Oturum sınırlamasının olduğu sunumlara ilk iki oturumda öncelik verilmesi gerekliliği nedeni ile oluşan bu küçük dengesizlikler normaldir. Yani elde edilen listeler yine en iyi listelerdir. Bir ve iki oturum sınırlamalı durumlar için algoritmaların en iyileme süresince gerçekleştirdikleri MSE ilerlemeleri Şekil 4.15'teki grafiklerde verilmiştir. Grafikler belirlenen maksimum iterasyon sayısının bu vaka çalışmasının bir ve iki oturum sınırlamalı durumları için de yeterli olduğunu göstermektedir.

Tablo 4.14. Algoritmaların 1 oturum sınırlamalı durum için başarılı bir örnek işletim sonrasında elde ettiği oturum dağılımları (P1, P2, …, P20: Sunumlar; S1, S2 ve S3: Oturumlar)

	DMBO			**DABC**			**DGA**		
	S1	**S2**	**S3**	**S1**	**S2**	**S3**	**S1**	**S2**	**S3**
P1	41	41	0	41	41	0	41	41	0
P2	20	20	21	20	19	22	20	19	22
P3	20	19	21	19	19	22	19	19	22
P4	21	21	24	21	21	24	21	21	24
P5	18	18	21	19	18	20	19	18	20
P6	11	11	12	10	11	13	11	10	13
P7	19	19	21	19	19	21	19	19	21
P8	14	13	16	14	14	15	13	14	16
P9	19	19	22	20	19	21	19	19	22
P10	18	18	21	18	19	20	18	18	21
P11	15	17	19	16	16	19	16	17	18
P12	13	13	16	13	14	15	13	13	16
P13	18	17	20	18	17	20	18	18	19
P14	13	13	15	13	13	15	13	13	15
P15	15	15	17	15	15	17	15	15	17
P16	15	14	16	14	15	16	14	15	16
P17	11	12	13	11	11	14	12	11	13
P18	14	15	16	14	15	16	14	15	16
P19	17	16	19	17	16	19	17	17	18
P20	15	16	17	15	15	18	15	15	18

Tablo 4.15. Algoritmaların 2 oturum sınırlamalı durum için başarılı bir örnek işletim sonrasında elde ettiği oturum dağılımları (P1, P2, …, P20: Sunumlar; S1, S2 ve S3: Oturumlar)

	DMBO			**DABC**			**DGA**		
	S1	**S2**	**S3**	**S1**	**S2**	**S3**	**S1**	**S2**	**S3**
P1	41	41	0	41	41	0	41	41	0
P2	31	30	0	31	30	0	30	31	0
P3	19	18	23	19	18	23	18	19	23
P4	20	21	25	21	21	24	20	21	25
P5	18	17	22	17	18	22	18	17	22
P6	10	10	14	10	10	14	10	10	14
P7	19	18	22	18	18	23	19	18	22
P8	13	13	17	13	13	17	13	13	17
P9	18	19	23	19	18	23	19	19	22
P10	17	18	22	18	18	21	17	18	22
P11	15	16	20	16	16	19	16	15	20
P12	13	13	16	12	13	17	13	13	16
P13	17	17	21	17	17	21	17	17	21
P14	12	13	16	13	12	16	12	12	17
P15	14	14	19	14	14	19	14	15	18
P16	14	14	17	13	14	18	14	13	18
P17	11	11	14	11	11	14	11	10	15
P18	14	14	17	13	14	18	14	14	17
P19	16	16	20	16	16	20	16	16	20
P20	15	14	19	15	15	18	15	15	18



Şekil 4.15. Algoritmaların (a) bir ve (b) iki oturum sınırlamalı durumlar için MSE ilerlemeleri

4.4.6. Genişletilmiş vaka çalışması ve sonuçları

Eşdeğer deneysel çalışma olarak adlandırabilecek temsili vaka çalışması tamamlandıktan sonra, önerilen DABC, DMBO ve DGA ayrık meta-sezgisel algoritmalarının katılımcı sayılarının ve paralel sunum sayılarının artırılması durumunda performansının nasıl etkileneceğini test etmek için problemin boyutu artırılmıştır. Meslek sunumu sayısı (konuşmacı sayısı) 40'a ve katılımcı sayısı 5000'e yükseltilerek aynı deneysel çalışma probleminin boyutu artırılarak bir kez daha tekrarlanmıştır.

Bu şekilde, büyük boyutlu problemlerdeki başarısı da test edilen algoritmalar için kullanılan algoritma parametreleri ve çalışmalarda ulaşılan hata değerleri Tablo 4.16'da verilmiştir. Her bir algoritmanın başarılı bir örnek işletimi sonrasında elde ettiği oturum dağılımları ise Tablo 4.17'de verilmiştir.

Tablo 4.16. Genişletilmiş temsili vaka çalışması için algoritma parametreleri ve elde edilen sonuçların maliyet değerleri

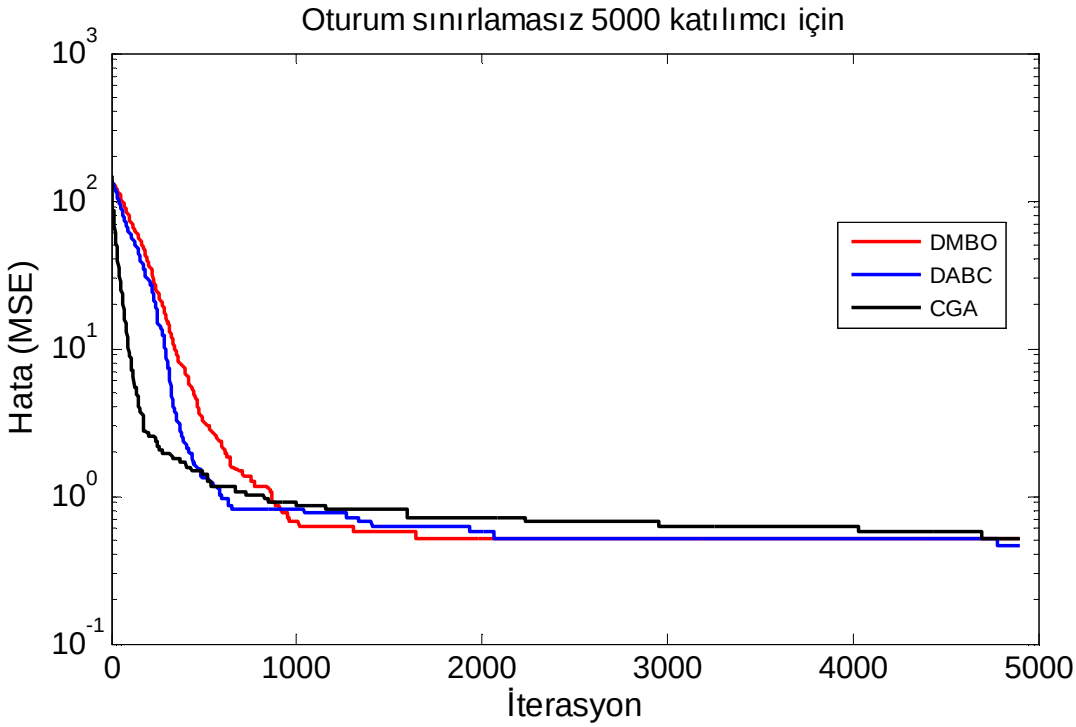
		DMBO	**DABC**	**DGA**
Algoritma parametreleri	İterasyon (*N*)	4902	5000	5000
	Popülasyon büyüklüğü (*n*)	25	50	50
	Değerlendirilen komşuluk (*k*)	3	-	-
	Paylaşılan komşuluk (*x*)	1	-	-
	Lider değişim periyodu (*m*)	10	-	-
	Mutasyon oranı (%)	-	-	0.1
Hesaplama maliyetleri	Toplam uygunluk hesabı sayısı	**250002**	**250000**	**250000**
	MSE	**0.5167**	**0.4667**	**0.5167**

Oturum dağılım sonuçları, oturum dağılımlarının algoritmalar tarafından oldukça iyi dengelendiğini göstermektedir. En iyi performans DABC algoritması tarafından elde edilmiştir. DABC algoritması bütün oturumlar için mükemmel bir dağılım elde ederek problemin küresel en iyi noktasına ulaşmıştır. DMBO algoritmasının ve DGA'nın performansları eşit seviyede olup DABC algoritmasının performansına oldukça yakın seviyededirler. DMBO algoritması ve DGA sırasıyla P32 ve P11 sunumları için bir alt en iyi çözüm üretmişlerdir.

Tablo 4.17. Algoritmaların 5000 katılımcı ve 40 paralel sunumlu oturum sınırlamasız durum için başarılı bir örnek işletim sonrasında elde ettiği oturum dağılımları (P1, P2, …, P20: Sunumlar; S1, S2 ve S3: Oturumlar)

	DMBO			**DABC**			**DGA**		
	S1	**S2**	**S3**	**S1**	**S2**	**S3**	**S1**	**S2**	**S3**
P1	66	67	67	66	67	67	67	66	67
P2	8	8	8	8	8	8	8	8	8
P3	77	77	77	77	77	77	77	77	77
P4	14	13	13	13	13	14	13	14	13
P5	74	74	74	74	74	74	74	74	74
P6	93	93	94	94	93	93	93	93	94
P7	110	109	110	110	109	110	110	109	110
P8	83	82	83	83	82	83	83	82	83
P9	136	136	137	136	136	137	136	137	136
P10	133	132	133	133	132	133	133	132	133
P11	121	121	122	121	122	121	120	122	122
P12	166	165	166	166	165	166	166	165	166
P13	107	106	106	107	106	106	107	106	106
P14	170	170	169	170	169	170	169	170	170
P15	75	75	76	75	76	75	75	75	76
P16	204	205	204	205	204	204	204	204	205
P17	199	199	200	200	199	199	200	199	199
P18	211	211	211	211	211	211	211	211	211
P19	148	147	148	148	148	147	147	148	148
P20	227	228	227	227	228	227	228	227	227
P21	137	138	137	138	137	137	137	138	137
P22	182	183	182	182	183	182	182	183	182
P23	274	275	274	274	275	274	275	274	274
P24	189	189	190	189	190	189	189	190	189
P25	200	201	200	200	201	200	201	200	200
P26	217	217	217	217	217	217	217	217	217
P27	86	86	85	85	86	86	86	86	85
P28	110	110	110	110	110	110	110	110	110
P29	172	171	171	171	172	171	171	172	171
P30	62	61	61	61	62	61	61	62	61
P31	133	133	133	133	133	133	133	133	133
P32	195	197	196	196	196	196	196	196	196
P33	72	72	72	72	72	72	72	72	72
P34	125	125	124	125	124	125	125	125	124
P35	63	63	63	63	63	63	63	63	63
P36	39	38	39	38	39	39	39	38	39
P37	80	80	80	80	80	80	80	80	80
P38	111	111	111	111	111	111	111	111	111
P39	63	63	62	63	62	63	63	62	63
P40	68	69	68	68	68	69	68	69	68

Şekil 4.16 algoritmaların en iyileme süresince elde ettikleri MSE değeri ilerlemesini vererek, onların en iyiye yakınsama karakteristiklerini vermektedir. 2000 İterasyon seviyelerinde DMBO ve DABC algoritmalarının her ikisi de makul en iyileme seviyelerine erişmiştir. DGA ise makul en iyileme seviyelerine daha yavaş yakınsamıştır. Sonuç olarak, bu büyüklükteki bir kombinatoryal en iyileme problemi için her üç algoritmanın da ürettiği sonuçlar makul seviyede en iyi çözümlerdir.



Şekil 4.16. Algoritmaların genişletilmiş vaka çalışmasının oturum sınırlamasız durumu için en iyileme süresince gerçekleştirdikleri MSE ilerlemeleri

4.5. Meta-Sezgisel Algoritma Kullanarak Karar Ağacı İnşası

Bu çalışmada bir karar ağacının oluşturulması hem kombinatoryal hem de nümerik en iyileme işlemlerini içeren bir en iyileme problemi olarak ele alınmıştır. Bu yüzden ABC ve MBO algoritmalarının ayrık ve sürekli versiyonları bu işlemler için seçilmiştir. Ayrık algoritmalar her bir düğüm noktasında hangi özelliğin kullanılacağının seçilmesinde, sürekli algoritmalar ise düğüm noktalarında seçilen özellikler için hangi ayrım değerlerinin kullanılacağının belirlenmesinde kullanılır. Önerilen yöntemin testleri UCI Machine Learning Repository web sitesinden alınan bir veri seti kullanılarak yapılmıştır [222].

4.5.1. ABC algoritması ile karar ağacı tasarımı

Geleneksel ABC algoritmasının Denklem (2.3) ile verilen komşu üretim yöntemi kullanılan değişkenlerin nümerik olmalarını gerektirir. Bu yüzden ayrık veya kombinatoryal değişkenleri kullanan problemlerde farklı bir komşu üretim mekanizmasının kurulması gerekir. Bir problemdeki ayrık değişkenler kümesi

$$X = [x_1, x_2, \ldots, x_N] \tag{4.17}$$

ile ifade edilebilir. Burada N olası ayrık verilerin toplam sayısıdır. X' in ayrık üyeleri için komşuluk üreten bir fonksiyon

$$\text{RandNbr}\ (x_k) = \left\{ x_{k'} \,\middle|\, x_{k'} \in X \ \wedge\ k \neq k' \right\} \tag{4.18}$$

olarak tanımlanabilir. RandNbr fonksiyonu kombinatoryal bir en iyileme probleminin kombinatoryal değişkenlerine komşuluklar üretmek için kullanılabilir. Bu yüzden, Denklem (2.3) kombinatoryal değişkenler için

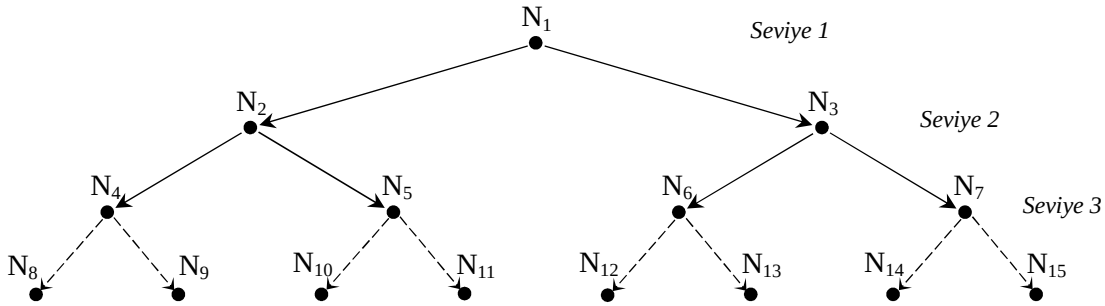
$$\hat{x}_{ij} = \text{RandNbr}\,(x_{ij}) \tag{4.19}$$

olarak değiştirilebilir. Burada x_{ij} i'inci çözümünün j'inci boyutundaki kombinatoryal bir değişkeni ve $\hat{x}_{ij}$ i'inci çözümüne j'inci boyutta üretilen bir komşu değerdir.

Burada önerilen ayrık ABC (DABC) algoritmasının komşu üretim yöntemi haricindeki diğer aşamaları geleneksel ABC algoritması ile aynıdır ve bir karar ağacının düğüm noktalarında kullanılacak özelliklerin değişik kombinasyonlarının denenmesinde kullanımı uygundur.

Karar ağacı tasarımındaki ana hedef yapraklardaki sınıflandırma hatalarını minimize etmektir. Bu düşünceden hareketle, karar ağacı inşa süreci bir en iyileme problemi olarak ele alınabilir. Ayrılma seviyesi sayısı, toplam düğüm noktası sayısı ve toplam yaprak sayısı kullanılacak olan en iyileme algoritmasının girdi parametreleridir. Şekil

4.17 önerilen yaklaşımla tasarlanacak olan üç ayrım seviyeli bir karar ağacı taslağını temsil etmektedir. Kök düğüm olarak adlandırılan N_1 düğümü ilk ayrım seviyesindedir. N_2 ve N_3 iç düğümleri ikinci ayrım seviyesindedirler. N_4'ten N_7'ye kadar olan iç düğümler ise üçüncü ayrım seviyesindedirler. N_8'den N_{15}'e kadar olan düğümler ise yapraklar olup buralarda ayrım işlemi gerçekleşmez.



Şekil 4.17. Üç ayrım seviyeli bir karar ağacı taslağı

Şekil 4.17'deki karar ağacı taslağı analiz edildiğinde ayrılma seviyesi sayısı, toplam düğüm noktası sayısı ve toplam yaprak sayısı arasında bazı matematiksel bağıntıların olduğu görülür. Bu bağıntılar

$$N_{leaf} = 2^{N_{SL}} \qquad \textbf{(4.20.a)}$$

$$N_{node} = 2^{N_{SL}} - 1 \qquad \textbf{(4.20.b)}$$

$$N_{tot} = N_{leaf} + N_{node} = 2^{(N_{SL}+1)} - 1 \qquad \textbf{(4.20.c)}$$

olarak sıralanabilir. Burada N_{SL} ayrılma seviyesi sayısı, N_{leaf} toplam yaprak sayısı, N_{node} yaprakların haricindeki toplam düğüm sayısı ve N_{tot} yapraklar dahil toplam düğüm sayısıdır. Bu taslak karar ağacı konfigürasyonunda her bir düğüm noktasında verilerin maksimum iki alt veri setine parçalandığı varsayılır. Bu alt veri setleri sol alt veri seti ve sağ alt veri seti olarak adlandırılabilirler. Eğer üzerinde işlem yapılan düğüm noktasının indeks numarası n ise sol alt veri setinin indeks numarası $2n$ ve sağ alt veri setinin indeks numarası $(2n + 1)$ olur. Bu ipuçları programlama aşamasında kural setlerinin oluşturulmasında faydalı olacaktır.

Herhangi bir düğüm noktası üzerinde odaklanıldığında, üzerinde karar verilmesi gereken iki önemli husus vardır. Bunlar o düğüm noktasında hangi özelliğin veri ayrımında kullanılacağına karar verilmesi ve seçilen özelliğin hangi eşik değerinin bu veri ayrımında esas alınacağının belirlenmesidir. Birinci husus kombinatoryal bir seçimi ikincisi ise nümerik hesaplamayı gerektirir. Kısacası süreç, hem kombinatoryal hem de nümerik işlemlerin beraberce kullanmasını gerektiren bir karmaşıklığa sahiptir. Bu işlemlerde kullanılan değişkenler

$$A = [a_1, a_2, \ldots, a_{N_{node}}] \tag{4.21}$$

$$T = [t_1, t_2, \ldots, t_{N_{node}}] \tag{4.22}$$

olarak vektör formunda verilebilir. Burada A ve T sırası ile özellik ve eşik değeri vektörleri, a_n ve t_n sırası ile n düğümünde kullanılan özelliği temsil eden kimlik numarası ve bu özelliğin eşik değeridir. Sonuç olarak seçilen en iyileme algoritması arzulanan çıkış elde edilene kadar A ve T matrislerinin ince ayarlarını yapar. Her bir düğüm noktasında ilgili veri setinin iki alt veri setine ayrıldığı düşünülürse, bu alt veri setleri

$$D = [d_1, d_2, \ldots, d_{N_{node}}, d_{N_{leaf}}, \ldots, d_{N_{tot}}] \tag{4.23}$$

gösterim formunda ifade edilebilir. Burada D alt veri setleri vektörü ve d_n n düğümündeki alt veri setidir. D vektörünün her bir d_n üyesinin ilgili n düğümündeki veri setini içeren bir matris olduğuna dikkat edilmelidir. Denklem (4.21) ve Denklem (4.22) birleştirerek, önerilen karar ağacı en iyileme yöntemi için

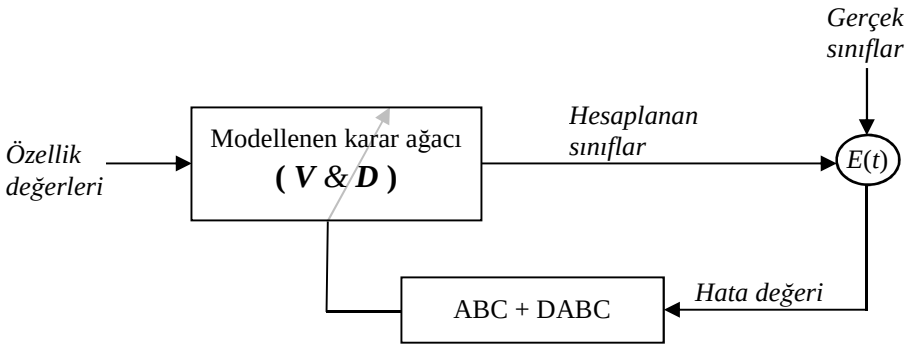
$$V = [a_1, t_1, a_2, t_2, \ldots, a_{N_{node}}\ t_{N_{node}}] \tag{4.24}$$

şeklinde tek bir değişken vektörü elde edilebilir. Bu aşamadan sonra model hatasının hesaplanması büyük önem taşır. Daha önceki yapılan değişken tanımlamalarına sadık kalınarak, eğitim sırasında oluşan karar ağacı kestirim hatası

$$E(t) = \sum_{i=N_{leaf}}^{N_{tot}} \frac{\left(|d_i| - |dc_i| \right)}{|d_i|} \qquad \textbf{(4.25)}$$

olarak hesaplanabilir. Burada $E(t)$ eğitim işleminin t iterasyonundaki sınıflandırma hatası, d_i i karar düğümündeki alt veri seti, mutlak değer işareti ilgili veri setinin içerdiği toplam veri sayısı ve dc_i ilgili alt veri setindeki baskın sınıfın oluşturduğu bir alt veri setidir.

Bu noktadan sonra karar ağacı inşasını bir en iyileme problemi olarak ele alan önerilen yöntem, Şekil 4.18'te verildiği gibi görselleştirilebilir. ABC ve DABC algoritmaları sistematik olarak değişik V vektörü varyasyonlarını deneyerek daha iyi kestirim sonuçları elde etmeye çalışırlar. DABC algoritması değişik özellik kombinasyonları için denemeler yaparken ABC algoritması denenen kombinasyonu oluşturan özelliklerin eşik değerlerini ayarlar. Yani DABC algoritması V vektörünün a_n değişkenleri üzerinde çalışırken ABC algoritması bu vektörün t_n değişkenleri üzerinde çalışır. Önerilen yöntemin her bir iterasyonunda önce V vektörü üzerine ABC ve DABC algoritmalarından hangisinin uygulanacağına karar verilir ve belirlenen algoritma ilgili değişkenler üzerinde işlem yapar.



Şekil 4.18. Önerilen karar ağacı inşa işlemi

Karar stratejisi şu şekilde çalışır. Önce [0,1] aralığında rasgele bir sayı üretilir. Daha sonra üretilen bu sayı önceden belirlenen bir olasılık değeri ile kıyaslanır. Üretilen sayı belirlenen olasılık değerinden küçük ise DABC algoritması V vektörünün kombinatoryal a_n değişkenlerini optimize etmek için kullanılır. Aksi taktirde ABC

algoritması *V* vektörünün nümerik t_n değişkenlerini optimize etmek için kullanılır. Önerilen yöntemin basit sözde kodu Şekil 4.19'da verilmiştir.

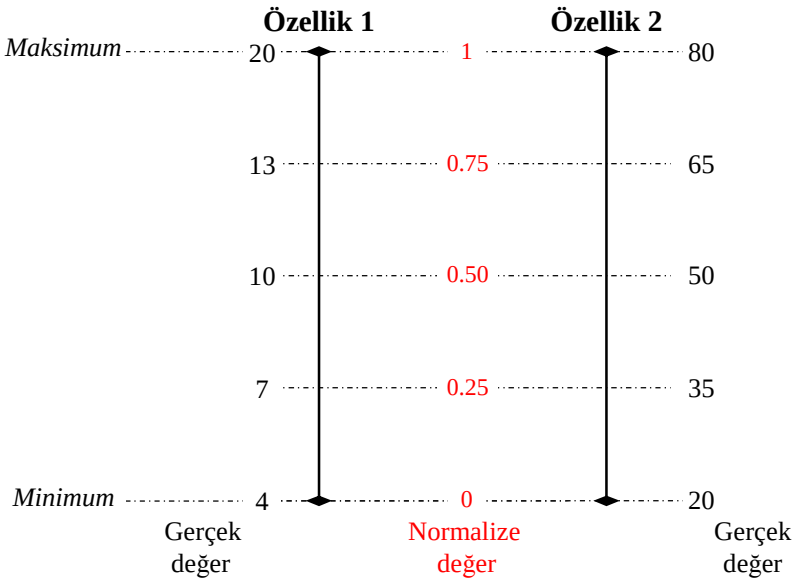
```
Veri seti özellik değerlerini normalize et
V vektörü formatında çözümlerden oluşan başlangıç kaynaklarını oluştur (Denklem 4.24)
Eğitim veri setini kullanarak her bir çözümün maliyet değerini hesapla (Denklem 4.25)
CP kombinatoryal en iyileme ihtimalini belirle
for  i = 1 to maxCycle
        for  j = 1 to PopülasyonNüfusu
                Çözüm j'yi seç
                 Kombinatoryal ve nümerik en iyileme arasında seçim yapmak için
                        [0,1] aralığında rasgele SW değerini üret
                if ( SW < CP )
                        Çözüm j için DABC kullanarak komşu çözüm üret
                else
                        Çözüm j için ABC kullanarak komşu çözüm üret
                endif
                Çözüm j için üretilen komşu çözümün maliyet değerini hesapla
                Çözüm j komşu çözüm ile iyileştirilebilmiş ise çözüm j'yi komşu
                        çözüm ile güncelle
        endfor
        Eğer iterasyonun en iyi çözümü algoritmanın en iyi çözümünden daha iyi ise
                algoritmanın en iyi çözümünü iterasyonun en iyi çözümü ile güncelle
endfor

En iyi çözümü kullanarak Karar Ağacı Kurallar Setini oluştur
Return "Karar Ağacı Kurallar Seti"
```

Şekil 4.19. Önerilen karar ağacı inşa yönteminin sözde kodu

Metodun işleyişi kısaca şöyledir. İlk olarak, kombinatoryal en iyilemeler gerçekleştirilirken meydana gelen özellik değişimlerinden kaynaklanan gerekli nümerik dönüşümlerden kurtulmak için bütün veri seti içeriği [0,1] aralığına normalize edilir. Aksi halde DABC tarafından bir noktada kullanılan özellik başka bir özellik ile değiştirildiğinde, önceki özelliğin değerinin yeni özellikteki karşılığının hesaplanıp yeni özelliğe atanması gerekir. Kombinatoryal en iyileme ile her özellik değişiminde bu işlemin tekrarlanmasının bir hesaplama maliyeti vardır.

Diğer taraftan özelliklere ait normalize edilmiş değerler zaten belirli bir aralıkta oransal bir ifadeyi temsil ettikleri için türden ve birimden bağımsızdırlar. Şekil 4.20'de görsel olarak da ifade edildiği gibi normalize edilmiş değerler için özellik değişimlerinde değer dönüşümlerine ihtiyaç duyulmaz.



Şekil 4.20. İki farklı özelliğin normalize edilmiş ve normalize edilmemiş değerleri

İkinci olarak, Denklem (4.24)'te verilen vektör formunda üyelerden oluşan çözüm seti oluşturulur. Üçüncü olarak ise her bir çözümün maliyet (hata) değeri Denklem (4.25) kullanılarak hesaplanır. Daha sonra ana döngü içerisinde iterasyon işlemleri başlar.

Ana döngü içerisinde bütün çözümler ABC algoritmasının kullanıldığı bir nümerik en iyileme vasıtasıyla veya DABC algoritmasının kullanıldığı bir kombinatoryal en iyileme ile iyileştirilmeye çalışılırlar. Bu maksatla sıradaki çözüm üzerinde işlem yapmadan önce [0,1] aralığında rasgele bir sayı (SW) üretilir. Bu sayı başlangıçta belirlenen CP limit olasılık değeri ile karşılaştırılır. Eğer $SW < CP$ ise DABC algoritması sıradaki çözüme uygulanır. Aksi halde ABC algoritması bütün çözümlere uygulanır. Böylece mevcut çözümler iyileştirilmeye çalışılır. Her bir iyileştirme denemesi sonrasında yeni üretilen komşu çözümün maliyet değeri hesaplanır. Eğer hata değerinde bir düşüş var ise elde edilen komşu çözüm ilgili çözüme atanır. Son olarak, çözüm seti içerisindeki en iyi çözümün kâşif arı evresinde daha kötü bir

çözüme dönüşme ihtimali göz önünde tutulur ve bu çözüm algoritma geçmişinde saklanan en iyi çözümden daha iyi ise algoritma geçmişinin en iyi çözümü olarak saklanır. Ana döngü tamamlandıktan sonra, algoritma geçmişinin en iyi çözümü karar ağacının kurallar setini oluşturmada kullanılır.

Karar ağacının kurallar setinin oluşturulmasında kullanılacak sistematik üzerinde durulması gereken diğer bir önemli konudur. Denklem (4.23)'te tanımlanan alt veri setlerinde boş olmayan alt veri setlerini takip ederek kurallar setinin oluşturulması mümkündür.

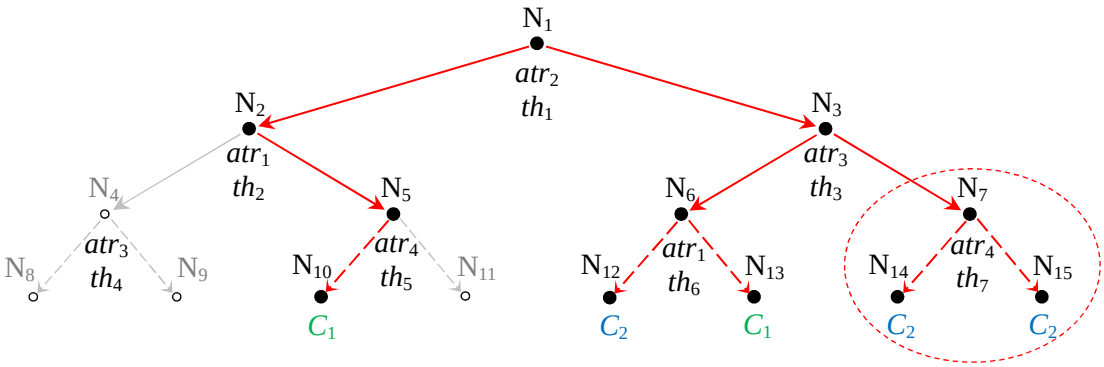
***Örnek*:** Üç seviyeli bir karar ağacının en iyileme işlemi tamamlandıktan sonra V vektörü için üretilen en iyi çözüm

$$V = [atr_2, th_1, atr_1, th_2, atr_3, th_3, atr_3, th_4, atr_4, th_5, atr_1, th_6, atr_4, th_7\ ] \quad \textbf{(4.26)}$$

olsun ve V vektörü içerisinde tanımlı özellik karşılaştırmalarını yaptıktan sonra elde edilen alt veri seti vektörü

$$D = [d_1, d_2, d_3, \phi, d_5, d_6, d_7, \phi, \phi, d_{10}, \phi, d_{12}, d_{13}, d_{14}, d_{15}] \quad \textbf{(4.27)}$$

olarak elde edilsin. Bu durumda, verilen örneğe ait basit karar ağacı gösterimi Şekil 4.21'deki gibi olur.



Şekil 4.21. Örnek uygulamanın taslak karar ağacı gösterimi

Şekilde, boş olmayan bir alt veri setine sahip olan düğüm noktaları "●" işareti ile, boş bir alt veri setine sahip olan düğüm noktaları ise "○" işareti ile temsil edilmiştir. C_1 ve C_2 verilen yapraklardaki baskın sınıflar olsun. Sadece boş olmayan alt veri setlerinin bulunduğu düğümleri kullanmak şartı ile kök düğümden başlayan, iç düğümlerden geçen ve yapraklarda son bulan toplam beş farklı yol vardır. Bu yollar ve yolların temsil ettiği kurallar Tablo 4.18'de verilmiştir.

Tablo 4.18. Taslak karar ağacındaki anlamlı yollar ve nihai karar ağacının kuralları

Yol	Güzergâh	Kural	Sadeleştirilmiş kural
1	$N_1N_2N_5N_{10}$	$(atr_2 < th_1) \wedge (atr_1 \geq th_2) \wedge (atr_4 < th_5) \rightarrow C_1$	Aynı
2	$N_1N_3N_6N_{12}$	$(atr_2 \geq th_1) \wedge (atr_3 < th_3) \wedge (atr_1 < th_6) \rightarrow C_2$	Aynı
3	$N_1N_3N_6N_{13}$	$(atr_2 \geq th_1) \wedge (atr_3 < th_3) \wedge (atr_1 \geq th_6) \rightarrow C_1$	Aynı
4	$N_1N_3N_7N_{14}$	$(atr_2 \geq th_1) \wedge (atr_3 \geq th_3) \wedge (atr_4 < th_7) \rightarrow C_2$	$(atr_2 \geq th_1) \wedge (atr_3 \geq th_3) \rightarrow C_2$
5	$N_1N_3N_7N_{15}$	$(atr_2 \geq th_1) \wedge (atr_3 \geq th_3) \wedge (atr_4 \geq th_7) \rightarrow C_2$	

4 ve 5 numaralı yolların, son ayrışmadan sonra aynı sınıfın baskın olduğu alt veri setlerinin bulunduğu yapraklara ulaştığı görülmektedir. Bu gibi durumlarda son kontrolün yapıldığı kural anlamını yitirdiğinden, kural çıkarımı son ayrışmanın gerçekleştiği düğümde (N_7) sonlandırılabilir. Dolayısıyla bu örnekte atr_4 özelliği için yapılan son kontroller ilgili kurallardan çıkarılarak kurallar sadeleştirilebilir. Bu durum N_{14} ve N_{15} yapraklarının budanması olarak da adlandırılabilir.

4.5.1.1. Kullanılan veri seti

Önerilen yöntemin performansının denenmesi için UCI veri tabanında yer alan bir tiroit veri seti kullanılmıştır [222]. Bu veri seti daha önce üzerinde yaygın olarak çalışılan ve performans testlerinde sıkça kullanılmış olan bir veri setidir [121-122,163,251-254]. Bu sebeple, hem önerilen yöntemin test edilmesi hem de yöntemin benzeri çalışmalarla kolayca mukayese edilebilmesi maksadıyla bu veri setinin kullanılması tercih edilmiştir. Veri seti, her biri 6 adet alan verisi içeren toplam 215 adet ölçüm değeri içermektedir. İçeriği Tablo 4.19'da özetlenen veri setinin alanlarından birisi sınıf verisi, diğerleri ise özellik verileridir.

Tablo 4.19. Tiroit veri seti alan bilgileri

Alan	Tür	Açıklama
1	Özellik	T3-reçine testi
2	Özellik	İzotopik değişim yöntemi ile ölçülen toplam serum tiroksin
3	Özellik	Radyoimüno deneyi ile ölçülen toplam serum triiyodotironin
4	Özellik	Radyoimüno deneyi ile ölçülen bazal TSH
5	Özellik	200 mg tirotropin salıveren hormon enjeksiyonu sonrasında bazal değerine kıyasla maksimum mutlak TSH değeri farkı
6	Sınıf	Sınıf 1 : Normal (150 örnek) Sınıf 2 : Hipertiroit (35 örnek) Sınıf 3 : Hipotiroit (30 örnek)

4.5.1.2. Test sonuçları

Tablo 4.20'de 5, 6, 7 ve 8 ayrım seviyeli karar ağacı tasarımlarından elde edilen sonuçlar görülmektedir. Tablodaki hata ve doğruluk değerleri 10-tekrarlı çapraz doğrulama yöntemi için elde edilen ortalama değerlerdir. Genel hata ve doğruluk değerleri hesaplanarak tablonun en alt satırında verilmiştir.

Tablo 4.20. Her bir alt (fold) uygulamanın ortalama hata ve doğruluk değerleri ile genel performans değerleri (*PS*: Popülasyon büyüklüğü, *MC*: Maksimum iterasyon)

	5 Ayrım Seviyeli (*PS* = 70, *MC* = 5000)		**6 Ayrım Seviyeli** (*PS* = 100, *MC* = 7000)		**7 Ayrım Seviyeli** (*PS* = 130, *MC* = 9000)		**8 Ayrım Seviyeli** (*PS* = 150, *MC* = 12000)	
Uygulama	***Hata***	***Doğruluk***	***Hata***	***Doğruluk***	***Hata***	***Doğruluk***	***Hata***	***Doğruluk***
F1	0	91.818	0	90.455	0	88.182	0	90.455
F2	5.1813E-04	100.000	0	100.000	0	98.182	0	100.000
F3	5.1813E-04	100.000	0	97.727	0	100.000	0	99.546
F4	5.1813E-04	94.091	0	97.273	0	96.364	0	94.091
F5	5.1813E-04	92.273	0	92.727	0	94.546	0	93.636
F6	5.1546E-04	95.714	0	95.714	0	95.238	0	94.762
F7	5.1546E-04	99.524	0	98.095	0	98.571	0	97.619
F8	0	96.191	0	94.286	0	98.571	0	95.714
F9	5.1546E-04	97.143	0	95.238	0	95.714	0	96.667
F10	1.0309E-03	91.429	0	91.905	0	93.333	0	91.429
Genel Ortalama	**4.6499E-04**	**95.818**	**0**	**95.342**	**0**	**95.870**	**0**	**95.392**

Tablo 4.20'de verilen her bir uygulamanın farklı ayrım seviyelerindeki sonuçları analiz edildiğinde ayrım seviyesi sayısındaki artışın karar ağacı doğruluk sonuçlarını

çok fazla etkilemediği görülmektedir. Eğitim sonrası erişilen hata oranı 5 ayrım seviyesi için makul seviyelerde iken 6, 7 ve 8 ayrım seviyelerinde sıfır olarak gerçekleşmiştir. Ayrım seviyesinin artışının eğitim esnasındaki hesaplama maliyetini artırdığı göz önünde bulundurulduğunda ayrım seviyesinin mümkün olduğunca düşük tutulması daha makul tasarımlar elde edilmesini sağlar. Dolayısıyla bu veri seti için 5 ayrım seviyeli karar ağacı tasarımı en makul tasarım olarak görülmektedir.

Tablo 4.21, Tablo 4.22, Tablo 4.23 ve Tablo 4.24'te önerilen yöntem ile elde edilmiş, sırasıyla 5, 6, 7 ve 8 ayrım seviyeli karar ağacı tasarımları için olası kural setleri örnek olarak verilmiştir. Bu kural setleri %100 doğruluk değerine sahip olup, ham haldeki karar ağacı tasarımlarıdır. Yani önerilen yöntemin oluşturduğu kural seti üzerinde herhangi bir sadeleştirilme (budama) işlemi yapılmamıştır.

Tablo 4.21. Tiroit veri seti için inşa edilen 5 Seviyeli karar ağacı kurallar setine bir örnek

Kural	**Kural Tanımı**	**Sınıf**
R_1	(A2<=0.24449)^(A5<=0.10602)^(A5<=0.099656)^(A2<=0.14643)^(A1<=0.76634)	C3
R_2	(A2<=0.24449)^(A5<=0.10602)^(A5<=0.099656)^(A2<=0.14643)^(A1>0.76634)	C3
R_3	(A2<=0.24449)^(A5<=0.10602)^(A5<=0.099656)^(A2>0.14643)^(A3<=0.84744)	C1
R_4	(A2<=0.24449)^(A5<=0.10602)^(A5>0.099656)^(A5<=0.10818)^(A1>0.27581)	C3
R_5	(A2<=0.24449)^(A5>0.10602)^(A4<=0.07895)^(A3<=0.30251)^(A3<=0.13593)	C3
R_6	(A2<=0.24449)^(A5>0.10602)^(A4<=0.07895)^(A3<=0.30251)^(A3>0.13593)	C1
R_7	(A2<=0.24449)^(A5>0.10602)^(A4>0.07895)^(A4>0)^(A2<=0.54449)	C3
R_8	(A2>0.24449)^(A3<=0.30794)^(A5<=0.019863)^(A2<=0.43783)^(A1<=0.38002)	C2
R_9	(A2>0.24449)^(A3<=0.30794)^(A5<=0.019863)^(A2<=0.43783)^(A1>0.38002)	C1
R_{10}	(A2>0.24449)^(A3<=0.30794)^(A5<=0.019863)^(A2>0.43783)^(A3<=1)	C2
R_{11}	(A2>0.24449)^(A3<=0.30794)^(A5>0.019863)^(A5>0)^(A3<=1)	C1
R_{12}	(A2>0.24449)^(A3>0.30794)^(A5<=0.62108)^(A1<=0.2942)^(A3>0.33712)	C2
R_{13}	(A2>0.24449)^(A3>0.30794)^(A5<=0.62108)^(A1>0.2942)^(A3<=0.50466)	C2
R_{14}	(A2>0.24449)^(A3>0.30794)^(A5<=0.62108)^(A1>0.2942)^(A3>0.50466)	C2

Tablo 4.22. Tiroit veri seti için inşa edilen 6 Seviyeli karar ağacı kurallar setine bir örnek

Kural	**Kural Tanımı**	**Sınıf**
R_1	(A4 <= 0.62514) ∧ (A2 <= 0.19988) ∧ (A3 <= 0.17061) ∧ (A2 <= 0.4823) ∧ (A3 <= 0.061847) ∧ (A2 <= 0.24212)	C3
R_2	(A4 <= 0.62514) ∧ (A2 <= 0.19988) ∧ (A3 <= 0.17061) ∧ (A2 <= 0.4823) ∧ (A3 > 0.061847) ∧ (A5 <= 0.045964)	C1
R_3	(A4 <= 0.62514) ∧ (A2 <= 0.19988) ∧ (A3 <= 0.17061) ∧ (A2 <= 0.4823) ∧ (A3 > 0.061847) ∧ (A5 > 0.045964)	C3
R_4	(A4 <= 0.62514) ∧ (A2 <= 0.19988) ∧ (A3 > 0.17061) ∧ (A2 <= 0.71161) ∧ (A1 > 0.39393) ∧ (A1 > 0.087603)	C3
R_5	(A4 <= 0.62514) ∧ (A2 > 0.19988) ∧ (A2 <= 0.38211) ∧ (A5 <= 0.1666) ∧ (A1 <= 0.96488) ∧ (A3 > 0)	C1
R_6	(A4 <= 0.62514) ∧ (A2 > 0.19988) ∧ (A2 <= 0.38211) ∧ (A5 > 0.1666) ∧ (A2 <= 0.3026) ∧ (A2 <= 0.5114)	C3
R_7	(A4 <= 0.62514) ∧ (A2 > 0.19988) ∧ (A2 <= 0.38211) ∧ (A5 > 0.1666) ∧ (A2 > 0.3026) ∧ (A3 <= 0.74249)	C1
R_8	(A4 <= 0.62514) ∧ (A2 > 0.19988) ∧ (A2 > 0.38211) ∧ (A2 <= 0.54963) ∧ (A3 <= 0.26908) ∧ (A1 <= 0.39371)	C2
R_9	(A4 <= 0.62514) ∧ (A2 > 0.19988) ∧ (A2 > 0.38211) ∧ (A2 <= 0.54963) ∧ (A3 <= 0.26908) ∧ (A1 > 0.39371)	C1
R_{10}	(A4 <= 0.62514) ∧ (A2 > 0.19988) ∧ (A2 > 0.38211) ∧ (A2 <= 0.54963) ∧ (A3 > 0.26908) ∧ (A4 <= 0.80908)	C2
R_{11}	(A4 <= 0.62514) ∧ (A2 > 0.19988) ∧ (A2 > 0.38211) ∧ (A2 > 0.54963) ∧ (A5 <= 0.037551) ∧ (A1 <= 0.5562)	C2
R_{12}	(A4 <= 0.62514) ∧ (A2 > 0.19988) ∧ (A2 > 0.38211) ∧ (A2 > 0.54963) ∧ (A5 <= 0.037551) ∧ (A1 > 0.5562)	C2
R_{13}	(A4 <= 0.62514) ∧ (A2 > 0.19988) ∧ (A2 > 0.38211) ∧ (A2 > 0.54963) ∧ (A5 > 0.037551) ∧ (A2 > 0.48348)	C1
R_{14}	(A4 > 0.62514) ∧ (A1 > 0.12434) ∧ (A3 <= 0.57205) ∧ (A3 <= 0.83808) ∧ (A3 <= 0.27506) ∧ (A4 <= 0.81554)	C3
R_{15}	(A4 > 0.62514) ∧ (A1 > 0.12434) ∧ (A3 <= 0.57205) ∧ (A3 <= 0.83808) ∧ (A3 <= 0.27506) ∧ (A4 > 0.81554)	C3

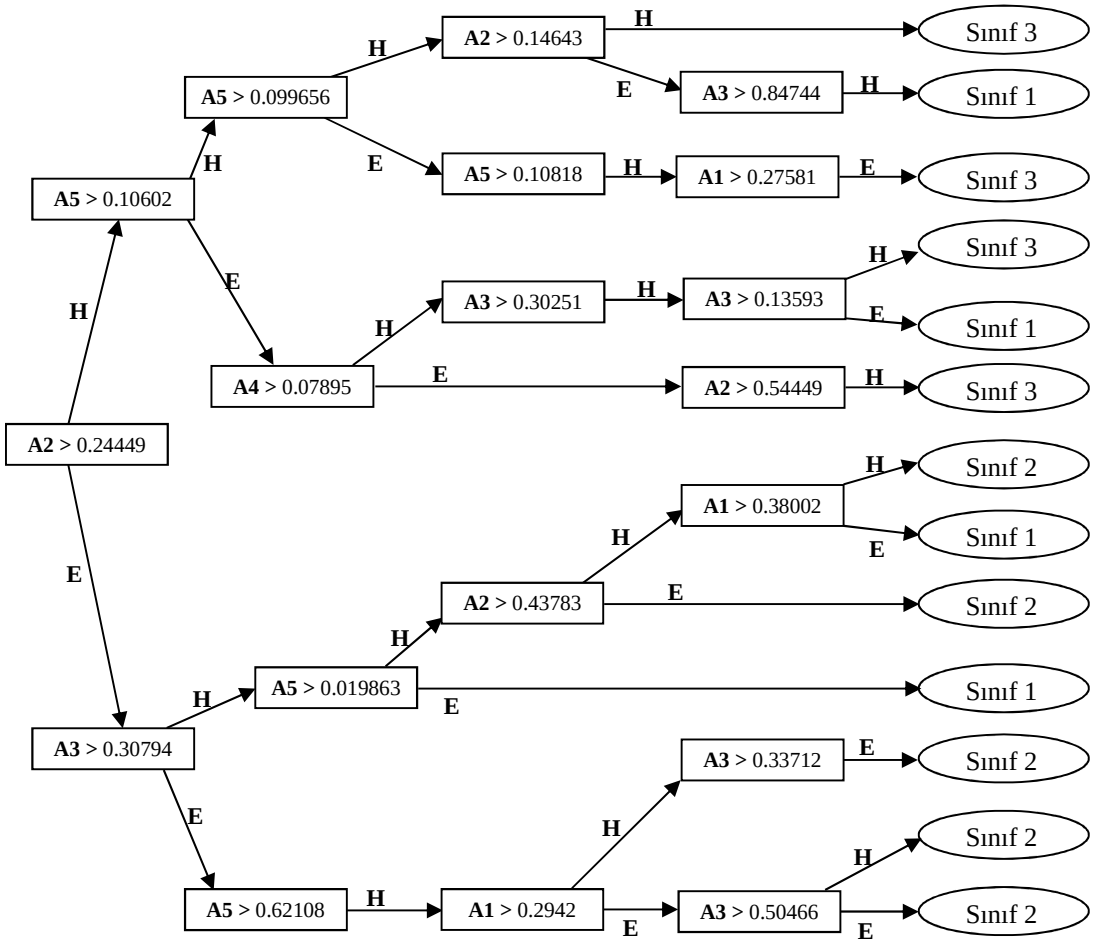
Tablo 4.23. Tiroit veri seti için inşa edilen 7 Seviyeli karar ağacı kurallar setine bir örnek

Kural	**Kural Tanımı**	**Sınıf**
R_1	(A3 <= 0.35176) ∧ (A3 <= 0.27703) ∧ (A2 <= 0.24308) ∧ (A5 <= 0.097381) ∧ (A5 <= 0.80924) ∧ (A4 <= 0.96568) ∧ (A2 <= 0.13607)	C3
R_2	(A3 <= 0.35176) ∧ (A3 <= 0.27703) ∧ (A2 <= 0.24308) ∧ (A5 <= 0.097381) ∧ (A5 <= 0.80924) ∧ (A4 <= 0.96568) ∧ (A2 > 0.13607)	C1
R_3	(A3 <= 0.35176) ∧ (A3 <= 0.27703) ∧ (A2 <= 0.24308) ∧ (A5 > 0.097381) ∧ (A3 <= 0.18007) ∧ (A1 <= 0.9147) ∧ (A4 <= 0.24346)	C3
R_4	(A3 <= 0.35176) ∧ (A3 <= 0.27703) ∧ (A2 <= 0.24308) ∧ (A5 > 0.097381) ∧ (A3 <= 0.18007) ∧ (A1 <= 0.9147) ∧ (A4 > 0.24346)	C3
R_5	(A3 <= 0.35176) ∧ (A3 <= 0.27703) ∧ (A2 <= 0.24308) ∧ (A5 > 0.097381) ∧ (A3 <= 0.18007) ∧ (A1 > 0.9147) ∧ (A5 <= 1)	C3
R_6	(A3 <= 0.35176) ∧ (A3 <= 0.27703) ∧ (A2 <= 0.24308) ∧ (A5 > 0.097381) ∧ (A3 > 0.18007) ∧ (A2 <= 0.38728) ∧ (A2 > 0.012741)	C1
R_7	(A3 <= 0.35176) ∧ (A3 <= 0.27703) ∧ (A2 > 0.24308) ∧ (A2 <= 0.55545) ∧ (A2 <= 0.40908) ∧ (A5 <= 0.65752) ∧ (A4 <= 0.37235)	C1
R_8	(A3 <= 0.35176) ∧ (A3 <= 0.27703) ∧ (A2 > 0.24308) ∧ (A2 <= 0.55545) ∧ (A2 > 0.40908) ∧ (A1 <= 0.38503) ∧ (A4 <= 0.33707)	C2
R_9	(A3 <= 0.35176) ∧ (A3 <= 0.27703) ∧ (A2 > 0.24308) ∧ (A2 <= 0.55545) ∧ (A2 > 0.40908) ∧ (A1 > 0.38503) ∧ (A4 <= 0.73614)	C1
R_{10}	(A3 <= 0.35176) ∧ (A3 <= 0.27703) ∧ (A2 > 0.24308) ∧ (A2 > 0.55545) ∧ (A1 <= 0.61425) ∧ (A3 <= 0.78209) ∧ (A3 <= 0.69625)	C2
R_{11}	(A3 <= 0.35176) ∧ (A3 <= 0.27703) ∧ (A2 > 0.24308) ∧ (A2 > 0.55545) ∧ (A1 > 0.61425) ∧ (A1 > 0.57327) ∧ (A3 <= 0.74016)	C1
R_{12}	(A3 <= 0.35176) ∧ (A3 > 0.27703) ∧ (A1 > 0.038357) ∧ (A3 <= 0.79043) ∧ (A4 <= 0.43675) ∧ (A2 <= 0.38237) ∧ (A5 <= 0.94109)	C1
R_{13}	(A3 <= 0.35176) ∧ (A3 > 0.27703) ∧ (A1 > 0.038357) ∧ (A3 <= 0.79043) ∧ (A4 <= 0.43675) ∧ (A2 > 0.38237) ∧ (A3 <= 0.79726)	C2
R_{14}	(A3 > 0.35176) ∧ (A2 <= 1) ∧ (A3 <= 0.80454) ∧ (A3 > 0) ∧ (A5 <= 0.17831) ∧ (A4 <= 1) ∧ (A2 <= 0.66182)	C2
R_{15}	(A3 > 0.35176) ∧ (A2 <= 1) ∧ (A3 <= 0.80454) ∧ (A3 > 0) ∧ (A5 <= 0.17831) ∧ (A4 <= 1) ∧ (A2 > 0.66182)	C2
R_{16}	(A3 > 0.35176) ∧ (A2 <= 1) ∧ (A3 > 0.80454) ∧ (A4 <= 0.16487) ∧ (A2 > 0.30573) ∧ (A4 <= 0.55064) ∧ (A4 <= 0.32369)	C2
R_{17}	(A3 > 0.35176) ∧ (A2 > 1) ∧ (A2 > 0.22656) ∧ (A4 <= 0.44187) ∧ (A3 > 0) ∧ (A1 <= 0.735) ∧ (A2 > 0.53386)	C2

Tablo 4.24. Tiroit veri seti için inşa edilen 8 Seviyeli karar ağacı kurallar setine bir örnek

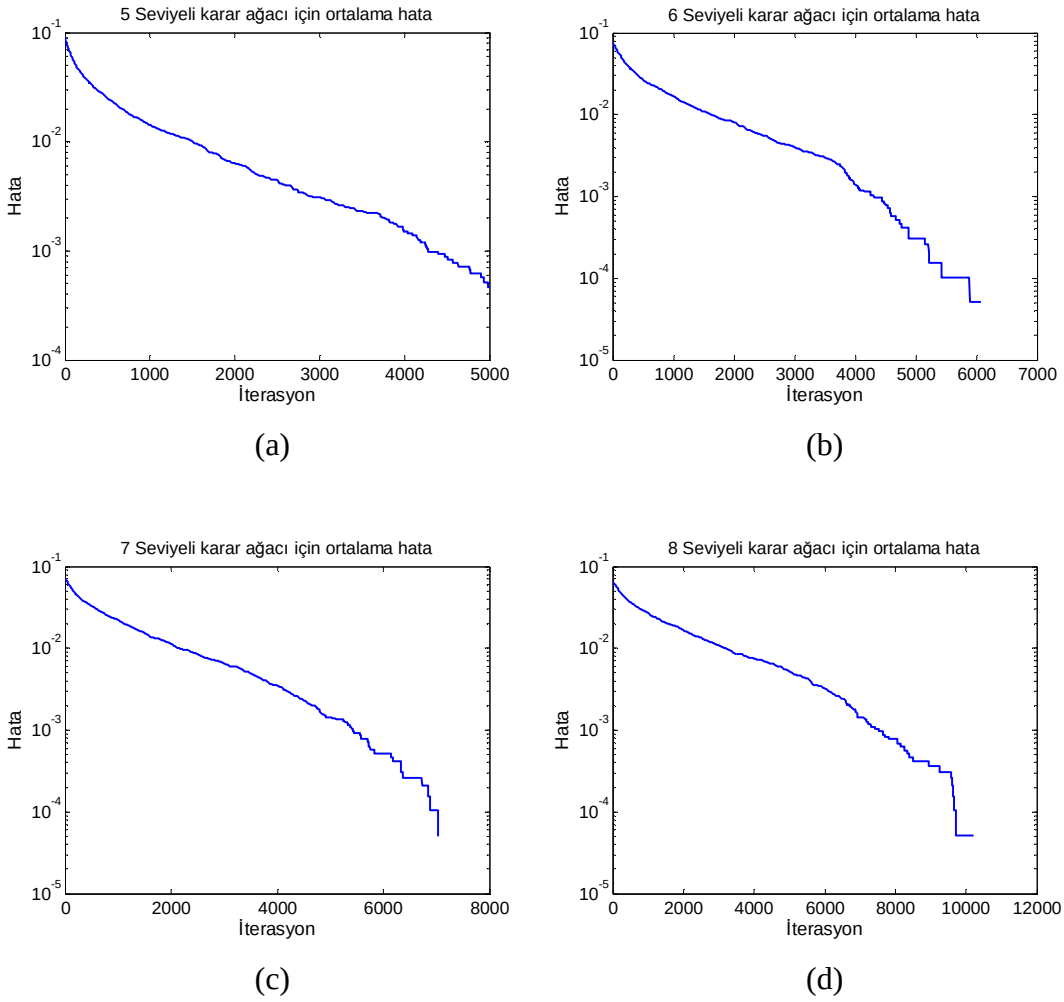
Kural	**Kural Tanımı**	**Sınıf**
R_1	(A2<=0.4092) ∧ (A3<=0.25131) ∧ (A5<=0.043841) ∧ (A4<=0.25632) ∧ (A4<=0.50517) ∧ (A5<=0.061599) ∧ (A2<=0.14898) ∧ (A4<=0.11319)	C3
R_2	(A2<=0.4092) ∧ (A3<=0.25131) ∧ (A5<=0.043841) ∧ (A4<=0.25632) ∧ (A4<=0.50517) ∧ (A5<=0.061599) ∧ (A2>0.14898) ∧ (A3<=0.51074)	C1
R_3	(A2<=0.4092) ∧ (A3<=0.25131) ∧ (A5>0.043841) ∧ (A3<=0.42563) ∧ (A5<=0.16507) ∧ (A2<=0.20568) ∧ (A3<=0.25104) ∧ (A4<=0.91406)	C3
R_4	(A2<=0.4092) ∧ (A3<=0.25131) ∧ (A5>0.043841) ∧ (A3<=0.42563) ∧ (A5<=0.16507) ∧ (A2>0.20568) ∧ (A5<=0.6957) ∧ (A3>0)	C1
R_5	(A2<=0.4092) ∧ (A3<=0.25131) ∧ (A5>0.043841) ∧ (A3<=0.42563) ∧ (A5>0.16507) ∧ (A2<=0.30595) ∧ (A2<=0.58359) ∧ (A5>0)	C3
R_6	(A2<=0.4092) ∧ (A3<=0.25131) ∧ (A5>0.043841) ∧ (A3<=0.42563) ∧ (A5>0.16507) ∧ (A2>0.30595) ∧ (A4<=0.82991) ∧ (A2<=0.80766)	C1
R_7	(A2<=0.4092) ∧ (A3>0.25131) ∧ (A1>0) ∧ (A5<=0.97516) ∧ (A3>0) ∧ (A5<=0.39706) ∧ (A2<=0.42954) ∧ (A5<=1)	C1
R_8	(A2>0.4092) ∧ (A5<=0.035297) ∧ (A1<=0.37657) ∧ (A3<=0.39394) ∧ (A2<=0.7243) ∧ (A5<=0.27889) ∧ (A5<=0.13069) ∧ (A5<=0.23242)	C2
R_9	(A2>0.4092) ∧ (A5<=0.035297) ∧ (A1<=0.37657) ∧ (A3<=0.39394) ∧ (A2>0.7243) ∧ (A2>0) ∧ (A2<=0.93859) ∧ (A4<=1)	C2
R_{10}	(A2>0.4092) ∧ (A5<=0.035297) ∧ (A1<=0.37657) ∧ (A3>0.39394) ∧ (A2>0.44283) ∧ (A5<=1) ∧ (A1<=0.70499) ∧ (A5<=0.43581)	C2
R_{11}	(A2>0.4092) ∧ (A5<=0.035297) ∧ (A1>0.37657) ∧ (A1<=1) ∧ (A2>0.22653) ∧ (A2<=0.53446) ∧ (A3<=0.24999) ∧ (A4<=0.66158)	C1
R_{12}	(A2>0.4092) ∧ (A5<=0.035297) ∧ (A1>0.37657) ∧ (A1<=1) ∧ (A2>0.22653) ∧ (A2<=0.53446) ∧ (A3>0.24999) ∧ (A2<=0.98929)	C2
R_{13}	(A2>0.4092) ∧ (A5<=0.035297) ∧ (A1>0.37657) ∧ (A1<=1) ∧ (A2>0.22653) ∧ (A2>0.53446) ∧ (A3>0) ∧ (A3<=0.74171)	C2
R_{14}	(A2>0.4092) ∧ (A5<=0.035297) ∧ (A1>0.37657) ∧ (A1>1) ∧ (A1>0.77544) ∧ (A5<=0.68887) ∧ (A1>1) ∧ (A3>0)	C2
R_{15}	(A2>0.4092) ∧ (A5>0.035297) ∧ (A4<=0.061566) ∧ (A4<=0.69771) ∧ (A5<=1) ∧ (A2>0.031248) ∧ (A3<=0.18601) ∧ (A3<=1)	C1
R_{16}	(A2>0.4092) ∧ (A5>0.035297) ∧ (A4<=0.061566) ∧ (A4<=0.69771) ∧ (A5<=1) ∧ (A2>0.031248) ∧ (A3>0.18601) ∧ (A5<=1)	C1

Tablo 4.21'de verilen kural setine ait 5 seviyeli karar ağacı Şekil 4.22'de verilmiştir. Bu karar ağacı oluşturulmadan önce Tablo 4.21'de verilen kural setinde bazı basit sadeleştirmeler yapılmıştır. Örneğin; R_1 ve R_2 Tablo 4.18'dekine benzer şekilde birleştirilerek budama işlemi yapılmıştır. Ayrıca, (0,1) aralığına normalize edilmiş değerler için "< 1" ve "> 0" koşulları her zaman doğru sonuç vereceğinden R_7, R_{10} ve R_{11} kuralları içerisinden bu şartlar çıkarılmıştır.



Şekil 4.22. Tiroit veri seti için, önerilen metot ile inşa edilmiş 5 seviyeli karar ağacına bir örnek (Tablo 4.21'deki kurallar setinin karar ağacı gösterimi)

Şekil 4.23 önerilen metodun, karar ağacı inşa (eğitim) sürecinde elde ettiği ortalama hata ilerlemelerini göstermektedir. Grafiklerde görüldüğü gibi, önerilen yöntemle 5 ayrım seviyeli karar ağacı tasarımında 5000 iterasyon makul bir hata seviyesine erişmek için yeterli gelmektedir. 6, 7 ve 8 ayrım seviyeli karar ağacı tasarımlarında ise sırasıyla 6000, 7000 ve 10500 iterasyon seviyelerinde sıfır hata değerine ulaşılmaktadır.



Şekil 4.23. Önerilen metot ile (a) 5, (b) 6, (c) 7 ve (d) 8 seviyeli karar ağacı inşa süreçlerindeki hata ilerlemeleri

4.5.2. MBO algoritması ile karar ağacı tasarımı ve test sonuçları

Bölüm 4.5.1'deki Denklem (4.19)'da ABC algoritması için önerilen ve kombinatoryal verilere komşu üretmek amacıyla kullanılan komşuluk üretme yöntemi MBO algoritması ile ayrık veriler üzerinde çalışırken de kullanıldı. Bu yöntemle elde edilen özelliklere ait nümerik ayrım değerleri için komşuluk değerleri üretilirken Bölüm 2.2'de anlatılan MBO algoritmasının kendi komşu üretme sistematiği kullanıldı.

En yüksek doğruluk değerine erişilen beş ayrım seviyeli karar ağacı yapısı oluşturulurken elde edilen ortalama 10-tekrarlı çapraz sorgulama neticeleri Tablo 4.25'te, bu yolla üretilen beş ayrım seviyeli örnek bir karar ağacının kurallar seti ise

Tablo 4.26'da verilmiştir. Elde edilen performans sonuçlarının ABC algoritması ile elde edilenlere yakın olduğu görülmektedir.

Tablo 4.25. Her bir alt (fold) uygulamanın ortalama hata ve doğruluk değerleri ile genel performans değerleri (*k*: komşuluk sayısı, *x*: paylaşılan komşuluk ayısı, *PS*: Popülasyon büyüklüğü, *MC*: Maksimum iterasyon)

	5 Ayrım Seviyeli (*k* = 3, *x* = 1, *PS* = 51, *MC* = 5000)	
Uygulama	***Hata***	***Doğruluk***
F1	0.0052	98.1818
F2	0.0309	91.8182
F3	0.0155	92.2727
F4	0.0052	96.3636
F5	0.0258	96.3636
F6	0.0155	96.6667
F7	0.0155	95.7143
F8	0.0155	97.6190
F9	0.0103	94.7619
F10	0.0052	90.9524
Genel Ortalama	**0.0144**	**95.0714**

Tablo 4.26. Tiroit veri seti için inşa edilen 5 Seviyeli karar ağacı kurallar setine bir örnek

Kural	**Kural Tanımı**	**Sınıf**
R_1	A2<=0.49221 & A2<=0.15 & A2<=0.57178 & A1<=0.54405 & A1<=0.70556	C1
R_2	A2<=0.49221 & A2<=0.15 & A2<=0.57178 & A1> 0.54405 & A1> 0.10143	C3
R_3	A2<=0.49221 & A2> 0.15 & A2<=0.28283 & A1> 0 & A5<=0.20577	C1
R_4	A2<=0.49221 & A2> 0.15 & A2<=0.28283 & A1> 0 & A5> 0.20577	C3
R_5	A2<=0.49221 & A2> 0.15 & A2> 0.28283 & A5<=0.55991 & A1<=0.98798	C1
R_6	A2> 0.49221 & A3<=0.36175 & A5<=0.036179 & A1<=0.67443 & A2> 0.11764	C2
R_7	A2> 0.49221 & A3<=0.36175 & A5<=0.036179 & A1> 0.67443 & A2<=0.64886	C1
R_8	A2> 0.49221 & A3<=0.36175 & A5<=0.036179 & A1> 0.67443 & A2> 0.64886	C2
R_9	A2> 0.49221 & A3<=0.36175 & A5> 0.036179 & A2<=0.94038 & A4<=0.30546	C1
R_{10}	A2> 0.49221 & A3> 0.36175 & A5<=0.63816 & A2> 0.24545 & A3<=0.83206	C2
R_{11}	A2> 0.49221 & A3> 0.36175 & A5<=0.63816 & A2> 0.24545 & A3> 0.83206	C2

4.5.3. Meta-sezgisel karar ağaçlarının başarı mukayesesi

Tablo 4.27'de önerilen meta-sezgisel karar ağacı tasarımlarından elde edilen ortalama sonuçlar ile aynı veri seti için geliştirilen değişik karar ağacı tasarım yöntemlerinin kullanıldığı çalışmalardan elde edilen sonuçlar verilmiştir. Sonuçlar karşılaştırıldığında, meta-sezgisel karar ağaçları ile elde edilen doğruluk değerlerinin benzer çalışmalar ile elde edilen doğruluk değerlerinden daha yüksek olduğu görülmektedir.

Tablo 4.27. Önerilen metodun aynı veri seti üzerinde gerçekleştirilen benzer çalışmalar ile mukayesesi

Çalışma	**Yöntem**	**Doğruluk (%)**
Yip ve Webb (1994) [251]	Fonksiyon özelliğini bulma algoritması (Function attribute finding algorithm - FAFA) + C4.5 (Budanmış)	94.38
Pasi (2004) [252]	C4.5-1 (Varsayılan öğrenme parametreleri ile) C4.5-2 (c = 5) C4.5-3 (c = 95)	93.26 92.81 92.94
Sun ve ark. (2007) [253]	C4.5 temel C4.5 AdaBoost	91.57 91.12
Yurtay ve ark. (2012) [254]	Gini algoritması	94.50
Önerilen metot	ABC Algoritması ile Karar Ağacı Tasarımı (5 Seviyeli) MBO Algoritması ile Karar Ağacı Tasarımı (5 Seviyeli)	95.82 95.07

4.6. Bir Sürekli Döküm Tesisinde Meta-Sezgisel Kesme Planı En İyilemesi

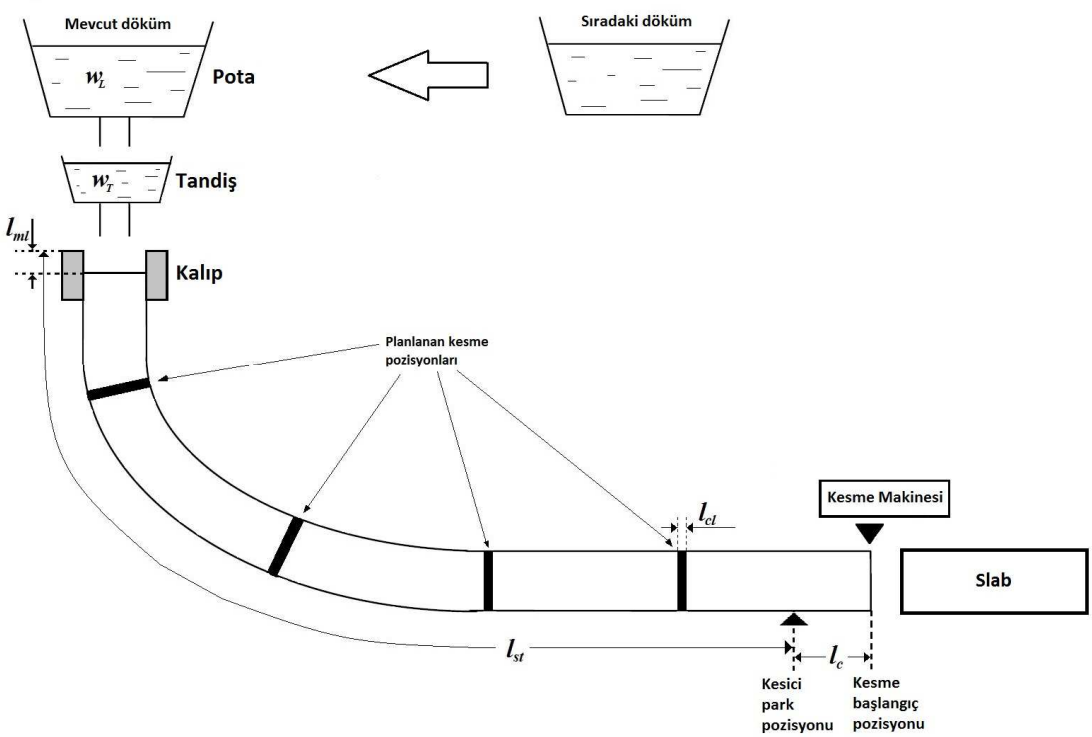
En iyileme üretimin her aşamasında ihtiyaç duyulan önemli bir süreçtir. Üreticiler hem bir taraftan kalite, üretim hızı, verim gibi parametreleri maksimize etmeye hem de diğer taraftan enerji kullanımı, ham madde tüketimi, üretim kayıpları, işçilik giderleri gibi parametreleri minimize etmeye çalışarak en iyi üretimi gerçekleştirmeyi hedeflerler. Bu işlemi gerçekleştirmek için ilgilenilen probleme özgü bir çözüm yöntemi her zaman geliştirilemeyebilir veya o probleme özgü geliştirilen çözüm yöntemleri ile her zaman küresel en iyiyi yakalamak mümkün olmayabilir. Bu durumlarda elde edilen alt en iyi çözümler istenmeyen ciddi üretim kayıplarına neden olabilir.

Bir sürekli döküm tesisinde üretim normal seyrinde devam ederken slab kesim pozisyonları herhangi bir planlamaya ihtiyaç duyulmaksızın siparişe ait hedef kesim uzunluklarına göre belirlenir ve kesme işlemi bu pozisyonlardan yapılır. Fakat kalıpta vuku bulan bazı kalite uygunsuzlukları hattaki malzemenin, planlı veya plansız olarak yapılan döküm sonları ise hem hattaki malzemenin hem de kalan sıvı çelikten elde edilecek slabların kesim pozisyonlarının yeniden planlanmasını gerektirir. Bu çalışmada önerilen yöntemde güçlü bir araştırma yeteneğine sahip olan yapay arı koloni algoritması kullanılarak slab kesim noktalarının yeniden planlanması sağlanmıştır. Majör hedef hurda kaybının minimize edilmesi olarak seçilmiş, bunun yanında hedef uzunluklardan sapmanın minimize edilmesi ve sipariş dışı üretilen slab adedinin minimize edilmesi de problemin minör hedefleri olarak seçilmiştir. Olması muhtemel altı değişik senaryo için çok sayıda rasgele örnek veri üretilmiş ve bu veriler üzerinde en iyileme denemeleri yapılarak en iyi sonuçlar elde edilmiştir. Sonuçlar, önerilen yöntemin sürekli döküm tesislerinde kesme planı en iyilemesi için başarı ile kullanılabileceğini doğrulamıştır.

4.6.1. Kesme planı en iyileme problemi tanımı

Şekil 4.24'te basit diyagramı verilen bir sürekli döküm tesisinin normal işleyişinde; mevcut döküm potasındaki sıvı çelik tandiş içerisine oradan da kalıp içerisine dökülür. Kalıp içerisinde ilk soğuma işlemine maruz kalan sıvı çeliğin kalıp ile temas eden kısmında katılaşma ve bir kabuk oluşumu meydana gelir. Kalıp dışına çıkan ve dış yüzeyi katılaşan fakat iç kısımları halen sıvı halde olan çelik üzerine su püskürtülmek suretiyle ikincil soğutma işlemine tabi tutulur. Çelik, kesme noktasına geldiğinde yaklaşık 600 ˚C sıcaklığa kadar soğumuş ve artık tamamen katılaşmış olur. Burada, katılaşan çelik belirlenen sıradaki kesme pozisyonundan kesilerek slab elde edilir. Mevcut döküm tamamen bittiğinde dökümün üzerinde bulunduğu taret mekanizması dönerek bu dökümü tandiş üzerinden çeker ve sıradaki dökümü tandiş üzerine getirir. Yeni gelen döküm açılarak içerisindeki sıvı çelik tandiş içerisine akmaya başlar. Önceki dökümün boş potası ve sıradaki dökümün dolu potası yer değiştirirken sürekli döküm prosesi tandiş içerisindeki sıvı çeliği kullanarak kesintisiz olarak çalışmaya devam eder. Proses normal seyrinde ilerlerken dökümün plan bilgisinde bulunan hedef uzunluklara göre kesme işlemi gerçekleşir. Yani

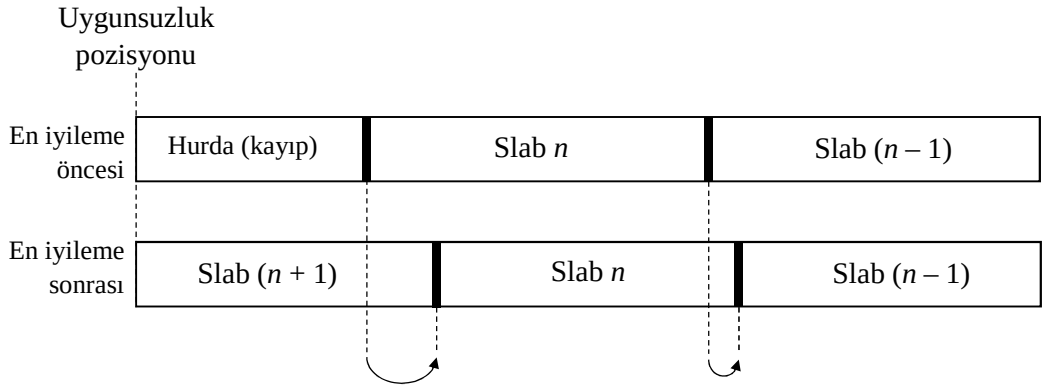
sıradaki slabın kesme uzunluğunu belirlemek için herhangi bir hesaplama yapmaya gerek yoktur.



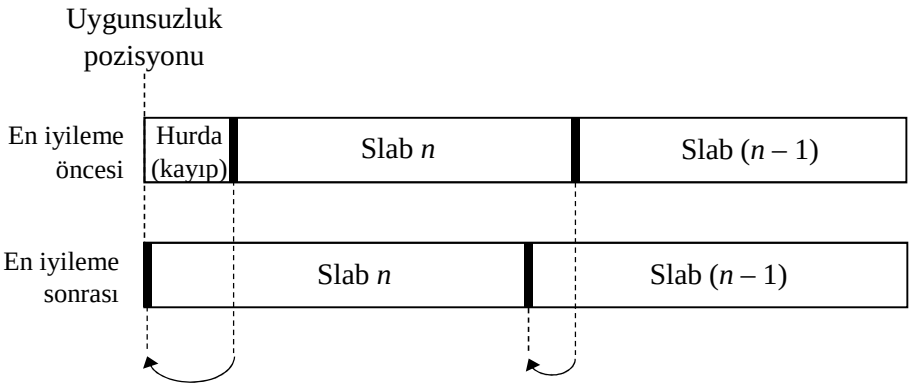
Şekil 4.24. Bir sürekli döküm prosesinin basit diyagramı ve planlanan kesme pozisyonları

Döküm devam ederken tandiş değişimi, nozul değişimi, slab yırtılma tehlikesi, kesme makinesi arızası gibi çeşitli nedenlerle kısa süreli de olsa döküm hızı sıfırlanabilir. Sorun giderilip döküme devam edildiğinde ise hattaki malzemenin kalıp seviyesi hizasında kalite sürekliliği bozulmuş olur. Bu kısım bir kalite uygunsuzluğu olarak kabul edilir ve hurda olarak değerlendirilir. Dolayısıyla bu kısmın hattaki hiçbir slabın içerisinde kalması istenmez. Bu durumda mevcut planlanan slab boyları ya Şekil 4.25'te gösterildiği gibi toleranslar dahilinde biraz daha kısaltılacak yada Şekil 4.26'da gösterildiği gibi toleranslar dahilinde biraz daha uzatılacaktır. Benzer durum planlı veya plansız olarak gerçekleşen döküm sonu uygulamalarında da söz konusudur. Bu durumlarda da döküm sonlanacağı için hattaki mevcut malzemenin en sonunda çıkacak olan ve döküm sonu hurdası diye tabir edilen kısmın minimize edilmesi gerekmektedir. Bunu başarabilmek için slab uzunlukları toleranslar dahilinde değiştirilecektir.

Bir slabın boyunun alabileceği minimum ve maksimum değerleri bir sonraki üretim hattı olan sıcak haddehanedeki slab fırınları belirler. Slab fırınının verimli kullanılması için slab uzunluklarının belirli bir aralığın dışına çıkmaması gerekir. Dolayısıyla bir slab için üç önemli uzunluk değeri vardır. Bunlar: hedef uzunluk, minimum uzunluk ve maksimum uzunluk değerleridir.



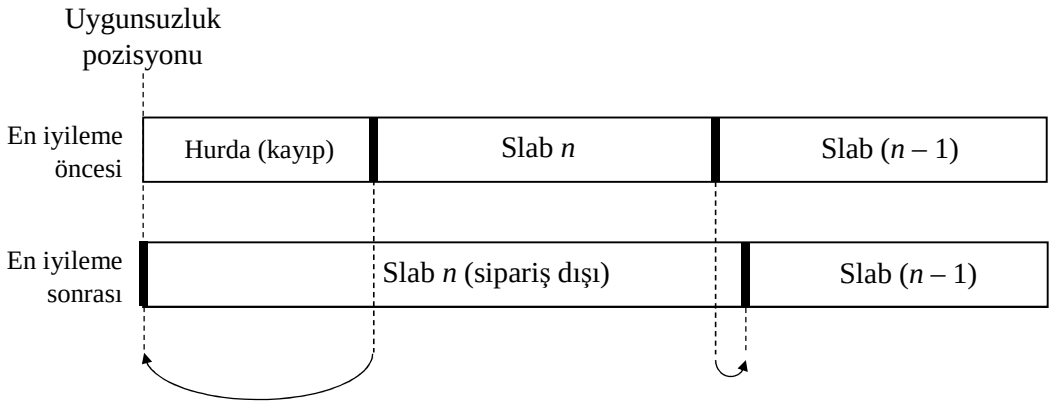
Şekil 4.25. Slab boylarının kısaltılarak hurda kaybının azaltılması



Şekil 4.26. Slab boylarının uzatılarak hurda kaybının azaltılması

Özet olarak, sürekli döküm prosesi normal seyrinde ilerlerken, kesme planı bütün slablar hedef uzunluklarından kesilecek şekilde yapılır. Planlı veya plansız döküm sonlarında ve üretim duraksamalarında ise hat üzerindeki slab boyları duruma göre minimum uzunluğa veya maksimum uzunluğa doğru kaydırılır. Fakat slab boyları bu iki sınırın dışına çıkarılmaz. Değişik slab boyları denenerek hurda miktarının azaltılması denenirken bir yandan da Şekil 4.27'de verildiği gibi planda olmayan fakat slab fırını ölçü kriterlerine uygun plan dışı kabul edilebilir slabların planlaması

da denenebilir. Eğer hurda miktarında dramatik bir düşme elde edilebiliyor ise plan dışı slabların da kullanıldığı yeni kesme planları yapılabilir.



Şekil 4.27. Plan dışı slab boylarının denenerek hurda kaybının azaltılması

Olaya bir en iyileme problemi olarak yaklaşırken hız sıfırlaması nedeni ile oluşan kalite uygunsuzlukları durumu, plan dışı döküm sonu durumu, planlı döküm sonu durumu, döküm başı hurdasının henüz kesilmiş olup olmaması durumları ayrı ayrı ele alınmalıdır. Bu durumda, her bir senaryo için optimize edilecek parametreler biraz farklılıklar içerir. Genel bir gösterim olarak, kesme pozisyonları optimize edilecek malzemenin toplam fiziki uzunluğu

$$l_{tot} = l_{st} + l_c - l_{ml} + l_T + l_L - l_{top} - l_{tail} + l_{cast} \qquad \textbf{(4.28)}$$

olarak tanımlanabilir. Burada l_{st} fiziki hat uzunluğu, l_c kesicinin park pozisyonuna uzaklığı, l_{ml} kalıp seviye değeri, l_T döküm ebadına göre hesaplanan tandişteki çelikten çıkacak olan slab uzunluğu, l_L döküm ebadına göre hesaplanan potadaki çelikten çıkacak olan slab uzunluğu, l_{top} döküm başı hurdası uzunluğu, l_{tail} döküm sonu hurdası uzunluğu ve l_{cast} henüz kesimi başlamamış ilk döküm için döküm uzunluğu. Dökümün bulunduğu statüye göre Denklem (4.28)'deki bazı terimler kullanılmayabilir veya sıfır olarak alınır. Olası durumlar ve bu durumlarda Denklem (4.28)'de hangi değişkenlerin kullanılıp kullanılmayacağı Tablo 4.28'de verilmiştir.

Tablo 4.28. Döküm durumları ve değişkenler (✓: değer var, ×: değer yok)

Durum	Statü	l_{st}	l_c	l_{ml}	l_T	l_L	l_{top}	l_{tail}	l_{cast}
1	Ara döküm ve kalite uygunsuzluğu	✓	✓	✓	×	×	×	×	×
2	Ara döküm ve plansız duruş	✓	✓	✓	✓	×	×	✓	×
3	Kesimi başlamamış ilk döküm ve kalite uygunsuzluğu	×	×	✓	×	×	✓	×	✓
4	Kesimi başlamamış ilk döküm ve plansız duruş	×	×	✓	✓	×	✓	✓	✓
5	Son döküm	✓	✓	✓	✓	✓	×	✓	×
6	Tek döküm	×	×	×	×	✓	✓	✓	×

Tablo 4.28'de verilen durumların biraz netleştirilmesi hesaplamaları daha anlaşılır hale getirecektir.

- Durum 1'deki senaryoda dökümü ve kesimi devam eden bir döküm katarı vardır. Oluşan kalite uygunsuzluğu sonrasında o anda hat üzerinde bulunan malzemenin kesim noktalarının optimize edilmesi gerekmektedir.

- Durum 2'deki senaryoda durum 1'deki gibi dökümü ve kesimi devam eden bir döküm katarı vardır. Herhangi bir nedenle potadan tandişe akış kesilmiştir. Tandişteki sıvı çelik ile birlikte hat üzerinde bulunan malzemenin kesim noktalarının optimize edilmesi gerekmektedir. Ayrıca döküm sonu hurdası oluşacaktır.

- Durum 3'teki senaryoda dökümü başlamış fakat henüz hiç kesim yapılmamış bir döküm vardır. Oluşan kalite uygunsuzluğu sonrasında o anda hat üzerinde bulunan malzemenin kesim noktalarının optimize edilmesi gerekmektedir. Ayrıca döküm başı hurdası oluşacaktır.

- Durum 4'teki senaryoda dökümü başlamış fakat durum 3'te olduğu gibi henüz hiç kesim yapılmamış bir döküm vardır. Herhangi bir nedenle potadan tandişe akış kesilmiştir. Tandişteki sıvı çelik ile birlikte hat üzerinde bulunan malzemenin kesim noktalarının optimize edilmesi gerekmektedir. Ayrıca hem döküm başı hurdası hem de döküm sonu hurdası oluşacaktır.

- Durum 5'teki senaryoda durum 1'deki gibi dökümü ve kesimi devam eden bir döküm katarı vardır. Dökülen döküm son dökümdür ve bu döküm bittiğinde döküm sonu yapılacaktır. Potadaki ve tandişteki sıvı çelik ile birlikte hat üzerinde bulunan malzemenin kesim noktalarının optimize edilmesi gerekmektedir. Ayrıca döküm sonu hurdası oluşacaktır.

- Durum 6'daki senaryoda henüz dökümü başlamamış tek dökümlük bir katar vardır. Bütün en iyileme potadaki sıvı çelikten elde edilecek malzeme uzunluğuna göre yapılacaktır. Ayrıca hem döküm başı hurdası hem de döküm sonu hurdası oluşacaktır. Bu durum için döküm başı hurdası kesildikten sonra Durum 5'teki senaryo ile en iyilemeye devam edilebilir. Döküm başı hurdası kesilmeden kalite uygunsuzluğu oluşursa veya plansız döküm sonu yapılırsa sırasıyla Durum 3 ve Durum 4 senaryoları uygulanır.

Yapılan kesme planının ve kesim kayıplarının toplam uzunluğu

$$l'_{tot} = \sum_{n=1}^{N} l'_{pl}(n) + \sum_{m=1}^{M} l'_{upl}(m) + (M+N-1)l_{torch} \qquad \textbf{(4.29)}$$

şeklinde tanımlanabilir. Burada l'_{pl} plan bilgisine uygun olarak planlanan slab uzunlukları, l'_{upl} ise plan bilgisi dışında fakat slab fırını limitlerine uygun olarak planlanan slab uzunlukları, N plan bilgisine uygun olarak planlanan slab adedi, M plan bilgisine uymayan slab adedi ve l_{torch} kesme makinesinin bir slabı kesmesi esnasında eriyerek kaybolan kesme kaybı uzunluğudur. Denklem (4.29)'un birinci ve ikinci terimlerindeki plan dahili ve plan harici slab uzunluklarının alabileceği değerler için uyulması gereken kısıtlar ise

$$l_{pl\_\min} \le l'_{pl}(n) \le l_{pl\_\max} \;, \quad 1 \le n \le N \qquad \textbf{(4.30)}$$

$$l_{\min}(type) \le l'_{upl}(m) \le l_{\max}(type) \;, \quad 1 \le m \le M \;, \; type = 1, 2, \ldots \qquad \textbf{(4.31)}$$

şeklinde tanımlanabilir. Burada $l'_{pl}(n)$ plana uyan *n*'inci slabın planlanan uzunluğu, $l_{pl\_\min}$ ve $l_{pl\_\max}$ sırasıyla minimum ve maksimum plan uzunlukları, $l'_{upl}(m)$ plan dışı *m*'inci slabın planlanan uzunluğu, *type* plan dışı üretilecek alternatif slab türü, $l_{\min}$ ve $l_{\max}$ sırası ile seçilen plan dışı slab türünün tanımlanmış minimum ve maksimum değerleridir.

Yapılan kesme planının sebep olacağı kayıp Denklem (4.32) ile ve hata olarak değerlendirilebilecek olan oransal kayıp değeri de Denklem (4.33) ile ifade edilebilir.

$$E_{loss} = l_{tot} - l'_{tot} \tag{4.32}$$

$$\Delta E_{loss} = \frac{E_{loss}}{l_{tot}} \tag{4.33}$$

Dolayısıyla kullanılacak en iyileme algoritmasından birinci hedef olarak elde edilen bu ΔE_{loss} değerini minimize etmesi beklenir.

Planlanan her bir slab uzunluğunun alabileceği minimum ve maksimum değerlerinin yanı sıra bu slaba ait bir de hedef uzunluk değeri vardır. Bu uzunluk değeri o slab için en iyi uzunluk değeridir ve prosesin normal akışı içerisinde kesme işleminde hedef uzunluk değerleri kullanılır. Dolayısıyla, algoritmanın ikinci hedefi kesme boyu planlamalarının hedef uzunluk değerlerinden mümkün olduğunca az sapmasını sağlamak olmalıdır. Hedef uzunluklardan toplam sapma miktarı

$$d = \sum_{n=1}^{N}\left[\frac{l_{pl\_aim} - l'_{pl}(n)}{l_{pl\_aim}}\right]^2 + \sum_{m=1}^{M}\left[\frac{l_{upl\_aim}(type) - l'_{upl}(m)}{l_{upl\_aim}(type)}\right]^2 \tag{4.34}$$

olarak tanımlanabilir. Burada l_{pl_aim} plan dahilinde planlanan slabların ve $l_{upl\_aim}(type)$ plan haricinde planlanan slabların hedef uzunluklarıdır.

Plan dışı slablar her ne kadar slab kaybını minimize etmek için oluşturulsa da bu slabların sayısının mümkün olduğunca az olması arzulanır. Bu yüzden kullanılacak en iyileme algoritmasının üçüncü ve son hedefi plan dışı üretilecek slab sayısı *M*'nin minimize edilmesi olacaktır. Bunun için Denklem (4.35) ile verilen plan dışı slabların tüm slablar içerisindeki oranı önemli bir kriter olarak kullanılabilir.

$$OFR = \frac{M}{N+M} \tag{4.35}$$

Burada *OFR* plan harici slab üretim oranını temsil etmektedir. Özet olarak kullanılacak en iyileme algoritmasının hedefleri topluca Denklem (4.36)'daki gibi ifade edilebilir.

$$\text{Hedefler} = \begin{cases} \text{minimize}(\Delta E_{loss}) & , \text{ birinci hedef} \\ \text{minimize}(d) & , \text{ ikinci hedef} \\ \text{minimize}(OFR) & , \text{ üçüncü hedef} \end{cases} \tag{4.36}$$

ABC algoritmasında yapay arıların ürettiği çözümlerin mevcut çözümler ile mukayese edilmesinde tek bir kriter kullanmak iyi olan çözümün seçilmesini kolaylaştırır. Bu yüzden Denklem (4.36)'da listelenen üç kriter önem sıraları göz önünde bulundurularak Denklem (4.37)'de verildiği gibi tek bir maliyet fonksiyonuna dönüştürülebilir.

$$E(t) = \Delta E_{loss}(t) + \alpha \cdot d(t) + \beta \cdot OFR(t) \tag{4.37}$$

Burada *E* toplam hata (maliyet) değeri, *t* iterasyon sayısı, α ve β önem katsayılarıdır. Deneysel çalışmalarda α = 0.01 ve β = 0.0001 değerleri için iyi sonuçlar alınmıştır.

4.6.2. Kesme planı en iyileme probleminin ABC algoritması ile çözümü

ABC algoritması ile kesme planı en iyilemesi problemini çözebilmek için önce en iyileme probleminin olası çözümlerini temsil eden yiyecek kaynağı pozisyonları için uygun bir tanımlama yapılması gerekir. Bir yiyecek kaynağının pozisyon bilgisini en

iyileme probleminin bilinmeyen değişkenleri oluşturur. Bu değişkenler içerisinde problemin hata fonksiyonunu minimum yapan değişkenler kombinasyonu, problemin uygunluk değeri en yüksek olan en iyi çözümüdür. Hata fonksiyonu içerisindeki bilinmeyen değişkenler şunlardır: plan bilgisine uygun slab adedi *N*, bu slabların uzunluk değerleri $l'_{pl}(n)$, plan bilgisine uymayan slab adedi *M* ve bu slabların uzunluk değerleri $l'_{upl}(m)$. Dolayısıyla çözümlere ait genel gösterim Şekil 4.28'de verildiği gibi olacaktır.

N	$l'_{pl}(1)$	$l'_{pl}(2)$	…	$l'_{pl}(N)$	M	$l'_{upl}(1)$	$l'_{upl}(2)$	…	$l'_{upl}(M)$

Şekil 4.28. Genel çözüm bilgisi

Bu genel çözüm bilgisi Şekil 4.28'de verildiği haliyle *N* ve *M* ayrık değişkenlerinin alacağı değere bağlı olarak dinamik bir uzunluğa sahiptir. Bu yüzden pozisyon bilgisi olarak kullanılması zordur. Problemin tesisin üretebileceği slab çeşitlerine göre özelleştirilmesi çözüm bilgisini basitleştirecek ve sabit uzunluklu hale getirecektir. İlgili slab fırınında *x* metre ve 2*x* metre hedef uzunluklarda iki farklı türden slabların kullanılabildiğini ve plan bilgisine uyan slabların uzunluk değerlerinin hepsinin birbirine eşit olduğunu varsayalım. Bu durumda iki farklı slab türü bulunması nedeniyle plan dışı slabların uzunluk değerleri de birbirine eşit olacaktır. Bu durumda hata fonksiyonu içerisindeki dört adet bilinmeyen değişken olacaktır. Bunlar: plan bilgisine uygun slab adedi *N*, bu slabların ortak uzunluk değeri l'_{pl}, plan bilgisine uymayan slab adedi *M* ve bu slabların ortak uzunluk değerleri l'_{upl}. Dolayısıyla genel çözüm bilgisi Şekil 4.29'daki gibi tesise uygun olarak özelleştirilmiş ve sadeleştirilmiş olur.

N	l'_{pl}	M	l'_{upl}

Şekil 4.29. İki farklı slab türü kabullenmesine göre sadeleştirilmiş genel çözüm bilgisi

Aynı zamanda Denklem (4.29) ve Denklem (4.34) sırasıyla Denklem (4.38) ve Denklem (4.39)'daki gibi sadeleştirilerek işlem maliyetleri azaltılmış olur.

$$l'_{tot} = N \cdot l'_{pl} + M \cdot l'_{upl} + (M + N - 1) l_{torch} \quad \textbf{(4.38)}$$

$$d = N\left[\frac{l_{pl\_aim} - l'_{pl}}{l_{pl\_aim}}\right]^2 + M\left[\frac{l_{upl\_aim} - l'_{upl}}{l_{upl\_aim}}\right]^2 \quad \textbf{(4.39)}$$

4.6.3. ABC algoritması ile yapılan kesme planı en iyilemesinin sonuçları

Önerilen yöntemin implementasyonları Ereğli Demir ve Çelik Fabrikaları T.A.Ş. (Erdemir) sürekli döküm tesis standartlarına göre yapıldı. Varsayımımızdaki farazi sürekli döküm tesisinde olduğu gibi Erdemir sürekli döküm tesislerinde de iki farklı uzunluk aralığında slablar üretilmektedir. Sürekli döküm tesis standartları ve rasgele üretilen ölçüm verilerinin olabilecekleri değer aralıkları Tablo 4.29'da verilmiştir.

Tablo 4.29. Sürekli döküm tesis standartları ve rasgele üretilen ölçüm verilerinin olabilecekleri değer aralıkları

	Unit	Minimum	Aim	Maximum
Tip 1 slab uzunluğu	mm	5500	6000	6090
Tip 2 slab uzunluğu	mm	11500	12000	12300
Hat uzunluğu (l_{st})	mm	-	32740 (fix)	-
Slab kalınlığı	mm	-	200 (fix)	-
Slab genişliği	mm	725	-	1500
Kalıp seviyesi(l_{ml})	mm	0	-	200
Kesme noktasının park pozisyonuna uzaklığı (l_c)	mm	0	-	6000
Pota ağırlığı (l_L)	kg	110000	-	120000
Tandiş ağırlığı (l_T)	kg	10000	-	20000
Kesme (erime) kaybı (l_{torch})	mm	-	10 (fix)	-
Döküm başı hurdası uzunluğu (l_{top})	mm	500	500	-
Döküm sonu hurdası uzunluğu (l_{tail})	mm	700	700	-
Slab yoğunluğu	kg/m^3	-	7900	-
Döküm uzunluğu (l_{cast})	mm	6000	-	38700

İlk olarak Tablo 4.28'de kurgulanan senaryolarda test verisi olarak kullanılmak maksadıyla rasgele 1000 farklı ölçüm verisi üretildi ve testlerde kullanılmak üzere veri dosyasına kaydedildi.

İkinci olarak önerilen yöntemin başarısı rasgele üretilen ve kaydedilen test verileri kullanılarak test edildi. Testlerde ABC algoritması 6 arı ve 10000 iterasyon ile kullanıldı. İmplementasyonlarda Intel i7-2600 Mhz CPU'ya sahip bir bilgisayar ve Windows 7 64 bit işletim sistemi altında çalışan MATLAB Versiyon: 8.1.0.604 (R2013a) programı kullanıldı. Algoritmanın toplam çalışma süresi yaklaşık 2 saniye olarak ölçüldü. Sürekli döküm prosesinin düşük hızı göz önünde bulundurulduğunda, bu hesaplama süresi makul bir gerçek zamanlı hesaplama süresidir. Kurgulanan 6 farklı senaryo için rasgele üretilen 1000 örnek durumdan bir kısmı için elde edilen sonuçlar, yöntemin başarısını göstermek amacıyla Tablo 4.30'dan Tablo 4.35'e kadar olan tablolarda verilmiştir.

Tablo 4.30. Ara döküm ve kalite uygunsuzluğu durumu (durum 1) test sonuçlarına örnekler

Durum 1 örnek senaryoları					Planlanan slablar				Kayıp (mm)	Kayıp (%)
					Normal slablar		Plan dışı slablar			
No	l_c (mm)	l_{ml} (mm)	l_{tot} (mm)	Hedef uzunluk (mm)	Slab adedi	Slab boyu (mm)	Slab adedi	Slab boyu (mm)		
1	4311	164	36887	12000	3	12289.00	0	0.00	0	0.00
2	2541	156	35125	6000	4	5767.04	1	12016.85	0	0.00
3	2745	106	35379	12000	3	11786.33	0	0.00	0	0.00
4	4057	49	36748	6000	2	6074.60	2	12284.40	0	0.00
5	4007	45	36702	6000	2	6090.00	2	12246.00	0	0.00
6	2321	121	34940	6000	6	5815.00	0	0.00	0	0.00
7	1935	155	34520	12000	2	11720.10	2	5524.90	0	0.00
8	917	95	33562	6000	6	5585.33	0	0.00	0	0.00
9	1615	184	34171	12000	2	12300.00	1	6090.00	3461	10.13
10	3280	137	35883	12000	3	11954.33	0	0.00	0	0.00
11	1254	190	33804	6000	6	5625.67	0	0.00	0	0.00
12	3972	92	36620	12000	3	12200.00	0	0.00	0	0.00
13	1539	167	34112	6000	6	5677.00	0	0.00	0	0.00
14	716	64	33392	12000	2	12300.00	1	6090.00	2682	8.03
15	3264	130	35874	6000	6	5970.67	0	0.00	0	0.00
16	382	22	33100	12000	2	12300.00	1	6090.00	2390	7.22
17	5598	155	38183	6000	0	0.00	3	12300.00	1263	3.31
18	3183	106	35817	12000	3	11932.33	0	0.00	0	0.00
19	2087	105	34722	12000	2	11804.79	2	5541.21	0	0.00
20	2655	49	35346	12000	3	11775.33	0	0.00	0	0.00
21	2654	141	35253	6000	4	5769.28	1	12135.87	0	0.00
22	2580	165	35155	12000	2	11765.06	2	5797.44	0	0.00
23	2265	152	34853	12000	2	11715.45	2	5696.05	0	0.00
35	256	38	32958	6000	1	6090.00	2	12300.00	2248	6.82
36	2672	108	35304	6000	2	5909.91	2	11727.09	0	0.00
:	:	:	:	:	:	:	:	:	:	:
998	853	22	33571	12000	2	12300.00	1	6090.00	2861	8.52
999	4428	146	37022	12000	3	12300.00	0	0.00	102	0.28
1000	3081	158	35663	12000	3	11881.00	0	0.00	0	0.00

Durum 1 senaryolarında döküm ve kesme işlemleri devam ederken bir kalite uygunsuzluğu oluşmaktadır. Kesme planı en iyilemesi hat üzerindeki toplam malzeme için yapılmaktadır. Limit değerlerin kullanıldığı uç örnekler hariç tutulduğunda kullanılan en iyileme algoritmasının iyi bir performans gösterdiği Tablo 4.30'da görülmektedir. Örnek veriler için elde edilen ortalama kayıp miktarı % 2.03 olarak gerçekleşmiştir. Kayıp miktarının yüksek olduğu senaryoların uç örnekler olduğu, limit değerler kullanılmasına rağmen kaybın azaltılamadığı ve daha iyi bir sonucun elde edilemeyeceği açıkça görülmektedir.

Tablo 4.31. Ara döküm ve plansız duruş durumu (durum 2) test sonuçlarına örnekler

Durum 2 örnek senaryoları								Planlanan slablar						
								Normal slablar		Plan dışı slablar				
No	l_c (mm)	l_{ml} (mm)	Slab Gen. (mm)	w_T (kg)	l_T (mm)	l_{tot} (mm)	Hedef uzunluk (mm)	Slab adedi	Slab boyu (mm)	Slab adedi	Slab boyu (mm)	l_{tail} (mm)	Kayıp (mm)	Kayıp (%)
1	5435	163	805	19134	15043.64	52355.64	6000	7	5745.07	1	12070.11	700.00	0.00	0.00
2	5746	110	1470	11577	4984.50	42660.50	6000	5	6074.28	1	12239.12	700.00	0.00	0.00
3	852	161	1470	19158	8248.51	40979.51	12000	3	11729.16	1	5762.03	700.00	0.00	0.00
4	5095	8	1305	16788	8142.00	45269.00	6000	6	5527.21	1	12045.72	700.00	0.00	0.00
5	1028	132	765	10319	8537.27	41473.27	6000	5	5839.50	1	12225.80	700.00	0.00	0.00
6	4169	165	1070	19503	11536.14	47580.14	6000	6	5929.62	1	11942.40	700.00	0.00	0.00
7	4772	154	1230	14898	7665.95	44323.95	12000	3	12300.00	1	6090.00	2003.95	1303.95	2.94
8	1657	151	820	16551	12774.78	46320.78	6000	4	5672.75	2	11789.88	700.00	0.00	0.00
9	2043	192	925	12239	8374.27	42265.27	12000	3	12086.51	1	5975.75	700.00	0.00	0.00
10	5346	140	845	15473	11589.39	48835.39	6000	6	6079.64	1	12297.56	700.00	0.00	0.00
11	1526	169	1000	12436	7870.89	41267.89	6000	7	5886.84	0	0.00	700.00	0.00	0.00
12	3697	51	1180	13517	7250.05	42936.05	12000	3	12273.59	1	6085.27	700.00	0.00	0.00
13	1716	184	1170	17538	9487.18	43059.18	6000	1	6090.00	3	12300.00	739.18	39.18	0.09
14	3185	11	1170	19341	10462.51	45676.51	6000	6	5600.18	1	12015.45	700.00	0.00	0.00
15	2023	3	1135	17943	10005.58	44065.58	6000	1	6090.00	3	12300.00	1745.58	1045.58	2.37
16	1578	121	1075	16893	9945.83	43442.83	6000	1	6090.00	3	12300.00	1122.83	422.83	0.97
17	5481	46	1500	18259	7704.22	45179.22	6000	8	5638.65	0	0.00	700.00	0.00	0.00
18	640	89	1360	10047	4675.63	37266.63	12000	3	12300.00	0	0.00	1046.63	346.63	0.93
19	2399	17	1435	18001	7939.40	42361.40	6000	7	6043.06	0	0.00	700.00	0.00	0.00
20	874	53	1155	18693	10243.30	43104.30	6000	1	6090.00	3	12300.00	784.30	84.30	0.20
21	3733	171	785	15133	12201.08	47803.08	6000	4	5992.50	2	11891.53	700.00	0.00	0.00
22	1104	25	1425	14173	6294.91	39413.91	12000	3	12300.00	0	0.00	3193.91	2493.91	6.33
23	2936	99	815	19001	14755.77	49632.77	12000	4	12300.00	0	0.00	1102.77	402.77	0.81
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
997	1390	105	1035	16241	9931.51	43256.51	6000	1	6090.00	3	12300.00	936.51	236.51	0.55
998	227	198	805	19133	15042.85	47111.85	6000	6	5847.17	1	11968.83	700.00	0.00	0.00
999	4079	68	1235	17213	8821.30	44872.30	6000	8	5600.29	0	0.00	700.00	0.00	0.00
1000	4291	156	1270	18910	9423.90	45598.90	6000	8	5691.11	0	0.00	700.00	0.00	0.00

Durum 2 senaryolarında döküm ve kesme işlemleri devam ederken bir döküm sonu operasyonu oluşmaktadır. Kesme planı en iyilemesi hat üzerinde bulunan ve tandişteki sıvı çelikten üretilecek toplam malzeme için yapılmaktadır. Limit değerlerin kullanıldığı uç örnekler hariç tutulduğunda kullanılan en iyileme algoritmasının durum 1 senaryolarından daha iyi bir performans gösterdiği Tablo 4.31'den görülmektedir. Örnek veriler için elde edilen ortalama kayıp miktarı % 1.03 olarak gerçekleşmiştir.

Tablo 4.32. Kesimi başlamamış ilk döküm ve kalite uygunsuzluğu durumu (durum 3) test sonuçlarına örnekler

Durum 3 örnek senaryoları					Planlanan slablar						
					Normal slablar		Plan dışı slablar				
No	l_{cast} (mm)	l_{ml} (mm)	l_{tot} (mm)	Hedef uzunluk (mm)	Slab adedi	Slab boyu (mm)	Slab adedi	Slab boyu (mm)	l_{top} (mm)	Kayıp (mm)	Kayıp (%)
1	29032	9	28523	12000	1	11769.72	3	5574.43	500	0.00	0.00
2	37552	110	36942	6000	0	0.00	3	12300.00	500	22.00	0.06
3	19792	161	19131	12000	1	12300.00	1	6090.00	500	731.00	3.82
4	36542	8	36034	6000	6	5997.33	0	0.00	500	0.00	0.00
5	29088	132	28456	6000	3	5543.17	1	11796.50	500	0.00	0.00
6	25436	164	24772	6000	0	0.00	2	12300.00	500	162.00	0.65
7	12111	154	11457	12000	0	0.00	2	5723.50	500	0.00	0.00
8	28227	151	27576	6000	5	5507.20	0	0.00	500	0.00	0.00
9	25139	192	24447	12000	2	12218.50	0	0.00	500	0.00	0.00
10	37369	140	36729	6000	2	6090.00	2	12259.50	500	0.00	0.00
11	32628	169	31959	6000	1	6090.00	2	12300.00	500	1249.00	3.91
12	21477	51	20926	12000	1	12300.00	1	6090.00	500	2526.00	12.07
13	30761	184	30077	6000	5	6007.40	0	0.00	500	0.00	0.00
14	31479	11	30968	6000	1	6090.00	2	12300.00	500	258.00	0.83
15	31661	155	31006	12000	2	12300.00	1	6090.00	500	296.00	0.95
16	27389	121	26768	6000	0	0.00	2	12300.00	500	2158.00	8.06
:	:	:	:	:	:	:	:	:	:	:	:
996	24918	147	24271	12000	2	12130.50	0	0.00	500	0.00	0.00
997	21987	105	21382	6000	1	6090.00	1	12300.00	500	2982.00	13.95
998	34945	198	34247	6000	4	5573.71	1	11912.16	500	0.00	0.00
999	10466	68	9898	6000	1	6090.00	0	0.00	500	3808.00	38.47
1000	35552	156	34896	6000	6	5807.67	0	0.00	500	0.00	0.00

Durum 3 senaryolarında ilk dökümün henüz kesimi başlamadan bir kalite uygunsuzluğu oluşmaktadır. Kesme planı en iyilemesi hat üzerindeki toplam malzeme için yapılmaktadır. Hat üzerindeki malzemenin toplam uzunluğu diğer durumların hepsinden daha kısa olduğu için ve bu durumda denenecek değişik kombinasyonlar da sınırlı olduğundan kayıp oranı Tablo 4.32'de görüldüğü gibi diğer durumlardan daha fazladır. Örnek veriler için elde edilen ortalama kayıp miktarı % 8.47 olarak makul bir seviyede gerçekleşmiştir.

Tablo 4.33. Kesimi başlamamış ilk döküm ve plansız duruş durumu (durum 4) test sonuçlarına örnekler

Durum 4 örnek senaryoları								Planlanan slablar							
								Normal slablar		Plan dışı slablar					
No	l_{cast} (mm)	l_{ml} (mm)	Slab gen. (mm)	w_T (kg)	l_T (mm)	l_{tot} (mm)	Hedef uzunluk (mm)	Slab adedi	Slab boyu (mm)	Slab adedi	Slab boyu (mm)	l_{top} (mm)	l_{tail} (mm)	Kayıp (mm)	Kayıp (%)
1	10153	163	805	19134	15043.64	23833.64	6000	2	5997.21	1	11819.21	500.00	700.00	0.00	0.00
2	37552	110	1470	11577	4984.50	41226.50	6000	5	5854.39	1	11904.55	500.00	700.00	0.00	0.00
3	19792	161	1470	19158	8248.51	26679.51	12000	2	12300.00	0	0.00	500.00	2769.51	2069.51	7.76
4	36542	8	1305	16788	8142.00	43476.00	6000	1	6090.00	3	12300.00	500.00	1156.00	456.00	1.05
5	29088	132	765	10319	8537.27	36293.27	6000	4	6020.02	1	12173.21	500.00	700.00	0.00	0.00
6	16370	165	1070	19503	11536.14	26541.14	6000	0	0.00	2	12300.00	500.00	2631.14	1931.14	7.28
7	12111	154	1230	14898	7665.95	18422.95	12000	1	12300.00	1	6090.00	500.00	722.95	22.95	0.12
8	28227	151	820	16551	12774.78	39650.78	6000	7	5655.83	0	0.00	500.00	700.00	0.00	0.00
9	25139	192	925	12239	8374.27	32121.27	12000	2	12300.00	1	6090.00	500.00	2111.27	1411.27	4.39
10	37369	140	845	15473	11589.39	47618.39	6000	8	5943.55	0	0.00	500.00	700.00	0.00	0.00
11	32628	169	1000	12436	7870.89	39129.89	6000	7	5581.41	0	0.00	500.00	700.00	0.00	0.00
12	21477	51	1180	13517	7250.05	27476.05	12000	2	12300.00	0	0.00	500.00	3566.05	2866.05	10.43
13	30761	184	1170	17538	9487.18	38864.18	6000	7	5543.45	0	0.00	500.00	700.00	0.00	0.00
14	31479	11	1170	19341	10462.51	40730.51	6000	7	5810.07	0	0.00	500.00	700.00	0.00	0.00
15	11304	3	1135	17943	10005.58	20106.58	6000	1	6090.00	1	12300.00	500.00	2406.58	1706.58	8.49
16	27389	121	1075	16893	9945.83	36013.83	6000	4	5966.36	1	12108.40	500.00	700.00	0.00	0.00
17	10983	46	1500	18259	7704.22	17441.22	6000	3	5807.07	0	0.00	500.00	700.00	0.00	0.00
18	37455	89	1360	10047	4675.63	40841.63	12000	2	11847.53	3	5702.19	500.00	700.00	0.00	0.00
19	14498	17	1435	18001	7939.40	21220.40	6000	1	6090.00	1	12300.00	500.00	3520.40	2820.40	13.29
20	10450	53	1155	18693	10243.30	19440.30	6000	1	6090.00	1	12300.00	500.00	1740.30	1040.30	5.35
21	17477	171	785	15133	12201.08	28307.08	6000	5	5653.42	0	0.00	500.00	700.00	0.00	0.00
22	13847	25	1425	14173	6294.91	18916.91	12000	1	12300.00	1	6090.00	500.00	1216.91	516.91	2.73
23	17044	99	815	19001	14755.77	30500.77	12000	2	12212.62	1	6055.52	500.00	700.00	0.00	0.00
24	19208	78	1460	10965	4753.34	22683.34	12000	1	12300.00	1	6090.00	500.00	4983.34	4283.34	18.88
25	13678	116	740	13532	11573.73	23935.73	6000	4	5976.43	0	0.00	500.00	700.00	0.00	0.00
26	29928	34	1150	16478	9068.79	37762.79	6000	0	0.00	3	12300.00	500.00	1542.79	842.79	2.23
998	34945	198	805	19133	15042.85	48589.85	6000	6	6038.31	1	12300.00	500.00	700.00	0.00	0.00
999	10466	68	1235	17213	8821.30	18019.30	6000	3	5999.77	0	0.00	500.00	700.00	0.00	0.00
1000	35552	156	1270	18910	9423.90	43619.90	6000	1	6090.00	3	12300.00	500.00	1299.90	599.90	1.38

Durum 4 senaryolarında ilk dökümün henüz kesimi başlamadan bir döküm sonu operasyonu oluşmaktadır. Kesme planı en iyilemesi hat üzerinde bulunan ve tandişteki sıvı çelikten üretilecek toplam malzeme için yapılmaktadır. Toplam malzeme uzunluğu iyi bir en iyileme elde etmek için uç örneklerde yetersiz kaldığından, kayıp oranı ancak Tablo 4.33'de verilen değerlere kadar düşürülebilmiştir. Uç örnekler için elde edilen yüksek kayıp değerleri olabilecek en iyi sonuçlar olduğundan tatmin edicidirler. Tandiş içerisindeki ilave sıvı çelik miktarı optimize edilecek malzeme miktarını artırdığından örnek veriler için elde edilen ortalama kayıp miktarı % 4.15 olarak durum 3'tekinden daha düşük oranda gerçekleşmiştir.

Tablo 4.34. Son döküm durumu (durum 5) test sonuçlarına örnekler

Durum 5 örnek senaryoları									Planlanan slablar						
									Normal slablar		Plan dışı slablar				
No	l_c (mm)	l_{ml} (mm)	Slab gen. (mm)	w_T (kg)	w_L (kg)	$l_T + l_L$ (mm)	l_{tot} (mm)	Hedef uzunluk (mm)	Slab adedi	Slab boyu (mm)	Slab adedi	Slab boyu (mm)	l_{tail} (mm)	Kayıp (mm)	Kayıp (%)
1	5435	163	805	19134	116324	106500.51	143812.51	6000	23	5734.02	1	11700.00	700.00	0.00	0.00
2	5746	110	1470	11577	119706	56524.15	94200.15	6000	10	5854.42	3	11845.32	700.00	0.00	0.00
3	852	161	1470	19158	117923	59020.49	91751.49	12000	7	12227.36	1	6090.00	700.00	0.00	0.00
4	5095	8	1305	16788	117578	65166.11	102293.11	6000	16	5652.07	1	11700.00	700.00	0.00	0.00
5	1028	132	765	10319	112770	101835.86	134771.86	6000	24	5605.91	0	0.00	700.00	0.00	0.00
6	4169	165	1070	19503	110345	76805.87	112849.87	6000	18	5609.44	1	11700.00	700.00	0.00	0.00
7	4772	154	1230	14898	114456	66560.67	103218.67	12000	8	12131.08	1	6090.00	700.00	0.00	0.00
8	1657	151	820	16551	111627	98933.31	132479.31	6000	21	5741.40	1	11700.00	700.00	0.00	0.00
9	2043	192	925	12239	117513	88780.02	122671.02	12000	9	11784.56	3	5500.00	700.00	0.00	0.00
10	5346	140	845	15473	111387	95019.10	132265.10	6000	21	5731.20	1	11700.00	700.00	0.00	0.00
11	1526	169	1000	12436	119293	83372.78	116769.78	6000	16	5793.01	2	11955.82	700.00	0.00	0.00
12	3697	51	1180	13517	118309	70706.93	106392.93	12000	9	11812.55	0	0.00	700.00	0.00	0.00
13	1716	184	1170	17538	113805	71049.98	104621.98	6000	16	5761.83	1	12272.67	700.00	0.00	0.00
14	3185	11	1170	19341	111300	70670.24	105884.24	6000	12	5842.01	3	11880.02	700.00	0.00	0.00
15	2023	3	1135	17943	113113	73080.91	107140.91	6000	17	5604.17	1	11700.00	700.00	0.00	0.00
16	1578	121	1075	16893	117482	79113.92	112610.92	6000	18	5596.16	1	11700.00	700.00	0.00	0.00
17	5481	46	1500	18259	115384	56389.45	93864.45	6000	14	5818.86	1	12260.42	700.00	0.00	0.00
18	640	89	1360	10047	117750	59473.66	92064.66	12000	7	12274.96	1	6069.93	700.00	0.00	0.00
19	2399	17	1435	18001	114315	58358.40	92780.40	6000	14	5742.44	1	12246.30	700.00	0.00	0.00
20	874	53	1155	18693	115798	73697.74	106558.74	6000	15	5533.25	2	11700.00	700.00	0.00	0.00
21	3733	171	785	15133	114019	104129.65	139731.65	6000	22	5809.62	1	11700.00	700.00	0.00	0.00
22	1104	25	1425	14173	110497	55371.97	88490.97	12000	7	11779.60	1	5963.79	700.00	0.00	0.00
23	2936	99	815	19001	113693	103047.29	137924.29	12000	11	11975.97	1	6078.68	700.00	0.00	0.00
24	1451	78	1460	10965	111320	53010.66	86423.66	12000	7	12300.00	0	0.00	963.66	263.66	0.31
25	359	116	740	13532	118212	112678.75	144961.75	6000	23	5783.99	1	11700.00	700.00	0.00	0.00
26	3895	34	1150	16478	114510	72090.26	107991.26	6000	15	5628.75	2	11700.00	700.00	0.00	0.00
27	1134	149	1210	11836	113685	65655.93	98680.93	12000	7	11713.49	3	5532.17	700.00	0.00	0.00
:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:
998	227	198	805	19133	117962	107787.56	139856.56	6000	22	5815.30	1	11700.00	700.00	0.00	0.00
999	4079	68	1235	17213	111068	65741.30	101792.30	6000	18	5645.68	0	0.00	700.00	0.00	0.00
1000	4291	156	1270	18910	113342	65908.50	102083.50	6000	16	5638.97	1	11700.00	700.00	0.00	0.00

Durum 1’den durum 4’e kadar olan durumlar önceden planlanmamış, birden bire vuku bulan ve üretimin normal seyri içerisinde arzu edilmeyen durumlardır. Fakat durum 5 senaryoları daha önceden planlanmış senaryolardır. Hat üzerindeki malzemeye ilave olarak tandişteki ve potadaki sıvı çelikten elde edilecek toplam malzeme miktarı da göz önünde bulundurulduğunda en iyilemesi yapılacak toplam malzeme uzunluğu oldukça iyi sonuçlar elde etmeye yetecek seviyelerdedir. Limit değerlerin kullanıldığı uç örnekler hariç tutulduğunda kullanılan en iyileme algoritmasının çok iyi bir performans gösterdiği Tablo 4.34’te görülmektedir. Örnek veriler için elde edilen ortalama kayıp miktarı % 0.01 olarak gerçekleşmiştir.

Tablo 4.35. Tek döküm durumu (durum 6) test sonuçlarına örnekler

Durum 6 örnek senaryoları						Planlanan slablar				l_{top} (mm)	l_{tail} (mm)	Kayıp (mm)	Kayıp (%)
						Normal slablar		Plan dışı slablar					
No	Slab gen. (mm)	w_L (kg)	l_L (mm)	l_{tot} (mm)	Hedef uzunluk (mm)	Slab adedi	Slab boyu (mm)	Slab adedi	Slab boyu (mm)				
1	805	116324	91456.88	90256.88	6000	14	5601.21	1	11700.00	500	700.00	0.00	0.00
2	1470	119706	51539.65	50339.65	6000	9	5584.41	0	0.00	500	700.00	0.00	0.00
3	1470	117923	50771.98	49571.98	12000	4	12300.00	0	0.00	500	1041.98	341.98	0.69
4	1305	117578	57024.10	55824.10	6000	8	5504.97	1	11704.35	500	700.00	0.00	0.00
5	765	112770	93298.59	92098.59	6000	14	5706.94	1	12061.36	500	700.00	0.00	0.00
6	1070	110345	65269.73	64069.73	6000	11	5815.43	0	0.00	500	700.00	0.00	0.00
7	1230	114456	58894.72	57694.72	12000	4	12300.00	1	6090.00	500	3064.72	2364.72	4.10
8	820	111627	86158.54	84958.54	6000	13	5625.27	1	11700.00	500	700.00	0.00	0.00
9	925	117513	80405.75	79205.75	12000	6	12176.95	1	6084.02	500	700.00	0.00	0.00
10	845	111387	83429.71	82229.71	6000	14	5864.26	0	0.00	500	700.00	0.00	0.00
11	1000	119293	75501.90	74301.90	6000	11	5639.54	1	12156.99	500	700.00	0.00	0.00
12	1180	118309	63456.88	62256.88	12000	5	12300.00	0	0.00	500	1416.88	716.88	1.15
13	1170	113805	61562.80	60362.80	6000	8	6032.43	1	12023.36	500	700.00	0.00	0.00
14	1170	111300	60207.72	59007.72	6000	10	5891.77	0	0.00	500	700.00	0.00	0.00
15	1135	113113	63075.34	61875.34	6000	9	5502.50	1	12262.79	500	700.00	0.00	0.00
16	1075	117482	69168.09	67968.09	6000	10	5561.24	1	12255.70	500	700.00	0.00	0.00
17	1500	115384	48685.23	47485.23	6000	6	5919.47	1	11908.43	500	700.00	0.00	0.00
18	1360	117750	54798.03	53598.03	12000	4	11878.60	1	6043.63	500	700.00	0.00	0.00
19	1435	114315	50419.00	49219.00	6000	4	6090.00	2	12300.00	500	909.00	209.00	0.42
20	1155	115798	63454.44	62254.44	6000	9	5581.35	1	11932.27	500	700.00	0.00	0.00
21	785	114019	91928.57	90728.57	6000	14	5634.90	1	11700.00	500	700.00	0.00	0.00
22	1425	110497	49077.06	47877.06	12000	3	11920.63	2	6037.58	500	700.00	0.00	0.00
23	815	113693	88291.53	87091.53	12000	6	11705.53	3	5592.78	500	700.00	0.00	0.00
24	1460	111320	48257.33	47057.33	12000	3	11827.73	2	5767.07	500	700.00	0.00	0.00
25	740	118212	101105.03	99905.03	6000	17	5540.83	0	0.00	500	700.00	0.00	0.00
26	1150	114510	63021.46	61821.46	6000	9	5503.08	1	12203.72	500	700.00	0.00	0.00
27	1210	113685	59464.90	58264.90	12000	4	11710.94	2	5685.58	500	700.00	0.00	0.00
28	1075	114359	67329.41	66129.41	6000	9	5973.85	1	12274.80	500	700.00	0.00	0.00
29	1020	116444	72253.66	71053.66	12000	5	11849.58	2	5872.89	500	700.00	0.00	0.00
30	1210	115502	60415.32	59215.32	12000	5	11835.06	0	0.00	500	700.00	0.00	0.00
31	880	118444	85187.00	83987.00	6000	15	5589.80	0	0.00	500	700.00	0.00	0.00
⋮													
997	1035	116792	71419.31	70219.31	6000	10	5792.64	1	12192.90	500	700.00	0.00	0.00
998	805	117962	92744.71	91544.71	6000	14	5693.19	1	11700.00	500	700.00	0.00	0.00
999	1235	111068	56920.00	55720.00	6000	10	5563.00	0	0.00	500	700.00	0.00	0.00
1000	1270	113342	56484.60	55284.60	6000	2	6090.00	3	12300.00	500	764.60	64.60	0.12

Durum 6 senaryoları da durum 5'te olduğu gibi planlı senaryolardır. Optimize edilecek toplam malzeme miktarı durum 5'e kıyasla daha az fakat iyi bir en iyileme neticesi elde etmek için yeterli miktardadır. En iyileme sonuçları Tablo 4.35'te verilen önerilen yöntem, bu durum için % 0.29 gibi iyi bir ortalama kayıp miktarı elde etmiştir.

4.7. ABC ve MBO Algoritmaları ile Melez ve Modifiye Uygulamalar

Literatüre ilk tanıtıldığı hali ile meta-sezgisel algoritmalar her ne kadar en iyileme problemlerinin çözümünde iyi performans sergileseler de araştırmacılar bu algoritmalara daha mükemmel çözümler üretebilme yeteneği katmayı arzulamışlardır. Bu istek araştırmacıları mevcut algoritmalar üzerinde sürekli iyileştirmeler yapmaya sevk etmiştir. Bu maksatla araştırmacıların bir kısmı mevcut algoritmaların parametrelerinin daha efektif bir şekilde ayarlanması üzerine [76,255-267], bazıları modifiye veya melez meta-sezgisel uygulamalar üzerine [121,221,268-279] ve bazıları da algoritmaları paralel olarak çalıştırma stratejileri üzerine [64,121,280-285] bir kısım çalışmalar yapmışlardır.

Tablo 4.36. Bir meta-sezgisel algoritmanın küresel arama (exploration) ve yerel arama (exploitation) özellikleri arasındaki denge

Küresel arama	**Yerel arama**	**Algoritma karakteristiği**
✓ Güçlü	✓ Güçlü	Algoritma önce küresel minimumun bulunduğu bölgeyi yerel minimumlardan kolayca kurtularak bulur. Daha sonra küresel minimuma arzulandığı gibi iyi bir yakınsama gerçekleştirir. Algoritma performansı başlangıç popülasyonunun kalitesinden ve problem karakteristiğinden bağımsızdır.
✓ Güçlü	× Zayıf	Algoritma önce küresel minimumun bulunduğu bölgeyi yerel minimumlardan kolayca kurtularak bulur. Fakat küresel minimuma arzulanan yakınsamayı çoğu kez gerçekleştiremez. Algoritma performansı başlangıç popülasyonunun kalitesinden ve problem karakteristiğinden bağımsızdır.
× Zayıf	✓ Güçlü	Algoritma, üzerinde çalışılan problemin karakteristiğine bağlı olarak küresel minimumun yer aldığı bölgeyi bulamaya bilir. Bu durumda en iyileme başarısızlıkla sonuçlanır. Algoritma eğer küresel minimumun yer aldığı bölgeyi keşfedebilirse bu durumda da küresel minimuma iyi bir yakınsama gerçekleştirir. Algoritma performansı başlangıç popülasyonunun karakteristiğine oldukça bağlıdır.
× Zayıf	× Zayıf	Her iki özelliğin de zayıf olması bir en iyileme algoritması için anlamsız olup arzulanmayan bir durumdur.

Bir meta-sezgisel algoritmadan beklenen güçlü yerel arama sistematiği ile popülasyon bireylerini bulundukları bölgelerin minimumlarına yönlendirmek ve aynı zamanda belirli bir oranda rasgele arama stratejilerini de kullanarak aramanın

yönünü problem uzayının daha önce araştırılmamış bölgelerine kanalize etmektir. İyi bir yerel arama (yakınsama) yeteneği algoritmanın problem uzayının minimumlarına mümkün olduğunca yaklaşmasını sağlarken, iyi bir küresel (rasgele) arama yeteneği de algoritmanın yerel minimumlardan kolayca kurtulmasını sağlar. Dolayısıyla bir algoritma içerisindeki küresel arama ve yerel arama özelliklerinin iyi bir şekilde dengelenmesi gerekmektedir. Melez ve modifiye algoritmalar tasarlanırken orijinal algoritmanın bu iki özelliğinin daha iyi dengelenmesi hedeflenir. Tablo 4.36 küresel arama veya keşif (exploration) özelliği ile yerel arama veya yakınsama (exploitation) özelliğinin bir algoritmaya kazandırdıklarını kısaca özetlemektedir. Bu bölümde, ABC ve MBO algoritmalarının başarımlarını artırmak için geliştirilen beş yeni uygulama önerilmiştir. Bölüm 4.1.1'de verilen test prosedürü uygulanarak bu algoritmalar test edilmiştir. Önerilen algoritmaların başarı performanslarını mukayese edebilmek için testlere GA, DE, ABC, MBO ve PSO algoritmaları ile daha önceki testlerde elde edilen sonuçlar da verilen tablolara ilave edilmiştir.

4.7.1. Modifiye meta-sezgisel algoritmalar

Modifiye algoritmalar orijinal algoritmaların komşuluk üretim yöntemi, yakınsama stratejileri ve keşif davranışları gibi temel aşamalarında kısmi veya köklü değişiklikler yapılarak elde edilen iyileştirilmiş algoritmalardır. Modifiye algoritmalarda orijinal algoritmanın ana felsefesine sadık kalınırken orijinal algoritmanın küresel arama veya yerel arama gibi yetenekleri geliştirilmek suretiyle performansında ve çözüm kalitesinde gelişmeler elde edilmesi hedeflenir.

4.7.1.1. Değişken komşuluklu göçmen kuşlar en iyileme algoritması

Bölüm 2.2'de detaylı olarak tanıtılan MBO algoritmasında, her bir çözüm için işlem yapılacak olan toplam komşuluk sayısı olarak ifade edilen *k* parametresinin algoritma performansı üzerinde büyük etkisi vardır. Bu parametre algoritmanın ilham aldığı kuş sürülerinin uçuş hızı ile ters orantılı olan bir kontrol parametresidir. Büyük *k* değerlerinin seçilmesi sürünün yavaş uçması ve küçük *k* değerlerinin seçilmesi sürünün hızlı uçması şeklinde yorumlanır. Sürünün hızlı uçması algoritmanın bütün

iterasyonları daha çabuk tamamlaması anlamına gelir. Dolayısıyla bu durumda algoritmanın işlem süresi de daha kısa olacaktır.

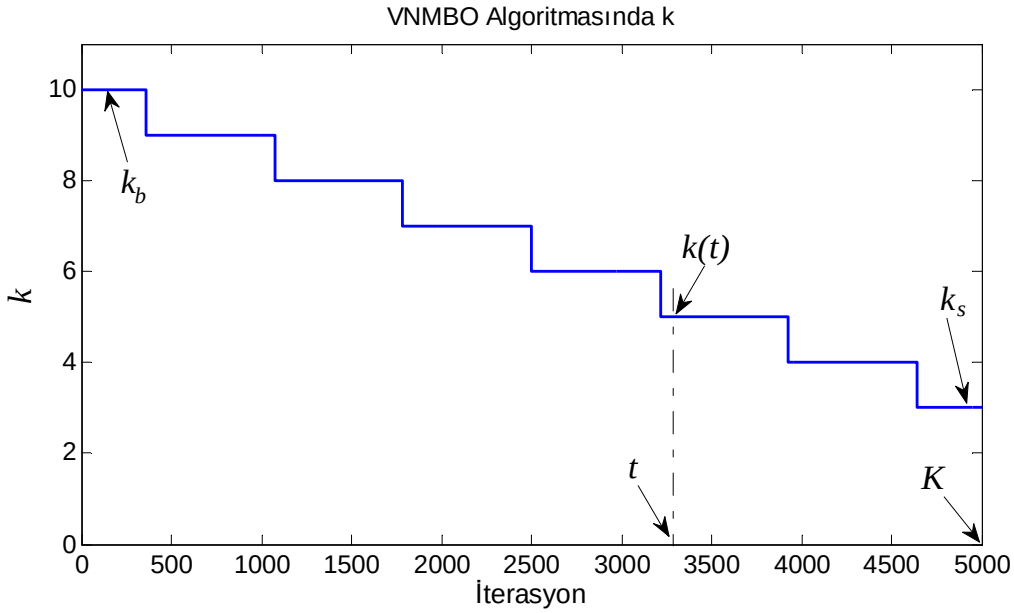
Olaya yapılan araştırmanın detayı noktasından yaklaşıldığında, MBO algoritmasının yaptığı araştırmanın derinliğinin *k* parametresi ile orantılı olduğu gözlemlenir. Her bir çözüm için üzerinde işlem yapılacak olan toplam komşu çözüm sayısını temsil eden *k* parametresinin değerinin büyük seçilmesi durumunda mevcut çözümler için daha fazla sayıda komşu çözüm üretilerek mevcut çözümlerin iyileştirilmesi denenecektir. Bu durumda her bir çözümün her bir iterasyon içerisindeki iyileşebilme ihtimali de artacaktır. Sonuç olarak, sürekli iyileşen çözümler yerel minimumlara takılmadan küresel minimuma daha kararlı bir şekilde yakınsayacaklardır. Tersi durumda, *k* parametresinin küçük değerlerinde her bir mevcut çözüm için daha az sayıda komşu üretileceğinden mevcut çözümlerin her bir iterasyon içerisinde iyileşme şansları da daha düşük olacaktır. Bu durumda çözümlerin yerel minimumlardan kurtulma şansları azalacak, algoritmanın küresel minimuma yönelmesi riske girecektir.

Kısaca özetleyecek olursak, *k* değeri azalırsa, algoritma işletim süresi kısaltır fakat araştırmanın detaycılığı da azalır. Tersi duruma, *k* değeri artırılırsa, araştırmanın detaycılığı artar fakat algoritma işletim süresi de artar. Uygulamalarda ise bir algoritmanın hem detaycı hem de hızlı olması arzulanır.

Değişken komşuluklu göçmen kuşlar en iyileme (Varying Neighbourhood Migrating Birds Optimization - VNMBO) algoritmasında *k* değeri Şekil 4.30'da gösterildiği gibi değişken tutulmuştur. Bu yöntemde *k* değeri

$$k(t) = \text{round}\left(k_b - \left(\frac{k_b - k_s}{K}\right)t\right) \qquad \textbf{(4.40)}$$

şeklinde hesaplanır. Burada k_b başlangıç *k* değeri, k_s bitiş *k* değeri, *K* maksimum iterasyon değeri, *t* o anki iterasyon değeri, *k*(*t*) *t*'inci iterasyonundaki *k* değeri ve round en yakın tamsayıya yuvarlama fonksiyonudur.



Şekil 4.30. VNMBO Algoritmasında *k* parametresi ilerlemesine bir örnek

Başlangıçta *k* değeri yüksek tutularak algoritmanın daha detaylı araştırma yapması ve bu sayede popülasyon üyelerinin (çözüm seti) ümit vadetmeyen çözüm bölgelerinden uzaklaştırılması sağlanır. Detaylı arama bir süre sonra popülasyon üyelerini küresel minimumun bulunduğu bölgede toplar.

İlerleyen iterasyonlarda popülasyon üyelerinin çoğu yerel minimum tuzaklarından uzaklaştırılmış olacağından artık detaylı arama yapmaya gerek kalmayacaktır. Algoritmaya daha fazla zaman kaybettirmemek için *k* değerine daha düşük değerler atanarak işlem süresinin kısaltılması sağlanır.

VNMBO algoritmasının çalışmasını özetleyen sözde kod Şekil 4.31'de verilmiştir. VNMBO algoritmasını MBO algoritmasından ayıran tek farklılık *k* değerinin sabit olmaması ve Denklem (4.40)'ta verilen strateji kullanılarak her iterasyonda yeniden hesaplanmasıdır. Algoritmanın geri kalan diğer kısımları aynen normal MBO algoritmasında olduğu gibidir.

Test fonksiyonlarının 2, 5, 10, 30 ve 50 boyutlu versiyonları için elde edilen test sonuçları Tablo 4.37'den Tablo 4.41'e kadar olan tablolarda verilmiştir. Her problem boyutunda her bir fonksiyon için problem boyutuna uygun tolerans değeri seçilmiş

ve bu tolerans değerinden daha az bir hata ile küresel minimuma yaklaşan işletimler başarılı diğerleri başarısız sayılmıştır. Başarılı işletim yüzdeleri de tablolarda verilmiştir.

Tablolarda test fonksiyonlarının ilgili problem boyutundaki küresel minimumları ve algoritmalar tarafından elde edilen sonuçların bu küresel minimumlara göre hata değerleri de verilmiştir. Anlamlı mukayeseler yapılabilmesi için, hata değerleri küresel minimumları sıfırdan farklı olan fonksiyonlarda (f_5 ve f_8) bağıl hata, diğerlerinde (f_1, f_2, f_3, f_4, f_6, f_7, f_9 ve f_{10}) ise mutlak hata olarak hesaplanmıştır.

```
Denklem (2.1) aracılığıyla bütün çözümlere rastgele başlangıç değerlerini ata
Rasgele seçilen bir çözümü lider olarak ata
Çözümleri farazi bir V formasyonuna rasgele yerleştir
sol = true
repeat
    for i = 1 to m
        Denklem (8.1)'i kullanarak mevcut iterasyonda kullanılacak k değerini hesapla
        Lider çözüme k adet komşu çözüm üreterek onu iyileştirmeyi dene
        Kullanılmayan en iyi çözümlerden x adedini sol arkadaki x adedini de sağ
            arkadaki çözümlerle paylaş
        for lider haricindeki çözümlerin her biri
            Sıradaki çözüme (k-x) adet komşu çözüm üret ve öndeki çözümden alınan x
                adet çözümle birlikte sıradaki çözümü iyileştirmeyi dene
            Kullanılmayan en iyi çözümlerden x adedini bir arkadaki çözümle paylaş
        end for
    end for
    if sol then
        lider çözümü farazi V formasyonun sol kolunun sonuna kaydır ve sol koldaki ilk
            çözümü lider olarak ata
    else
        lider çözümü farazi V formasyonun sağ kolunun sonuna kaydır ve sağ koldaki
            ilk çözümü lider olarak ata
    end if
    sol = not (sol)
until Durdurma kriterleri
return Mevcut en iyi çözüm
```

Şekil 4.31. VNMBO algoritması sözde kodu

Tablo 4.37. VNMBO algoritmasının 2 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	**ABC**	**PSO**	**DE**	**GA**	**VNMBO**
f_1	*KM*	0	*M*	0	2.6992E-17	1.7508E-149	1.7525E-161	4.2287E-17	0
	T	5.0000E-03	*H*	0	2.6992E-17	1.7508E-149	1.7525E-161	4.2287E-17	0
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	0	1.2448E-07	0	0	1.4211E-16	0
	T	5.0000E-03	*H*	0	1.2448E-07	0	0	1.4211E-16	0
			B	95	90	80	96	97	99
f_3	*KM*	0	*M*	0	3.6367E-17	4.3904E-149	2.1397E-161	4.9386E-19	0
	T	5.0000E-03	*H*	0	3.6367E-17	4.3904E-149	2.1397E-161	4.9386E-19	0
			B	100	100	100	100	100	100
f_4	*KM*	0	*M*	7.4873E-04	1.2457E-02	1.9230E-03	0	6.7057E-03	0
	T	5.0000E-03	*H*	7.4873E-04	1.2457E-02	1.9230E-03	0	6.7057E-03	0
			B	45	8	37	68	10	67
f_5	*KM*	-1.8010E+00	*M*	-1.8013E+00	-1.8013E+00	-1.8013E+00	-1.8013E+00	-1.8013E+00	-1.8013E+00
	T	3.5000E-04	*H*	1.6844E-04	1.6844E-04	1.6844E-04	1.6844E-04	1.6844E-04	1.6844E-04
			B	100	100	93	100	99	100
f_6	*KM*	0	*M*	9.0640E-244	4.8596E-07	9.4008E-85	8.6804E-104	8.7397E-15	0
	T	5.0000E-03	*H*	9.0640E-244	4.8596E-07	9.4008E-85	8.6804E-104	8.7397E-15	0
			B	100	100	100	100	100	100
f_7	*KM*	0	*M*	0	3.5134E-17	0	0	3.8941E-20	0
	T	5.0000E-03	*H*	0	3.5134E-17	0	0	3.8941E-20	0
			B	100	100	100	100	97	100
f_8	*KM*	-8.3797E+02	*M*	-8.3797E+02	-8.3797E+02	-7.6690E+02	-8.3797E+02	-1.2035E+04	-8.3797E+02
	T	1.0000E-01	*H*	5.0425E-06	5.0435E-06	9.2668E-02	5.0425E-06	9.3038E-01	5.0425E-06
			B	91	87	20	88	22	98
f_9	*KM*	0	*M*	8.8818E-16	1.8829E-15	8.8818E-16	8.8818E-16	1.2773E-07	8.8818E-16
	T	5.0000E-03	*H*	8.8818E-16	1.8829E-15	8.8818E-16	8.8818E-16	1.2773E-07	8.8818E-16
			B	100	97	100	100	87	100
f_{10}	*KM*	0	*M*	6.1752E-130	2.2923E-03	3.5879E-74	7.7725E-71	7.4712E-04	8.8019E-195
	T	5.0000E-03	*H*	6.1752E-130	2.2923E-03	3.5879E-74	7.7725E-71	7.4712E-04	8.8019E-195
			B	100	40	100	100	56	100
Genel Ortalama Hata				**9.2221E-05**	**1.4923E-03**	**9.4759E-03**	**1.7348E-05**	**9.3800E-02**	**1.7348E-05**
Genel Ortalama Başarı(%)				**93.1**	**82.2**	**83.0**	**95.2**	**76.8**	**96.4**

Tablo 4.38. VNMBO algoritmasının 5 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	**ABC**	**PSO**	**DE**	**GA**	**VNMBO**
f_1	*KM*	0	*M*	3.1987E-222	6.4359E-17	4.8685E-246	7.6169E-171	2.5427E-07	0
	T	1.0000E-02	*H*	3.1987E-222	6.4359E-17	4.8685E-246	7.6169E-171	2.5427E-07	0
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	0	0	1.2139E+00	0	5.2194E-05	0
	T	1.0000E-02	*H*	0	0	1.2139E+00	0	5.2194E-05	0
			B	69	100	13	100	100	95
f_3	*KM*	0	*M*	5.0794E-222	6.6576E-17	8.8594E-246	1.9272E-170	9.4855E-07	0
	T	1.0000E-02	*H*	5.0794E-222	6.6576E-17	8.8594E-246	1.9272E-170	9.4855E-07	0
			B	100	100	100	100	100	100
f_4	*KM*	0	*M*	1.3097E-02	4.5754E-07	8.7264E-02	0	2.2388E-02	8.9845E-03
	T	1.0000E-02	*H*	1.3097E-02	4.5754E-07	8.7264E-02	0	2.2388E-02	8.9845E-03
			B	20	96	2	99	4	31
f_5	*KM*	-4.6870E+00	*M*	-4.6877E+00	-4.6877E+00	-4.5523E+00	-4.6877E+00	-4.6877E+00	-4.6877E+00
	T	1.6000E-03	*H*	1.4041E-04	1.4041E-04	2.9585E-02	1.4041E-04	1.3958E-04	1.4041E-04
			B	62	97	9	96	100	82
f_6	*KM*	0	*M*	5.7515E-148	1.8877E-16	7.1054E-17	4.0330E-135	8.8746E-05	2.8234E-249
	T	1.0000E-02	*H*	5.7515E-148	1.8877E-16	7.1054E-17	4.0330E-135	8.8746E-05	2.8234E-249
			B	99	100	100	100	100	100
f_7	*KM*	0	*M*	0	6.5295E-17	0	0	1.3775E-04	0
	T	1.0000E-02	*H*	0	6.5295E-17	0	0	1.3775E-04	0
			B	100	100	100	100	97	100
f_8	*KM*	-2.0949E+03	*M*	-2.0949E+03	-2.0949E+03	-1.5773E+03	-2.0949E+03	-1.5074E+04	-2.0949E+03
	T	2.5000E-01	*H*	6.8911E-06	6.8911E-06	3.2814E-01	6.8911E-06	8.6103E-01	6.8911E-06
			B	70	98	0	87	1	91
f_9	*KM*	0	*M*	3.5172E-15	4.3698E-15	4.1567E-15	8.8818E-16	7.8859E-03	3.3040E-15
	T	1.0000E-02	*H*	3.5172E-15	4.3698E-15	4.1567E-15	8.8818E-16	7.8859E-03	3.3040E-15
			B	98	100	100	100	31	99
f_{10}	*KM*	0	*M*	4.4960E-43	8.1871E-06	2.1329E-115	6.4222E-56	8.7774E-02	6.3102E-76
	T	1.0000E-02	*H*	4.4960E-43	8.1871E-06	2.1329E-115	6.4222E-56	8.7774E-02	6.3102E-76
			B	96	85	100	100	1	99
Genel Ortalama Hata				**1.3245E-03**	**1.5594E-05**	**1.6588E-01**	**1.4730E-05**	**9.7949E-02**	**9.1318E-04**
Genel Ortalama Başarı(%)				**81.4**	**97.6**	**62.4**	**98.2**	**63.4**	**89.7**

Tablo 4.39. VNMBO algoritmasının 10 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

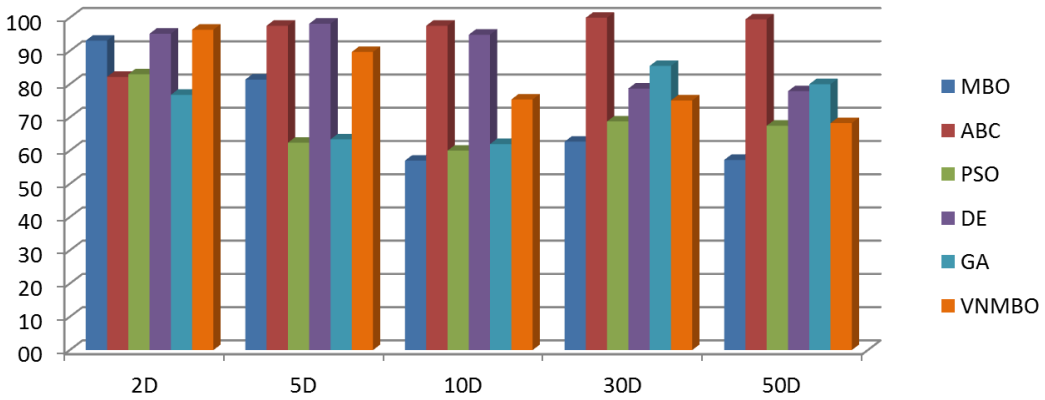
				MBO	**ABC**	**PSO**	**DE**	**GA**	**VNMBO**
f_1	*KM*	0	*M*	1.7398E-122	9.0681E-17	2.9633E-269	3.4394E-249	1.4506E-06	1.9776E-293
	T	1.1000E-02	*H*	1.7398E-122	9.0681E-17	2.9633E-269	3.4394E-249	1.4506E-06	1.9776E-293
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	3.8541E-01	0	5.2733E+00	0	2.4249E-04	0
	T	1.1000E-02	*H*	3.8541E-01	0	5.2733E+00	0	2.4249E-04	0
			B	30	100	0	96	100	76
f_3	*KM*	0	*M*	3.8799E-140	9.2792E-17	1.0598E-275	1.4532E-248	7.2226E-06	9.8877E-293
	T	1.1000E-02	*H*	3.8799E-140	9.2792E-17	1.0598E-275	1.4532E-248	7.2226E-06	9.8877E-293
			B	100	100	100	100	100	100
f_4	*KM*	0	*M*	3.1914E-02	2.9584E-04	3.8357E-01	0	3.7359E-02	2.3741E-02
	T	1.1000E-02	*H*	3.1914E-02	2.9584E-04	3.8357E-01	0	3.7359E-02	2.3741E-02
			B	5	76	0	98	2	5
f_5	*KM*	-9.6600E+00	*M*	-9.5849E+00	-9.6602E+00	-8.4746E+00	-9.6602E+00	-9.6601E+00	-9.6477E+00
	T	5.0000E-03	*H*	7.8346E-03	1.5705E-05	1.3987E-01	1.5705E-05	9.8244E-06	1.2730E-03
			B	17	100	0	75	100	36
f_6	*KM*	0	*M*	3.7558E-16	2.7008E-16	4.5853E-16	3.7114E-202	3.2727E-04	3.5577E-17
	T	1.1000E-02	*H*	3.7558E-16	2.7008E-16	4.5853E-16	3.7114E-202	3.2727E-04	3.5577E-17
			B	90	100	100	100	100	98
f_7	*KM*	0	*M*	0	9.3529E-17	0	0	5.5265E-04	0
	T	1.1000E-02	*H*	0	9.3529E-17	0	0	5.5265E-04	0
			B	99	100	100	100	99	100
f_8	*KM*	-4.1898E+03	*M*	-4.1211E+03	-4.1898E+03	-2.6082E+03	-4.1898E+03	-1.6321E+04	-4.1898E+03
	T	3.0000E-01	*H*	1.6662E-02	6.8911E-06	6.0642E-01	6.8911E-06	7.4329E-01	6.8911E-06
			B	21	100	0	92	0	59
f_9	*KM*	0	*M*	8.3489E-15	7.9936E-15	7.3541E-15	3.8014E-15	1.1570E-02	5.7199E-15
	T	1.1000E-02	*H*	8.3489E-15	7.9936E-15	7.3541E-15	3.8014E-15	1.1570E-02	5.7199E-15
			B	67	100	100	100	19	95
f_{10}	*KM*	0	*M*	1.9667E-02	5.4424E-11	2.2224E-67	4.5729E-08	1.4122E-01	5.6030E-33
	T	1.1000E-02	*H*	1.9667E-02	5.4424E-11	2.2224E-67	4.5729E-08	1.4122E-01	5.6030E-33
			B	41	100	100	88	0	85
Genel Ortalama Hata				**4.6149E-02**	**3.1844E-05**	**6.4032E-01**	**2.2642E-06**	**9.3458E-02**	**2.5021E-03**
Genel Ortalama Başarı(%)				**57.0**	**97.6**	**60.0**	**94.9**	**62.0**	**75.4**

Tablo 4.40. VNMBO algoritmasının 30 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	**ABC**	**PSO**	**DE**	**GA**	**VNMBO**
f_1	*KM*	0	*M*	1.3000E-23	4.9316E-16	2.6521E-79	2.9023E-263	8.7879E-06	1.0072E-165
	T	5.0000E-01	*H*	1.3000E-23	4.9316E-16	2.6521E-79	2.9023E-263	8.7879E-06	1.0072E-165
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	3.1244E+00	0	3.1202E+01	0	1.9920E-03	5.5718E-01
	T	5.0000E-01	*H*	3.1244E+00	0	3.1202E+01	0	1.9920E-03	5.5718E-01
			B	2	100	0	73	100	25
f_3	*KM*	0	*M*	3.7380E-19	4.8509E-16	2.3738E-78	7.2600E-262	1.3585E-04	9.6940E-138
	T	5.0000E-01	*H*	3.7380E-19	4.8509E-16	2.3738E-78	7.2600E-262	1.3585E-04	9.6940E-138
			B	98	100	100	100	100	100
f_4	*KM*	0	*M*	5.9400E-06	7.3275E-17	2.9088E-03	0	1.4791E-02	0
	T	5.0000E-01	*H*	5.9400E-06	7.3275E-17	2.9088E-03	0	1.4791E-02	0
			B	98	100	100	100	100	100
f_5	*KM*	-2.8980E+01	*M*	-2.9121E+01	-2.9631E+01	-2.3443E+01	-2.6569E+01	-2.9627E+01	-2.9490E+01
	T	7.0000E-01	*H*	4.8288E-03	2.1963E-02	2.3617E-01	9.0743E-02	2.1853E-02	1.7288E-02
			B	83	100	0	0	100	100
f_6	*KM*	0	*M*	1.0433E-04	5.5956E-16	5.3999E-11	6.1784E-156	1.2395E-03	2.7889E-15
	T	5.0000E-01	*H*	1.0433E-04	5.5956E-16	5.3999E-11	6.1784E-156	1.2395E-03	2.7889E-15
			B	97	100	100	100	100	100
f_7	*KM*	0	*M*	1.2803E-24	4.6608E-16	2.6501E-33	0	3.9747E-03	0
	T	5.0000E-01	*H*	1.2803E-24	4.6608E-16	2.6501E-33	0	3.9747E-03	0
			B	99	100	100	100	100	100
f_8	*KM*	-1.2569E+04	*M*	-1.2247E+04	-1.2569E+04	-6.3747E+03	-1.2569E+04	-4.2350E+04	-1.2487E+04
	T	1.0000E+00	*H*	2.6264E-02	3.8714E-05	9.7171E-01	3.8714E-05	7.0321E-01	6.6007E-03
			B	3	100	0	88	0	15
f_9	*KM*	0	*M*	5.6005E-02	3.1086E-14	3.8050E-14	4.4409E-15	1.4639E-02	2.7534E-14
	T	5.0000E-01	*H*	5.6005E-02	3.1086E-14	3.8050E-14	4.4409E-15	1.4639E-02	2.7534E-14
			B	47	100	88	100	100	87
f_{10}	*KM*	0	*M*	1.2274E+01	3.1423E-02	8.1776E-06	5.3910E-01	4.2169E-01	9.9659E-01
	T	5.0000E-01	*H*	1.2274E+01	3.1423E-02	8.1776E-06	5.3910E-01	4.2169E-01	9.9659E-01
			B	0	100	100	26	55	24
Genel Ortalama Hata				**1.5486E+00**	**5.3425E-03**	**3.2413E+00**	**6.2989E-02**	**1.1835E-01**	**1.5777E-01**
Genel Ortalama Başarı(%)				**62.7**	**100.0**	**68.8**	**78.7**	**85.5**	**75.1**

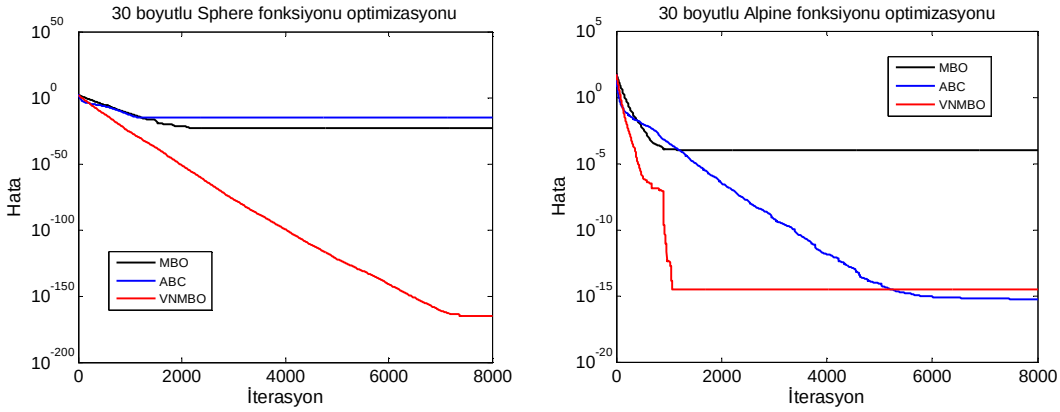
Tablo 4.41. VNMBO algoritmasının 50 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	**ABC**	**PSO**	**DE**	**GA**	**VNMBO**
f_1	*KM*	0	*M*	7.6778E-10	8.9649E-16	1.9490E-48	0	4.5167E-05	4.8848E-101
	T	6.0000E-01	*H*	7.6778E-10	8.9649E-16	1.9490E-48	0	4.5167E-05	4.8848E-101
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	7.6024E+00	0	5.9996E+01	0	8.3003E-03	2.2685E+00
	T	6.0000E-01	*H*	7.6024E+00	0	5.9996E+01	0	8.3003E-03	2.2685E+00
			B	0	100	0	75	100	3
f_3	*KM*	0	*M*	5.0462E-10	8.2798E-16	8.9623E-48	0	9.9204E-04	1.3245E-97
	T	6.0000E-01	*H*	5.0462E-10	8.2798E-16	8.9623E-48	0	9.9204E-04	1.3245E-97
			B	96	100	97	100	100	98
f_4	*KM*	0	*M*	9.8167E-04	7.9936E-17	4.2093E-14	0	2.2371E-02	0
	T	6.0000E-01	*H*	9.8167E-04	7.9936E-17	4.2093E-14	0	2.2371E-02	0
			B	87	100	100	100	100	99
f_5	*KM*	-4.8300E+01	*M*	-4.8606E+01	-4.9623E+01	-3.9379E+01	-3.4137E+01	-4.9602E+01	-4.9230E+01
	T	1.3500E+00	*H*	6.2906E-03	2.6654E-02	2.2653E-01	4.1487E-01	2.6244E-02	1.8889E-02
			B	77	100	0	0	100	100
f_6	*KM*	0	*M*	1.4317E-02	9.3362E-16	1.0707E-08	1.9308E-04	3.1337E-03	4.4280E-05
	T	6.0000E-01	*H*	1.4317E-02	9.3362E-16	1.0707E-08	1.9308E-04	3.1337E-03	4.4280E-05
			B	90	100	100	100	100	100
f_7	*KM*	0	*M*	5.8536E-08	8.4834E-16	9.7930E-32	0	1.6281E-02	0
	T	6.0000E-01	*H*	5.8536E-08	8.4834E-16	9.7930E-32	0	1.6281E-02	0
			B	90	100	100	100	100	95
f_8	*KM*	-2.0949E+04	*M*	-2.0160E+04	-2.0949E+04	-9.7364E+03	-2.0949E+04	-6.5528E+04	-2.0594E+04
	T	1.5000E+00	*H*	3.9148E-02	6.8911E-06	1.1516E+00	6.8911E-06	6.8030E-01	1.7246E-02
			B	0	100	0	86	0	6
f_9	*KM*	0	*M*	3.3808E-01	5.8371E-14	1.3061E-12	7.7804E-15	2.3689E-02	5.7803E-14
	T	6.0000E-01	*H*	3.3808E-01	5.8371E-14	1.3061E-12	7.7804E-15	2.3689E-02	5.7803E-14
			B	32	100	78	100	100	82
f_{10}	*KM*	0	*M*	3.1367E+01	2.4628E-01	3.1404E-02	9.3642E-01	9.3579E-01	1.2684E+01
	T	6.0000E-01	*H*	3.1367E+01	2.4628E-01	3.1404E-02	9.3642E-01	9.3579E-01	1.2684E+01
			B	0	95	100	18	0	0
Genel Ortalama Hata				**3.9368E+00**	**2.7294E-02**	**6.1405E+00**	**1.3515E-01**	**1.7172E-01**	**1.4989E+00**
Genel Ortalama Başarı(%)				**57.2**	**99.5**	**67.5**	**77.9**	**80.0**	**68.3**



Şekil 4.32. VNMBO ve diğer algoritmaların başarı dağılımları

Tablolarda verilen başarı yüzdeleri Şekil 4.32'de bar grafiği ile ifade edilmiştir. Bölüm 4.7 içerisinde ABC ve MBO algoritmalarının birlikte kullanımlarına sıkça yer verildiğinden önerilen her bir yöntemin bu iki algoritma ile mukayeselerinde yakınsama karakteristikleri önemli bir kriter olarak ele alınmıştır. Bu maksatla ABC ve MBO algoritmaları ile VNMBO algoritmasının tek ve çok modlu fonksiyonlardaki yakınsama davranışlarına Şekil 4.33'te birer örnek verilmiştir.



(a) (b)

Şekil 4.33. ABC, MBO ve VNMBO algoritmaların (a) 30 boyutlu tek modlu Sphere fonksiyonunun ve (b) 30 boyutlu çok modlu Alpine fonksiyonunun küresel minimumuna yakınsamaları

Öne çıkan sonuçlardan bazıları şunlardır:

- Özetle MBO algoritmasındaki komşuluk üretim sayısının bir sistematik dahilinde değişken hale getirilmesi ile elde edilen VNMBO algoritmasında MBO

algoritmasının performansının iyileştirilmesi sağlanmıştır. Şekil 4.32'de görüldüğü gibi, bütün problem boyutlarında VNMBO algoritmasının başarı yüzdesi MBO algoritmasına kıyasla daha iyidir.

- Tablo 4.37'den Tablo 4.41'e kadar verilen sonuçlardan, VNMBO algoritmasının MBO algoritmasına kıyasla küresel minimuma daha fazla yakınsadığı görülmektedir. Bu yakınsamadaki iyileşme tek modlu fonksiyonlarda diğer algoritmaları geride bırakırken çok modlu fonksiyonlarda onlarla rekabet edebilir düzeye gelmiştir.

- Bir algoritmanın küresel minimuma yakınsama davranışı da önemli bir mukayese kriterleridir. Bu bağlamda, VNMBO algoritmasının tek ve çok modlu problemler üzerindeki davranışları Şekil 4.33'te olduğu gibi ayrı ayrı incelenebilir. ABC ve MBO algoritmaları tek modlu fonksiyonlar için makul hata seviyelerinde doyuma ulaşırken, VNMBO hata değerini düşürmeye devam ederek çok daha düşük değerlerde doyuma ulaşır. Çok modlu fonksiyonlarda ise VNMBO algoritmasının küresel minimuma yakınsama miktarı MBO algoritmasından daha iyidir fakat ABC algoritmasının nihai yakınsama değeri daha iyidir. Çok modlu fonksiyonlardaki yakınsama karakteristikleri detaylı incelendiğinde VNMBO algoritmasının MBO ve ABC algoritmalarından daha önce makul hata seviyelerine eriştiği gözlemlenir.

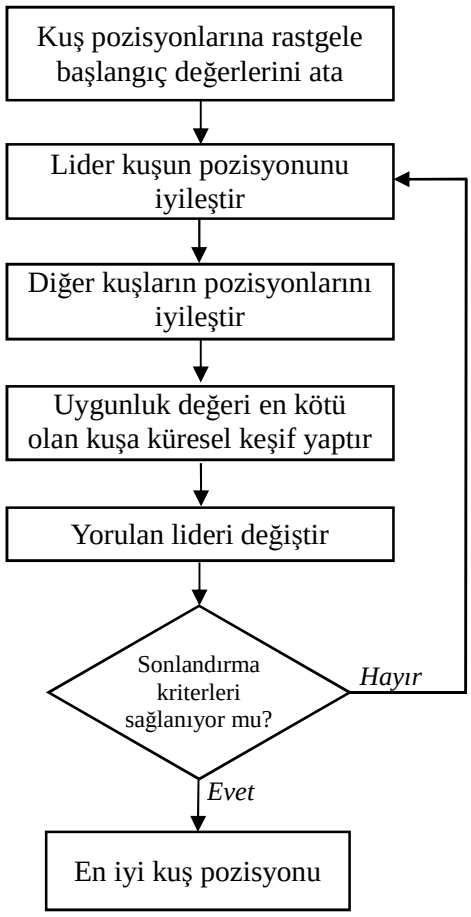
- MBO algoritmasından onun modifiye bir formu olan VNMBO elde edilirken algoritmanın hem keşif hem de yakınsama yetenekleri artırılmıştır. MBO algoritmasına kıyasla elde edilen daha düşük nihai hata değerleri algoritmanın yakınsama kabiliyetinin arttığının, daha yüksek başarı oranları ise algoritmanın keşif kabiliyetinin arttığının birer kanıtıdır.

4.7.1.2. Küresel araştırma ilaveli göçmen kuşlar en iyileme algoritması

Meta-sezgisel algoritmalar küresel ve yerel aramaları dengelemek için değişik yöntemler kullanırlar. Bir algoritmanın keşif yeteneği olarak da adlandırılan küresel arama yeteneği algoritmanın çözüm geliştiremediği veya yerel minimum bölgelerine

takıldığı zamanlarda, algoritmanın arama yönünü değiştirerek araştırmayı problem uzayının daha önce araştırılmamış bölgelerine yönlendirir. Böylece algoritma küresel minimuma yakınsayabilmek için yeni şanslar elde eder. ABC algoritmasındaki kâşif arı evresinde yapılan rasgele araştırmalar ve GA'da mutasyon evresinde kromozom genlerinde gerçekleşen rasgele değişimler kontrollü bir şekilde yönetilen küresel arama stratejileridir.

MBO algoritmasında ise bir popülasyon üyesinin daha iyi çözüm üretememe durumu için kullandığı tek mekanizma algoritmanın kendine özgü çözüm paylaşım yöntemidir. Bu yöntemde çözüm geliştiremeyen popülasyon üyesi diğer üyelerden yardım alır. Fakat bu mekanizma, algoritmanın genelinin çözüm üretememesi durumunda veya bütün popülasyon üyelerinin yerel minimumlar etrafında toplandığı durumlarda bir fayda sağlamaz. Bu gibi durumlarda MBO algoritmasının farklı bir rasgele arama stratejisine ihtiyacı vardır.



Şekil 4.34. MBOGE algoritması akış diyagramı

Küresel araştırma ilaveli göçmen kuşlar en iyileme (Migrating Birds Optimization with Global Exploration – MBOGE) algoritmasında, MBO algoritmasına rasgele küresel arama yeteneğinin ilave edilmesi hedeflenmiştir. Şekil 4.34'te akış diyagramı şematik olarak verilen MBOGE algoritmasının ilk üç adımı geleneksel MBO algoritması ile aynıdır.

Her iterasyonda problemin olası çözümlerini temsil eden kuş pozisyonlarının iyileştirilmesi denendikten sonra en kötü uygunluk değerine sahip kuşa Denklem (2.1) kullanılarak rasgele küresel arama yaptırılır. Eğer bu arama neticesinde elde edilen çözüm küresel arama yapan kuşun temsil ettiği çözümden daha iyi ise bu çözüm ilgili kuşun pozisyonuna atanır. Aksi halde küresel arama ile bulunan çözüm kullanılmadan terkedilir. Geleneksel MBO algoritmasına ilave edilen bu küresel arama evresinden sonraki kısımlar MBO algoritmasında olduğu gibi lider değişimi ve sonlandırma kriterlerinin kontrolü şeklinde olup temel MBO algoritmasındaki ile birebir aynıdır.

MBOGE algoritmasının çalışmasını özetleyen sözde kod bütün bu anlatılanlar doğrultusunda Şekil 4.35'teki gibi oluşturulmuştur. Verilen sözde kodda da açıkça görüldüğü gibi problemin olası çözümlerini temsil eden bütün kuş pozisyonları için kullanılan komşu üretme stratejisi aracılığı ile kuş pozisyonlarının iyileştirilmesi denenmektedir. Daha sonra problemin çözümü için gelişim vadetmediği düşünülen en kötü uygunluktaki çözüm seçilir ve onun için bir kez de mevcut komşu üretme sistematiğinin dışına çıkılarak rasgele küresel arama yapılır. Eğer bu yolla üretilen çözüm seçilen çözümden daha iyi uygunluk değerine sahip ise seçilen çözüm üretilen bu çözüm ile değiştirilir.

Bu uygulama ABC algoritmasının kâşif arı evresine benzemektedir. Fakat uygulamada bazı farklılıklar vardır. ABC algoritması rasgele arama işlemini belirli başarısızlık sayısına erişildiğinde yapar. MBOGE algoritması ise bu işlemi her iterasyonda yapar. ABC algoritması üretilen rasgele çözümü ilgili çözüme mutlaka atar. MBOGE algoritması ise üretilen çözüm eğer seçilen çözümden daha iyi uygunluk değerine sahip ise o zaman ilgili çözüme atar. Aksi taktirde mevcut çözümü korur.

```
Denklem (2.1) aracılığıyla bütün çözümlere rastgele başlangıç değerlerini ata
Rasgele seçilen bir çözümü lider olarak ata
Çözümleri farazi bir V formasyonuna rasgele yerleştir
sol = true
repeat
    for i = 1 to m
        Lider çözüme k adet komşu çözüm üreterek onu iyileştirmeyi dene
        Kullanılmayan en iyi çözümlerden x adedini sol arkadaki x adedini de sağ
            arkadaki çözümlerle paylaş
        for lider haricindeki çözümlerin her biri
            Sıradaki çözüme (k-x) adet komşu çözüm üret ve öndeki çözümden alınan x
                adet çözümle birlikte sıradaki çözümü iyileştirmeyi dene
            Kullanılmayan en iyi çözümlerden x adedini bir arkadaki çözümle paylaş
        end for
    end for
    Uygunluk değeri en düşük olan çözümü belirle ve onun için Denklem (2.1)
        aracılığıyla küresel arama yap
    Küresel arama ile elde edilen çözümün uygunluğu ilgili çözümünkinden daha iyi
        ise elde edilen çözümü ilgili çözüme ata
    if sol then
        lider çözümü farazi V formasyonun sol kolunun sonuna kaydır ve sol koldaki ilk
            çözümü lider olarak ata
    else
        lider çözümü farazi V formasyonun sağ kolunun sonuna kaydır ve sağ koldaki
            ilk çözümü lider olarak ata
    end if
    sol = not (sol)
until Durdurma kriterleri
return Mevcut en iyi çözüm
```

Şekil 4.35. MBOGE algoritması sözde kodu

Test fonksiyonlarının 2, 5, 10, 30 ve 50 boyutlu versiyonları için elde edilen test sonuçları Tablo 4.42'den Tablo 4.46'ya kadar olan tablolarda verilmiştir. Her problem boyutunda her bir fonksiyon için problem boyutuna uygun tolerans değeri seçilmiş ve bu tolerans değerinden daha az bir hata ile küresel minimuma yaklaşan işletimler başarılı diğerleri başarısız sayılmıştır. Başarılı işletim yüzdeleri de tablolarda verilmiştir.

Tablo 4.42. MBOGE algoritmasının 2 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	ABC	PSO	DE	GA	MBOGE
f_1	*KM*	0	*M*	0	2.6992E-17	1.7508E-149	1.7525E-161	4.2287E-17	0
	T	5.0000E-03	*H*	0	2.6992E-17	1.7508E-149	1.7525E-161	4.2287E-17	0
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	0	1.2448E-07	0	0	1.4211E-16	0
	T	5.0000E-03	*H*	0	1.2448E-07	0	0	1.4211E-16	0
			B	95	90	80	96	97	100
f_3	*KM*	0	*M*	0	3.6367E-17	4.3904E-149	2.1397E-161	4.9386E-19	0
	T	5.0000E-03	*H*	0	3.6367E-17	4.3904E-149	2.1397E-161	4.9386E-19	0
			B	100	100	100	100	100	100
f_4	*KM*	0	*M*	7.4873E-04	1.2457E-02	1.9230E-03	0	6.7057E-03	1.8566E-09
	T	5.0000E-03	*H*	7.4873E-04	1.2457E-02	1.9230E-03	0	6.7057E-03	1.8566E-09
			B	45	8	37	68	10	54
f_5	*KM*	-1.8010E+00	*M*	-1.8013E+00	-1.8013E+00	-1.8013E+00	-1.8013E+00	-1.8013E+00	-1.8013E+00
	T	3.5000E-04	*H*	1.6844E-04	1.6844E-04	1.6844E-04	1.6844E-04	1.6844E-04	1.6844E-04
			B	100	100	93	100	99	100
f_6	*KM*	0	*M*	9.0640E-244	4.8596E-07	9.4008E-85	8.6804E-104	8.7397E-15	1.3991E-227
	T	5.0000E-03	*H*	9.0640E-244	4.8596E-07	9.4008E-85	8.6804E-104	8.7397E-15	1.3991E-227
			B	100	100	100	100	100	100
f_7	*KM*	0	*M*	0	3.5134E-17	0	0	3.8941E-20	0
	T	5.0000E-03	*H*	0	3.5134E-17	0	0	3.8941E-20	0
			B	100	100	100	100	97	100
f_8	*KM*	-8.3797E+02	*M*	-8.3797E+02	-8.3797E+02	-7.6690E+02	-8.3797E+02	-1.2035E+04	-8.3797E+02
	T	1.0000E-01	*H*	5.0425E-06	5.0435E-06	9.2668E-02	5.0425E-06	9.3038E-01	5.0425E-06
			B	91	87	20	88	22	100
f_9	*KM*	0	*M*	8.8818E-16	1.8829E-15	8.8818E-16	8.8818E-16	1.2773E-07	8.8818E-16
	T	5.0000E-03	*H*	8.8818E-16	1.8829E-15	8.8818E-16	8.8818E-16	1.2773E-07	8.8818E-16
			B	100	97	100	100	87	100
f_{10}	*KM*	0	*M*	6.1752E-130	2.2923E-03	3.5879E-74	7.7725E-71	7.4712E-04	1.7333E-120
	T	5.0000E-03	*H*	6.1752E-130	2.2923E-03	3.5879E-74	7.7725E-71	7.4712E-04	1.7333E-120
			B	100	40	100	100	56	100
Genel Ortalama Hata				**9.2221E-05**	**1.4923E-03**	**9.4759E-03**	**1.7348E-05**	**9.3800E-02**	**1.7348E-05**
Genel Ortalama Başarı(%)				**93.1**	**82.2**	**83.0**	**95.2**	**76.8**	**95.4**

Tablo 4.43. MBOGE algoritmasının 5 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	**ABC**	**PSO**	**DE**	**GA**	**MBOGE**
f_1	*KM*	0	*M*	3.1987E-222	6.4359E-17	4.8685E-246	7.6169E-171	2.5427E-07	3.4869E-218
	T	1.0000E-02	*H*	3.1987E-222	6.4359E-17	4.8685E-246	7.6169E-171	2.5427E-07	3.4869E-218
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	0	0	1.2139E+00	0	5.2194E-05	0
	T	1.0000E-02	*H*	0	0	1.2139E+00	0	5.2194E-05	0
			B	69	100	13	100	100	100
f_3	*KM*	0	*M*	5.0794E-222	6.6576E-17	8.8594E-246	1.9272E-170	9.4855E-07	9.6984E-218
	T	1.0000E-02	*H*	5.0794E-222	6.6576E-17	8.8594E-246	1.9272E-170	9.4855E-07	9.6984E-218
			B	100	100	100	100	100	100
f_4	*KM*	0	*M*	1.3097E-02	4.5754E-07	8.7264E-02	0	2.2388E-02	1.0302E-02
	T	1.0000E-02	*H*	1.3097E-02	4.5754E-07	8.7264E-02	0	2.2388E-02	1.0302E-02
			B	20	96	2	99	4	26
f_5	*KM*	-4.6870E+00	*M*	-4.6877E+00	-4.6877E+00	-4.5523E+00	-4.6877E+00	-4.6877E+00	-4.6877E+00
	T	1.6000E-03	*H*	1.4041E-04	1.4041E-04	2.9585E-02	1.4041E-04	1.3958E-04	1.4041E-04
			B	62	97	9	96	100	100
f_6	*KM*	0	*M*	5.7515E-148	1.8877E-16	7.1054E-17	4.0330E-135	8.8746E-05	1.9223E-138
	T	1.0000E-02	*H*	5.7515E-148	1.8877E-16	7.1054E-17	4.0330E-135	8.8746E-05	1.9223E-138
			B	99	100	100	100	100	100
f_7	*KM*	0	*M*	0	6.5295E-17	0	0	1.3775E-04	0
	T	1.0000E-02	*H*	0	6.5295E-17	0	0	1.3775E-04	0
			B	100	100	100	100	97	100
f_8	*KM*	-2.0949E+03	*M*	-2.0949E+03	-2.0949E+03	-1.5773E+03	-2.0949E+03	-1.5074E+04	-2.0949E+03
	T	2.5000E-01	*H*	6.8911E-06	6.8911E-06	3.2814E-01	6.8911E-06	8.6103E-01	6.8911E-06
			B	70	98	0	87	1	99
f_9	*KM*	0	*M*	3.5172E-15	4.3698E-15	4.1567E-15	8.8818E-16	7.8859E-03	2.5935E-15
	T	1.0000E-02	*H*	3.5172E-15	4.3698E-15	4.1567E-15	8.8818E-16	7.8859E-03	2.5935E-15
			B	98	100	100	100	31	100
f_{10}	*KM*	0	*M*	4.4960E-43	8.1871E-06	2.1329E-115	6.4222E-56	8.7774E-02	1.4778E-44
	T	1.0000E-02	*H*	4.4960E-43	8.1871E-06	2.1329E-115	6.4222E-56	8.7774E-02	1.4778E-44
			B	96	85	100	100	1	97
Genel Ortalama Hata				**1.3245E-03**	**1.5594E-05**	**1.6588E-01**	**1.4730E-05**	**9.7949E-02**	**1.0450E-03**
Genel Ortalama Başarı(%)				**81.4**	**97.6**	**62.4**	**98.2**	**63.4**	**92.2**

Tablo 4.44. MBOGE algoritmasının 10 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

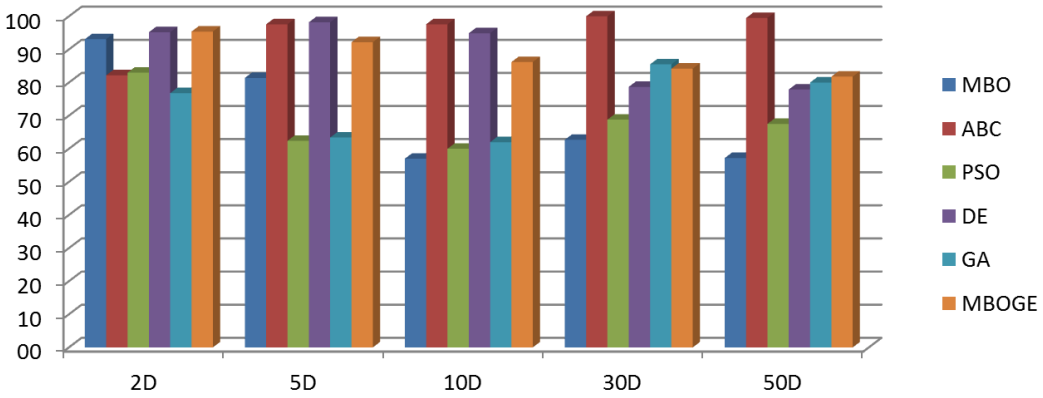
				MBO	**ABC**	**PSO**	**DE**	**GA**	**MBOGE**
f_1	*KM*	0	*M*	1.7398E-122	9.0681E-17	2.9633E-269	3.4394E-249	1.4506E-06	9.4716E-187
	T	1.1000E-02	*H*	1.7398E-122	9.0681E-17	2.9633E-269	3.4394E-249	1.4506E-06	9.4716E-187
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	3.8541E-01	0	5.2733E+00	0	2.4249E-04	0
	T	1.1000E-02	*H*	3.8541E-01	0	5.2733E+00	0	2.4249E-04	0
			B	30	100	0	96	100	100
f_3	*KM*	0	*M*	3.8799E-140	9.2792E-17	1.0598E-275	1.4532E-248	7.2226E-06	4.5581E-144
	T	1.1000E-02	*H*	3.8799E-140	9.2792E-17	1.0598E-275	1.4532E-248	7.2226E-06	4.5581E-144
			B	100	100	100	100	100	100
f_4	*KM*	0	*M*	3.1914E-02	2.9584E-04	3.8357E-01	0	3.7359E-02	2.2663E-02
	T	1.1000E-02	*H*	3.1914E-02	2.9584E-04	3.8357E-01	0	3.7359E-02	2.2663E-02
			B	5	76	0	98	2	8
f_5	*KM*	-9.6600E+00	*M*	-9.5849E+00	-9.6602E+00	-8.4746E+00	-9.6602E+00	-9.6601E+00	-9.6602E+00
	T	5.0000E-03	*H*	7.8346E-03	1.5705E-05	1.3987E-01	1.5705E-05	9.8244E-06	1.5705E-05
			B	17	100	0	75	100	98
f_6	*KM*	0	*M*	3.7558E-16	2.7008E-16	4.5853E-16	3.7114E-202	3.2727E-04	2.5428E-16
	T	1.1000E-02	*H*	3.7558E-16	2.7008E-16	4.5853E-16	3.7114E-202	3.2727E-04	2.5428E-16
			B	90	100	100	100	100	100
f_7	*KM*	0	*M*	0	9.3529E-17	0	0	5.5265E-04	0
	T	1.1000E-02	*H*	0	9.3529E-17	0	0	5.5265E-04	0
			B	99	100	100	100	99	100
f_8	*KM*	-4.1898E+03	*M*	-4.1211E+03	-4.1898E+03	-2.6082E+03	-4.1898E+03	-1.6321E+04	-4.1898E+03
	T	3.0000E-01	*H*	1.6662E-02	6.8911E-06	6.0642E-01	6.8911E-06	7.4329E-01	6.8911E-06
			B	21	100	0	92	0	93
f_9	*KM*	0	*M*	8.3489E-15	7.9936E-15	7.3541E-15	3.8014E-15	1.1570E-02	6.1462E-15
	T	1.1000E-02	*H*	8.3489E-15	7.9936E-15	7.3541E-15	3.8014E-15	1.1570E-02	6.1462E-15
			B	67	100	100	100	19	100
f_{10}	*KM*	0	*M*	1.9667E-02	5.4424E-11	2.2224E-67	4.5729E-08	1.4122E-01	3.8272E-08
	T	1.1000E-02	*H*	1.9667E-02	5.4424E-11	2.2224E-67	4.5729E-08	1.4122E-01	3.8272E-08
			B	41	100	100	88	0	63
Genel Ortalama Hata				**4.6149E-02**	**3.1844E-05**	**6.4032E-01**	**2.2642E-06**	**9.3458E-02**	**2.2685E-03**
Genel Ortalama Başarı(%)				**57.0**	**97.6**	**60.0**	**94.9**	**62.0**	**86.2**

Tablo 4.45. MBOGE algoritmasının 30 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	ABC	PSO	DE	GA	MBOGE
f_1	*KM*	0	*M*	1.3000E-23	4.9316E-16	2.6521E-79	2.9023E-263	8.7879E-06	1.5620E-33
	T	5.0000E-01	*H*	1.3000E-23	4.9316E-16	2.6521E-79	2.9023E-263	8.7879E-06	1.5620E-33
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	3.1244E+00	0	3.1202E+01	0	1.9920E-03	1.4140E-14
	T	5.0000E-01	*H*	3.1244E+00	0	3.1202E+01	0	1.9920E-03	1.4140E-14
			B	2	100	0	73	100	100
f_3	*KM*	0	*M*	3.7380E-19	4.8509E-16	2.3738E-78	7.2600E-262	1.3585E-04	2.0227E-36
	T	5.0000E-01	*H*	3.7380E-19	4.8509E-16	2.3738E-78	7.2600E-262	1.3585E-04	2.0227E-36
			B	98	100	100	100	100	100
f_4	*KM*	0	*M*	5.9400E-06	7.3275E-17	2.9088E-03	0	1.4791E-02	1.9518E-07
	T	5.0000E-01	*H*	5.9400E-06	7.3275E-17	2.9088E-03	0	1.4791E-02	1.9518E-07
			B	98	100	100	100	100	100
f_5	*KM*	-2.8980E+01	*M*	-2.9121E+01	-2.9631E+01	-2.3443E+01	-2.6569E+01	-2.9627E+01	-2.9612E+01
	T	7.0000E-01	*H*	4.8288E-03	2.1963E-02	2.3617E-01	9.0743E-02	2.1853E-02	2.1339E-02
			B	83	100	0	0	100	100
f_6	*KM*	0	*M*	1.0433E-04	5.5956E-16	5.3999E-11	6.1784E-156	1.2395E-03	1.3356E-10
	T	5.0000E-01	*H*	1.0433E-04	5.5956E-16	5.3999E-11	6.1784E-156	1.2395E-03	1.3356E-10
			B	97	100	100	100	100	100
f_7	*KM*	0	*M*	1.2803E-24	4.6608E-16	2.6501E-33	0	3.9747E-03	1.0444E-28
	T	5.0000E-01	*H*	1.2803E-24	4.6608E-16	2.6501E-33	0	3.9747E-03	1.0444E-28
			B	99	100	100	100	100	100
f_8	*KM*	-1.2569E+04	*M*	-1.2247E+04	-1.2569E+04	-6.3747E+03	-1.2569E+04	-4.2350E+04	-1.2553E+04
	T	1.0000E+00	*H*	2.6264E-02	3.8714E-05	9.7171E-01	3.8714E-05	7.0321E-01	1.2927E-03
			B	3	100	0	88	0	42
f_9	*KM*	0	*M*	5.6005E-02	3.1086E-14	3.8050E-14	4.4409E-15	1.4639E-02	3.0210E-12
	T	5.0000E-01	*H*	5.6005E-02	3.1086E-14	3.8050E-14	4.4409E-15	1.4639E-02	3.0210E-12
			B	47	100	88	100	100	100
f_{10}	*KM*	0	*M*	1.2274E+01	3.1423E-02	8.1776E-06	5.3910E-01	4.2169E-01	4.1266E+00
	T	5.0000E-01	*H*	1.2274E+01	3.1423E-02	8.1776E-06	5.3910E-01	4.2169E-01	4.1266E+00
			B	0	100	100	26	55	0
Genel Ortalama Hata				**1.5486E+00**	**5.3425E-03**	**3.2413E+00**	**6.2989E-02**	**1.1835E-01**	**4.1492E-01**
Genel Ortalama Başarı(%)				**62.7**	**100.0**	**68.8**	**78.7**	**85.5**	**84.2**

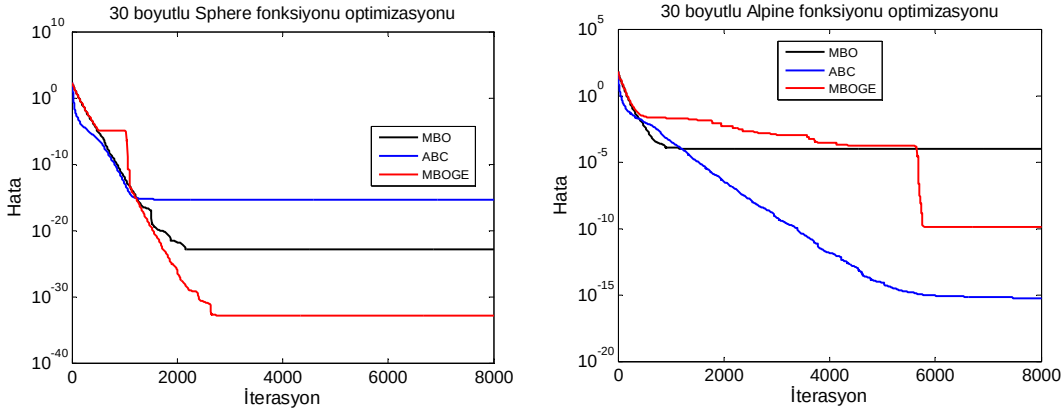
Tablo 4.46. MBOGE algoritmasının 50 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	ABC	PSO	DE	GA	MBOGE
f_1	*KM*	0	*M*	7.6778E-10	8.9649E-16	1.9490E-48	0	4.5167E-05	4.9539E-16
	T	6.0000E-01	*H*	7.6778E-10	8.9649E-16	1.9490E-48	0	4.5167E-05	4.9539E-16
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	7.6024E+00	0	5.9996E+01	0	8.3003E-03	1.6330E-10
	T	6.0000E-01	*H*	7.6024E+00	0	5.9996E+01	0	8.3003E-03	1.6330E-10
			B	0	100	0	75	100	100
f_3	*KM*	0	*M*	5.0462E-10	8.2798E-16	8.9623E-48	0	9.9204E-04	9.4265E-14
	T	6.0000E-01	*H*	5.0462E-10	8.2798E-16	8.9623E-48	0	9.9204E-04	9.4265E-14
			B	96	100	97	100	100	100
f_4	*KM*	0	*M*	9.8167E-04	7.9936E-17	4.2093E-14	0	2.2371E-02	1.6761E-08
	T	6.0000E-01	*H*	9.8167E-04	7.9936E-17	4.2093E-14	0	2.2371E-02	1.6761E-08
			B	87	100	100	100	100	100
f_5	*KM*	-4.8300E+01	*M*	-4.8606E+01	-4.9623E+01	-3.9379E+01	-3.4137E+01	-4.9602E+01	-4.9575E+01
	T	1.3500E+00	*H*	6.2906E-03	2.6654E-02	2.2653E-01	4.1487E-01	2.6244E-02	2.5710E-02
			B	77	100	0	0	100	100
f_6	*KM*	0	*M*	1.4317E-02	9.3362E-16	1.0707E-08	1.9308E-04	3.1337E-03	1.2352E-05
	T	6.0000E-01	*H*	1.4317E-02	9.3362E-16	1.0707E-08	1.9308E-04	3.1337E-03	1.2352E-05
			B	90	100	100	100	100	100
f_7	*KM*	0	*M*	5.8536E-08	8.4834E-16	9.7930E-32	0	1.6281E-02	8.5398E-14
	T	6.0000E-01	*H*	5.8536E-08	8.4834E-16	9.7930E-32	0	1.6281E-02	8.5398E-14
			B	90	100	100	100	100	100
f_8	*KM*	-2.0949E+04	*M*	-2.0160E+04	-2.0949E+04	-9.7364E+03	-2.0949E+04	-6.5528E+04	-2.0860E+04
	T	1.5000E+00	*H*	3.9148E-02	6.8911E-06	1.1516E+00	6.8911E-06	6.8030E-01	4.2458E-03
			B	0	100	0	86	0	18
f_9	*KM*	0	*M*	3.3808E-01	5.8371E-14	1.3061E-12	7.7804E-15	2.3689E-02	9.5420E-05
	T	6.0000E-01	*H*	3.3808E-01	5.8371E-14	1.3061E-12	7.7804E-15	2.3689E-02	9.5420E-05
			B	32	100	78	100	100	100
f_{10}	*KM*	0	*M*	3.1367E+01	2.4628E-01	3.1404E-02	9.3642E-01	9.3579E-01	8.9985E+00
	T	6.0000E-01	*H*	3.1367E+01	2.4628E-01	3.1404E-02	9.3642E-01	9.3579E-01	8.9985E+00
			B	0	95	100	18	0	0
Genel Ortalama Hata				**3.9368E+00**	**2.7294E-02**	**6.1405E+00**	**1.3515E-01**	**1.7172E-01**	**9.0286E-01**
Genel Ortalama Başarı(%)				**57.2**	**99.5**	**67.5**	**77.9**	**80.0**	**81.8**



Şekil 4.36. MBOGE ve diğer algoritmaların başarı dağılımları

Tablolarda test fonksiyonlarının ilgili problem boyutundaki küresel minimumları ile algoritmalar tarafından elde edilen sonuçların bu küresel minimumlara göre Bölüm 4.7.1.1'deki yaklaşımla hesaplanan hata değerleri de verilmiştir. Tablolarda verilen başarı yüzdeleri Şekil 4.36'da bar grafiği ile gösterilmiş, ABC, MBO ve VNMBO algoritmalarının tek modlu (Sphere fonksiyonu) ve çok modlu (Alpine fonksiyonu) fonksiyonlardaki yakınsama davranışları Şekil 4.37'de birer örnek ile gösterilmiştir.



(a) (b)

Şekil 4.37. ABC, MBO ve MBOGE algoritmaların (a) 30 boyutlu tek modlu Sphere fonksiyonunun ve (b) 30 boyutlu çok modlu Alpine fonksiyonunun küresel minimumuna yakınsamaları

Öne çıkan sonuçlardan bazıları şunlardır:

- Özetle MBO algoritmasına rasgele küresel arama evresi eklenerek elde edilen MBOGE algoritmasında MBO algoritmasının performansının iyileştirilmesi sağlanmıştır.

- Tablo 4.42'den Tablo 4.46'ya kadar verilen sonuçlardan MBOGE algoritması MBO algoritması ile kıyaslanıp genel bir değerlendirme yapıldığında düşük problem boyutlarında küresel minimuma yakınsama açısından önemli bir gelişme gözlemlenmemesine rağmen problem boyutu arttıkça elde edilen sonuçların MBO algoritması sonuçlarından daha fazla küresel minimuma yakınsadığı gözlemlenmektedir.

- Şekil 4.36'da görüldüğü gibi, bütün problem boyutlarında MBOGE algoritmasının başarı yüzdesi MBO algoritmasına kıyasla daha iyidir. Bu durum algoritmaya ilave edilen keşif yeteneği sayesinde algoritmanın yerel minimumlardan daha başarılı bir şekilde uzaklaşabildiğini doğrulamaktadır.

- Algoritmanın küresel minimuma yakınsama davranışı kriter alındığında, MBOGE algoritmasının tek ve çok modlu problemler üzerindeki davranışları Şekil 4.37'de olduğu gibi ayrı ayrı incelenebilir. ABC ve MBO algoritmaları tek modlu fonksiyonlar için makul hata seviyelerinde doyuma ulaşırken, MBOGE algoritması hata değerini düşürmeye devam ederek çok daha düşük hata değerlerinde doyuma ulaşır. Çok modlu fonksiyonlarda ise MBOGE algoritmasının küresel minimuma yakınsama miktarı MBO algoritmasından daha iyidir. ABC algoritmasının yakınsama değeri ise MBOGE algoritmasından daha iyidir.

- MBO algoritmasından onun modifiye bir formu olan MBOGE elde edilirken algoritmanın hem keşif hem de yakınsama yetenekleri artmıştır. MBO algoritmasına kıyasla elde edilen daha düşük nihai hata değerleri algoritmanın yakınsama kabiliyetinin arttığının, daha yüksek başarı oranları ise algoritmanın keşif kabiliyetinin arttığının birer göstergesidir.

4.7.2. Paralel veya sıralı çalışan meta-sezgisel algoritmalar

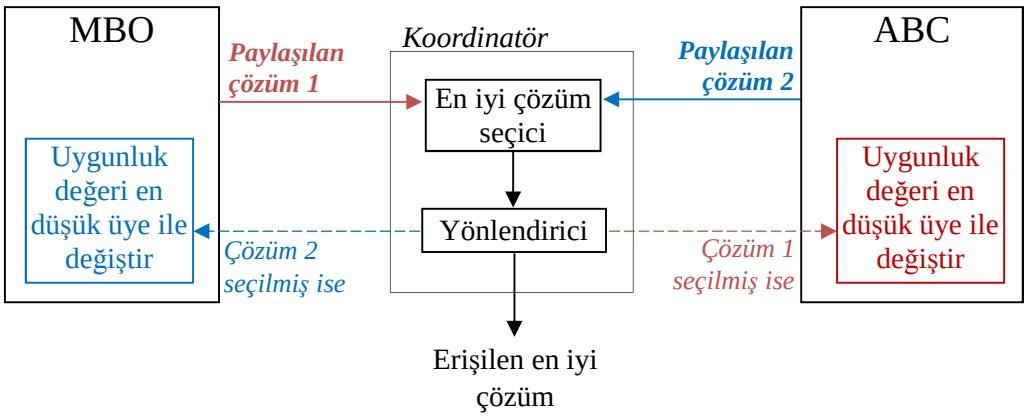
Bu bölümdeki paralel çalışan meta-sezgisel algoritmalarda, kurulan mekanizmalar ile birden fazla aynı veya farklı meta-sezgisel algoritmanın birbiri ile eş zamanlı çalışması ve birbirleri ile çözümlerini paylaşmaları hedeflenmiştir. Böylece iyi

çözüm üretemeyen algoritmanın diğerlerinin paylaştığı daha kaliteli çözümleri kullanarak kendi çözüm kalitelerini artırması sağlanmıştır. Bu yöntemlerin en büyük avantajı, özellikle yerel minimumlara takılma ve kötü başlangıç koşullarından kaynaklanan olumsuzluklar nedeni ile meydana gelen başarısız en iyilemelerin oranını önemli ölçüde azaltmalarıdır. En büyük dezavantajları ise implementasyonlarında daha fazla sistem kaynağına ihtiyaç duyulması veya zaman paylaşımlı çalışmadan kaynaklanan artan işlem süresi maliyetidir.

Sıralı çalışan algoritmada hedeflenen ise keşif yeteneği iyi olan bir algoritma ile yakınsama yeteneği iyi olan iki farklı algoritmanın makul bir sıra ile peş peşe çalıştırılmaları neticesinde bu iki özelliğin tek bir yapı altında toplanmasının sağlanmasıdır. Bu yaklaşımda sistem kaynağı gereksinimi ve işlem süresi maliyeti açısından herhangi bir dezavantaj görülmemektedir.

4.7.2.1. Göçmen kuşlar ve yapay arı koloni kooperatif en iyileme algoritması

Göçmen kuşlar ve yapay arı koloni kooperatif en iyileme (Migrating Birds and Artificial Bee Colony Cooperative Optimization – MBABC-CO) algoritması MBO ve ABC algoritmalarının paralel çalıştığı ve her iterasyonda algoritmaların en iyi çözümlerini birbirleri ile paylaştıkları bir işbirliği stratejisi ile çalışırlar. Şekil 4.38 MBABC-CO algoritmasının çalışma felsefesini şematik olarak ifade etmektedir. Şekil 8.39'da ise MBABC-CO algoritması için oluşturulan sözde kod verilmiştir.



Şekil 4.38. MBABC-CO Algoritması şematik diyagramı

MBO ve ABC algoritmalarının başlangıç popülasyonlarını Denklem (2.1) ile üret
MBABC-CO maliyet değerine ($C_{MBABC-CO}$) başlangıç değerini ata
Önceki paylaşılan en iyi ABC çözümüne (X_{ABC_Hist})başlangıç değeri ata
Önceki paylaşılan en iyi MBO çözümüne (X_{MBO_Hist})başlangıç değeri ata
for *i = 1* ***to*** *K*
    ABC algoritmasını kendi popülasyonuna uygula
    ABC popülasyonundaki en iyi üyeyi (X_{ABC}) ve maliyetini (C_{ABC}) hesapla
    MBO algoritmasını kendi popülasyonuna uygula
    MBO popülasyonundaki en iyi üyeyi (X_{MBO}) ve maliyetini (C_{MBO}) hesapla
    if $C_{ABC} < C_{MBO}$ ***ve*** $X_{ABC} \neq X_{ABC\_Hist}$ ***then***
        MBO popülasyonunun en kötü üyesini X_{ABC} ile değiştir
    elseif $C_{ABC} > C_{MBO}$ ***ve*** $X_{MBO} \neq X_{MBO\_Hist}$ *then*
        ABC popülasyonunun en kötü üyesini X_{MBO} ile değiştir
    endif
    if $C_{ABC} < C_{MBABC-CO}$ ***then***
        $C_{MBABC-CO} = C_{ABC}$; $X_{MBABC-CO} = X_{ABC}$;
    endif
    if $C_{MBO} < C_{MBABC-CO}$ ***then***
        $C_{MBABC-CO} = C_{MBO}$; $X_{MBABC-CO} = X_{MBO}$;
    endif
    $X_{ABC\_Hist} = X_{ABC}$; $X_{MBO\_Hist} = X_{MBO}$;
endfor
return *Mevcut en iyi çözüm ($X_{MBABC-CO}$)*

Şekil 4.39. Zaman paylaşımlı MBABC-CO algoritması sözde kodu

MBABC-CO algoritmasının çalışması şu şekildedir. Paralel çalışan MBO ve ABC algoritmaları kendi popülasyonları içerisindeki en yüksek uygunluklu çözümü mukayese edilmek üzere paylaşırlar. Paylaşılan en iyi çözümlerinden hangisinin uygunluk değeri daha iyi ise o çözüm iterasyonun en iyi çözümü olarak seçilir. Bu seçilmiş çözüm karşı taraftaki algoritmaya gönderilir. Seçilmiş çözümü alan algoritma bu çözümü kontrol eder. Eğer, paylaşılan çözüm daha önce kendisi ile paylaşılmamış yeni bir çözüm ise kendisinin en kötü çözümü ile değiştirerek sahip olduğu çözüm kalitesini yükseltir. MBABC-CO algoritması küresel arama yeteneği iyi olan ABC algoritması ile yerel yakınsama özelliği iyi olan MBO algoritmasının bu iyi özelliklerini bir araya getirir. İki algoritmanın yardımlaşmasıyla yerel minimumlardan kolayca kaçabilen ve küresel minimuma başarılı bir şekilde yakınsayabilen bir yöntem elde edilir.

Tablo 4.47. MBABC-CO algoritmasının 2 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	**ABC**	**PSO**	**DE**	**GA**	**MBOABC-CO**
f_1	*KM*	0	*M*	0	2.6992E-17	1.7508E-149	1.7525E-161	4.2287E-17	0
	T	5.0000E-03	*H*	0	2.6992E-17	1.7508E-149	1.7525E-161	4.2287E-17	0
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	0	1.2448E-07	0	0	1.4211E-16	0
	T	5.0000E-03	*H*	0	1.2448E-07	0	0	1.4211E-16	0
			B	95	90	80	96	97	100
f_3	*KM*	0	*M*	0	3.6367E-17	4.3904E-149	2.1397E-161	4.9386E-19	0
	T	5.0000E-03	*H*	0	3.6367E-17	4.3904E-149	2.1397E-161	4.9386E-19	0
			B	100	100	100	100	100	100
f_4	*KM*	0	*M*	7.4873E-04	1.2457E-02	1.9230E-03	0	6.7057E-03	0
	T	5.0000E-03	*H*	7.4873E-04	1.2457E-02	1.9230E-03	0	6.7057E-03	0
			B	45	8	37	68	10	100
f_5	*KM*	-1.8010E+00	*M*	-1.8013E+00	-1.8013E+00	-1.8013E+00	-1.8013E+00	-1.8013E+00	-1.8013E+00
	T	3.5000E-04	*H*	1.6844E-04	1.6844E-04	1.6844E-04	1.6844E-04	1.6844E-04	1.6844E-04
			B	100	100	93	100	99	100
f_6	*KM*	0	*M*	9.0640E-244	4.8596E-07	9.4008E-85	8.6804E-104	8.7397E-15	6.8456E-247
	T	5.0000E-03	*H*	9.0640E-244	4.8596E-07	9.4008E-85	8.6804E-104	8.7397E-15	6.8456E-247
			B	100	100	100	100	100	100
f_7	*KM*	0	*M*	0	3.5134E-17	0	0	3.8941E-20	0
	T	5.0000E-03	*H*	0	3.5134E-17	0	0	3.8941E-20	0
			B	100	100	100	100	97	100
f_8	*KM*	-8.3797E+02	*M*	-8.3797E+02	-8.3797E+02	-7.6690E+02	-8.3797E+02	-1.2035E+04	-8.3797E+02
	T	1.0000E-01	*H*	5.0425E-06	5.0435E-06	9.2668E-02	5.0425E-06	9.3038E-01	5.0425E-06
			B	91	87	20	88	22	100
f_9	*KM*	0	*M*	8.8818E-16	1.8829E-15	8.8818E-16	8.8818E-16	1.2773E-07	8.8818E-16
	T	5.0000E-03	*H*	8.8818E-16	1.8829E-15	8.8818E-16	8.8818E-16	1.2773E-07	8.8818E-16
			B	100	97	100	100	87	100
f_{10}	*KM*	0	*M*	6.1752E-130	2.2923E-03	3.5879E-74	7.7725E-71	7.4712E-04	1.0413E-128
	T	5.0000E-03	*H*	6.1752E-130	2.2923E-03	3.5879E-74	7.7725E-71	7.4712E-04	1.0413E-128
			B	100	40	100	100	56	100
Genel Ortalama Hata				**9.2221E-05**	**1.4923E-03**	**9.4759E-03**	**1.7348E-05**	**9.3800E-02**	**1.7348E-05**
Genel Ortalama Başarı(%)				**93.1**	**82.2**	**83.0**	**95.2**	**76.8**	**100.0**

Tablo 4.48. MBABC-CO algoritmasının 5 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	**ABC**	**PSO**	**DE**	**GA**	**MBOABC-CO**
f_1	*KM*	0	*M*	3.1987E-222	6.4359E-17	4.8685E-246	7.6169E-171	2.5427E-07	3.8988E-224
	T	1.0000E-02	*H*	3.1987E-222	6.4359E-17	4.8685E-246	7.6169E-171	2.5427E-07	3.8988E-224
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	0	0	1.2139E+00	0	5.2194E-05	0
	T	1.0000E-02	*H*	0	0	1.2139E+00	0	5.2194E-05	0
			B	69	100	13	100	100	100
f_3	*KM*	0	*M*	5.0794E-222	6.6576E-17	8.8594E-246	1.9272E-170	9.4855E-07	1.6659E-223
	T	1.0000E-02	*H*	5.0794E-222	6.6576E-17	8.8594E-246	1.9272E-170	9.4855E-07	1.6659E-223
			B	100	100	100	100	100	100
f_4	*KM*	0	*M*	1.3097E-02	4.5754E-07	8.7264E-02	0	2.2388E-02	0
	T	1.0000E-02	*H*	1.3097E-02	4.5754E-07	8.7264E-02	0	2.2388E-02	0
			B	20	96	2	99	4	100
f_5	*KM*	-4.6870E+00	*M*	-4.6877E+00	-4.6877E+00	-4.5523E+00	-4.6877E+00	-4.6877E+00	-4.6877E+00
	T	1.6000E-03	*H*	1.4041E-04	1.4041E-04	2.9585E-02	1.4041E-04	1.3958E-04	1.4041E-04
			B	62	97	9	96	100	100
f_6	*KM*	0	*M*	5.7515E-148	1.8877E-16	7.1054E-17	4.0330E-135	8.8746E-05	3.9693E-161
	T	1.0000E-02	*H*	5.7515E-148	1.8877E-16	7.1054E-17	4.0330E-135	8.8746E-05	3.9693E-161
			B	99	100	100	100	100	100
f_7	*KM*	0	*M*	0	6.5295E-17	0	0	1.3775E-04	0
	T	1.0000E-02	*H*	0	6.5295E-17	0	0	1.3775E-04	0
			B	100	100	100	100	97	100
f_8	*KM*	-2.0949E+03	*M*	-2.0949E+03	-2.0949E+03	-1.5773E+03	-2.0949E+03	-1.5074E+04	-2.0949E+03
	T	2.5000E-01	*H*	6.8911E-06	6.8911E-06	3.2814E-01	6.8911E-06	8.6103E-01	6.8911E-06
			B	70	98	0	87	1	100
f_9	*KM*	0	*M*	3.5172E-15	4.3698E-15	4.1567E-15	8.8818E-16	7.8859E-03	2.6645E-15
	T	1.0000E-02	*H*	3.5172E-15	4.3698E-15	4.1567E-15	8.8818E-16	7.8859E-03	2.6645E-15
			B	98	100	100	100	31	100
f_{10}	*KM*	0	*M*	4.4960E-43	8.1871E-06	2.1329E-115	6.4222E-56	8.7774E-02	8.8906E-45
	T	1.0000E-02	*H*	4.4960E-43	8.1871E-06	2.1329E-115	6.4222E-56	8.7774E-02	8.8906E-45
			B	96	85	100	100	1	100
Genel Ortalama Hata				**1.3245E-03**	**1.5594E-05**	**1.6588E-01**	**1.4730E-05**	**9.7949E-02**	**1.4730E-05**
Genel Ortalama Başarı(%)				**81.4**	**97.6**	**62.4**	**98.2**	**63.4**	**100.0**

Tablo 4.49. MBABC-CO algoritmasının 10 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	**ABC**	**PSO**	**DE**	**GA**	**MBOABC-CO**
f_1	*KM*	0	*M*	1.7398E-122	9.0681E-17	2.9633E-269	3.4394E-249	1.4506E-06	1.4516E-162
	T	1.1000E-02	*H*	1.7398E-122	9.0681E-17	2.9633E-269	3.4394E-249	1.4506E-06	1.4516E-162
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	3.8541E-01	0	5.2733E+00	0	2.4249E-04	0
	T	1.1000E-02	*H*	3.8541E-01	0	5.2733E+00	0	2.4249E-04	0
			B	30	100	0	96	100	100
f_3	*KM*	0	*M*	3.8799E-140	9.2792E-17	1.0598E-275	1.4532E-248	7.2226E-06	3.9767E-169
	T	1.1000E-02	*H*	3.8799E-140	9.2792E-17	1.0598E-275	1.4532E-248	7.2226E-06	3.9767E-169
			B	100	100	100	100	100	100
f_4	*KM*	0	*M*	3.1914E-02	2.9584E-04	3.8357E-01	0	3.7359E-02	0
	T	1.1000E-02	*H*	3.1914E-02	2.9584E-04	3.8357E-01	0	3.7359E-02	0
			B	5	76	0	98	2	100
f_5	*KM*	-9.6600E+00	*M*	-9.5849E+00	-9.6602E+00	-8.4746E+00	-9.6602E+00	-9.6601E+00	-9.6602E+00
	T	5.0000E-03	*H*	7.8346E-03	1.5705E-05	1.3987E-01	1.5705E-05	9.8244E-06	1.5705E-05
			B	17	100	0	75	100	100
f_6	*KM*	0	*M*	3.7558E-16	2.7008E-16	4.5853E-16	3.7114E-202	3.2727E-04	1.0165E-80
	T	1.1000E-02	*H*	3.7558E-16	2.7008E-16	4.5853E-16	3.7114E-202	3.2727E-04	1.0165E-80
			B	90	100	100	100	100	100
f_7	*KM*	0	*M*	0	9.3529E-17	0	0	5.5265E-04	0
	T	1.1000E-02	*H*	0	9.3529E-17	0	0	5.5265E-04	0
			B	99	100	100	100	99	100
f_8	*KM*	-4.1898E+03	*M*	-4.1211E+03	-4.1898E+03	-2.6082E+03	-4.1898E+03	-1.6321E+04	-4.1898E+03
	T	3.0000E-01	*H*	1.6662E-02	6.8911E-06	6.0642E-01	6.8911E-06	7.4329E-01	6.8911E-06
			B	21	100	0	92	0	100
f_9	*KM*	0	*M*	8.3489E-15	7.9936E-15	7.3541E-15	3.8014E-15	1.1570E-02	4.4409E-15
	T	1.1000E-02	*H*	8.3489E-15	7.9936E-15	7.3541E-15	3.8014E-15	1.1570E-02	4.4409E-15
			B	67	100	100	100	19	100
f_{10}	*KM*	0	*M*	1.9667E-02	5.4424E-11	2.2224E-67	4.5729E-08	1.4122E-01	4.3519E-16
	T	1.1000E-02	*H*	1.9667E-02	5.4424E-11	2.2224E-67	4.5729E-08	1.4122E-01	4.3519E-16
			B	41	100	100	88	0	100
Genel Ortalama Hata				**4.6149E-02**	**3.1844E-05**	**6.4032E-01**	**2.2642E-06**	**9.3458E-02**	**2.2596E-06**
Genel Ortalama Başarı(%)				**57.0**	**97.6**	**60.0**	**94.9**	**62.0**	**100.0**

Tablo 4.50. MBABC-CO algoritmasının 30 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	**ABC**	**PSO**	**DE**	**GA**	**MBOABC-CO**
f_1	*KM*	0	*M*	1.3000E-23	4.9316E-16	2.6521E-79	2.9023E-263	8.7879E-06	2.0634E-26
	T	5.0000E-01	*H*	1.3000E-23	4.9316E-16	2.6521E-79	2.9023E-263	8.7879E-06	2.0634E-26
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	3.1244E+00	0	3.1202E+01	0	1.9920E-03	0
	T	5.0000E-01	*H*	3.1244E+00	0	3.1202E+01	0	1.9920E-03	0
			B	2	100	0	73	100	100
f_3	*KM*	0	*M*	3.7380E-19	4.8509E-16	2.3738E-78	7.2600E-262	1.3585E-04	9.1927E-26
	T	5.0000E-01	*H*	3.7380E-19	4.8509E-16	2.3738E-78	7.2600E-262	1.3585E-04	9.1927E-26
			B	98	100	100	100	100	100
f_4	*KM*	0	*M*	5.9400E-06	7.3275E-17	2.9088E-03	0	1.4791E-02	0
	T	5.0000E-01	*H*	5.9400E-06	7.3275E-17	2.9088E-03	0	1.4791E-02	0
			B	98	100	100	100	100	100
f_5	*KM*	-2.8980E+01	*M*	-2.9121E+01	-2.9631E+01	-2.3443E+01	-2.6569E+01	-2.9627E+01	-2.9631E+01
	T	7.0000E-01	*H*	4.8288E-03	2.1963E-02	2.3617E-01	9.0743E-02	2.1853E-02	2.1966E-02
			B	83	100	0	0	100	100
f_6	*KM*	0	*M*	1.0433E-04	5.5956E-16	5.3999E-11	6.1784E-156	1.2395E-03	1.8016E-17
	T	5.0000E-01	*H*	1.0433E-04	5.5956E-16	5.3999E-11	6.1784E-156	1.2395E-03	1.8016E-17
			B	97	100	100	100	100	100
f_7	*KM*	0	*M*	1.2803E-24	4.6608E-16	2.6501E-33	0	3.9747E-03	3.7983E-26
	T	5.0000E-01	*H*	1.2803E-24	4.6608E-16	2.6501E-33	0	3.9747E-03	3.7983E-26
			B	99	100	100	100	100	100
f_8	*KM*	-1.2569E+04	*M*	-1.2247E+04	-1.2569E+04	-6.3747E+03	-1.2569E+04	-4.2350E+04	-1.2569E+04
	T	1.0000E+00	*H*	2.6264E-02	3.8714E-05	9.7171E-01	3.8714E-05	7.0321E-01	3.8714E-05
			B	3	100	0	88	0	100
f_9	*KM*	0	*M*	5.6005E-02	3.1086E-14	3.8050E-14	4.4409E-15	1.4639E-02	2.4052E-14
	T	5.0000E-01	*H*	5.6005E-02	3.1086E-14	3.8050E-14	4.4409E-15	1.4639E-02	2.4052E-14
			B	47	100	88	100	100	100
f_{10}	*KM*	0	*M*	1.2274E+01	3.1423E-02	8.1776E-06	5.3910E-01	4.2169E-01	9.9586E-03
	T	5.0000E-01	*H*	1.2274E+01	3.1423E-02	8.1776E-06	5.3910E-01	4.2169E-01	9.9586E-03
			B	0	100	100	26	55	100
Genel Ortalama Hata				**1.5486E+00**	**5.3425E-03**	**3.2413E+00**	**6.2989E-02**	**1.1835E-01**	**3.1963E-03**
Genel Ortalama Başarı(%)				**62.7**	**100.0**	**68.8**	**78.7**	**85.5**	**100.0**

Tablo 4.51. MBABC-CO algoritmasının 50 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

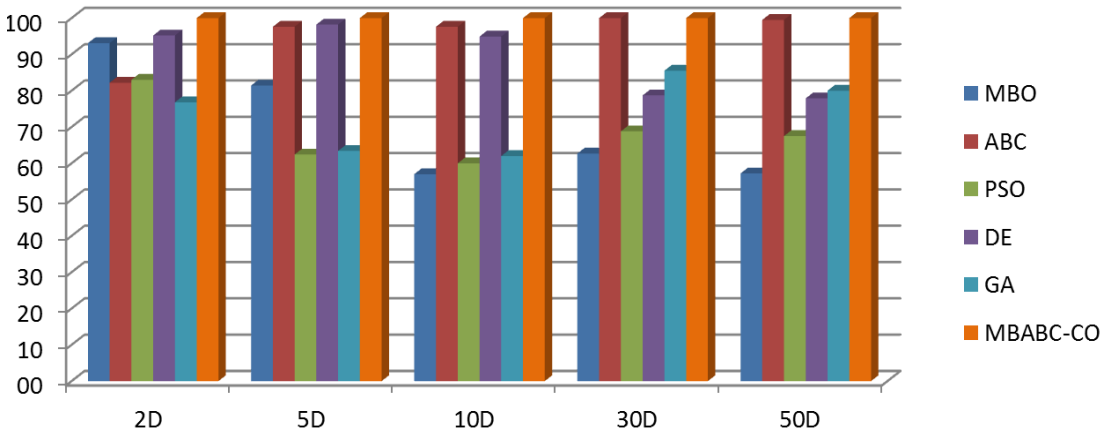
				MBO	**ABC**	**PSO**	**DE**	**GA**	**MBOABC-CO**
f_1	*KM*	0	*M*	7.6778E-10	8.9649E-16	1.9490E-48	0	4.5167E-05	2.4504E-20
	T	6.0000E-01	*H*	7.6778E-10	8.9649E-16	1.9490E-48	0	4.5167E-05	2.4504E-20
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	7.6024E+00	0	5.9996E+01	0	8.3003E-03	0
	T	6.0000E-01	*H*	7.6024E+00	0	5.9996E+01	0	8.3003E-03	0
			B	0	100	0	75	100	100
f_3	*KM*	0	*M*	5.0462E-10	8.2798E-16	8.9623E-48	0	9.9204E-04	5.8451E-20
	T	6.0000E-01	*H*	5.0462E-10	8.2798E-16	8.9623E-48	0	9.9204E-04	5.8451E-20
			B	96	100	97	100	100	100
f_4	*KM*	0	*M*	9.8167E-04	7.9936E-17	4.2093E-14	0	2.2371E-02	0
	T	6.0000E-01	*H*	9.8167E-04	7.9936E-17	4.2093E-14	0	2.2371E-02	0
			B	87	100	100	100	100	100
f_5	*KM*	-4.8300E+01	*M*	-4.8606E+01	-4.9623E+01	-3.9379E+01	-3.4137E+01	-4.9602E+01	-4.9623E+01
	T	1.3500E+00	*H*	6.2906E-03	2.6654E-02	2.2653E-01	4.1487E-01	2.6244E-02	2.6663E-02
			B	77	100	0	0	100	100
f_6	*KM*	0	*M*	1.4317E-02	9.3362E-16	1.0707E-08	1.9308E-04	3.1337E-03	8.2873E-17
	T	6.0000E-01	*H*	1.4317E-02	9.3362E-16	1.0707E-08	1.9308E-04	3.1337E-03	8.2873E-17
			B	90	100	100	100	100	100
f_7	*KM*	0	*M*	5.8536E-08	8.4834E-16	9.7930E-32	0	1.6281E-02	5.1521E-21
	T	6.0000E-01	*H*	5.8536E-08	8.4834E-16	9.7930E-32	0	1.6281E-02	5.1521E-21
			B	90	100	100	100	100	100
f_8	*KM*	-2.0949E+04	*M*	-2.0160E+04	-2.0949E+04	-9.7364E+03	-2.0949E+04	-6.5528E+04	-2.0949E+04
	T	1.5000E+00	*H*	3.9148E-02	6.8911E-06	1.1516E+00	6.8911E-06	6.8030E-01	6.8911E-06
			B	0	100	0	86	0	100
f_9	*KM*	0	*M*	3.3808E-01	5.8371E-14	1.3061E-12	7.7804E-15	2.3689E-02	4.5866E-14
	T	6.0000E-01	*H*	3.3808E-01	5.8371E-14	1.3061E-12	7.7804E-15	2.3689E-02	4.5866E-14
			B	32	100	78	100	100	100
f_{10}	*KM*	0	*M*	3.1367E+01	2.4628E-01	3.1404E-02	9.3642E-01	9.3579E-01	7.9944E-02
	T	6.0000E-01	*H*	3.1367E+01	2.4628E-01	3.1404E-02	9.3642E-01	9.3579E-01	7.9944E-02
			B	0	95	100	18	0	100
Genel Ortalama Hata				**3.9368E+00**	**2.7294E-02**	**6.1405E+00**	**1.3515E-01**	**1.7172E-01**	**1.0661E-02**
Genel Ortalama Başarı(%)				**57.2**	**99.5**	**67.5**	**77.9**	**80.0**	**100.0**

Test fonksiyonlarının 2, 5, 10, 30 ve 50 boyutlu versiyonları için elde edilen test sonuçları Tablo 4.47'den Tablo 4.51'e kadar olan tablolarda verilmiştir. Her problem boyutunda her bir fonksiyon için problem boyutuna uygun tolerans değeri seçilmiş ve bu tolerans değerinden daha az bir hata ile küresel minimuma yaklaşan işletimler başarılı diğerleri başarısız sayılmıştır. Başarılı işletim yüzdeleri de tablolarda verilmiştir.

Öne çıkan sonuçlardan bazıları şunlardır:

- Özetle MBO algoritması ile ABC algoritmasının paralel çalışırken birbiri ile çözüm paylaşarak yardımlaşmaları fikrinden yola çıkılarak elde edilen MBABC-CO algoritmasında hem MBO algoritmasının hem de ABC algoritmasının performansının iyileştirilmesi sağlanmıştır.

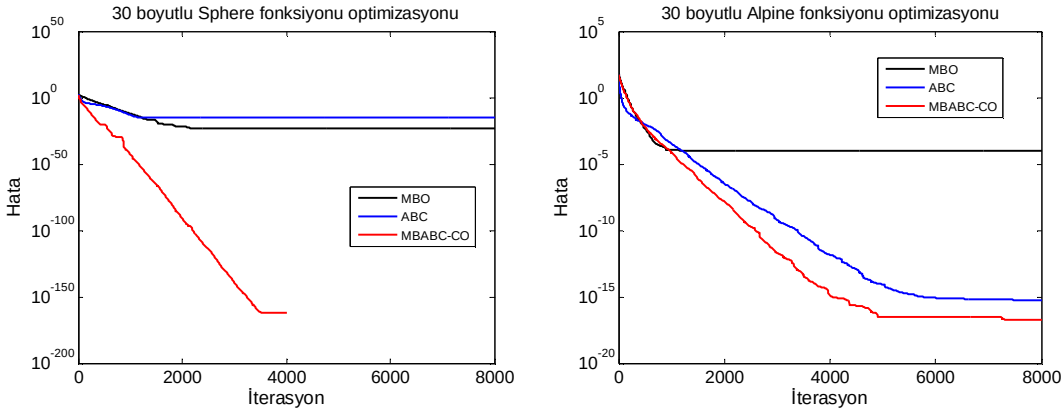
- Tablo 4.47'den Tablo 4.51'e kadar verilen sonuçlardan MBABC-CO algoritması MBO ve ABC algoritmaları ile kıyaslanıp genel bir değerlendirme yapıldığında, bütün problem boyutlarında küresel minimuma yakınsama açısından önemli bir gelişme gözlemlenmektedir. Her iki algoritma tarafında da paylaşılan en iyi çözümler algoritmaların yerel minimumlardan kurtulmak için fazla efor (iterasyon) harcamamalarını mevcut eforlarını küresel minimuma yakınsamada kullanmalarını sağlar. Böylece küresel minimuma daha fazla yakınsama sağlanır.



Şekil 4.40. MBABC-CO ve diğer algoritmaların başarı dağılımları

- Şekil 4.40'ta görüldüğü gibi, bütün problem boyutlarında MBABC-CO algoritmasının başarı yüzdesi diğer bütün algoritmalardan daha iyidir. Bu durum, iki algoritmanın kurulan sistematik ile icra ettiği işbirliği sayesinde yerel minimum tuzaklarından başarılı bir şekilde kurtulduğunu ve iyi bir keşif yeteneği kazandığını doğrulamaktadır.

- Algoritmanın küresel minimuma yakınsama davranışı incelendiğinde MBABC-CO algoritmasının tek ve çok modlu problemler üzerindeki davranışları Şekil 4.41'de olduğu gibi ayrı ayrı incelenebilir. ABC ve MBO algoritmaları tek modlu fonksiyonlar için makul hata seviyelerinde doyuma ulaşırken, MBABC-CO algoritması hata değerini düşürmeye devam ederek çok daha düşük hata değerlerinde doyuma ulaşır. Özellikle küçük boyutlu problemlerde küresel minimum değerine tam olarak erişilmiştir. Çok modlu fonksiyonlarda ise MBABC-CO algoritmasının küresel minimuma yakınsama miktarı hem MBO hem de ABC algoritmasından daha iyidir.



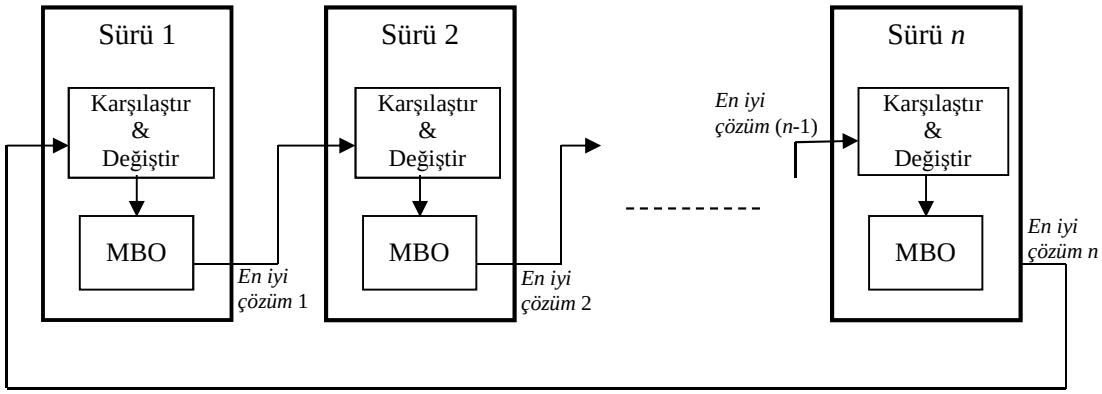
(a) (b)

Şekil 4.41. ABC, MBO ve MBMBABC-CO algoritmalarının (a) 30 boyutlu tek modlu Sphere fonksiyonunun ve (b) 30 boyutlu çok modlu Alpine fonksiyonunun küresel minimumuna yakınsamaları

- Paralel olarak çalışan MBO ve ABC algoritmalarının bir yardımlaşma örneği olan MBABC-CO elde edilirken algoritmanın hem keşif hem de yakınsama yetenekleri artmıştır. Algoritma bütün problem boyutlarında %100 başarı ile tamamladığı en iyileme işlemleri neticesinde küresel minimuma başarılı bir yakınsama gerçekleştirmiştir.

4.7.2.2. Çok sürülü göçmen kuşlar en iyileme algoritması

Çok sürülü göçmen göçmen kuşlar en iyileme (Multi-Flock Migrating Birds Optimization – MFMBO) algoritmasında birden fazla MBO algoritmasının paralel çalıştırılması ve Şekil 4.42'te verildiği gibi bir halka yapı içerisinde birbirleri ile çözüm paylaşımı suretiyle yardımlaşmaları hedeflenmiştir.



Şekil 4.42. MFMBO Algoritmasında çözüm paylaşım sistematiği

MFMBO algoritmasının çalışması şöyledir. Paralel çalışan MBO algoritmalarından her biri kendi popülasyonu içerisindeki en yüksek uygunluklu çözümü halka yapı içerisinde kendisinden sonra gelen MBO algoritması ile paylaşır. Her bir MBO algoritması, halka yapıda kendisinden önceki MBO algoritmasından aldığı çözümü kendi en iyi çözümü ile karşılaştırır. Eğer paylaşım yolu ile alınan çözüm kendi en iyi çözümünden daha iyi bir uygunluk değerine sahip ise alınan çözüm mevcut en iyi çözüm ile değiştirilir ve MBO algoritması bundan sonraki adım için bu popülasyon ile işletilir.

Bu paylaşım mekanizması paralel çalışan bütün MBO algoritmaları için işletilir. Böylece bir iterasyon tamamlanır. Bu yöntemin avantajı birden fazla paralel çalışan algoritma arasındaki yardımlaşma sayesinde bütün yapının yerel minimumlardan kolayca kurtulabilmesi ve başlangıç popülasyonlarından kaynaklanan olumsuzlukların bertaraf edilmesidir. Yöntemin dezavantajı ise birden fazla algoritmanın paralel çalışırken ekstra sistem kaynağına ihtiyaç duyması veya aynı sistem kaynağının zaman paylaşımlı kullanımının getireceği yüksek hesaplama

maliyetidir. Zaman paylaşımlı MFMBO algoritmasının çalışmasını özetleyen sözde kod Şekil 4.43'te verilmiştir.

for *i = 1* ***to*** *n*
    MBO(i) algoritması için i'inci başlangıç popülasyonunu Denklem (2.1) ile üret
    i'inci popülasyonun uygunluk değerlerini hesapla
    i'inci popülasyonun en iyi üyesi Xbest(i)'yi bul
    Bir sonraki popülasyonu belirle $p_{next} = mod(i,n) + 1$
    Xbest(i)'yi p_{next} *popülasyonu ile paylaş*
endfor
for *t = 1* ***to*** *K*
    for *i = 1* ***to*** *n*
        Paylaşılan çözüm, i'inci popülasyonun en iyi üyesinden daha iyi ise
            paylaşılan çözümü i'inci popülasyonun en iyi üyesi yap
        MBO(i) algoritmasını i'inci popülasyona uygula
        i'inci popülasyonun uygunluk değerlerini hesapla
        i'inci popülasyonun en iyi üyesi Xbest(i)'yi bul
        Bir sonraki popülasyonu belirle $p_{next} = mod(i,n) + 1$
        Xbest(i)'yi p_{next} *popülasyonu ile paylaş*
    endfor
    En iyi Xbest çözümleri içerisinde uygunluğu en iyi olanı en iyi çözüm olarak seç
endfor
return *Seçilen en iyi Xbest çözümü*

Şekil 4.43. Zaman paylaşımlı MFMBO algoritması sözde kodu

Test fonksiyonlarının 2, 5, 10, 30 ve 50 boyutlu versiyonları için elde edilen test sonuçları Tablo 4.52'den Tablo 4.56'ya kadar olan tablolarda verilmiştir. Her problem boyutunda her bir fonksiyon için problem boyutuna uygun tolerans değeri seçilmiş ve bu tolerans değerinden daha az bir hata ile küresel minimuma yaklaşan işletimler başarılı diğerleri başarısız sayılmıştır. Başarılı işletim yüzdeleri de tablolarda verilmiştir.

Tablo 4.52. MFMBO algoritmasının 2 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	**ABC**	**PSO**	**DE**	**GA**	**MFMBO**
f_1	*KM*	0	*M*	0	2.6992E-17	1.7508E-149	1.7525E-161	4.2287E-17	0
	T	5.0000E-03	*H*	0	2.6992E-17	1.7508E-149	1.7525E-161	4.2287E-17	0
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	0	1.2448E-07	0	0	1.4211E-16	0
	T	5.0000E-03	*H*	0	1.2448E-07	0	0	1.4211E-16	0
			B	95	90	80	96	97	100
f_3	*KM*	0	*M*	0	3.6367E-17	4.3904E-149	2.1397E-161	4.9386E-19	0
	T	5.0000E-03	*H*	0	3.6367E-17	4.3904E-149	2.1397E-161	4.9386E-19	0
			B	100	100	100	100	100	100
f_4	*KM*	0	*M*	7.4873E-04	1.2457E-02	1.9230E-03	0	6.7057E-03	0
	T	5.0000E-03	*H*	7.4873E-04	1.2457E-02	1.9230E-03	0	6.7057E-03	0
			B	45	8	37	68	10	84
f_5	*KM*	-1.8010E+00	*M*	-1.8013E+00	-1.8013E+00	-1.8013E+00	-1.8013E+00	-1.8013E+00	-1.8013E+00
	T	3.5000E-04	*H*	1.6844E-04	1.6844E-04	1.6844E-04	1.6844E-04	1.6844E-04	1.6844E-04
			B	100	100	93	100	99	100
f_6	*KM*	0	*M*	9.0640E-244	4.8596E-07	9.4008E-85	8.6804E-104	8.7397E-15	5.4372E-270
	T	5.0000E-03	*H*	9.0640E-244	4.8596E-07	9.4008E-85	8.6804E-104	8.7397E-15	5.4372E-270
			B	100	100	100	100	100	100
f_7	*KM*	0	*M*	0	3.5134E-17	0	0	3.8941E-20	0
	T	5.0000E-03	*H*	0	3.5134E-17	0	0	3.8941E-20	0
			B	100	100	100	100	97	100
f_8	*KM*	-8.3797E+02	*M*	-8.3797E+02	-8.3797E+02	-7.6690E+02	-8.3797E+02	-1.2035E+04	-8.3797E+02
	T	1.0000E-01	*H*	5.0425E-06	5.0435E-06	9.2668E-02	5.0425E-06	9.3038E-01	5.0425E-06
			B	91	87	20	88	22	100
f_9	*KM*	0	*M*	8.8818E-16	1.8829E-15	8.8818E-16	8.8818E-16	1.2773E-07	8.8818E-16
	T	5.0000E-03	*H*	8.8818E-16	1.8829E-15	8.8818E-16	8.8818E-16	1.2773E-07	8.8818E-16
			B	100	97	100	100	87	100
f_{10}	*KM*	0	*M*	6.1752E-130	2.2923E-03	3.5879E-74	7.7725E-71	7.4712E-04	6.4286E-142
	T	5.0000E-03	*H*	6.1752E-130	2.2923E-03	3.5879E-74	7.7725E-71	7.4712E-04	6.4286E-142
			B	100	40	100	100	56	100
Genel Ortalama Hata				**9.2221E-05**	**1.4923E-03**	**9.4759E-03**	**1.7348E-05**	**9.3800E-02**	**1.7348E-05**
Genel Ortalama Başarı(%)				**93.1**	**82.2**	**83.0**	**95.2**	**76.8**	**98.4**

Tablo 4.53. MFMBO algoritmasının 5 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	**ABC**	**PSO**	**DE**	**GA**	**MFMBO**
f_1	*KM*	0	*M*	3.1987E-222	6.4359E-17	4.8685E-246	7.6169E-171	2.5427E-07	1.4172E-243
	T	1.0000E-02	*H*	3.1987E-222	6.4359E-17	4.8685E-246	7.6169E-171	2.5427E-07	1.4172E-243
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	0	0	1.2139E+00	0	5.2194E-05	0
	T	1.0000E-02	*H*	0	0	1.2139E+00	0	5.2194E-05	0
			B	69	100	13	100	100	100
f_3	*KM*	0	*M*	5.0794E-222	6.6576E-17	8.8594E-246	1.9272E-170	9.4855E-07	2.2779E-242
	T	1.0000E-02	*H*	5.0794E-222	6.6576E-17	8.8594E-246	1.9272E-170	9.4855E-07	2.2779E-242
			B	100	100	100	100	100	100
f_4	*KM*	0	*M*	1.3097E-02	4.5754E-07	8.7264E-02	0	2.2388E-02	4.7820E-03
	T	1.0000E-02	*H*	1.3097E-02	4.5754E-07	8.7264E-02	0	2.2388E-02	4.7820E-03
			B	20	96	2	99	4	53
f_5	*KM*	-4.6870E+00	*M*	-4.6877E+00	-4.6877E+00	-4.5523E+00	-4.6877E+00	-4.6877E+00	-4.6877E+00
	T	1.6000E-03	*H*	1.4041E-04	1.4041E-04	2.9585E-02	1.4041E-04	1.3958E-04	1.4041E-04
			B	62	97	9	96	100	96
f_6	*KM*	0	*M*	5.7515E-148	1.8877E-16	7.1054E-17	4.0330E-135	8.8746E-05	1.5663E-200
	T	1.0000E-02	*H*	5.7515E-148	1.8877E-16	7.1054E-17	4.0330E-135	8.8746E-05	1.5663E-200
			B	99	100	100	100	100	100
f_7	*KM*	0	*M*	0	6.5295E-17	0	0	1.3775E-04	0
	T	1.0000E-02	*H*	0	6.5295E-17	0	0	1.3775E-04	0
			B	100	100	100	100	97	100
f_8	*KM*	-2.0949E+03	*M*	-2.0949E+03	-2.0949E+03	-1.5773E+03	-2.0949E+03	-1.5074E+04	-2.0949E+03
	T	2.5000E-01	*H*	6.8911E-06	6.8911E-06	3.2814E-01	6.8911E-06	8.6103E-01	6.8911E-06
			B	70	98	0	87	1	100
f_9	*KM*	0	*M*	3.5172E-15	4.3698E-15	4.1567E-15	8.8818E-16	7.8859E-03	1.9540E-15
	T	1.0000E-02	*H*	3.5172E-15	4.3698E-15	4.1567E-15	8.8818E-16	7.8859E-03	1.9540E-15
			B	98	100	100	100	31	100
f_{10}	*KM*	0	*M*	4.4960E-43	8.1871E-06	2.1329E-115	6.4222E-56	8.7774E-02	2.5428E-55
	T	1.0000E-02	*H*	4.4960E-43	8.1871E-06	2.1329E-115	6.4222E-56	8.7774E-02	2.5428E-55
			B	96	85	100	100	1	100
Genel Ortalama Hata				**1.3245E-03**	**1.5594E-05**	**1.6588E-01**	**1.4730E-05**	**9.7949E-02**	**4.9293E-04**
Genel Ortalama Başarı(%)				**81.4**	**97.6**	**62.4**	**98.2**	**63.4**	**94.9**

Tablo 4.54. MFMBO algoritmasının 10 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	**ABC**	**PSO**	**DE**	**GA**	**MFMBO**
f_1	*KM*	0	*M*	1.7398E-122	9.0681E-17	2.9633E-269	3.4394E-249	1.4506E-06	3.9063E-219
	T	1.1000E-02	*H*	1.7398E-122	9.0681E-17	2.9633E-269	3.4394E-249	1.4506E-06	3.9063E-219
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	3.8541E-01	0	5.2733E+00	0	2.4249E-04	0
	T	1.1000E-02	*H*	3.8541E-01	0	5.2733E+00	0	2.4249E-04	0
			B	30	100	0	96	100	98
f_3	*KM*	0	*M*	3.8799E-140	9.2792E-17	1.0598E-275	1.4532E-248	7.2226E-06	4.6488E-218
	T	1.1000E-02	*H*	3.8799E-140	9.2792E-17	1.0598E-275	1.4532E-248	7.2226E-06	4.6488E-218
			B	100	100	100	100	100	100
f_4	*KM*	0	*M*	3.1914E-02	2.9584E-04	3.8357E-01	0	3.7359E-02	8.2758E-03
	T	1.1000E-02	*H*	3.1914E-02	2.9584E-04	3.8357E-01	0	3.7359E-02	8.2758E-03
			B	5	76	0	98	2	31
f_5	*KM*	-9.6600E+00	*M*	-9.5849E+00	-9.6602E+00	-8.4746E+00	-9.6602E+00	-9.6601E+00	-9.6602E+00
	T	5.0000E-03	*H*	7.8346E-03	1.5705E-05	1.3987E-01	1.5705E-05	9.8244E-06	1.5705E-05
			B	17	100	0	75	100	85
f_6	*KM*	0	*M*	3.7558E-16	2.7008E-16	4.5853E-16	3.7114E-202	3.2727E-04	2.4893E-147
	T	1.1000E-02	*H*	3.7558E-16	2.7008E-16	4.5853E-16	3.7114E-202	3.2727E-04	2.4893E-147
			B	90	100	100	100	100	100
f_7	*KM*	0	*M*	0	9.3529E-17	0	0	5.5265E-04	0
	T	1.1000E-02	*H*	0	9.3529E-17	0	0	5.5265E-04	0
			B	99	100	100	100	99	100
f_8	*KM*	-4.1898E+03	*M*	-4.1211E+03	-4.1898E+03	-2.6082E+03	-4.1898E+03	-1.6321E+04	-4.1898E+03
	T	3.0000E-01	*H*	1.6662E-02	6.8911E-06	6.0642E-01	6.8911E-06	7.4329E-01	6.8911E-06
			B	21	100	0	92	0	92
f_9	*KM*	0	*M*	8.3489E-15	7.9936E-15	7.3541E-15	3.8014E-15	1.1570E-02	4.4409E-15
	T	1.1000E-02	*H*	8.3489E-15	7.9936E-15	7.3541E-15	3.8014E-15	1.1570E-02	4.4409E-15
			B	67	100	100	100	19	100
f_{10}	*KM*	0	*M*	1.9667E-02	5.4424E-11	2.2224E-67	4.5729E-08	1.4122E-01	1.3226E-35
	T	1.1000E-02	*H*	1.9667E-02	5.4424E-11	2.2224E-67	4.5729E-08	1.4122E-01	1.3226E-35
			B	41	100	100	88	0	100
Genel Ortalama Hata				**4.6149E-02**	**3.1844E-05**	**6.4032E-01**	**2.2642E-06**	**9.3458E-02**	**8.2984E-04**
Genel Ortalama Başarı(%)				**57.0**	**97.6**	**60.0**	**94.9**	**62.0**	**90.6**

Tablo 4.55. MFMBO algoritmasının 30 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	**ABC**	**PSO**	**DE**	**GA**	**MFMBO**
f_1	*KM*	0	*M*	1.3000E-23	4.9316E-16	2.6521E-79	2.9023E-263	8.7879E-06	6.8638E-139
	T	5.0000E-01	*H*	1.3000E-23	4.9316E-16	2.6521E-79	2.9023E-263	8.7879E-06	6.8638E-139
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	3.1244E+00	0	3.1202E+01	0	1.9920E-03	0
	T	5.0000E-01	*H*	3.1244E+00	0	3.1202E+01	0	1.9920E-03	0
			B	2	100	0	73	100	90
f_3	*KM*	0	*M*	3.7380E-19	4.8509E-16	2.3738E-78	7.2600E-262	1.3585E-04	7.5468E-138
	T	5.0000E-01	*H*	3.7380E-19	4.8509E-16	2.3738E-78	7.2600E-262	1.3585E-04	7.5468E-138
			B	98	100	100	100	100	100
f_4	*KM*	0	*M*	5.9400E-06	7.3275E-17	2.9088E-03	0	1.4791E-02	0
	T	5.0000E-01	*H*	5.9400E-06	7.3275E-17	2.9088E-03	0	1.4791E-02	0
			B	98	100	100	100	100	100
f_5	*KM*	-2.8980E+01	*M*	-2.9121E+01	-2.9631E+01	-2.3443E+01	-2.6569E+01	-2.9627E+01	-2.9625E+01
	T	7.0000E-01	*H*	4.8288E-03	2.1963E-02	2.3617E-01	9.0743E-02	2.1853E-02	2.1766E-02
			B	83	100	0	0	100	100
f_6	*KM*	0	*M*	1.0433E-04	5.5956E-16	5.3999E-11	6.1784E-156	1.2395E-03	1.0321E-15
	T	5.0000E-01	*H*	1.0433E-04	5.5956E-16	5.3999E-11	6.1784E-156	1.2395E-03	1.0321E-15
			B	97	100	100	100	100	100
f_7	*KM*	0	*M*	1.2803E-24	4.6608E-16	2.6501E-33	0	3.9747E-03	0
	T	5.0000E-01	*H*	1.2803E-24	4.6608E-16	2.6501E-33	0	3.9747E-03	0
			B	99	100	100	100	100	100
f_8	*KM*	-1.2569E+04	*M*	-1.2247E+04	-1.2569E+04	-6.3747E+03	-1.2569E+04	-4.2350E+04	-1.2569E+04
	T	1.0000E+00	*H*	2.6264E-02	3.8714E-05	9.7171E-01	3.8714E-05	7.0321E-01	3.8714E-05
			B	3	100	0	88	0	67
f_9	*KM*	0	*M*	5.6005E-02	3.1086E-14	3.8050E-14	4.4409E-15	1.4639E-02	1.6662E-14
	T	5.0000E-01	*H*	5.6005E-02	3.1086E-14	3.8050E-14	4.4409E-15	1.4639E-02	1.6662E-14
			B	47	100	88	100	100	100
f_{10}	*KM*	0	*M*	1.2274E+01	3.1423E-02	8.1776E-06	5.3910E-01	4.2169E-01	1.1303E-09
	T	5.0000E-01	*H*	1.2274E+01	3.1423E-02	8.1776E-06	5.3910E-01	4.2169E-01	1.1303E-09
			B	0	100	100	26	55	100
Genel Ortalama Hata				**1.5486E+00**	**5.3425E-03**	**3.2413E+00**	**6.2989E-02**	**1.1835E-01**	**2.1805E-03**
Genel Ortalama Başarı(%)				**62.7**	**100.0**	**68.8**	**78.7**	**85.5**	**95.7**

Tablo 4.56. MFMBO algoritmasının 50 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

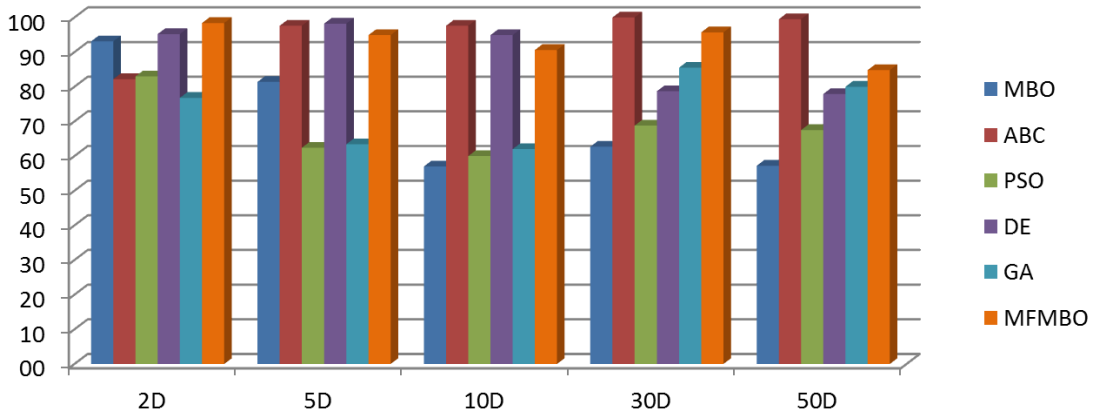
				MBO	**ABC**	**PSO**	**DE**	**GA**	**MFMBO**
f_1	*KM*	0	*M*	7.6778E-10	8.9649E-16	1.9490E-48	0	4.5167E-05	5.1347E-159
	T	6.0000E-01	*H*	7.6778E-10	8.9649E-16	1.9490E-48	0	4.5167E-05	5.1347E-159
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	7.6024E+00	0	5.9996E+01	0	8.3003E-03	0
	T	6.0000E-01	*H*	7.6024E+00	0	5.9996E+01	0	8.3003E-03	0
			B	0	100	0	75	100	81
f_3	*KM*	0	*M*	5.0462E-10	8.2798E-16	8.9623E-48	0	9.9204E-04	1.4092E-157
	T	6.0000E-01	*H*	5.0462E-10	8.2798E-16	8.9623E-48	0	9.9204E-04	1.4092E-157
			B	96	100	97	100	100	100
f_4	*KM*	0	*M*	9.8167E-04	7.9936E-17	4.2093E-14	0	2.2371E-02	0
	T	6.0000E-01	*H*	9.8167E-04	7.9936E-17	4.2093E-14	0	2.2371E-02	0
			B	87	100	100	100	100	100
f_5	*KM*	-4.8300E+01	*M*	-4.8606E+01	-4.9623E+01	-3.9379E+01	-3.4137E+01	-4.9602E+01	-4.9592E+01
	T	1.3500E+00	*H*	6.2906E-03	2.6654E-02	2.2653E-01	4.1487E-01	2.6244E-02	2.6048E-02
			B	77	100	0	0	100	100
f_6	*KM*	0	*M*	1.4317E-02	9.3362E-16	1.0707E-08	1.9308E-04	3.1337E-03	3.2574E-15
	T	6.0000E-01	*H*	1.4317E-02	9.3362E-16	1.0707E-08	1.9308E-04	3.1337E-03	3.2574E-15
			B	90	100	100	100	100	100
f_7	*KM*	0	*M*	5.8536E-08	8.4834E-16	9.7930E-32	0	1.6281E-02	0
	T	6.0000E-01	*H*	5.8536E-08	8.4834E-16	9.7930E-32	0	1.6281E-02	0
			B	90	100	100	100	100	100
f_8	*KM*	-2.0949E+04	*M*	-2.0160E+04	-2.0949E+04	-9.7364E+03	-2.0949E+04	-6.5528E+04	-2.0944E+04
	T	1.5000E+00	*H*	3.9148E-02	6.8911E-06	1.1516E+00	6.8911E-06	6.8030E-01	2.1930E-04
			B	0	100	0	86	0	48
f_9	*KM*	0	*M*	3.3808E-01	5.8371E-14	1.3061E-12	7.7804E-15	2.3689E-02	3.2863E-14
	T	6.0000E-01	*H*	3.3808E-01	5.8371E-14	1.3061E-12	7.7804E-15	2.3689E-02	3.2863E-14
			B	32	100	78	100	100	100
f_{10}	*KM*	0	*M*	3.1367E+01	2.4628E-01	3.1404E-02	9.3642E-01	9.3579E-01	7.6570E-01
	T	6.0000E-01	*H*	3.1367E+01	2.4628E-01	3.1404E-02	9.3642E-01	9.3579E-01	7.6570E-01
			B	0	95	100	18	0	19
Genel Ortalama Hata				**3.9368E+00**	**2.7294E-02**	**6.1405E+00**	**1.3515E-01**	**1.7172E-01**	**7.9197E-02**
Genel Ortalama Başarı(%)				**57.2**	**99.5**	**67.5**	**77.9**	**80.0**	**84.8**

Öne çıkan sonuçlardan bazıları şunlardır:

- Özet olarak birden fazla MBO algoritmasının paralel çalışırken birbiri ile çözüm paylaşarak yardımlaşmaları fikrinden yola çıkılarak elde edilen MFMBO algoritmasında MBO algoritmasının elde ettiği çözümlerin kalitesinin artırılması sağlanmıştır.

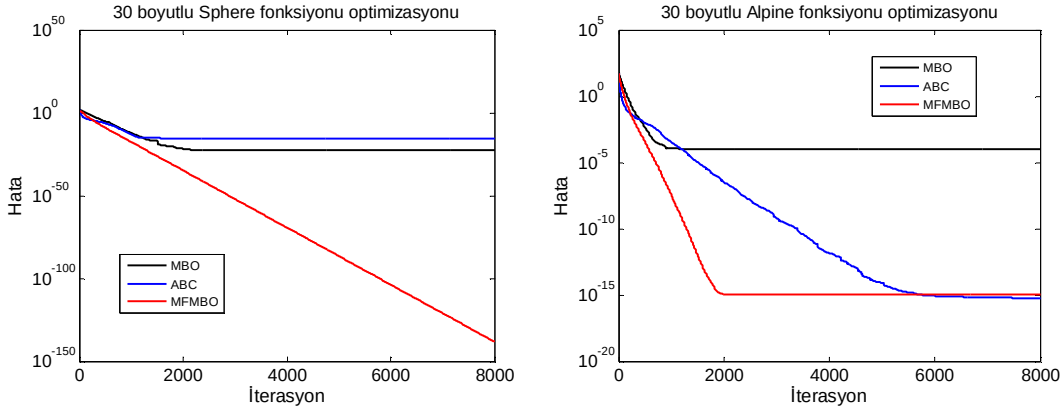
- Tablo 4.52'den Tablo 4.56'ya kadar verilen sonuçlardan MFMBO algoritması MBO algoritması ile kıyaslanıp genel bir değerlendirme yapıldığında bütün problem boyutlarında küresel minimuma yakınsama açısından önemli bir gelişme gözlemlenmektedir. Algoritmaların paylaştıkları en iyi çözümler algoritmaların yerel minimumlardan kurtulmak için fazla efor (iterasyon) harcamamalarını ve mevcut eforlarını küresel minimuma yakınsamada kullanmalarını sağlamıştır. Böylece küresel minimuma daha fazla yakınsama elde edilmiştir.

- Şekil 4.44'te görüldüğü gibi, bütün problem boyutlarında MFMBO algoritmasının başarı yüzdesi iyi seviyelerdedir. Özellikle problem boyutunun yüksek olduğu durumlarda ABC algoritması hariç diğer bütün algoritmalardan daha iyidir. Bu durum algoritmanın kurulan sistematik ile icra ettiği işbirliği sayesinde yerel minimum tuzaklarını başarılı bir şekilde bertaraf ettiğini ve iyi bir keşif yeteneği kazandığını doğrulamaktadır.



Şekil 4.44. MFMBO ve diğer algoritmaların başarı dağılımları

- Algoritmanın küresel minimuma yakınsama davranışı incelendiğinde MFMBO algoritmasının tek ve çok modlu problemler üzerindeki davranışları Şekil 4.45'te olduğu gibi ayrı ayrı incelenebilir. ABC ve MBO algoritmaları tek modlu fonksiyonlar için makul hata seviyelerinde doyuma ulaşırken, MFMBO algoritması hata değerini düşürmeye devam ederek çok daha düşük hata değerlerinde doyuma ulaşmaktadır. Çok modlu fonksiyonlarda ise MFMBO algoritmasının küresel minimuma yakınsama miktarı MBO algoritmasından daha iyi ve ABC algoritmasının yakınsama miktarına yakın seviyelerdedir.



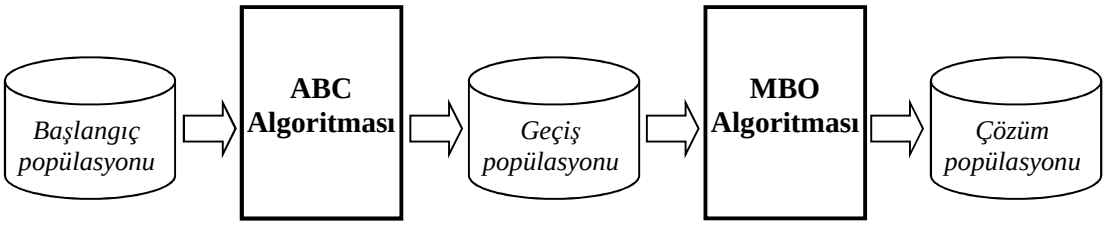
(a) (b)

Şekil 4.45. ABC, MBO ve MFMBO algoritmalarının (a) 30 boyutlu tek modlu Sphere fonksiyonunun ve (b) 30 boyutlu çok modlu Alpine fonksiyonunun küresel minimumuna yakınsamaları

- Üç MBO algoritmasının (kuş sürüsünün) kullanıldığı deneysel çalışmalarda paralel çalışmayı destekleyen sistem kaynağı temin edilemediğinde zaman paylaşımlı programlama tekniğinin kullanılması gerekecektir. Bu durumda da doğal olarak aynı iterasyon sayısı için toplam işlem süresi üç kat uzayacaktır. Fakat Şekil 4.45'teki hata trendleri incelendiğinde MFMBO algoritmasının ABC ve MBO algoritmalarının elde ettiği yakınsamaya eşdeğer bir yakınsamayı daha düşük iterasyonlarda elde ettiği görülmektedir. Bu yaklaşımla MFMBO algoritmasının hesaplama maliyetini artırmadan MBO algoritmasının başarı oranında bir artış sağladığı söylenebilir. Diğer taraftan yapay sinir ağı eğitimi gibi hesaplama süresinin fazla önemli olmadığı problem türleri için MFMBO algoritması rahatlıkla kullanılabilir.

4.7.2.3. Yapay arı koloni ve göçmen kuşlar en iyileme algoritmalarının sıralı işletimi

Şekil 4.46'da şematik olarak gösterilen, önce ABC algoritmasının daha sonra ise MBO algoritmalarının kullanımı esasına dayalı olan sıralı yapay arı koloni ve göçmen kuşlar en iyileme (Sequential Artificial Bee Colony and Migrating Birds Optimization – SABCMBO) algoritmasında ABC algoritmasının iyi keşif özelliği ile MBO algoritmasının iyi yakınsama özelliklerinin bir araya getirilmesi hedeflenmiştir.



Şekil 4.46. SABCMBO algoritmasının şematik gösterimi

ABC algoritması kâşif arı evresinde uyguladığı küresel arama stratejisi sayesinde yerel minimumları rahatlıkla aşıp popülasyonu küresel minimuma doğru yöneltmektedir. Fakat yakınsama yeteneğinin çok iyi olmaması nedeni ile ilerleyen iterasyonlarda karşılaşılan daha iyi çözüm üretememe nedeni ile kâşif arı evresinde popülasyon üyeleri yeniden çözüm uzayının rasgele pozisyonlarına yönlenmektedirler. Bu durum küresel minimuma arzulanan yakınsamayı engellemektedir. Dolayısıyla algoritmanın yakınsama yeteneğinin iyileştirilmeye ihtiyacı vardır.

MBO algoritması minimumlara yakınsamada oldukça başarılı, fakat çok boyutlu ve çok modlu problem uzaylarında eğer başlangıç popülasyonu iyi seçilmemiş ise yerel minimumlara takılabilmektedir. Dolayısıyla, eğer başlangıç popülasyonundaki uygunsuz dağılım önlenebilir ise algoritma küresel minimuma arzulanan ölçüde yakınsayabilmektedir.

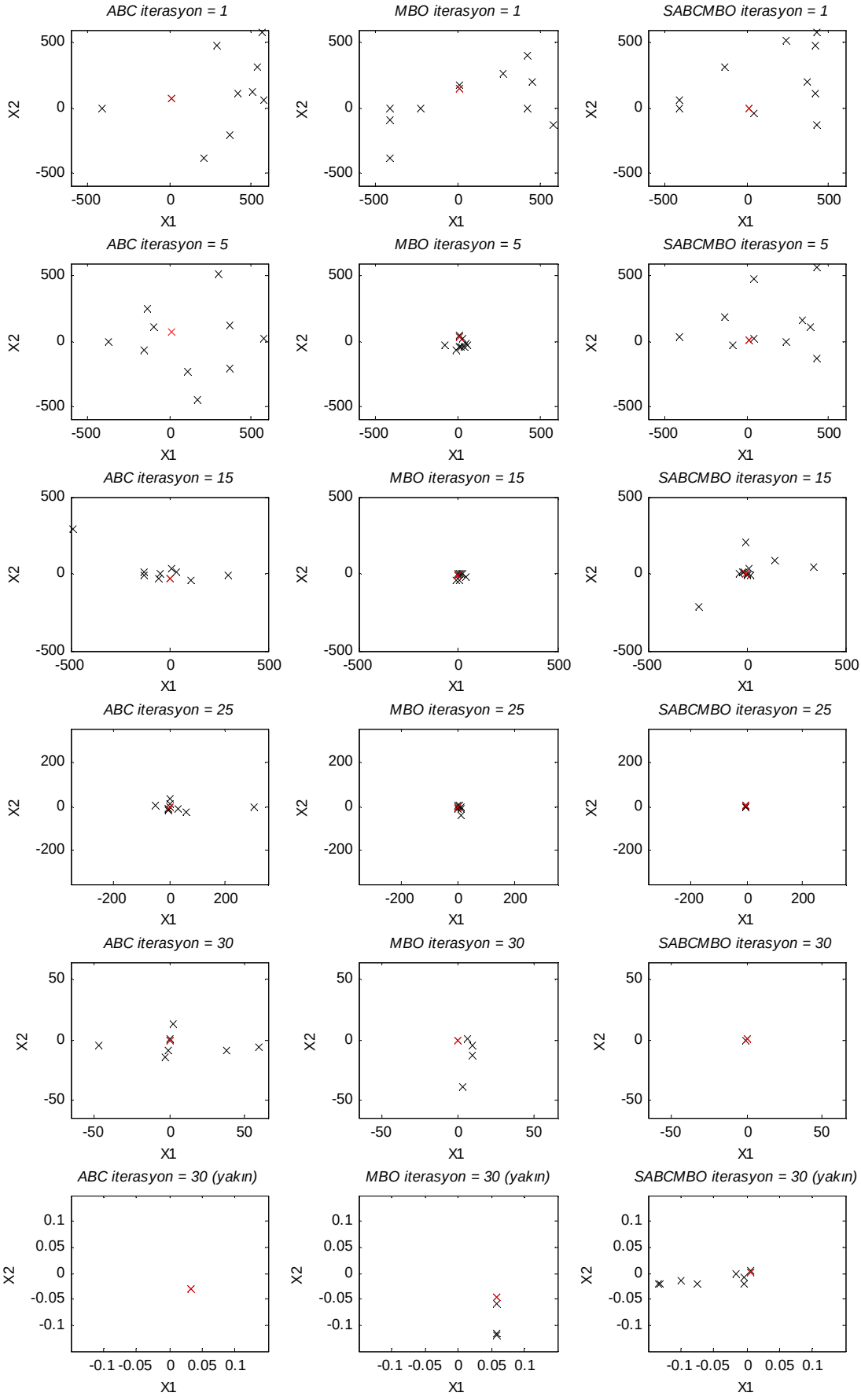
Bu iki gerçeğin göz önünde bulundurulduğu SABCMBO algoritmasında başlangıç popülasyonuna önce ABC algoritması uygulanarak popülasyon üyelerinin küresel

minimum çevresinde kümelenmesi sağlanmaktadır. Bu yolla elde edilen geçiş popülasyonuna daha sonra MBO algoritması uygulanarak küresel minimum çevresinde kümelenen popülasyon üyelerinin küresel minimuma mümkün olduğunca yakınsamaları sağlanmaktadır.

Bu gerçek, sonuçları Şekil 4.47'de verilen bir örnek üzerinde görsel olarak da kanıtlanmaktadır. Şekildeki grafikler ABC, MBO ve SABCMBO algoritmaları arasında gerçekleştirilen deneysel bir yarışmanın ilerleyişini özetlemektedir. Deneysel çalışmada MBO algoritmasının hesaplamalarında işlem yapılan komşuluk sayısı $k = 3$ ve paylaşılan komşu çözüm sayısı $x = 1$ olarak seçilmiştir. Bu parametre seçimleri ile her iterasyonda eşit uygunluk hesaplamalarının gerçekleştirilmesi ve böylece adil bir yarışma yapılması için MBO algoritmasının popülasyonunun yaklaşık olarak ABC algoritmasının popülasyonunun yarısı kadar olması gerekir. Dolayısıyla, bu basit deneyde ABC algoritması ve SABCMBO algoritmasının ABC kısmı için popülasyon büyüklüğü 20 üye olarak, MBO algoritması ve SABCMBO algoritmasının MBO kısmı için popülasyon büyüklüğü 11 üye olarak seçilmiştir.

Deneyde minimumu (0,0) noktasında olan iki boyutlu ve çoklu modlu Griewangk test fonksiyonunun en iyilenmesi üzerinde çalışılmıştır. Popülasyon üyelerinin 1'inci, 5'inci, 15'inci, 25'inci ve 30'uncu iterasyonların sonundaki pozisyonları Şekil 4.47'de koordinatlarına göre verilmiştir.

Griewangk test fonksiyonunun araştırma uzayı [-600,600] olarak verilmesine rağmen popülasyon üyelerinin ilerleyen iterasyonlarda çözüm kalitelerini iyileştirerek (0,0) noktasındaki küresel minimuma doğru yakınsadıkları görülmektedir. Grafiklerdeki kırmızı pozisyonlar algoritmaların o ana kadar ulaştıkları en iyi çözümü temsil etmektedirler. Her üç algoritmanın da popülasyon üyelerinin küresel minimuma doğru hareket ettirmedeki başarılı ilerleyişleri açıkça görülmektedir. Dolayısıyla iterasyonlar tamamlandıktan sonra elde edilen görüntüde kırmızı pozisyonun küresel minimuma yakınsaması ve küresel minimum çevresinde toplanan üye sayısı algoritmanın başarısını ifade eden değerlendirme unsurlarıdır.



Şekil 4.47. ABC, MBO ve SABCMBO algoritmalarının yakınsama davranışlarına bir örnek

SABCMBO algoritması daha detaylı analiz edildiğinde ilk 15 iterasyonda ABC algoritması aracılığı ile popülasyonun önce yerel minimumlardan kurtarılarak küresel minimum çevresinde toparlandığı, daha sonra da MBO algoritması aracılığı ile küresel minimuma başarılı bir yakınsama gerçekleştirildiği görülmektedir. Araştırma uzayının $[-0.15, 0.15]^2$ aralığı Şekil 4.47'de en alt sıradaki grafiklerde yakınlaştırılarak detayların daha net görülmesi sağlanmıştır. SABCMBO algoritması ile elde edilen en iyi çözümün pozisyonu küresel minimuma ABC ve MBO algoritmalarında elde edilenden daha yakındır. Ayrıca, 30 iterasyon sonrasında yakınlaştırılan $[-0.15, 0.15]^2$ aralığına ABC algoritması 1, MBO algoritması 4 ve SABCMBO algoritması 9 popülasyon üyesini yaklaştırmayı başarmıştır. Bu basit deneysel çalışmadan elde edilen sonuçlar algoritmanın ana motivasyonunu desteklemektedir. SABCMBO algoritmasının sözde kodu Şekil 4.48'de verilmiştir.

```
n elemanlı başlangıç popülasyonunu Denklem (2.1) aracılığıyla üret
Popülasyon üyelerinin uygunluk değerlerini hesapla
for t = 1 to round(K/2)
        ABC algoritmasını kullanarak popülasyonu iyileştirmeyi dene
        Yeni popülasyonun üyelerinin uygunluk değerlerini hesapla
endfor
Popülasyon içerisinden en iyi uygunluk değerine sahip (n/2 + 1) tane üyeyi seçerek
        geçiş popülasyonunu oluştur.
for t = round(K/2)+1 to K
        MBO algoritmasını kullanarak geçiş popülasyonunu iyileştirmeyi dene
        Yeni popülasyonun üyelerinin uygunluk değerlerini hesapla
endfor
return Mevcut en iyi çözüm
```

Şekil 4.48. SABCMBO algoritması sözde kodu

Test fonksiyonlarının 2, 5, 10, 30 ve 50 boyutlu versiyonları için elde edilen test sonuçları Tablo 4.57'den Tablo 4.61'e kadar olan tablolarda verilmiştir. Her problem boyutunda her bir fonksiyon için problem boyutuna uygun tolerans değeri seçilmiş ve bu tolerans değerinden daha az bir hata ile küresel minimuma yaklaşan işletimler başarılı diğerleri başarısız sayılmıştır. Başarılı işletim yüzdeleri de tablolarda verilmiştir.

Tablo 4.57. SABCMBO algoritmasının 2 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	**ABC**	**PSO**	**DE**	**GA**	**SABCMBO**
f_1	*KM*	0	*M*	0	2.6992E-17	1.7508E-149	1.7525E-161	4.2287E-17	1.9152E-183
	T	5.0000E-03	*H*	0	2.6992E-17	1.7508E-149	1.7525E-161	4.2287E-17	1.9152E-183
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	0	1.2448E-07	0	0	1.4211E-16	0
	T	5.0000E-03	*H*	0	1.2448E-07	0	0	1.4211E-16	0
			B	95	90	80	96	97	100
f_3	*KM*	0	*M*	0	3.6367E-17	4.3904E-149	2.1397E-161	4.9386E-19	1.8324E-183
	T	5.0000E-03	*H*	0	3.6367E-17	4.3904E-149	2.1397E-161	4.9386E-19	1.8324E-183
			B	100	100	100	100	100	100
f_4	*KM*	0	*M*	7.4873E-04	1.2457E-02	1.9230E-03	0	6.7057E-03	0
	T	5.0000E-03	*H*	7.4873E-04	1.2457E-02	1.9230E-03	0	6.7057E-03	0
			B	45	8	37	68	10	99
f_5	*KM*	-1.8010E+00	*M*	-1.8013E+00	-1.8013E+00	-1.8013E+00	-1.8013E+00	-1.8013E+00	-1.8013E+00
	T	3.5000E-04	*H*	1.6844E-04	1.6844E-04	1.6844E-04	1.6844E-04	1.6844E-04	1.6844E-04
			B	100	100	93	100	99	100
f_6	*KM*	0	*M*	9.0640E-244	4.8596E-07	9.4008E-85	8.6804E-104	8.7397E-15	8.3909E-147
	T	5.0000E-03	*H*	9.0640E-244	4.8596E-07	9.4008E-85	8.6804E-104	8.7397E-15	8.3909E-147
			B	100	100	100	100	100	100
f_7	*KM*	0	*M*	0	3.5134E-17	0	0	3.8941E-20	0
	T	5.0000E-03	*H*	0	3.5134E-17	0	0	3.8941E-20	0
			B	100	100	100	100	97	100
f_8	*KM*	-8.3797E+02	*M*	-8.3797E+02	-8.3797E+02	-7.6690E+02	-8.3797E+02	-1.2035E+04	-8.3797E+02
	T	1.0000E-01	*H*	5.0425E-06	5.0435E-06	9.2668E-02	5.0425E-06	9.3038E-01	5.0425E-06
			B	91	87	20	88	22	100
f_9	*KM*	0	*M*	8.8818E-16	1.8829E-15	8.8818E-16	8.8818E-16	1.2773E-07	8.8818E-16
	T	5.0000E-03	*H*	8.8818E-16	1.8829E-15	8.8818E-16	8.8818E-16	1.2773E-07	8.8818E-16
			B	100	97	100	100	87	100
f_{10}	*KM*	0	*M*	6.1752E-130	2.2923E-03	3.5879E-74	7.7725E-71	7.4712E-04	1.4780E-82
	T	5.0000E-03	*H*	6.1752E-130	2.2923E-03	3.5879E-74	7.7725E-71	7.4712E-04	1.4780E-82
			B	100	40	100	100	56	100
Genel Ortalama Hata				**9.2221E-05**	**1.4923E-03**	**9.4759E-03**	**1.7348E-05**	**9.3800E-02**	**1.7348E-05**
Genel Ortalama Başarı(%)				**93.1**	**82.2**	**83.0**	**95.2**	**76.8**	**99.9**

Tablo 4.58. SABCMBO algoritmasının 5 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	**ABC**	**PSO**	**DE**	**GA**	**SABCMBO**
f_1	*KM*	0	*M*	3.1987E-222	6.4359E-17	4.8685E-246	7.6169E-171	2.5427E-07	3.1710E-129
	T	1.0000E-02	*H*	3.1987E-222	6.4359E-17	4.8685E-246	7.6169E-171	2.5427E-07	3.1710E-129
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	0	0	1.2139E+00	0	5.2194E-05	0
	T	1.0000E-02	*H*	0	0	1.2139E+00	0	5.2194E-05	0
			B	69	100	13	100	100	100
f_3	*KM*	0	*M*	5.0794E-222	6.6576E-17	8.8594E-246	1.9272E-170	9.4855E-07	1.6118E-129
	T	1.0000E-02	*H*	5.0794E-222	6.6576E-17	8.8594E-246	1.9272E-170	9.4855E-07	1.6118E-129
			B	100	100	100	100	100	100
f_4	*KM*	0	*M*	1.3097E-02	4.5754E-07	8.7264E-02	0	2.2388E-02	0
	T	1.0000E-02	*H*	1.3097E-02	4.5754E-07	8.7264E-02	0	2.2388E-02	0
			B	20	96	2	99	4	99
f_5	*KM*	-4.6870E+00	*M*	-4.6877E+00	-4.6877E+00	-4.5523E+00	-4.6877E+00	-4.6877E+00	-4.6877E+00
	T	1.6000E-03	*H*	1.4041E-04	1.4041E-04	2.9585E-02	1.4041E-04	1.3958E-04	1.4041E-04
			B	62	97	9	96	100	100
f_6	*KM*	0	*M*	5.7515E-148	1.8877E-16	7.1054E-17	4.0330E-135	8.8746E-05	1.0724E-102
	T	1.0000E-02	*H*	5.7515E-148	1.8877E-16	7.1054E-17	4.0330E-135	8.8746E-05	1.0724E-102
			B	99	100	100	100	100	100
f_7	*KM*	0	*M*	0	6.5295E-17	0	0	1.3775E-04	0
	T	1.0000E-02	*H*	0	6.5295E-17	0	0	1.3775E-04	0
			B	100	100	100	100	97	100
f_8	*KM*	-2.0949E+03	*M*	-2.0949E+03	-2.0949E+03	-1.5773E+03	-2.0949E+03	-1.5074E+04	-2.0949E+03
	T	2.5000E-01	*H*	6.8911E-06	6.8911E-06	3.2814E-01	6.8911E-06	8.6103E-01	6.8911E-06
			B	70	98	0	87	1	100
f_9	*KM*	0	*M*	3.5172E-15	4.3698E-15	4.1567E-15	8.8818E-16	7.8859E-03	2.8066E-15
	T	1.0000E-02	*H*	3.5172E-15	4.3698E-15	4.1567E-15	8.8818E-16	7.8859E-03	2.8066E-15
			B	98	100	100	100	31	100
f_{10}	*KM*	0	*M*	4.4960E-43	8.1871E-06	2.1329E-115	6.4222E-56	8.7774E-02	2.8837E-34
	T	1.0000E-02	*H*	4.4960E-43	8.1871E-06	2.1329E-115	6.4222E-56	8.7774E-02	2.8837E-34
			B	96	85	100	100	1	100
Genel Ortalama Hata				**1.3245E-03**	**1.5594E-05**	**1.6588E-01**	**1.4730E-05**	**9.7949E-02**	1.4730E-05
Genel Ortalama Başarı(%)				**81.4**	**97.6**	**62.4**	**98.2**	**63.4**	99.9

Tablo 4.59. SABCMBO algoritmasının 10 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	**ABC**	**PSO**	**DE**	**GA**	**SABCMBO**
f_1	*KM*	0	*M*	1.7398E-122	9.0681E-17	2.9633E-269	3.4394E-249	1.4506E-06	3.5321E-112
	T	1.1000E-02	*H*	1.7398E-122	9.0681E-17	2.9633E-269	3.4394E-249	1.4506E-06	3.5321E-112
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	3.8541E-01	0	5.2733E+00	0	2.4249E-04	0
	T	1.1000E-02	*H*	3.8541E-01	0	5.2733E+00	0	2.4249E-04	0
			B	30	100	0	96	100	100
f_3	*KM*	0	*M*	3.8799E-140	9.2792E-17	1.0598E-275	1.4532E-248	7.2226E-06	3.3282E-111
	T	1.1000E-02	*H*	3.8799E-140	9.2792E-17	1.0598E-275	1.4532E-248	7.2226E-06	3.3282E-111
			B	100	100	100	100	100	100
f_4	*KM*	0	*M*	3.1914E-02	2.9584E-04	3.8357E-01	0	3.7359E-02	0
	T	1.1000E-02	*H*	3.1914E-02	2.9584E-04	3.8357E-01	0	3.7359E-02	0
			B	5	76	0	98	2	92
f_5	*KM*	-9.6600E+00	*M*	-9.5849E+00	-9.6602E+00	-8.4746E+00	-9.6602E+00	-9.6601E+00	-9.6602E+00
	T	5.0000E-03	*H*	7.8346E-03	1.5705E-05	1.3987E-01	1.5705E-05	9.8244E-06	1.5705E-05
			B	17	100	0	75	100	100
f_6	*KM*	0	*M*	3.7558E-16	2.7008E-16	4.5853E-16	3.7114E-202	3.2727E-04	2.5297E-19
	T	1.1000E-02	*H*	3.7558E-16	2.7008E-16	4.5853E-16	3.7114E-202	3.2727E-04	2.5297E-19
			B	90	100	100	100	100	100
f_7	*KM*	0	*M*	0	9.3529E-17	0	0	5.5265E-04	0
	T	1.1000E-02	*H*	0	9.3529E-17	0	0	5.5265E-04	0
			B	99	100	100	100	99	100
f_8	*KM*	-4.1898E+03	*M*	-4.1211E+03	-4.1898E+03	-2.6082E+03	-4.1898E+03	-1.6321E+04	-4.1898E+03
	T	3.0000E-01	*H*	1.6662E-02	6.8911E-06	6.0642E-01	6.8911E-06	7.4329E-01	6.8911E-06
			B	21	100	0	92	0	100
f_9	*KM*	0	*M*	8.3489E-15	7.9936E-15	7.3541E-15	3.8014E-15	1.1570E-02	4.4409E-15
	T	1.1000E-02	*H*	8.3489E-15	7.9936E-15	7.3541E-15	3.8014E-15	1.1570E-02	4.4409E-15
			B	67	100	100	100	19	100
f_{10}	*KM*	0	*M*	1.9667E-02	5.4424E-11	2.2224E-67	4.5729E-08	1.4122E-01	3.2672E-06
	T	1.1000E-02	*H*	1.9667E-02	5.4424E-11	2.2224E-67	4.5729E-08	1.4122E-01	3.2672E-06
			B	41	100	100	88	0	100
Genel Ortalama Hata				**4.6149E-02**	**3.1844E-05**	**6.4032E-01**	**2.2642E-06**	**9.3458E-02**	**2.5864E-06**
Genel Ortalama Başarı(%)				**57.0**	**97.6**	**60.0**	**94.9**	**62.0**	**99.2**

Tablo 4.60. SABCMBO algoritmasının 30 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

				MBO	**ABC**	**PSO**	**DE**	**GA**	**SABCMBO**
f_1	*KM*	0	*M*	1.3000E-23	4.9316E-16	2.6521E-79	2.9023E-263	8.7879E-06	1.5550E-27
	T	5.0000E-01	*H*	1.3000E-23	4.9316E-16	2.6521E-79	2.9023E-263	8.7879E-06	1.5550E-27
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	3.1244E+00	0	3.1202E+01	0	1.9920E-03	0
	T	5.0000E-01	*H*	3.1244E+00	0	3.1202E+01	0	1.9920E-03	0
			B	2	100	0	73	100	100
f_3	*KM*	0	*M*	3.7380E-19	4.8509E-16	2.3738E-78	7.2600E-262	1.3585E-04	3.2025E-32
	T	5.0000E-01	*H*	3.7380E-19	4.8509E-16	2.3738E-78	7.2600E-262	1.3585E-04	3.2025E-32
			B	98	100	100	100	100	100
f_4	*KM*	0	*M*	5.9400E-06	7.3275E-17	2.9088E-03	0	1.4791E-02	0
	T	5.0000E-01	*H*	5.9400E-06	7.3275E-17	2.9088E-03	0	1.4791E-02	0
			B	98	100	100	100	100	100
f_5	*KM*	-2.8980E+01	*M*	-2.9121E+01	-2.9631E+01	-2.3443E+01	-2.6569E+01	-2.9627E+01	-2.9631E+01
	T	7.0000E-01	*H*	4.8288E-03	2.1963E-02	2.3617E-01	9.0743E-02	2.1853E-02	2.1961E-02
			B	83	100	0	0	100	100
f_6	*KM*	0	*M*	1.0433E-04	5.5956E-16	5.3999E-11	6.1784E-156	1.2395E-03	1.7094E-16
	T	5.0000E-01	*H*	1.0433E-04	5.5956E-16	5.3999E-11	6.1784E-156	1.2395E-03	1.7094E-16
			B	97	100	100	100	100	100
f_7	*KM*	0	*M*	1.2803E-24	4.6608E-16	2.6501E-33	0	3.9747E-03	1.4143E-28
	T	5.0000E-01	*H*	1.2803E-24	4.6608E-16	2.6501E-33	0	3.9747E-03	1.4143E-28
			B	99	100	100	100	100	100
f_8	*KM*	-1.2569E+04	*M*	-1.2247E+04	-1.2569E+04	-6.3747E+03	-1.2569E+04	-4.2350E+04	-1.2569E+04
	T	1.0000E+00	*H*	2.6264E-02	3.8714E-05	9.7171E-01	3.8714E-05	7.0321E-01	3.8714E-05
			B	3	100	0	88	0	100
f_9	*KM*	0	*M*	5.6005E-02	3.1086E-14	3.8050E-14	4.4409E-15	1.4639E-02	2.1565E-14
	T	5.0000E-01	*H*	5.6005E-02	3.1086E-14	3.8050E-14	4.4409E-15	1.4639E-02	2.1565E-14
			B	47	100	88	100	100	100
f_{10}	*KM*	0	*M*	1.2274E+01	3.1423E-02	8.1776E-06	5.3910E-01	4.2169E-01	2.3964E-01
	T	5.0000E-01	*H*	1.2274E+01	3.1423E-02	8.1776E-06	5.3910E-01	4.2169E-01	2.3964E-01
			B	0	100	100	26	55	82
Genel Ortalama Hata				**1.5486E+00**	**5.3425E-03**	**3.2413E+00**	**6.2989E-02**	**1.1835E-01**	**2.6164E-02**
Genel Ortalama Başarı(%)				**62.7**	**100.0**	**68.8**	**78.7**	**85.5**	**98.2**

Tablo 4.61. SABCMBO algoritmasının 50 boyutlu test fonksiyonlarının en iyilenmesindeki performans sonuçları ve diğer algoritmalar ile performans mukayesesi (*KM*: küresel minimum, *T*: tolerans, *M*: ortalama minimum, *H*: ortalama hata, *B*: başarı yüzdesi)

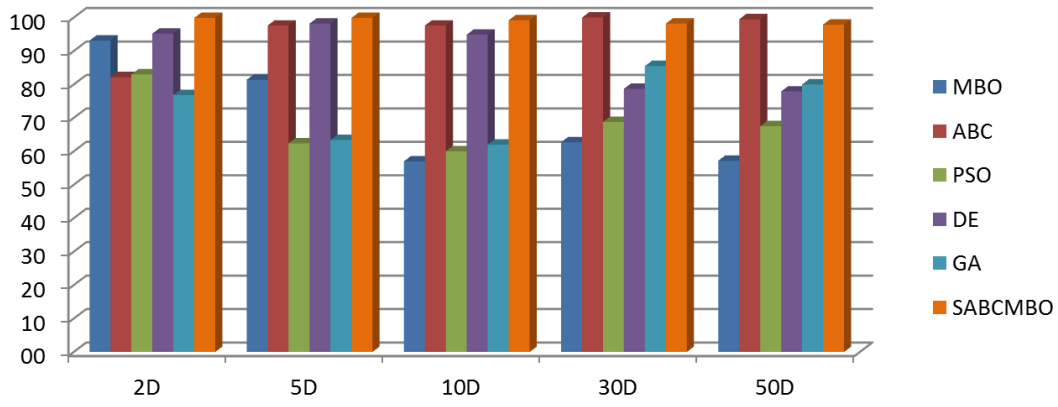
				MBO	**ABC**	**PSO**	**DE**	**GA**	**SABCMBO**
f_1	*KM*	0	*M*	7.6778E-10	8.9649E-16	1.9490E-48	0	4.5167E-05	6.5402E-20
	T	6.0000E-01	*H*	7.6778E-10	8.9649E-16	1.9490E-48	0	4.5167E-05	6.5402E-20
			B	100	100	100	100	100	100
f_2	*KM*	0	*M*	7.6024E+00	0	5.9996E+01	0	8.3003E-03	0
	T	6.0000E-01	*H*	7.6024E+00	0	5.9996E+01	0	8.3003E-03	0
			B	0	100	0	75	100	100
f_3	*KM*	0	*M*	5.0462E-10	8.2798E-16	8.9623E-48	0	9.9204E-04	1.7180E-20
	T	6.0000E-01	*H*	5.0462E-10	8.2798E-16	8.9623E-48	0	9.9204E-04	1.7180E-20
			B	96	100	97	100	100	100
f_4	*KM*	0	*M*	9.8167E-04	7.9936E-17	4.2093E-14	0	2.2371E-02	0
	T	6.0000E-01	*H*	9.8167E-04	7.9936E-17	4.2093E-14	0	2.2371E-02	0
			B	87	100	100	100	100	100
f_5	*KM*	-4.8300E+01	*M*	-4.8606E+01	-4.9623E+01	-3.9379E+01	-3.4137E+01	-4.9602E+01	-4.9625E+01
	T	1.3500E+00	*H*	6.2906E-03	2.6654E-02	2.2653E-01	4.1487E-01	2.6244E-02	2.6693E-02
			B	77	100	0	0	100	100
f_6	*KM*	0	*M*	1.4317E-02	9.3362E-16	1.0707E-08	1.9308E-04	3.1337E-03	2.8134E-16
	T	6.0000E-01	*H*	1.4317E-02	9.3362E-16	1.0707E-08	1.9308E-04	3.1337E-03	2.8134E-16
			B	90	100	100	100	100	100
f_7	*KM*	0	*M*	5.8536E-08	8.4834E-16	9.7930E-32	0	1.6281E-02	2.3448E-20
	T	6.0000E-01	*H*	5.8536E-08	8.4834E-16	9.7930E-32	0	1.6281E-02	2.3448E-20
			B	90	100	100	100	100	100
f_8	*KM*	-2.0949E+04	*M*	-2.0160E+04	-2.0949E+04	-9.7364E+03	-2.0949E+04	-6.5528E+04	-2.0949E+04
	T	1.5000E+00	*H*	3.9148E-02	6.8911E-06	1.1516E+00	6.8911E-06	6.8030E-01	6.8911E-06
			B	0	100	0	86	0	100
f_9	*KM*	0	*M*	3.3808E-01	5.8371E-14	1.3061E-12	7.7804E-15	2.3689E-02	4.1531E-14
	T	6.0000E-01	*H*	3.3808E-01	5.8371E-14	1.3061E-12	7.7804E-15	2.3689E-02	4.1531E-14
			B	32	100	78	100	100	100
f_{10}	*KM*	0	*M*	3.1367E+01	2.4628E-01	3.1404E-02	9.3642E-01	9.3579E-01	3.1306E-01
	T	6.0000E-01	*H*	3.1367E+01	2.4628E-01	3.1404E-02	9.3642E-01	9.3579E-01	3.1306E-01
			B	0	95	100	18	0	79
Genel Ortalama Hata				**3.9368E+00**	**2.7294E-02**	**6.1405E+00**	**1.3515E-01**	**1.7172E-01**	**3.3976E-02**
Genel Ortalama Başarı(%)				**57.2**	**99.5**	**67.5**	**77.9**	**80.0**	**97.9**

Öne çıkan sonuçlardan bazıları şunlardır:

- Özetle ABC ve MBO algoritmalarının birbiri ardından çalıştırılması ve böylece her iki algoritmanın da güçlü özelliklerini kullanmayı hedefleyen SABCMBO algoritmasında başarı oranı ve yakınsama değerleri oldukça iyi olan sonuçlar elde edilmiştir.

- Tablo 4.57'den Tablo 4.61'e kadar verilen sonuçlardan SABCMBO algoritmasının bütün problem boyutlarında küresel minimuma yakınsama başarısının iyi olduğu gözlemlenmektedir.

- Şekil 4.49'da görüldüğü gibi, bütün problem boyutlarında SABCMBO algoritmasının başarı yüzdesi çok iyi seviyelerdedir. Özellikle problem boyutunun yüksek olduğu durumlarda ABC algoritması hariç diğer bütün algoritmalardan daha iyidir. Bu durum algoritmanın kurulan sistematik ile icra ettiği işbirliği sayesinde yerel minimum tuzaklarını başarılı bir şekilde bertaraf ettiğini ve böylece iyi bir keşif yeteneği elde ettiğini doğrulamaktadır.

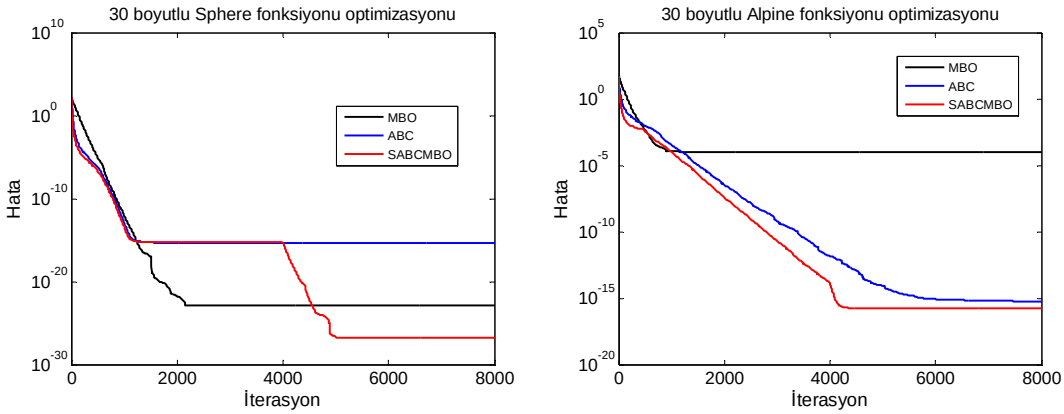


Şekil 4.49. SABCMBO ve diğer algoritmaların başarı dağılımları

- Algoritmanın küresel minimuma yakınsama davranışı incelendiğinde SABCMBO algoritmasının tek ve çok modlu problemler üzerindeki davranışları Şekil 4.50'de olduğu gibi ayrı ayrı incelenebilir. ABC ve MBO algoritmaları tek modlu fonksiyonlar için makul hata seviyelerinde doyuma ulaşırlarken, SABCMBO algoritması hata değerini düşürmeye devam ederek çok daha düşük

hata değerlerinde doyuma ulaşır. Çok modlu fonksiyonlarda ise SABCMBO algoritmasının küresel minimuma yakınsama miktarı MBO ve ABC algoritmalarının yakınsama miktarından daha iyi seviyelerdedir.

- Şekil 4.50 incelendiğinde iterasyonların ilk yarısında SABCMBO algoritmasının ABC algoritmasına paralel bir seyir içerisinde olduğu ve nihai yakınsama atağını iterasyonların ikinci yarısında ortaya koyduğu gözlemlenir. Algoritmanın bu davranış biçimi, popülasyonun ABC algoritması vasıtasıyla ilk yarı evrede küresel minimum yakınlarında toparlandığını ve MBO algoritması vasıtasıyla ikinci yarı evrede küresel minimuma mümkün olduğunca yakınsadığını kanıtlamaktadır.



(a) (b)

Şekil 4.50. ABC, MBO ve SABCMBO algoritmalarının (a) 30 boyutlu tek modlu Sphere fonksiyonunun ve (b) 30 boyutlu çok modlu Alpine fonksiyonunun küresel minimumuna yakınsamaları

- Ayrıca bu işlemleri gerçekleştirirken toplam iterasyon sayısı algoritmalar arasında bölüştürüldüğünden toplam işlem maliyeti de sabit kalmaktadır.

4.7.3. Algoritmaların İşlem Maliyetleri

Genel olarak üretilen her komşu çözümün uygunluk değeri hesaplanarak algoritmadaki değerlendirmeleri yapılır. Meta-sezgisel algoritmaların işleme süresinin belirlenmesinde toplam uygunluk hesaplama adedi önemli bir ölçüdür. Araştırma sistematiklerinin farklılığı nedeni ile bu hesaplama miktarı algoritmadan

algoritmaya değişebilmektedir. Tezin bu bölümündeki testlerde kullanılan algoritmalar için toplam uygunluk hesaplaması adetleri

$$tnfc_{ABC} = tnfc_{PSO} = tnfc_{DE} = tnfc_{GA} = PS \times K \quad \textbf{(4.41.a)}$$

$$tnfc_{MBO} = \left((k - x) \times (PS - 1) + k\right) \times K \quad \textbf{(4.41.b)}$$

$$tnfc_{VNMBO} = \left(\left(\frac{k_b + k_s}{2} - x\right) \times (PS - 1) + \frac{k_b + k_s}{2}\right) \times K \quad \textbf{(4.41.c)}$$

$$tnfc_{MBOGE} = \left((k - x) \times (PS - 1) + k + 1\right) \times K \quad \textbf{(4.41.d)}$$

$$tnfc_{MBABC-CO} = \left((k - x) \times (PS - 1) + k\right) \times K + 2 \times PS \times K \quad \textbf{(4.41.e)}$$

$$tnfc_{MFMBO} = N_F \times \left((k - x) \times (PS - 1) + k\right) \times K \quad \textbf{(4.41.f)}$$

$$tnfc_{SABCMBO} = \left((k - x) \times (PS - 1) + k\right) \times K / 2 + PS \times K \quad \textbf{(4.41.g)}$$

şeklinde hesaplanabilir. Burada *tnfc* toplam uygunluk hesaplama sayısı, *PS* popülasyon nüfusu, *K* toplam iterasyon sayısı, *k* çözüm kalitesini iyileştirmede denenecek toplam komşuluk sayısı, k_b başlangıç *k* değeri, k_s bitiş *k* değeri, *x* sonraki üyelerle paylaşılacak komşuluk sayısı ve N_F toplam sürü sayısıdır.

Yapılan testlerde hesaplama maliyetini düşürmek için mümkün olan en küçük $k = 3$ ve $x = 1$ değerleri seçilmiştir. Kullanılan *PS* ve *K* değerleri Tablo 4.62'de verilmiştir. VNMBO algoritması için k_b ve k_s sırasıyla 7 ve 3 olarak kullanılmıştır. MFMBO algoritmasının N_F değeri 3 alınmıştır. Her bir test düzeneği için hesaplanan *tnfc* değerleri Tablo 4.62'de verilmiştir. Testlerde kullanılan algoritma parametreleri seçilirken Tablo 4.62'de hesaplanan *tnfc* değerlerinin birbirine yakın olmasına dikkat edilmiştir. Bu yapılacak mukayeselerin adil olabilmesi için gerekli bir şarttır. MBABC-CO ve MFMBO algoritmalarında birden fazla paralel çalışan algoritma bulunması nedeniyle *tnfc* değeri algoritma sayısıyla doğru orantılı olarak artmaktadır.

Dolayısıyla, bu algoritmaların hedefindeki problemler, hesaplama maliyetinin önemsenmediği ve hatanın mümkün olan en küçük değere çekilmesinin amaçlandığı problemlerdir.

Tablo 4.62. Performans testlerindeki işlem maliyetleri (*PS*: popülasyon nüfusu, *K*: toplam iterasyon sayısı, *tnfc*: toplam uygunluk hesaplama sayısı)

		Problem Boyutu				
		2	**5**	**10**	**30**	**50**
MBO	*PS*	11	15	21	39	51
	K	1000	2000	4000	8000	15000
	tnfc	23000	62000	172000	632000	1545000
ABC	*PS*	22	30	40	80	100
	K	1000	2000	4000	8000	15000
	tnfc	22000	60000	160000	640000	1500000
PSO	*PS*	22	30	40	80	100
	K	1000	2000	4000	8000	15000
	tnfc	22000	60000	160000	640000	1500000
DE	*PS*	22	30	40	80	100
	K	1000	2000	4000	8000	15000
	tnfc	22000	60000	160000	640000	1500000
GA	*PS*	22	30	40	80	1000
	K	1000	2000	4000	8000	15000
	tnfc	22000	60000	160000	640000	15000000
VNMBO	*PS*	7	9	11	25	33
	K	1000	2000	4000	8000	15000
	tnfc	29000	74000	180000	808000	1995000
MBOGE	*PS*	11	15	21	39	51
	K	1000	2000	4000	8000	15000
	tnfc	24000	64000	176000	640000	1560000
MBABC-CO	*PS*	11	15	21	39	51
	K	1000	2000	4000	8000	15000
	tnfc	45000	122000	340000	1256000	3075000
MFMBO	*PS*	11	15	21	39	51
	K	1000	2000	4000	8000	15000
	tnfc	69000	186000	516000	1896000	4635000
SABCMBO	*PS*	11	15	21	39	51
	K	1000	2000	4000	8000	15000
	tnfc	22500	61000	170000	628000	1537500

BÖLÜM 5. SONUÇLAR VE ÖNERİLER

Bu tez çalışmasında meta-sezgisel algoritmalardan en iyileme problemlerinin çözümünde sıkça kullanılan GA, DE, PSO, ABC ve MBO algoritmasının tanıtımı yapılarak bir giriş yapılmış, çalışmalarda kullanılan test fonksiyonları ve yapılan testlerde izlenilen test prosedürü hakkında bilgi verilmiş ve performans testlerinden elde edilen sonuçlar mukayeseli olarak sunulmuştur.

Uygulamalar kısmında, tanıtımı yapılan meta-sezgisel en iyileme algoritmaları kullanılarak sistem kimliklendirme, YSA eğitimi, sempozyum oturum listeleri en iyilemesi, karar ağacı inşası ve bir sürekli döküm tesisinde kesme planı en iyilemesi gibi gerçek dünya problemlerine çözümler bulunmuştur.

Bilinmeyen bir sistemin davranışlarını inceleyebilmek ve anlayabilmek için önemli olan sistem kimliklendirme işlemi gerçek sistem ile onun matematiksel modeli arasında bir arayüz kurmayı amaçlar. Bu maksatla geliştirilen geleneksel yöntemler sistem girişlerini ve onlara ait çıkışları kullanırlar ve bilinmeyen sistem ile model çıkışındaki hata arzulanan değere minimize edilene kadar parametre ayarlaması yaparlar. Bu yöntemler genellikle eğime dayalı hesaplamaları kullanılırlar. Bu yüzden yumuşak geçişli ve türevi alınabilir hata fonksiyonlarına ihtiyaç duyarlar. Türevi alınamayan çok modlu hata fonksiyonlarında başarılı sistem kimliklendirmesi gerçekleştiremezler. Sistematik araştırma teknikleri olarak bilinen meta-sezgisel en iyileme algoritmaları ise bu tür çözümsüzlüklere en iyi çözüm üretebilen yöntemlerdir. Sistem kimliklendirme işlemine model çıkışı ile gerçek sistem çıkışı arasındaki farkın minimize edileceği bir en iyileme problemi olarak yaklaşıldığında meta-sezgisel algoritmalar aracılığı ile sistem kimliklendirme işleminin başarılı bir şekilde gerçekleştirildiği deneysel olarak görülmüştür.

YSA eğitim algoritmalarının temel felsefesi eğitim işlemi süresince ağ çıkışında oluşan MSE değerinin minimize edilmesidir. Literatürde bu iş için geliştirilen geleneksel eğitim algoritmaları sistem kimliklendirme sürecinde olduğu gibi türevi alınabilir hata fonksiyonlarına ihtiyaç duyarlar. Aksi durumlarda problemin yerel minimumlarına takılarak başarısız veya başarı seviyesi düşük eğitimler gerçekleştirirler. Yirmi farklı deney seti üzerinde değişik YSA topolojileri ile gerçekleştirilen deneysel çalışmalardan elde edilen sonuçlar meta-sezgisel algoritmaların geleneksel ve başarılı bir YSA eğitim algoritması olan LM'den daha iyi sonuçlar elde ettiğini göstermiştir. Özellikle MBO algoritmasının elde ettiği sonuçların diğer meta-sezgisel algoritmaları da geride bıraktığı deneysel olarak gözlemlenmiştir.

Kombinatoryal en iyileme algoritmaları endüstriyel uygulamalardan bilimsel uygulamalara varıncaya kadar pek çok alanda hızla gelişen ve giderek artan ilgi odağı haline gelen bir araştırma alanını oluşturmaktadır. Bu problemlerin çözümünde karşılaşılan en büyük sorun probleme özgü ve yüksek hesaplama maliyetine sahip olan metotların kullanılıyor olmasıdır. Bu metotların kullanımında karşılaşılan dezavantajlar nedeni ile meta-sezgiseller algoritmaların kombinatoryal en iyileme problemlerinin çözümündeki kullanımları son 20 yıl içerisinde giderek artmıştır. Tezde sempozyum oturum listelerinin optimize edildiği bir vaka çalışması için GA, ABC ve MBO algoritmalarının kombinatoryal versiyonları önerilmiştir. Önerilen yöntemler ile arzulanan sonuçların elde edildiği gözlemlenmiş ve ilgili vaka çalışmasındaki sempozyumun katılımcı listeleri verilen yöntemler kullanılarak dengeli dağılımlarla hazırlanmıştır.

Alt veri setlerinin her biri tek bir sınıfını içerene kadar eğitim setini art arda daha küçük parçalara ayıran kurallar seti olarak tanımlanan karar ağaçlarının tasarımında kullanılan değişik yöntemler vardır. Bu yöntemlerin veri setini alt veri setlerine ayırmasında kullanılan entropi, kazanç oranı, histogram, istatistiksel anlamlılık gibi değişik kriterler geliştirilmiştir. Bu tez çalışmasında karar ağacının kullandığı ayrım kuralları için bir hata hesaplama yöntemi önerilmiş, daha sonra karar ağacı tasarımına hem kombinatoryal hem de nümerik en iyileme işlemlerini bir arada içeren bir en iyileme problemi olarak yaklaşılmış ve bu problemin çözümüne yönelik

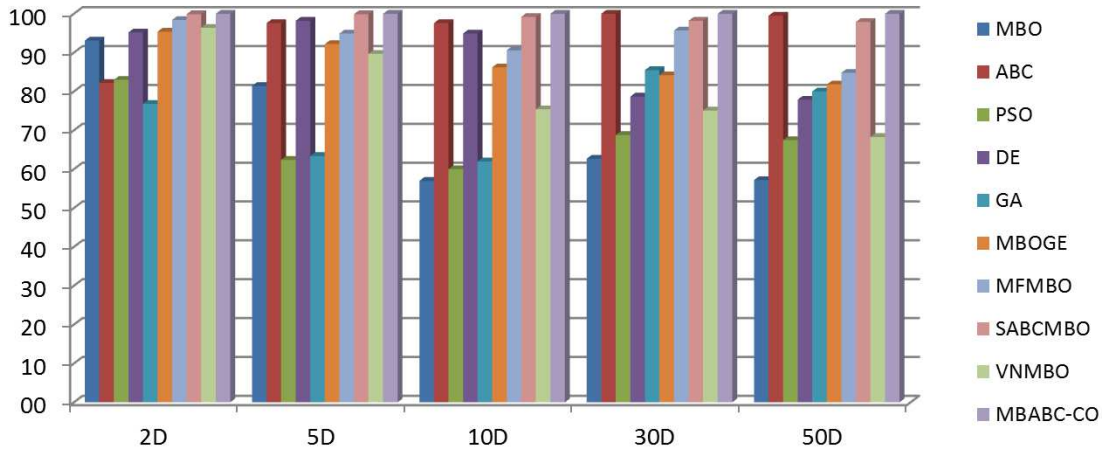
bir yöntem geliştirilmiştir. Bu yöntemde, ayrık ABC ve MBO algoritmaları her bir düğüm noktasında hangi özelliğin kullanılacağının belirlenmesi için önerilmiş, düğüm noktalarında belirlenen özelliklerin ayrım değerlerinin belirlenmesinde ise temel ABC ve MBO algoritmaları kullanılmıştır. UCI Machine Learning Repository web sitesinden alınan bir tiroit veri seti kullanılarak testler yapılmış, geliştirilen yöntemin %95.82 başarı oranı ile bu veri seti için geliştirilen literatürdeki diğer karar ağacı çalışmalarından daha iyi sonuçlar verdiği görülmüştür.

Demir çelik fabrikalarının sürekli döküm tesislerinde, kalıpta vuku bulan bazı kalite uygunsuzlukları ile planlı veya plansız olarak yapılan döküm sonu uygulamaları sonrasında üretim kayıplarını minimize etmek için hat üzerindeki malzemenin kesim pozisyonlarının yeniden planlanmasını gerektirir. Tezde, ABC algoritması kullanılarak slab kesim noktalarının yeniden planlanmasını ve bu sayede üretim kaybının minimize edilmesini sağlayan bir sistematik geliştirilmiştir. Majör hedef hurda kaybının minimize edilmesi olarak seçilmiş, bunun yanında hedef uzunluklardan sapmanın minimize edilmesi ve sipariş dışı üretilen slab adedinin minimize edilmesi de problemin minör hedefleri olarak seçilmiştir. Elde edilen sonuçlar, önerilen yöntemin sürekli döküm tesislerinde kesme planı en iyilemesi için başarı ile kullanılabileceğini doğrulamıştır.

Tezin son kısmında ise beş adet yeni yöntem önerilmiş ve bunların belirlenen test fonksiyonlarını optimize etmedeki başarıları test edilmiştir. Test sonuçları değerlendirilirken önerilen her bir yöntem ayrı ayrı diğer algoritmalarla mukayese edilmiş sonuçlar üzerinde yorumlar yapılmış ve değerlendirmeler sunulmuştur. Daha geniş bir perspektiften bir performans değerlendirmesi yapılması için önerilen yöntemlerin hepsinin bir arada bulunduğu grafikler aşağıda verilmiştir.

Şekil 5.1 algoritmaların testlerdeki başarılı en iyileme yüzdelerini toplu olarak göstermektedir. Önerilen algoritmaların başarı oranlarının MBO algoritmasından daha iyi seviyede olduğu grafikten görülmektedir. VNMBO algoritması haricindeki önerilen algoritmaların başarı oranları diğer popüler algoritmalarla rekabet edebilir düzeydedir. En iyi başarı oranlarının elde edildiği SABCMBO algoritması ABC algoritmasına yakın başarı performansı sergilerken diğer bütün algoritmaları geride

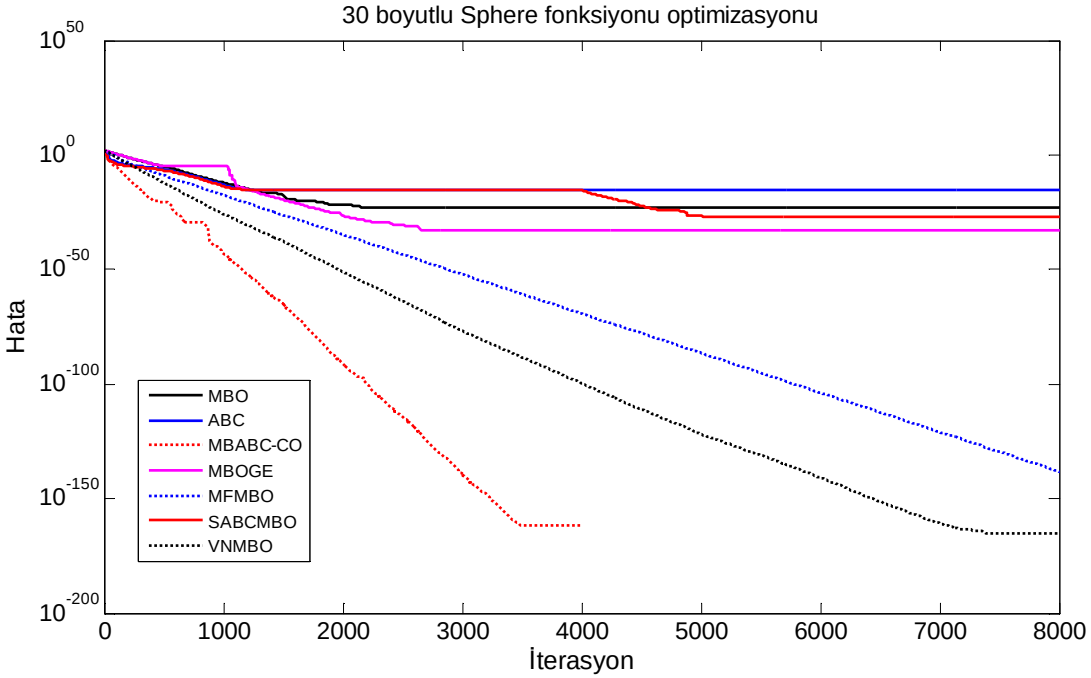
bırakmıştır. MBABC-CO algoritması ise bu bölümde test edilen algoritmalar içerisinde bütün testlerden %100 başarı oranı ile çıkan ve hiçbir işletimde yerel minimuma takılmayan tek algoritmadır.



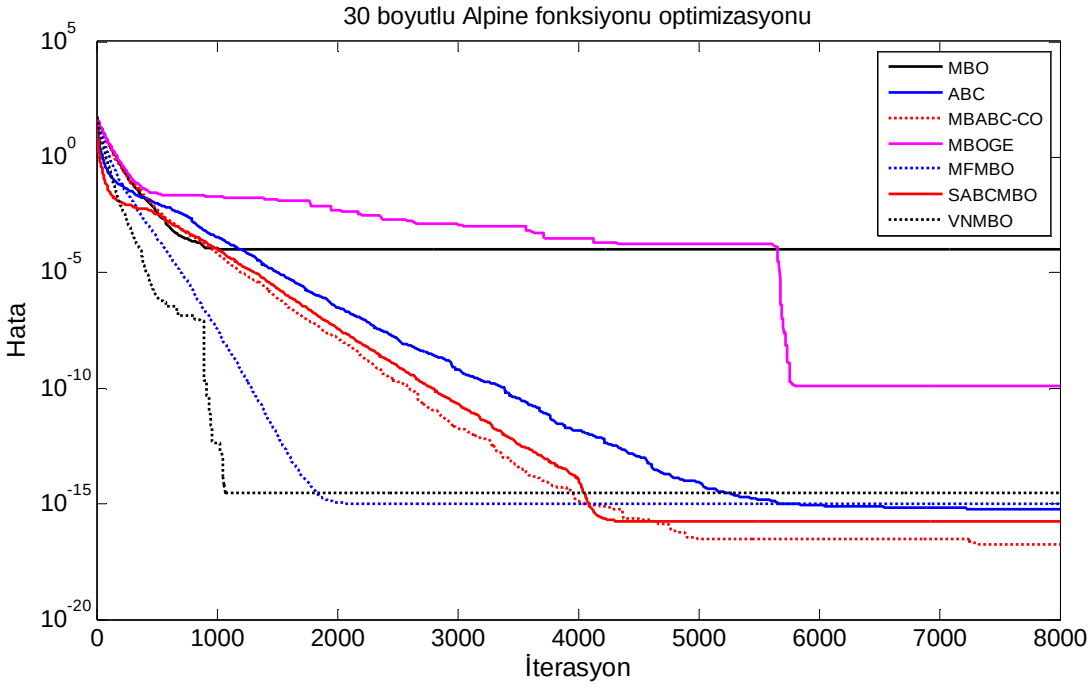
Şekil 5.1. Algoritmaların problem boyutlarına göre başarı dağılımları

Önerilen algoritmalar MBO algoritmasının performansını daha iyi seviyelere çekmek maksadıyla MBO algoritmasının modifikasyonu, kendisi veya ABC algoritması ile paralel kullanımı veya ABC algoritması ile birlikte sıralı kullanımı gibi çeşitli yöntemleri içermektedir. Bu yüzden MBO ve ABC algoritmaları ile önerilen algoritmaların hepsinin tek modlu ve çok modlu problemlerin çözümünde küresel minimuma yakınsama davranışları daha detaylı incelenmesi maksadıyla sırasıyla Şekil 5.2 ve Şekil 5.3'te tek bir grafik üzerinde verilmişlerdir.

Algoritmaların, tek modlu problemler içerisinden seçilen 30 boyutlu Sphere fonksiyonunun küresel minimumuna yakınsama davranışları Şekil 5.2'deki gibidir. Önerilen algoritmaların hepsi bu tek modlu test fonksiyonu testinde orijinal MBO ve ABC algoritmalarından daha fazla küresel minimuma yakınsamışlardır. SABCMBO algoritması daha az iterasyonda küresel minimumu yakalayarak en iyilemeyi tamamlamış, böylece en iyi yakınsama performansını sergilemiştir. VNMBO ve MFMBO algoritmaları da benzer yakınsamayı daha fazla iterasyon ile gerçekleştirmişlerdir. SABCMBO ve MBOGE algoritmaları ise en iyileme sürecini makul hatalarla tamamlamışlardır.



Şekil 5.2. Algoritmalarının 30 boyutlu tek modlu Sphere fonksiyonunun küresel minimumuna yakınsamaları



Şekil 5.3. Algoritmalarının 30 boyutlu çok modlu Alpine fonksiyonunun küresel minimumuna yakınsamaları

Çok modlu test fonksiyonları karakteristik olarak gerçek dünya problemlerine daha yakın olan davranışlara sahiptirler. Algoritmaların, çok modlu problemler içerisinden seçilen 30 boyutlu Alpine fonksiyonunun küresel minimumuna yakınsama

davranışları Şekil 5.3'te verilmiştir. Bu çok modlu problem testinin analizi hem önerilen algoritmalarla orijinal MBO algoritması üzerinde elde edilen iyileştirmeyi ortaya koyması hem de orijinal ABC algoritmasının güçlü keşif yeteneğinin bir kez daha gözlemlenmesi açısından oldukça önemlidir.

ABC algoritmasının güçlü keşif yeteneği onun yerel minimumlardan sıyrılarak rahatlıkla küresel minimuma doğru ilerlemesini sağlamaktadır. Orijinal MBO algoritması yerel minimum tuzaklarına takılabilirken önerilen algoritmaların elde ettikleri yüksek başarı oranı eriştikleri düşük hata değerleri onların bu tuzaklardan daha rahat kaçabildiklerini göstermektedir. Küresel minimuma en hızlı yakınsamayı MFMBO ve VNMBO algoritmaları elde ederken en mükemmel yakınsamayı SABCMBO ve MBABC-CO algoritmaları gerçekleştirmiştir. Bu algoritmalarda ABC algoritmasının güçlü keşif yeteneği ile MBO algoritmasının güçlü yakınsama yeteneği iyi bir entegrasyon ile birleştirilmiştir.

İleriye yönelik olarak:

- Bu çalışmada elde edilen melez algoritmaların paralel kullanımları ve bu yolla elde edilen yöntemlerin detaylı performans testlerinin yapılması,

- Mevcut algoritmalarda statik olarak kullanılan bazı algoritma parametrelerinin adaptif hale getirilmesi ve bu yolla elde edilen yöntemlerin detaylı performans testlerinin yapılması,

- Meta-sezgisellerle tasarlanan karar ağaçlarının budanarak sadeleştirilmesi ve böylece işlem maliyetinin azaltılması için bir sistematik geliştirilmesi,

- YSA topolojilerinin meta-sezgisel algoritmalar yardımı ile optimize edilmesi

gibi çalışmalar denenebilir.

KAYNAKLAR

[1] MURTY, KG., Optimization Models For Decision Making. Vol. 1, Internet Edition, 2003.

[2] AKAY, B., Performance analysis of artificial bee colony algorithm on numerical optimization problems. Ph.D. Thesis, Erciyes University, Turkey, 2009.

[3] CHANDLER, PR., PATCHER, M., Research Issues in Autonomous Control of Tactical UAVs. In Proc American Control Conference, Philadelphia, Pennsylvania, pp. 394-398, 1998

[4] DUMAN, E., UYSAL, M., ALKAYA, AF., Migrating Birds Optimization: A new metaheuristic approach and its performance on quadratic assignment problem. Information Sciences, 217:65–77, 2012.

[5] AHUJA, RK., ERGUN, O.¸ ORLIN, JB., PUNNEN, AP., A survey of very large scale neighborhood search techniques. Discrete Applied Mathematics, 123:75–102, 2002.

[6] BOUSSAÏD, I., LEPAGNOT, J., SIARRY, P., A survey on optimization metaheuristics. Information Sciences, 237:82–117, 2013.

[7] YAN, TS., TAO, YQ., CUI, DW., Research on Handwritten Numeral Recognition Method Based On Improved Genetic Algorithm and Neural Network. In Proc Int. Conf. on wavelet analysis and pattern recognition, Beijing, China, pp. 1271-1276, 2007.

[8] HOLLAND, JH., Adaptation in Natural and Artificial Systems. The MIT Press, Ann Arbor, MI, 1992.

[9] KIRKPATRICK, S., GELATT, CD., VECCHI, MP., Optimization by simulated annealing. Science, 220(4598):671–680, 1983.

[10] GLOVER, F., Future paths for integer programming and links to artificial intelligence. Computers & Operations Research, 13(5):533–549, 1986.

[11] GLOVER, F., Tabu search - part i. ORSA Journal on Computing, 1(3):190-206, 1989.

[12] GLOVER, F., Tabu search - part ii. ORSA Journal on Computing, 2(1):4-32, 1990.

[13] DORIGO, M., Optimization, learning and natural algorithms. Ph.D. thesis, Dipartimento di Elettronica, Politecnico di Milano, Italy, 1992.

[14] EBERHART, RC., KENNEDY, J., A new optimizer using particle swarm theory. In Proc 6th Int. Sym. on micromachine and human science. Nagoya, Japan, pp. 39–43, 1995.

[15] STORN, R., PRICE, K., Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization, 11:341–359, 1997.

[16] GEEM, ZW., KIM, JH., LOGANATHAN, GV., A new heuristic optimization algorithm: harmony search. Simulation, 76:60–68, 2001.

[17] MUCHERINO, A., SEREF, O., A novel meta-heuristic approach for global optimization. In Proc Conf. on data mining, system analysis and optimization in biomedicine, Gainesville, Florida, pp. 162–173, 2007.

[18] KARABOĞA, D., An idea based on honey bee swarm for numerical optimization. Technical Report TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[19] KARABOGA, D., BASTURK, B., A powerful and efficient algorithm for numerical function optimization: Artificial Bee Colony (ABC) Algorithm. Journal of Global Optimization, 39(3):459–471, 2007.

[20] YANG, XS., Firefly algorithm, In Nature-Inspired Metaheuristic Algorithms, Luniver Press, Frome, UK, pp. 79–90, 2008.

[21] SHAH-HOSSEINI, H., The intelligent water drops algorithm: a nature-inspired swarm-based optimization algorithm. International Journal of Bio-inspired Computation (IJBIC), 1:71–79, 2009.

[22] YANG, XS., DEB, S., Cuckoo search via Lévy flights. In Proc World Congress on Nature & Biologically Inspired Computing (NaBIC), Coimbatore, India, pp. 210-214, 2009.

[23] YANG, XS., DEB, S., Engineering optimisation by cuckoo search. Int. J. Math. Modelling & Num. Optimisation, 1:330-343, 2010.

[24] YANG, XS., A New Metaheuristic Bat-Inspired Algorithm. In JR. GONZALEZ et al. (Eds.), Nature Inspired Cooperative Strategies for Optimization (NICSO 2010), Vol. 284, pp. 65-74, 2010.

[25] YANG, XS., Bat algorithm for multi-objective optimization. Int. J. Bio-Inspired Computation, 3(5):267-274, 2011.

[26] LI, X., A new intelligent optimization—artificial fish swarm algorithm. Ph. D. Thesis, Zhejiang University of Zhejiang, China, 2003.

[27] DASGUPTA, D., Artificial Immune Systems and their Applications. Springer- Verlag, 1999.

[28] DASGUPTA, D., YU, S., NINO, F., Recent advances in artificial immune systems: models and applications. Applied Soft Computing, 11:1574–1587, 2011.

[29] PASSINO, K., Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Syst. Mag., 22(3):52–67, 2002.

[30] TANG, WJ., WU, QH., SAUNDERS, JR., Bacterial foraging algorithm for dynamic environments. In IEEE Congress on Evolutionary Computation, CEC, pp. 1324–1330, 2006.

[31] GREENSMITH, J., The dendritic cell algorithm. Ph.D. Thesis, University of Nottingham, 2007.

[32] GREENSMITH, J., AICKELIN, U., TWYCROSS, J., Articulation and clarification of the dendritic cell algorithm. In 5th Int. Conf. on Artificial Immune Systems (ICARIS2006), Lecture Notes in Computer Science, Vol. 4163, Springer-Verlag, pp.404–417, 2006.

[33] CHU, SC., TSAI, PW., PAN, JS., Cat Swarm Optimization. In Proc 9th Pacific Rim International Conference on Artificial Intelligence, Lecture Notes in Computer Science, Vol. 4099, Springer-Verlag, pp.854–858, 2006.

[34] CHU, SC., TSAI, PW., Computational intelligence based on the behavior of cats. Int. J. Innov. Comput. Inf. Control, 3:163–173, 2007.

[35] RASHEDI, E., Gravitational search algorithm. M.Sc. Thesis, Shahid Bahonar University of Kerman, Kerman, Iran, 2007.

[36] RASHEDI, E., NEZAMABADI-POUR, H., SARYAZDI, S., GSA: a gravitational search algorithm. Inf. Sci., 179:2232–2248, 2009.

[37] CERNY, V., Thermodynamical approach to the traveling salesman problem: an efficientsimulation algorithm. Journal of Optimization Theory and Applications, 45:41–51, 1985.

[38] LAARHOVEN, PV., AARTS, E., Simulated Annealing: Theory and Applications. 1st ed., D. Reidel Publishing Company, 1987.

[39] COLLINS, NE., EGLESE, RW., GOLDEN, B., Simulated annealing – an annotated bibliography. American Journal of Mathematical and Management Sciences, 8:209–307, 1988.

[40] KOULAMAS, C., ANTONY, SR., JAEN, R., A survey of simulated annealing applications to operations research problems. Omega, 22:41–56, 1994.

[41] FLEISCHER, M., Simulated annealing: past, present and future. In Proc 27th Conf. on Winter Simulation, Arlington, VA, USA, pp. 155–161, 1995.

[42] TAN, CM., Simulated Annealing. 1st ed., IN-TECH Education and Publishing, 2008.

[43] GLOVER, F., LAGUNA, M., Tabu Search. Kluwer Academic Publishers, 1997.

[44] GENDREAU, M., Chapter 2: An introduction to tabu search. In: GLOVER, F., KOCHENBERGER, GA. (Eds.), Handbook of Metaheuristics, Kluwer Academic Publishers, pp. 37–54, 2003.

[45] GENDREAU, M., POTVIN, JY., Chapter 6: Tabu search. In BURKE, EK., KENDALL . G (Eds.), Search Methodologies, Springer, pp. 165–186, 2006.

[46] GLOVER, F., Tabu search for nonlinear and parametric optimization (with links to genetic algorithms). Discrete Applied Mathematics, 49:231–255, 1994.

[47] MLADENOVIC, N., A variable neighborhood algorithm – a new metaheuristic for combinatorial optimization. In Abstracts of Papers Presented at Optimization Days, Montréal, Canada, p. 112, 1995.

[48] MLADENOVIC, N., DRAZIC, M., KOVACEVIC-VUJCIC, V., CANGALOVIC, M., General variable neighborhood search for the continuous optimization. European Journal of Operational Research, 191:753–770, 2008.

[49] MLADENOVIC, N., HANSEN, P., Variable neighborhood search. Computers and Operations Research, 24:1097–1100, 1997.

[50] HANSEN, P., MLADENOVIC, N., PÉREZ, JAM., Variable neighbourhood search: methods and applications. 4OR, 6(4):319–360, 2008.

[51] HANSEN, P., MLADENOVIC, N., PÉREZ, JAM., Variable neighbourhood search: algorithms and applications. Annals of Operations Research, 175:367–407, 2010.

[52] VOUDOURIS, C., Guided Local Search for Combinatorial Optimization Problems. Ph. D Thesis, University of Essex, 1997.

[53] VOUDOURIS, C., Guided local search: an illustrative example in function optimization. BT Technology Journal, 16:46–50, 1998.

[54] VOUDOURIS, C., TSANG, E., Guided local search. European Journal of Operational Research, 113:469–499, 1999.

[55] VOUDOURIS, C., TSANG, EPK., ALSHEDDY, A., Effective application of guided local search. In COCHRAN et al. (Eds.), Wiley Encyclopedia of Operations Research and Management Science, John Wiley & Sons, 2010.

[56] VOUDOURIS, C., TSANG, EPK., ALSHEDDY, A., Guided local search. In COCHRAN et al. (Eds.),Wiley Encyclopedia of Operations Research and Management Science, John Wiley & Sons, 2010.

[57] STÜTZLE, T., Local Search Algorithms for Combinatorial Problems: Analysis, Improvements, and New Applications. Ph.D. Thesis, Darmstadt University of Technology, 1998.

[58] LOURENÇO, HR., MARTIN, O., STÜTZLE, T., Chapter 12: iterated local search: framework and applications. In GENDREAU, M., POTVIN Y. (Eds.), Handbook of Metaheuristics, 2nd ed., Springer, pp. 363–397, 2010.

[59] FEO, TA., RESENDE, MGC., A probabilistic heuristic for a computationally difficult set covering problem. Operations Research Letters, 8:67–71, 1989.

[60] FEO, TA., RESENDE, MGC., Greedy randomized adaptive search procedures. Journal of Global Optimization, 6:109–133, 1995.

[61] RESENDE, MGC., RIBEIRO, CC., GRASP: Greedy Randomized Adaptive Search Procedures. Technical Report SGRASP2010, AT & T Labs Research, 2010.

[62] BEASLEY, D., BULL, D., MARTIN, RR., An overview of genetic algorithms. Part i: fundamentals. University Computing, 15:58–69, 1993.

[63] BEASLEY, D., BULL, D., MARTIN, RR., An overview of genetic algorithms. Part ii: research topics. University Computing, 15:170–181, 1993.

[64] ALBA, E., TROYA, JM., A survey of parallel distributed genetic algorithms. Complexity, 4:31–52, 1999.

[65] SINHA, A., GLODBERG, D., A Survey of Hybrid Genetic and Evolutionary algorithms. Technical Report 2003004, Illinois Genetic Algorithms Laboratory (IlliGAL), 2003.

[66] KONAK, A., COIT, D., SMITH, A., Multi-objective optimization using genetic algorithms: a tutorial. Reliability Engineering & System Safety in Special Issue – Genetic Algorithms and Reliability, 92:992–1007, 2006.

[67] KOZA, JR., Genetic Programming: On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems). 1st ed., The MIT Press, 1992.

[68] POLI, R., LANGDON, WB., MCPHEE, NF., A Field Guide to Genetic Programming. Lulu Enterprises, UK Ltd., 2008.

[69] WILLIAM, BL., RICCARDO, P., NICHOLAS, FM., JOHN, RK., Genetic programming: an introduction and tutorial with a survey of techniques and applications. In FULCHER, J. and JAIN LC. (Eds.), Computational Intelligence: A Compendium, Studies in Computational Intelligence (SCI), Vol. 115, Springer-Verlag, pp. 927–1028, 2008.

[70] MCKAY, RI., HOAI, NX., WHIGHAM, PA., SHAN, Y., O'NEILL, M., Grammar-based genetic programming: a survey. Genetic Programming and Evolvable Machines, 11:365–396, 2010.

[71] RECHENBERG, I., Cybernetic Solution Path of an Experimental Problem. Technical Report, Royal Air Force Establishment, 1965.

[72] RECHENBERG, I., Evolutionsstrategie: Optimierung Technischer Systeme Nach Prinzipien der Biologischen Evolution. Frommann-Holzboog, Stuttgart, 1973.

[73] BÄCK, T., HOFFMEISTER, F., HP. Schwefel, A survey of evolution strategies. In Proc 4th Int. Conf. on Genetic Algorithms, pp. 2–9, 1991.

[74] BÄCK, T., SCHWEFEL, HP., An overview of evolutionary algorithms for parameter optimization, Evolutionary Computation, 1:1–23, 1993.

[75] BEYER, HG., SCHWEFEL, HP., Evolution strategies – a comprehensive introduction. Natural Computing, 1:3–52, 2002.

[76] KRAMER, O., A review of constraint-handling techniques for evolution strategies. Applied Computational Intelligence and Soft Computing, 2010:1–19, 2010.

[77] FOGEL, LJ., OWENS, AJ., WALSH, MJ., Artificial Intelligence through Simulated Evolution. John Wiley, New York, USA, 1966.

[78] FOGEL, DB., System Identification through Simulated Evolution: A Machine Learning Approach to Modeling. Ginn Press, 1991.

[79] FOGEL, DB., Evolutionary Computation: Toward a New Philosophy of Machine Intelligence. IEEE Press, Piscataway, NJ, USA, 1995.

[80] LIU, J., LAMPINEN, J., A fuzzy adaptive differential evolution algorithm. Soft Computing, 9:448–462, 2005.

[81] CHAKRABORTY, U., Advances in Differential Evolution. 1st ed., Springer Publishing Company, 2008.

[82] NERI, F., TIRRONEN, V., Recent advances in differential evolution: a survey and experimental analysis. Artificial Intelligence Review, 33:61–106, 2010.

[83] DAS, S., SUGANTHAN, PN., Differential evolution: a survey of the state-of-the-art. IEEE Transactions on Evolutionary Computation, 15:4–31, 2011.

[84] HILLIS, WD., Co-evolving parasites improve simulated evolution as an optimization procedure. Physica D, 42:228–234, 1990.

[85] POTTER, MA., JONG, KAD., A cooperative coevolutionary approach to function optimization. In Proc Int. Conf. on Evolutionary Computation, The Third Conference on Parallel Problem Solving from Nature: Parallel Problem Solving from Nature, PPSN III, Springer- Verlag, London, UK, pp. 249–257, 1994.

[86] REYNOLDS, RG., An adaptive computer model of plan collection and early agriculture in the eastern valley of oaxaca. In Guila Naquitz (ed) Archaic Foraging and Early Agriculture in Oaxaca, Mexico, pp. 439–500, 1986.

[87] REYNOLDS, RG., An introduction to cultural algorithms. In SEBALK, AV., FOGEL LJ. (Eds.), Proc 3rd Annual Conf. on Evolutionary Programming, World Scientific Publishing, River Edge, NJ, pp. 131–139, 1994.

[88] RYCHTYCKYJ, N., REYNOLDS, RG., Using cultural algorithms to re-engineer large-scale semantic networks. International Journal of Software Engineering and Knowledge Engineering, 15:665–694, 2005.

[89] RIVERA, DC., BECERRA, RL., COELLO COELLO, CA., Cultural algorithms, an alternative heuristic to solve the job shop scheduling problem. Engineering Optimization, 39:69–85, 2007.

[90] REYNOLDS, RG., PENG, B., ALI, MZ., The role of culture in the emergence of decision-making roles: an example using cultural algorithms. Complexity, 13:27–42, 2008.

[91] OCHOA, A., PONCE, J., HERNÁNDEZ, A., LI, L., Resolution of a combinatorial problem using cultural algorithms. JCP, 4:738–741, 2009.

[92] REYNOLDS, RG., LIU, D., Multi-objective cultural algorithms. In IEEE Congress on Evolutionary Computation, pp. 1233–1241, 2011.

[93] AKYOL, S., ALATAŞ, B., Güncel Sürü Zekâsı Optimizasyon Algoritmaları. Nevşehir Üniversitesi FBE Dergisi, 1:36-50, 2012.

[94] ENGELBRECHT, A., Fundamentals of Computational Swarm Intelligence, Wiley, 2006.

[95] CHAN, FTS., TIWARI, M., Swarm Intelligence, Focus on Ant and Particle Swarm Optimization. I-Tech Education and Publishing, Vienna, Austria, 2007.

[96] KENNEDY, J., EBERHART, R., Particle swarm optimization. In IEEE Int. Conf. on Neural Networks, 4:1942–1948, 1995.

[97] KENNEDY, J., EBERHART, R., SHI, Y., Swarm Intelligence. Morgan Kaufman, San Francisco, 2001.

[98] CLERC, M., Particle Swarm Optimization. ISTE, London, UK, 2006.

[99] BANKS, A., VINCENT, J., ANYAKOHA, C., A review of particle swarm optimization - Part ii: hybridisation, combinatorial, multicriteria and constrained optimization, and indicative applications. Natural Computing, 7:109–124, 2008.

[100] THANGARAJ, R., PANT, M., ABRAHAM, A., BOUVRY, P., Particle swarm optimization: hybridization perspectives and experimental illustrations. Applied Mathematics and Computation, 217:5208–5226, 2011.

[101] POLI, R., KENNEDY, J., BLACKWELL, T., Particle swarm optimization - An overview. Swarm Intelligence, 1:33–57, 2007.

[102] CASTILLO, O., MELIN, P., Optimization of type-2 fuzzy systems based on bio-inspired methods: a concise review. Information Sciences, 205:1–19, 2012.

[103] DORIGO, M., MANIEZZO, V., COLORNI, A., Positive Feedback as a Search Strategy. Technical Report 91-016, Dipartimento di Elettronica, Politecnico di Milano, Milan, Italy, 1991.

[104] DORIGO, M., MANIEZZO, V., COLORNI, A., The ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics – Part B, 26:29–41, 1996.

[105] DORIGO, M., STÜTZLE, T., The ant colony optimization metaheuristic: algorithms, applications, and advances. In GLOVER, F., KOCHENBERGER G. (Eds.), Handbook of Metaheuristics, International Series in Operations Research & Amp; Management Science, Vol. 57, Springer, New York, pp. 250–285, 2003.

[106] BLUM, C., Ant colony optimization: Introduction and recent trends. Physics of Life Reviews, 2:353–373, 2005.

[107] DORIGO, M., BLUM, C., Ant colony optimization theory: a survey. Theoretical Computer Science, 344:243–278, 2005.

[108] DORIGO, M., BIRATTARI, M., STÜTZLE, T., LIBRE, U., BRUXELLES, D., ROOSEVELT, AFD., Ant colony optimization –artificial ants as a computational intelligence technique. IEEE Computational Intelligence Magazine, 1:28–39, 2006.

[109] ANGUS, D., WOODWARD, C., Multiple objective ant colony optimization. Swarm Intelligence, 3:69–85, 2009.

[110] DORIGO, M., THOMAS, S., Ant colony optimization: overview and recent advances. In GENDREAU, M., POTVIN JY. (Eds.), Handbook of Metaheuristics, International Series in Operations Research & Management Science, Springer, US, pp. 227–263, 2010.

[111] FARMER, JD., PACKARD, NH., PERELSON, AS., The immune system, adaptation, and machine learning. Physica D, 2:187–204, 1986.

[112] HART, E., TIMMIS, J., Application areas of AIS: the past, the present and the future. Applied Soft Computing, 8:191–201, 2008.

[113] ZHENG, J., CHEN, Y., ZHANG, W., A survey of artificial immune applications. Artificial Intelligence Review, 34:19–34, 2010.

[114] TIMMIS, J., ANDREWS, P., HART, E., On artificial immune systems and swarm intelligence. Swarm Intelligence, 4:247–273, 2010.

[115] HART, E., MCEWAN, C., TIMMIS, J., A. Hone, Advances in artificial immune systems. Evolutionary Intelligence, 4:67–68, 2011.

[116] PASSINO, KM., Bacterial foraging optimization. International Journal of Swarm Intelligence Research, 1:1–16, 2010.

[117] DAS, S., BISWAS, A., DASGUPTA, S., ABRAHAM, A., Bacterial foraging optimization algorithm: theoretical foundations, analysis, and applications. In: ABRAHAM et al (Eds.), Foundations of Computational Intelligence, Studies in Computational Intelligence, Vol. 3, Springer, Berlin, Heidelberg, pp. 23–55, 2009.

[118] SIMON, D., Biogeography-based optimization. IEEE Transactions on Evolutionary Computation, 12:702–713, 2008.

[119] MACARTHUR, R., WILSON, E., The Theory of Biogeography. Princeton University Press, Princeton, NJ, 1967.

[120] GAO, KZ., SUGANTHAN, PN., CHUA, TJ. , An enhanced migrating birds optimization algorithm for no-wait flow shop scheduling problem. IEEE Symp. on Computational Intelligence in Scheduling (SCIS), Singapore, pp. 9 – 13, 16-19 April, 2013.

[121] MAKAS, H., YUMUŞAK, N., New Cooperative and Modified Variants of the Migrating Birds Optimization Algorithm. In Proc 10th Int. Conf. on Electronics, Computer and Computation (ICECCO'13), Ankara, Turkey, pp. 176–179, 2013.

[122] MAKAS, H., YUMUŞAK, N., A Comprehensive Study on Thyroid Diagnosis by Neural Networks and Swarm Intelligence. In Proc 10th Int. Conf. on Electronics, Computer and Computation (ICECCO'13), Ankara, Turkey, pp. 180–183, 2013.

[123] DUMAN, E., BÜYÜKKAYA, A., ELİKÜÇÜK, I., A Novel and Successful Credit Card Fraud Detection System Implemented in a Turkish Bank. In Proc IEEE 13th Int. Conf. on Data Mining Workshops, Dallas, Texaspp, pp.162 - 171, 7-10 December 2013.

[124] TAŞGETİREN, MF., PAN, QK., SUGANTHAN, PN., DIZBAY, IE., Metaheuristic algorithms for the quadratic assignment problem. In Proc IEEE Symp. on Computational Intelligence in Production and Logistics Systems (CIPLS), Singapore, pp. 131-137, 16-19 April 2013.

[125] PAN, QK., DONG, Y., An improved migrating birds optimisation for a hybrid flow shop scheduling with total flow time minimization. Information Sciences, in press, 2014.

[126] DUMAN, E., UYSAL, M., ALKAYA, AF., Migrating Birds Optimization: A New Meta-heuristic Approach and Its Application to the Quadratic Assignment Problem. In Applications of Evolutionary Computation, DI CHIO et al. (Eds.), Springer-Verlag, Berlin Heidelberg, pp. 254-263, 2011.

[127] DUMAN, E., ELIKÜCÜK, İ., Solving Credit Card Fraud Detection Problem by the New Metaheuristics Migrating Birds Optimization. In Advances in Computational Intelligence, ROJAS et al. (Eds.), Springer-Verlag, Berlin Heidelberg, pp. 62-71, 2013.

[128] DUMAN, E., ELİKÜÇÜK, İ., Applying Migrating Birds Optimization to Credit Card Fraud Detection. In Trends and Applications in Knowledge Discovery and Data Mining, LI et al. (Eds.), Springer-Verlag, Berlin Heidelberg, pp. 416-427, 2013.

[129] JUNG, S., Queen-bee evolution for genetic algorithms. Electronics Letters 39:575–576, 2003.

[130] GORDON, N., WAGNER, IA., BRUCKS, AM., Discrete bee dance algorithms for pattern formation on a grid. In Proc IEEE/WIC Int. Conf. on Intelligent Agent Technology, IAT'03, IEEE Computer Society, Washington, DC, USA, pp. 545–549, 2003.

[131] WEDDE, HF., FAROOQ, M., ZHANG, Y., Bee Hive: an efficient fault-tolerant routing algorithm inspired by honey bee behavior. In DORIGO et al. (Eds.), Ant Colony Optimization and Swarm Intelligence, 4th Int. Workshop, ANTS 2004, Brussels, Belgium, September 5–8, 2004 (Proc, Number 3172 in Lecture Notes in Computer Science, Springer, pp. 83–94, 2004).

[132] ABBASS, HA., MBO: marriage in honey bees optimisation: a haplometrosis polygynous swarming approach. In CEC'2001 Congress on Evolutionary Computation, pp. 207–214, 2001.

[133] LUCIC, P., TEODOROVIC, D., Computing with bees: attacking complex transportation engineering problems. International Journal on Artificial Intelligence Tools, 12:375–394, 2003.

[134] YANG, XS., Engineering optimizations via nature-inspired virtual bee algorithms. In Artificial Intelligence and Knowledge Engineering Applications: A Bio inspired Approach: 1st International Work-Conference on the Interplay Between Natural and Artificial Computation, IWINAC 2005, volume 3562 of Lecture Notes in Computer Science, Springer, pp. 317–323, 2005.

[135] KARABOĞA, D., AKAY, B., A survey: algorithms simulating bee swarm intelligence. Artificial Intelligence Review, 31:61–85, 2009.

[136] KARABOĞA, D., AKAY, B., A comparative study of Artificial Bee Colony algorithm. Applied Mathematics and Computation, 214:108–132, 2009.

[137] KARABOĞA, D., Yapay Zekâ Optimizasyon Algoritmaları. Nobel Yayın, Ankara, 2011.

[138] KARABOĞA, D., BASTÜRK, B., On the performance of artificial bee colony (ABC) algorithm. Applied Soft Computing, 8:687 – 697, 2008.

[139] LISSAMAN, PBS., SCHOLLENBERGER, CA., Formation Flight of Bird. Science, 168:1003–1005, 1970.

[140] SHI, Y., EBERHART, RC., Empirical Study of Particle Swarm Optimization. In Proc of CEC'99 Congress of Evolutionary Computation, pp. 1945-1950, 1999.

[141] EBERHART RC, SHI Y. Comparing inertia weights and constriction factors in Particle Swarm Optimization. In Proc IEEE Int. Congress on Evolutionary Computation, Washington, USA., pp. 84-88, 16-19 July, 2000.

[142] KARABOĞA, D., ÖKDEM, S., A Simple and Global Optimization Algorithm for Engineering Problems: Differential Evolution Algorithm. Turkish Journal of Electrical Engineering 12:53-60, 2004.

[143] HAUPT RL., HAUPT, SE., Practical Genetic Algorithms. Wiley, New York, NY, USA, 2004.

[144] JAMIL M., YANG, XS., A literature survey of benchmark functions for global optimization problems. Int. Journal of Mathematical Modelling and Numerical Optimisation, 4(2):150–194, 2013.

[145] WINSTON, PH., Artificial Intelligence. 3rd ed., Addison-Wesley, 1992.

[146] CHUNG, CJ., REYNOLDS, RG., CAEP: An Evolution-Based Tool for Real-Valued Function Optimization Using Cultural Algorithms. International Journal on Artificial Intelligence Tool, 7(3):239-291, 1998.

[147] BOYER, DO., MARTFNEZ, CH., PEDRAJAS, NG., Crossover Operator for Evolutionary Algorithms Based on Population Features. Journal of Artificial Intelligence Research, 24:1-48, 2005.

[148] DE JONG, KA., An analysis of the behavior of a class of genetic adaptive systems. Ph.D. Thesis, Computer and Communication Sciences, University of Michigan, August, 1975.

[149] SCHUMER, MA., STEIGLITZ, K., Adaptive Step Size Random Search. IEEE Transactions on Automatic Control, 13(3):270-276, 1968.

[150] RASTRIGIN, LA., Systems of extremal control, Nauka, Moscow, Russian, 1974.

[151] GRIEWANK, AO. , Generalized Descent for Global Optimization. Journal of Optimization Theory and Applications, 34(1):11-39, 1981.

[152] MICHALEWICZ, Z., Genetic Algorithms C Data Structures D Evolution Programs. Springer, New York, 1992.

[153] RAHNAMYAN, S., TIZHOOSH, HR., SALAMA, NMM. , A Novel Population Initialization Method for Accelerating Evolutionary Algorithms. Computers and Mathematics with Applications, 53(10):1605-1614, 2007.

[154] SCHWEFEL, HP., Numerical optimization of computer models, Wiley, Chichester, 1981.

[155] ACKLEY, DH., A connectionist machine for genetic hill climbing. Kluwer Academic Publishers, Boston, 1987.

[156] YAO, X., Evolving artificial neural networks. In Proc of the IEEE 87(9):1423-1447, 1999.

[157] RUMELHART, DE., HINTON, GE., WILLIAMS, RJ., Learning representations by back-propagating errors. Nature, 323:533–536, 1986.

[158] WERBOS, PJ., Back-propagation: Past and future. In Proc Int. Conf. on Neural Networks, San Diego, CA, Vol. 1, pp. 343–354, 1988.

[159] LEVENBERG, K., A method for the solution of certain problems in least squares. Quarterly of Applied Mathematics, 5:164–168, 1944.

[160] MARQUARDT, D., An algorithm for least-squares estimation of nonlinear parameters. SIAM Journal on Applied Mathematics, 11(2):431–441, 1963.

[161] MOLLER, MF., A Scaled Conjugate Gradient Algorithm for Fast Supervised Learning. Neural Networks, 6:525-533, 1993.

[162] WATKINS, A., AIRS: A resource limited artificial immune classifier. Ms. D. Thesis, Mississippi State University, 2001.

[163] TEMURTAŞ, F., A comparative study on thyroid disease diagnosis using neural networks. Expert Systems with Applications, 36:944–949, 2009.

[164] KOTHARI, R., DONG, M., Decision trees for classification: a review and some new results. In Lecture Notes in Pattern Recognition, World Scientific Publishing Company, Singapore, 2001.

[165] QUINLAN, JR., Learning efficient classification procedures and their application to chess end games. In MICHALSHI et al. (Ed.), Machine Learning: An Artificial Intelligence Approach. Morgan Kaufmann, 1983

[166] QUINLAN, JR., Induction of decision trees. Machine learning, 1:81–106, 1986.

[167] LÓPEZ-CHAU, A., CERVANTES, J., LÓPEZ-GARCÍA, L., LAMONT, FG., Fisher's decision tree. Expert Systems with Applications, 40:6283–6291, 2013.

[168] AITKENHEAD, MJ., A co-evolving decision tree classification method. Expert Systems with Applications, 34:18–25, 2008.

[169] MIKUCIONIEN, R., MARTINAITIS, V., KERAS, Evaluation of energy efficiency measures sustainability by decision tree method. Energy and Buildings, 76:64–71, 2014.

[170] SHARMA, A., SUGUMARAN, V., DEVASENAPATI, SB., Misfire detection in an IC engine using vibration signal and decision tree algorithms. Measurement, 50:370–380, 2014.

[171] TABAKOV, M., KOZAK, P., Segmentation of histopathology HER2/neu images with fuzzy decision tree and Takagi–Sugeno reasoning. Computers in Biology and Medicine, 49:19–29, 2014.

[172] AYDIN, I., KARAKOSE, M., AKIN, E., An approach for automated fault diagnosis based on a fuzzy decision tree and boundary analysis of a reconstructed phase space. ISA Transactions, 53:220–229, 2014.

[173] KARBASSI, A., MOHEBI, B., REZAEE, S., LESTUZZI, P., Damage prediction for regular reinforced concrete buildings using the decision tree algorithm. Computers and Structures, 130:46–56, 2014.

[174] JEGADEESHWARAN, R., SUGUMARAN, V., Comparative study of decision tree classifier and best first tree classifier for fault diagnosis of automobile hydraulic brake system using statistical features. Measurement, 46:3247–3260, 2013.

[175] KIM, SY., UPNEJA, A., Predicting restaurant financial distress using decision tree and AdaBoosted decision tree models. Economic Modelling, 36:354–362, 2014.

[176] DUCHESSI, P., LAURÍA, EJM., Decision tree models for profiling ski resorts' promotional and advertising strategies and the impact on sales. Expert Systems with Applications, 40,5822–5829, 2013.

[177] NOWAK, M., NOWAKA, B., An application of the multiple criteria decision tree in project planning. Procedia Technology, 9:826-835, 2013.

[178] SAHIN, Y., BULKAN, S., DUMAN, E., A cost-sensitive decision tree approach for fraud detection. Expert Systems with Applications, 40:5916–5923, 2013.

[179] ABELLÁN, J., LÓPEZ, G., DE OÑA, J., Analysis of traffic accident severity using Decision Rules via Decision Trees. Expert Systems with Applications, 40:6047–6054, 2013.

[180] DELPHIN, S., ESCOBEDO, FJ., ABD-ELRAHMAN, A., CROPPER JR., W., Mapping potential carbon and timber losses from hurricanes using a decision tree and ecosystem services driver model. Journal of Environmental Management, 129:599-607, 2013.

[181] TEHRANY, MS., PRADHAN, B., JEBUR, MN., Spatial prediction of flood susceptible areas using rule based decision tree (DT) and a novel ensemble bivariate and multivariate statistical models in GIS. Journal of Hydrology, 504:69–79, 2013.

[182] QUINLAN, JR., C4.5: Programs for Machine Learning. Morgan Kaufmann, Los Altos, 1993.

[183] BREIMAN, L., FRIEDMAN, J., OLSHEN, R., STONE C., Classification and Regression Trees. Wadsworth Int. Group., 1984.

[184] KASS, GV., An Exploratory Technique for Investigating Large Quantities of Categorical Data. Applied Statistics, 29(2):119–127, 1980.

[185] FRIEDMAN, JH., Multivariate Adaptive Regression Splines. The Annals of Statistics, 19:1-141, 1991.

[186] MEHTA, M., AGRAWAL, R., RIASSNEN, J., SLIQ: a fast scalable classifier for data mining. In Extending Database Technology, Avignon, France, 1996.

[187] SHAFER, JC., AGARWAL, R., MEHTA, M., SPRINT: a scalable parallel classifier for data mining. In Proc 24th Int. Conf. Very Large Databases, 1996.

[188] MITCHELL, TM., Machine Learning. McGraw-Hill International, 1997.

[189] SAFFAVIAN, SR., LANDGREBE, D., A survey of decision tree classifier methodology. IEEE Transactions on Systems, Man, and Cybernetics, 21(3):660–674, 1991.

[190] ATTNEAVE, F., Applications of Information Theory to Psychology. Holt, Rinehart and Winston, 1959.

[191] DIETTERICH, TG., KEARNS, M., MANSOUR, Y., Applying the weak learning framework to understand and improve C4.5. In Proc 13th Int. Conf. Machine Learning, Morgan Kaufmann, San Francisco, pp. 96-104, 1996.

[192] KEARNS, M., MANSOUR, Y., On the boosting ability of top-down decision tree learning algorithms. Journal of Computer and Systems Sciences, 58(1):109-128, 1999.

[193] LOPEZ DE MANTRAS, R., A distance-based attribute selection measure for decision tree induction. Machine Learning, 6:81-92, 1991.

[194] FAYYAD, U., IRANI KB., The attribute selection problem in decision tree generation. In Proc 10th National Conf. Artificial Intelligence, Cambridge, MA: AAAI Press/MIT Press, pp. 104–110, 1992.

[195] FRIEDMAN, JH., A recursive partitioning decision rule for nonparametric classifiers. IEEE Trans. on Comp., C26:404-408, 1977.

[196] ROUNDS, E., A combined non-parametric approach to feature selection and binary decision tree design. Pattern Recognition, 12:313-317, 1980.

[197] FERRI, C., FLACH, P., HERN´ANDEZ-ORALLO, J., Learning Decision Trees Using the Area Under the ROC Curve. In SAMMUT, C., HOFFMANN A. (Ed.), Proc 19th Int. Conf. Machine Learning, Morgan Kaufmann, pp. 139-146, 2002

[198] MAIMON, O., ROKACH, L., Data Mining and Knowledge Discovery Handbook. Springer, New York Dordrecht Heidelberg London, 2010.

[199] PAGALLO, G. HUASSLER, D., Boolean feature discovery in empirical learning. Machine Learning, 5(1):71-99, 1990.

[200] FARID, DM., ZHANG, L., RAHMAN, CM., HOSSAIN, MA., STRACHAN, R., Hybrid decision tree and naïve Bayes classifiers for multi-class classification tasks. Expert Systems with Applications, 41:1937–1946, 2014.

[201] LENNART, L., System Identification: Theory for the User. Prentice Hall, New Jersey, 1997.

[202] ROSENQVIST, F., KARLSTRÖM, A., Realisation and estimation of piecewise-linear output-error models. Automatica, 41:545-551, 2005.

[203] PINTELON R, SCHOUKENS J. Box-Jenkins identification revisited - Part I: Theory. Automatica, 42:63-75, 2006.

[204] STROBACH, P., Pure order recursive least-squares ladder algorithms. IEEE Transactions on Acoustics, Speech and Signal Processing, 34(4):880-897, 1986.

[205] ÖZER, Ş., GÜNEY, K., MAKAS. H., Erciyes Üniversitesi FBE Dergisi 16(1-2):101-107, 2000.

[206] KARABOĞA, N., A new design method based on artificial bee colony algorithm for digital IIR filters. Journal of the Franklin Institute, 346: 328-348, 2009.

[207] KARABOĞA, N, ÇETINKAYA, MB. A novel and efficient algorithm for adaptive filtering: Artificial bee colony algorithm. Turk J Elec Eng & Comp Sci, 19(1):175-190, 2011.

[208] MAKAS, H., YUMUSAK, N., System identification by using migrating birds optimization algorithm: a comparative performance analysis. Turk J Elec Eng & Comp Sci, DOI: 10.3906/elk-1311-45 (Accepted paper).

[209] ERÇIN, O., ÇOBAN, R., Identification of linear dynamic systems using the artificial bee colony algorithm. Turk J Elec Eng & Comp Sci, 20(1):1175-1188, 2012

[210] KALINLI, A., Simulated annealing algorithm-based Elman network for dynamic system identification. Turk J Elec Eng & Comp Sci, 20(4):569-582, 2012.

[211] MAKAS, H., The performance analysis of PORLA method in system modeling. Ms. D. Thesis, Erciyes University, Kayseri, Turkey, 1998.

[212] SHENA, Q., SHIA, W., YANGC, X., YEA, B., Particle swarm algorithm trained neural network for QSAR studies of inhibitors of platelet-derived growth factor receptor phosphorylation. European Journal of Pharmaceutical Sciences, 28:369–376, 2006.

[213] ASKARZADEH, A., REZAZADEH, A., Artificial neural network training using a new efficient optimization algorithm. Applied Soft Computing, 13:1206–1213, 2013.

[214] MIRARAB, M., SHARIFI, M., GHAYYEM, MA., MIRARAB, F., Prediction of solubility of CO2 in ethanol–[EMIM][Tf2N] ionic liquid mixtures using artificial neural networks based on genetic algorithm. Fluid Phase Equilibria, 371:6–14, 2014.

[215] DAS, G., PATTNAIK, PK., PADHY, SK., Artificial Neural Network trained by Particle Swarm Optimization for non-linear channel equalization. Expert Systems with Applications, 41:3491–3496, 2014.

[216] KHAJEH, M., KAYKHAII, M., HASHEMI, SH., SHAKERI, M., Particle swarm optimization–artificial neural network modeling and optimization of leachable zinc from flour samples by miniaturized homogenous liquid–liquid micro extraction. Journal of Food Composition and Analysis, 33:32–38, 2014.

[217] PIOTROWSKI, AP., OSUCH, M., NAPIORKOWSKI, MJ., ROWINSKI, PM., NAPIORKOWSKI, JJ., Comparing large number of metaheuristics for artificial neural networks training to predict water temperature in a natural river. Computers & Geosciences, 64:136–151, 2014.

[218] LU, Q., SHEN, G., YU, R., A chaotic approach to maintain the population diversity of genetic algorithm in network training, Computational Biology and Chemistry, 27:363-371, 2003.

[219] CHEN, M., YAO, Z., Classification Techniques of Neural Networks Using Improved Genetic Algorithms. In Proc IEEE 2nd Int. Conf. on Genetic and Evolutionary Computing, pp. 115-119, 2008.

[220] ÖZTÜRK, C., KARABOĞA, D., Hybrid Artificial Bee Colony Algorithm for Neural Network Training. In Proc IEEE Congress on Evolutionary Computation (CEC), pp. 84-88, 2011.

[221] YAGHINI, M., KHOSHRAFTAR, MM., FALLAHI, M., A hybrid algorithm for artificial neural network training. Engineering Applications of Artificial Intelligence, 26:293–301, 2013.

[222] http://archive.ics.uci.edu/ml/index.html, UCI Machine Learning Repository web site, Erişim Tarihi: 10.06.13.

[223] http://sci2s.ugr.es/keel/category.php?cat=clas#sub2, KEEL Data Set Repository web site, Erişim Tarihi: 04.03.14.

[224] CZERNIAK, J., ZARZYCKI, H., Application of rough sets in the presumptive diagnosis of urinary system diseases. In Proc 9th Int. Conf. Artificial Intelligence and Security in Computing Systems ACS'2002, pp. 41-51, 2003.

[225] YEH, I., YANG, K., TING, T., Knowledge discovery on RFM model using Bernoulli sequence. Expert Systems with Applications, 36:5866-5871, 2009.

[226] WOLBERG, WH., MANGASARIAN, OL., Multi-surface method of pattern separation for medical diagnosis applied to breast cytology. Proc of the National Academy of Sciences, 87:9193-9196, 1990.

[227] ZHANG, J., Selecting typical instances in instance-based learning. In Proc 9th Int. Machine Learning Conf., pp. 470-479, 1992.

[228] GIL, D., GIRELA, JL., JUAN, JD., GOMEZ-TORRES, MJ., JOHNSSON, M., Predicting seminal quality with artificial intelligence methods. Expert Systems with Applications, 39:12564-12573, 2012.

[229] RAMANA, BV., BABU, MSP., VENKATESWARLU, NB., A Critical Comparative Study of Liver Patients from USA and INDIA: An Exploratory Analysis. International Journal of Computer Science Issues, Vol. 9, Iss. 3, No. 2, pp. 506-516, 2012.

[230] RAMANA, BV., BABU, MSP., VENKATESWARLU, NB., A Critical Study of Selected Classification Algorithms for Liver Disease Diagnosis. International Journal of Database Management Systems (IJDMS), 3(2):101-114, 2011.

[231] WITTEN, IH., MACDONALD, BA., Using concept learning for knowledge acquisition. International Journal of Man-Machine Studies, 27:349-370, 1988.

[232] CENDROWSKA, J., PRISM: An algorithm for inducing modular rules. International Journal of Man-Machine Studies, 27:349-370, 1987.

[233] FORSYTH, RS., PC/BEAGLE User's Guide. BUPA Medical Research Ltd, 1990.

[234] SMITH, JW., EVERHART, JE., DICKSON, WC., KNOWLER, WC., JOHANNES, RS., Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. In Proc Symp. on Computer Applications and Medical Care, pp. 261-265, 1988.

[235] BHATT, RB., GOPAL, M., FRCT: Fuzzy-Rough Classification Trees. Pattern Analysis and Applications, 11(1):73-88, 2008.

[236] SAHU, S., BHATT, RB., Automatic classification of Electroencephalography Signals using Wavelet Packet Analysis and Fuzzy Decision Trees. In Proc 28th National Systems Conference, Vellore, India, 16-18 December, 2004.

[237] KURGAN, LA., CIOS, KJ., TADEUSIEWICZ, R., OGIELA, M., GOODENDAY, LS., Knowledge Discovery Approach to Automated Cardiac SPECT Diagnosis. Artificial Intelligence in Medicine, 23:149-169, 2001.

[238] CIOS, KJ., WEDDING, DK., LIU, N., CLIP3: cover learning using integer programming. Kybernetes, 26(5):513-536, 1997.

[239] POLAT, K., ŞAHAN, S., GÜNEŞ, S., A novel hybrid method based on artificial immune recognition system (AIRS) with fuzzy weighted pre-processing for thyroid disease diagnosis. Expert Systems with Applications, 32:1141-1147, 2007.

[240] BERTHONNAUD, E., DIMNET, J., ROUSSOULY, P., LABELLE, H., Analysis of the sagittal balance of the spine and pelvis using shape and orientation parameters. Journal of Spinal Disorders & Techniques, 18(1):40-47, 2005.

[241] NETO, ARR., BARRETO, GA., On the Application of Ensembles of Classifiers to the Diagnosis of Pathologies of the Vertebral Column: A Comparative Analysis. IEEE Latin America Transactions 7(4):487-496, 2009.

[242] WEISS, SM., KULIKOWSKI, CA., Computer Systems that Learn: Techniques from Statistics, Neural Nets, Machine Learning, and Expert Systems. Morgan Kaufmann, Palo Alto, CA, 1991.

[243] DAWSON, RJM., The Unusual Episode Data Revisited. Journal of Statistics Education [Online], 3(3), Available at: www.amstat.org/publications/jse/v3n3/datasets.dawson.html, 1995.

[244] KOTSIANTIS, SB., PINTELAS, PE., Logitboost of Simple Bayesian Classifier. Informatica, 29(1):53-59, 2005.

[245] ELTER, M., SCHULZ-WENDTLAND, R., WITTENBERG, T., The prediction of breast cancer biopsy outcomes using two CAD approaches that both emphasize an intelligible decision process. Medical Physics 34(11):4164-4172, 2007.

[246] LANGLEY, P., A General Theory of Discrimination Learning. In KLAHR et al (Eds.), Production System Models of Learning and Development, Cambridge, MA: MIT Press, pp. 99-161, 1987.

[247] HABERMAN, SJ., Generalized Residuals for Log-Linear Models. In Proc 9th Int. Biometrics Conf. Boston, pp. 104-122, 1976.

[248] LANDWEHR, JM., PREGIBON, D., SHOEMAKER, AC., Graphical Models for Assessing Logistic Regression Models (with discussion). Journal of the American Statistical Association, 79:61-83, 1984.

[249] PAPADIMITRIOU, CH., STEIGLITZ, K., Combinatorial Optimization – Algorithms and Complexity. Dover Publications Inc., New York, 1982.

[250] BLUM, C., ROLI, A., Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. ACM Computing Surveys, 35:268–308, 2003.

[251] YIP, SP., WEBB, GI., Empirical function attribute construction in classification learning. In Joint Conf. Artificial Intelligence (AI'94), pp. 29–36, 1994.

[252] PASI, L., Similarity classifier applied to medical data sets. In Int. Conf. on Soft Computing. Helsinki, Finland & Gulf of Finland & Tallinn, Estonia, 2004.

[253] SUN, Y., KAMEL, MS., WONG, AKC., WANG, Y., Cost-sensitive boosting for classification of imbalanced data. Pattern Recognition, 40:3358–3378, 2007.

[254] YURTAY, N., ADAK, MF., DURAL, D., SERTTAS, S., A study on use of decision tree method in the diagnosıs of thyroid disease. In Proc Int. Science and Technology Conference (ISTEC) , Dubai, pp. 768-774, 2012.

[255] BIRATTARI, M., Tuning Metaheuristics: A Machine Learning Perspective. Springer Publishing Company, 1st ed., 2005.

[256] EIBEN, AE., SMIT, SK., Parameter tuning for configuring and analyzing evolutionary algorithms. Swarm and Evolutionary Computation, 1:19–31, 2011.

[257] YEGUAS, E., LUZÓN, MV., PAVÓN, R., LAZA, R., ARROYO, G., DÍAZ, F., Automatic parameter tuning for Evolutionary Algorithms using a Bayesian Case-Based Reasoning system. Applied Soft Computing, 18:185–195, 2014.

[258] MATURANA, J., LARDEUX, F., SAUBION, F., Controlling behavioral and structural parameters in evolutionary algorithms. In Int. Conf. Artificial Evolution, EA'2009, In Lecture Notes in Computer Science, Vol. 5926, Springer, pp. 110–121, 2009.

[259] LOBO, F., LIMA, C., MICHALEWICZ, Z., Parameter Setting in Evolutionary Algorithms. Springer, 2007.

[260] SMIT, SK., EIBEN, AE., Comparing parameter tuning methods for evolutionary algorithms. In Proc IEEE Congress on Evolutionary Computation, IEEE Press, Trondheim, pp. 399–406, 2009.

[261] SMIT, SK., EIBEN, AE., Using entropy for parameter analysis of evolutionary algorithms. In: BARTZ-BEIELSTEIN (Eds.), Experimental Methods for the Analysis of Optimization Algorithms, In Natural Computing Series, Springer, pp. 287–310, 2010.

[262] BARTZ-BEIELSTEIN, T., LASARCZYK, C., PREUSS, M., The sequential parameter optimization toolbox. In BARTZ-BEIELSTEIN (Eds.), Empirical Methods for the Analysis of Optimization Algorithms, Springer, Berlin, Heidelberg, New York, pp. 337–360, 2009.

[263] GREFFENSTETTE, J., Optimisation of control parameters for genetic algorithms. In SAGE A. (Ed.), IEEE Transactions on Systems, Man and Cybernetics, Vol. 16 (1), IEEE Press, Piscataway, NJ, pp. 122–128, 1986.

[264] NANNEN, V., EIBEN, AE., Relevance Estimation and Value Calibration of evolutionary algorithm parameters. In VELOSO MM. (Eds.), Proc 20th Int. Joint Conf. Artificial Intelligence, IJCAI, Hyderabad, India, pp. 1034–1039, 2007.

[265] NANNEN, V., EIBEN, AE., Efficient Relevance Estimation and Value Calibration of evolutionary algorithm parameters. In IEEE Congress on Evolutionary Computation, pp. 103–110, 2007.

[266] SCHAFFER, J., CARUANA, R., ESHELMAN, L., DAS, R., A study of control parameters affecting online performance of genetic algorithms for function optimization. In Proc 3rd Int. Conf. Genetic Algorithms, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 51–60, 1989.

[267] COY, SP., GOLDEN, BL., RUNGER, GC., WASIL, EA., Using experimental design to find effective parameter settings for heuristics. Journal of Heuristics, 7:77–97, 2001.

[268] YI, H., DUAN, Q., LIAO, TW., Three improved hybrid metaheuristic algorithms for engineering design optimization. Applied Soft Computing, 13:2433–2444, 2013.

[269] LIEFOOGHE, A., VEREL, S., HAO, JK., A hybrid metaheuristic for multi objective unconstrained binary quadratic programming. Applied Soft Computing, 16:10–19, 2014.

[270] PANDIAN, SMV., THANUSHKODI, K., Considering transmission loss for an economic dispatch problem without valve - point loading using an EP - EPSO algorithm. Turk J Elec Eng & Comp Sci, 20:1259-1267, 2012.

[271] MUTLUER, M., BILGIN, O., Application of a hybrid evolutionary technique for efficiency determination of a submersible induction motor. Turk J Elec Eng & Comp Sci, 19:877-890, 2011.

[272] JADDI, NS., ABDULLAH, S., Hybrid of genetic algorithm and great deluge algorithm for rough set attribute reduction. Turk J Elec Eng & Comp Sci, 21:1737-1750, 2013.

[273] BU, TM., YU, SN., GUAN, HW., Binary-Coding-Based Ant Colony Optimization and Its Convergence. Journal of Computer Science and Technology, 19(4):472-478, 2004.

[274] HU, XM., ZHANG, J., LI, Y., Orthogonal Methods Based Ant Colony Search for Solving Continuous Optimization Problems. Journal of Computer Science and Technology, 23(1): 2-18, 2008.

[275] CHU, ZF., XIA, YS., WANG, LY., Cell Mapping for Nanohybrid Circuit Architecture Using Genetic Algorithm. Journal of Computer Science and Technology, 27(1):113-120, 2012.

[276] LI, J., LIU, X., Melt index prediction by RBF neural network optimized with an MPSO-SA hybrid algorithm. Neurocomputing, 74:735–740, 2011.

[277] POP, PC., MATEI, O., SITAR, CP., An improved hybrid algorithm for solving the generalized vehicle routing problem. Neurocomputing, 109:76–83, 2013.

[278] SUN, Y., ZHANG, L., GU, X., A hybrid co-evolutionary cultural algorithm based on particle swarm optimization for solving global optimization problems. Neurocomputing, 98:76–89, 2012.

[279] BLUM, C., PUCHINGER, J., RAIDL, GR., ROLI, A., Hybrid metaheuristics in combinatorial optimization: a survey. Applied Soft Computing, 11:4135–4151, 2011.

[280] FORNARELLI, G., GIAQUINTO, A. An unsupervised multi - swarm clustering technique for image segmentation. Swarm and Evolutionary Computation, 11:31 – 45, 2013.

[281] WANG, Y., WANG, H., LEI, X., JIANG, Y., SONG, X., Flood simulation using parallel genetic algorithm integrated wavelet neural networks. Neurocomputing, 74:2734–2744, 2011.

[282] BAÑOS, R., ORTEGA, J., GIL, C., FERNÁNDEZ, A., TORO F., A Simulated Annealing-based parallel multi objective approach to vehicle routing problems with time windows. Expert Systems with Applications, 40:1696–1707, 2013.

[283] YUSOF, R., KHALID, M., HUI, GT., YUSOF, SM, OTHMAN, MF., Solving job shop scheduling problem using a hybrid parallel micro genetic algorithm. Applied Soft Computing, 11:5782–5792, 2011.

[284] YU, B, YANG, Z, SUN, X, YAO, B, ZENG, Q, JEPPESEN, E., Parallel genetic algorithm in bus route headway optimization. Applied Soft Computing, 11:5081–5091, 2011.

[285] ALBA, E., Parallel Metaheuristics: A New Class of Algorithms. Wiley-Interscience, 2005.

EKLER

EK A: ABC Algoritması MATLAB Test Programı

```
function [fBest, bestSolution, fHistABC] = ...
    ABC_Benchmark(benchmarkFunction, limits, param, dimension, ...
                        colonySize, maxCycleLimit, trial)

% benchmarkFunction : Test fonksiyonu numarası
% limits            : [l_alt l_ust] Pronlem uzayı alt ve üst sınırı
% param             : Test fonksiyonu parametre vektörü
% dimension         : Problem boyutu
% colonySize        : Koloni büyüklüğü
% maxCycleLimit     : Maksimum iterasyon
% trial             : Deneme sayısı
% fBest             : En iyi çözümün maliyet değeri
% bestSolution      : En iyi çözümün
% fHistABC          : Maliyet değeri trend verisi

% Algoritma parametrelerinin ayarlanması
limitForScout = colonySize*dimension/2;
numOfEmpBee = colonySize/2;
numOfOnlookerBee = numOfEmpBee;

% Popülasyonların saklandığı uygun dosyanın seçilmesi
nSamp = 100;
sampFileName = cat(2, ...
    'InitSol_Fun', num2str(benchmarkFunction), ...
    '_Dim', num2str(dimension), ...
    '_Samp', num2str(nSamp), ...
    '.mat');
load(sampFileName);

% Seçilen dosyadan başlangıç popülasyonun alınması
x = xSamp(trial).x(1:colonySize,:);

% Maliyet ve uygunluk değerlerinin hesaplanması
[f, fitness] = EvalSolution(x, benchmarkFunction, param);

% Çözüm geliştirememe sayaçlarının sıfırlanması
failCount = zeros(numOfEmpBee, 1);

% Ana döngü sayacının sıfırlanması
currentCycle = 1;

fitBest = 0;
while currentCycle <= maxCycleLimit
    % Görevli Arı Evresi Başlangıcı --------------------------------
```

```
for i=1:numOfEmpBee
    k=i;
    while k == i                    % Rasgele k indeksi üretimi
        k = fix(1+numOfEmpBee*rand); % (i' den farklı)
    end
    j = fix(1+dimension*rand);      % Rasgele j indeksi üretimi
    phi = -1 + 2*rand;              % Rasgele phi değeri üret.

    v = x(i,:);
    v(1,j) = x(i,j) + phi*(x(i,j)-x(k,j)); % Yeni komşu üretimi

    % Limit değer kontrolü
    v = LimitCheck(v, limits, j);

    % Komşu çözümün maliyet ve uygunluk değer. hesaplanması
    [fnew, fitnew] = EvalSolution(v, benchmarkFunction, param);

    % Aç gözlü seçim uygulama
    [fitness, x, failCount] = ...
        GreedySelection(fitnew, fitness, i, x, v, failCount);
end

% Çözümlerin seçilebilme olasılıklarının hesaplanması
fitSum = sum(fitness);
p = fitness ./ fitSum;

% Gözcü Arı Evresi Başlangıcı ---------------------------------
rValues = rand(1,numOfOnlookerBee).'; % Rulet değerlerinin üret.

for olb=1:numOfOnlookerBee
    % Rulet tekeri yöntemi ile çözüm seçimi
    partialSum = 0;
    selSourceIndex = 1;
    for pIndex=1:numOfEmpBee
        partialSum = partialSum + p(pIndex);
        if partialSum > rValues(olb)
            selSourceIndex = pIndex;
            break;
        end
    end

    i = selSourceIndex;             % Görevli arı evresi gibi
    k = i;                          % -------------------------
    while k == i                    % Rasgele k indeksi üretimi
        k = fix(1+numOfEmpBee*rand); % (i' den farklı)
    end
    j = fix(1+dimension*rand);      % Rasgele j indeksi üretimi
    phi = -1 + 2*rand;              % Rasgele phi değeri üret.

    v = x(i,:);
    v(1,j) = x(i,j) + phi*(x(i,j)-x(k,j)); % Yeni komşu üretimi

    % Limit değer kontrolü
    v = LimitCheck(v, limits, j);

    % Komşu çözümün maliyet ve uygunluk değer. hesaplanması
    [fnew, fitnew] = EvalSolution(v, benchmarkFunction, param);
```

```
        % Aç gözlü seçim uygulama
        [fitness, x, failCount] = ...
            GreedySelection(fitnew, fitness, i, x, v, failCount);
    end

    % İterasyonun en iyi çözümünün belirlenmesi
    [fitBestOFiteration, bestIndex] = max(fitness);
    bestSolutionOFiteration = x(bestIndex,:);

    % En iyi çözümün saklanması
    if fitBestOFiteration>fitBest
        fitBest = fitBestOFiteration;
        bestSolution = bestSolutionOFiteration;
        [bestCost, ~] = ...
              EvalSolution(bestSolution, benchmarkFunction, param);
    end

    % Trend verisinin elde edilmesi
    fHistABC(currentCycle) = bestCost;

    % Kâşif Arı Evresi Başlangıcı --------------------------------

    % Biten kaynakların tespiti ve kâşif arı üretimi
    [maxFail, maxFailIndex] = max(failCount);
    if maxFail > limitForScout
        % biten kaynak için rasgele arama
        v = ProdRandSolution(1, dimension, limits);
        x(maxFailIndex,:) = v ;
        failCount(maxFailIndex) = 0;
    end

    currentCycle = currentCycle + 1 ;
end

[fBest, fitBest] = ...
             EvalSolution(bestSolution, benchmarkFunction, param);
```

EK B: MBO Algoritması MATLAB Test Programı

```
function [fBest, bestSolution, fHistMBO] = ...
    MBO_Benchmark(bFunc, limits, param, dimension, n, k, x, m, K,
trial)

% bFunc        : Test fonksiyonu numarası
% limits       : [l_alt l_ust] Pronlem uzayı alt ve üst sınırları
% param        : Test fonksiyonu parametre vektörü
% dimension    : Problem boyutu
% n            : Poülasyon büyüklüğü
% k            : İlgilenilecek komşuluk sayısı
% x            : Paylaşılacak komşuluk sayısı
% m            : Lider değişimi için gerekli tur sayısı
% K            : Maksimum iterasyon
% trial        : Deneme sayısı
% fBest        : En iyi çözümün maliyet değeri
% bestSolution : En iyi çözümün
% fHistMBO     : Maliyet değeri trend verisi

% Popülasyonların saklandığı uygun dosyanın seçilmesi
% Seçilen dosyadan başlangıç popülasyonun alınması
% ve popülasyonun farazi vir V formasyonuna yerleştirilmesi
[leader, sLeft, sRight] = InitializeBirds(n, dimension, bFunc,
trial);

leftSide = true;
i = 1;
while i<K
    for j=1:m
        % Lider çözümün iyileştirilmesi ve oluşturulan k çözümden 2x
        % adedinin 2. sıradaki kuşlarla paylaşılması
        [leader, sLeft1, sRight1] = ...
            ImproveLeaderSolution...
            (leader,k,x,limits,[sLeft;sRight],bFunc,param);

        %Diğer çözümlerin iyileştirilmesi ve elde edilen k çözümün x
        % adedinin sıradaki kuşlarla paylaşılması
        [sLeft, sRight] = ...
            ImproveOtherSolutions...
                        (leader,sLeft,sRight,sLeft1,sRight1, ...
                                             k,x,limits,bFunc,param);

        % En iyi çözümün seçilmesi
        [f, solution] = SortSolutions...
                                (leader,sLeft,sRight,bFunc,param);
        fHistMBO(i) = f(1);
        i = i + 1;
    end

    % Yorulan kuşun değiştirilmesi
    [leftSide, leader, sLeft, sRight] =...
        ReplaceLeader(leftSide, leader, sLeft, sRight);
end
[f, solution] = SortSolutions(leader,sLeft,sRight,bFunc,param);
bestSolution = solution(1,:);
fBest = f(1);
```

```
function [leader, sLeft, sRight] = ...
    InitializeBirds(n, dimension, bFunc, trial)

% n          : Kuş sayısı
% dimension  : Problem boyutu
% bFunc      : Test fonksiyonu numarası
% trial      : Deneme sayısı
% leader     : Leader solution
% sLeft      : Left-side solutions
% sRight     : Right-side solutions

% Popülasyonların saklandığı uygun dosyanın seçilmesi
nSamp = 100;
sampFileName = cat(2, ...
    'InitSol_Fun', num2str(fN), ...
    '_Dim', num2str(dimension), ...
    '_Samp', num2str(nSamp), ...
    '.mat');
load(sampFileName);

% Seçilen dosyadan başlangıç popülasyonun alınması
s = xSamp(trial).x(1:n,:);

% Kuşlardan birinin lider olarak seçilmesi
leaderIndex = fix(n*rand)+1;
leader = s(leaderIndex,:);
s(leaderIndex,:) = [];

% Sağ ve sol kollardaki kuşların V formasyonuna yerleştirilmesi
sLeft = s(1:(n-1)/2,:);
sRight = s((n-1)/2+1:n-1,:);

function [leader, sLeft1, sRight1] = ...
    ImproveLeaderSolution(leader,k,x,limits,others,bFunc,param)

% leader   : Lider kuş
% others   : diğer kuşlar
% k        : İlgilenilecek komşuluk sayısı
% x        : Paylaşılacak komşuluk sayısı
% limits   : [l_alt l_ust] Pronlem uzayı alt ve üst sınırları
% sLeft1   : Sol tarafta paylaşılacak çözümler
% sRight1  : Sağ tarafta paylaşılacak çözümler
% bFunc    : Test fonksiyonu numarası
% param    : Test fonksiyonu parametre vektörü

% Lider çözüm için k adet komşu çözüm üretilmesi
nb = GenerateNeighbor(leader, others, k, limits);

% Komşu çözümlerin maliyet değerlerinin hesaplanması
[f, ~] = EvalSolution(nb, bFunc, param);

% Komşu çözümlerin artan maliyet sıralamasına sokulması
[f, ix] = sort(f);
nb = nb(ix,:);

% Liderden daha iyi komşu varsa onun lider olarak atanması
[fLeader, fitLeader] = EvalSolution(leader, bFunc, param);
if f(1) < fLeader
```

```
    leader = nb(1,:);
    nb(1,:) = [];
end

% Paylaşılacak 2x komşu çözümün belirlenmesi
for j=1:x
    sLeft1(j,:) = nb(2*j-1,:);
    sRight1(j,:) = nb(2*j,:);
end

function [sLeft, sRight] =  ...
  ImproveOtherSolutions(leader, sLeft, sRight, sLeft1, sRight1, ...
                                        k, x, limits, bFunc, param)
% leader   : Lider kuş
% sLeft    : Sol taraf çözümleri
% sRight   : Sağ taraf çözümleri
% sLeft1   : Liderin sol taraf ile paylaştığı çözümler
% sRight1  : Liderin sağ taraf ile paylaştığı çözümler
% k        : İlgilenilecek komşuluk sayısı
% x        : Paylaşılacak komşuluk sayısı
% limits   : [l_alt l_ust] Problem uzayı alt ve üst sınırları
% bFunc    : Test fonksiyonu numarası
% param    : Test fonksiyonu parametre vektörü

% Sol ve sağ taraftaki kuş sayısının tespit edilmesi
dim = size(sLeft);
p = dim(1);

% Sol ve sağ taraftaki her bir kuşun pozisyonunun iyileştirilmesi
for t=1:p
    % sLeft(t) ve sRight(t) kuşları için (k-x) adet komşunun üert.
    tempL = sLeft;
    tempL(t,:) = [];
    otherL = [leader; tempL; sRight];

    tempR = sRight;
    tempR(t,:) = [];
    otherR = [leader; sLeft; tempR];

    nbLeft = GenerateNeighbor(sLeft(t,:), otherL, k-x, limits);
    nbRight = GenerateNeighbor(sRight(t,:), otherR, k-x, limits);

    % Üretilen ve paylaşılan komşulukların birleştirilmesi
    nbLeft = [nbLeft; sLeft1];
    nbRight = [nbRight; sRight1];

    % Komşu çözümlerin maliyet değerlerinin hesaplanması
    [fLeft, ~] = EvalSolution(nbLeft,bFunc,param);
    [fRight, fitRght] = EvalSolution(nbRight,bFunc,param);

    % Komşu çözümlerin artan maliyet sıralamasına sokulması
    [fLeft, ixL] = sort(fLeft);
    nbLeft = nbLeft(ixL,:);

    [fRight, ixR] = sort(fRight);
    nbRight = nbRight(ixR,:);

    % sLeft(t) veya sRight(t) kuşlarından daha iyi komşu varsa
```

```
    % bu komşuların ilgili çözümlere atanması
    [cmpLeft, fitLft] = EvalSolution(sLeft(t,:),bFunc,param);
    if fLeft(1) < cmpLeft
        sLeft(t,:) = nbLeft(1,:);
        nbLeft(1,:) = [];
    end

    [cmpRight, fitLft] = EvalSolution(sRight(t,:),bFunc,param);
    if fRight(1) < cmpRight
        sRight(t,:) = nbRight(1,:);
        nbRight(1,:) = [];
    end

    % Paylaşılacak komşu setlerinin belirlenmesi
    sLeft1 = nbLeft(1:x,:);
    sRight1 = nbRight(1:x,:);

end
```

EK C: PSO Algoritması MATLAB Test Programı

```
function [fBest, gBest, fHistPSO] =
        PSO_Benchmark(bFunc, limits, param, dimension, n, c1, ...
                      Vmax, wLim, K, trial)

% bFunc      : Test fonksiyonu numarası
% limits     : [l_alt l_ust] Pronlem uzayı alt ve üst sınırları
% param      : Test fonksiyonu parametre vektörü
% dimension  : Problem boyutu
% n          : Koloni büyüklüğü
% c1         : ivme katsayısı
% Vmax       : Hız limiti
% wLim       : [w_alt w_ust] Eylemsizlik ağırlığı limitleri
% K          : Maksimum iterasyon
% trial      : Deneme sayısı
% fBest      : En iyi çözümün maliyet değeri
% gBest      : En iyi çözümün
% fHistPSO   : Maliyet değeri trend verisi

% Algoritma parametrelerinin ayarlanması
c2 = c1;
w = linspace(wLim(1,1), wLim(1,2), K); %Lineer azalan eylemsiz. ağ.

% Popülasyonların saklandığı uygun dosyanın seçilmesi
nSamp = 100;
sampFileName = cat(2, ...
    'InitSol_Fun', num2str(bFunc), ...
    '_Dim', num2str(dimension), ...
    '_Samp', num2str(nSamp), ...
    '.mat');
load(sampFileName);

% Seçilen dosyadan başlangıç popülasyonun alınması
X = xSamp(trial).x(1:n,:);
% Parçacıklara en iyi değer başlangıç eğerlerinin atanması
P = X;
% Hız vektörlerinin başlangıç değerlerinin sıfırlanması
V = zeros(n, dimension);
% En iyi sosyal bileşenin tespiti
[gBest, fBest, f] = GetBestAndFitness(X, bFunc, param);

for iter=1:K
    % Hız vektörünün güncellenmesi
    for i=1:n
        for j=1:dimension
            % i çözümünün j boyutundaki hız vektörünün hesaplanması
            V(i,j) = w(iter)*V(i,j) + c1*rand*(P(i,j)-X(i,j)) + ...
                                      c2*rand*(gBest(1,j)-X(i,j));

            % Hız vektörü için limit kontrolü
            if V(i,j)>Vmax
                V(i,j) = Vmax;
            elseif V(i,j)<-Vmax
                V(i,j) = -Vmax;
            end
        end
    end
```

```
    % Sürü üyelerinin yeni pozisyonlarının hesaplanması
    X = X + V;

    % Pozisyonlar için limit kontrolü
    X = chkPopLmt(X, limits);

    % En iyi sosyal bileşenin güncellenmesi
    [gBestNew, fBestNew, fNew] = GetBestAndFitness(X, bFunc, param);
    if fBestNew<fBest
        gBest = gBestNew;
        fBest = fBestNew;
    end

    % Bireysel en iyi pozisyonların güncellenmesi
    for i=1:n
        if fNew(i)<f(i)
            P(i,:) = X(i,:);
        end
    end

    f = fNew;
    fHistPSO(iter) = fBest;
end
```

EK D: DE Algoritması MATLAB Test Programı

```
function [fBest, bestSolution, errDE] = ...
    DE_Benchmark(bFunc,lim,prm,dim,pSize,F,CR,maxCyc, trial)

% bFunc        : Test fonksiyonu numarası
% lim          : [l_alt l_ust] Pronlem uzayı alt ve üst sınırları
% prm          : Test fonksiyonu parametre vektörü
% dim          : Problem boyutu
% pSize        : Poülasyon büyüklüğü
% F            : Amplifikasyon sabiti
% CR           : Rekombinasyon sabiti
% maxCyc       : Maksimum iterasyon
% trial        : Deneme sayısı
% fBest        : En iyi çözümün maliyet değeri
% bestSolution : En iyi çözümün
% errDE        : Maliyet değeri trend verisi

% Popülasyonların saklandığı uygun dosyanın seçilmesi
nSamp = 100;
sampFileName = cat(2, ...
    'InitSol_Fun', num2str(bFunc), ...
    '_Dim', num2str(dim), ...
    '_Samp', num2str(nSamp), ...
    '.mat');
load(sampFileName);

% Seçilen dosyadan başlangıç popülasyonun alınması
x = xSamp(trial).x(1:pSize,:);

% Maliyet değerlerinin hesaplanması
[f, ~] = EvalSolution(x, bFunc, prm);

for t = 1:maxCyc
    %Mutant vectör oluşturulması
    for i = 1:pSize
        r = randperm(pSize);
        for j = 1:3
            if r(j) == i
                r(j) = [];
            end
        end
        v(i,:) = x(r(1),:) + F*(x(r(2),:) - x(r(3),:));

        %Limit kontrolü
        for j = 1:dim
            if v(i,j)<lim(1)
                v(i,j) = lim(1);
            elseif v(i,j)>lim(2)
                v(i,j) = lim(2);
            end
        end
    end

    % Deneme vektörünün oluşturulması
    for i = 1:pSize
        for j = 1:dim
```

```
            if rand()<CR || j == randi(dim)
                u(i,j) = v(i,j);
            else
                u(i,j) = x(i,j);
            end
        end
    end

    % Deneme vektörünün maliyet değerlerinin hesaplanması
    [f_u, ~] = EvalSolution(u, bFunc, prm);

    % Aç gözlü seçim uygulama
    for i = 1:pSize
        if f_u(i)<f(i)
            f(i) = f_u(i);
            x(i,:) = u(i,:);
        end
    end
    errDE(t) = min(f);
end

% En iyi çözümün seçilmesi
[fBest, bestIndex] = min(f);
bestSolution = x(bestIndex,:);
```

EK E: GA Algoritması MATLAB Test Programı

```
function [fBest, bestSolution, fHistGA] = ...
    GA_Benchmark(bFunc, lim, prm, dim, pSize, Mu,maxCyc, trial)

% bFunc        : Test fonksiyonu numarası
% lim          : [l_alt l_ust] Pronlem uzayı alt ve üst sınırları
% prm          : Test fonksiyonu parametre vektörü
% dim          : Problem boyutu
% pSize        : Poülasyon büyüklüğü
% Mu           : Mutasyon yüzdesi
% maxCyc       : Maksimum iterasyon
% trial        : Deneme sayısı
% fBest        : En iyi çözümün maliyet değeri
% bestSolution : En iyi çözümün
% fHistGA      : Maliyet değeri trend verisi

% Popülasyonların saklandığı uygun dosyanın seçilmesi
nSamp = 100;
sampFileName = cat(2, ...
    'InitSol_Fun', num2str(bFunc), ...
    '_Dim', num2str(dim), ...
    '_Samp', num2str(nSamp), ...
    '.mat');
load(sampFileName);

% Seçilen dosyadan başlangıç popülasyonun alınması
x = xSamp(trial).x(1:pSize,:);

% Maliyet değerlerinin hesaplanması
[f, ~] = EvalSolution(x, bFunc, prm);

% Maliyet değerlerine göre popülasyon üyelerinin sıralanması
[f, ix] = sort(f);
x = x(ix,:);

for t = 1:maxCyc
    % Pn olasılıklarının hesaplanması
    Psum = sum(f);
    Pn = f ./ Psum ;

    % Pi değerlerinin hesaplanması
    sumP = 0;
    for i=1:pSize
        sumP = sumP + Pn(i);
        Pi(i) = sumP;
    end

    % Eşleştirme işlemi
    PA = rand(pSize/2,1);
    PB = rand(pSize/2,1);
    for i=1:pSize/2
        for j=1:pSize
            if PA(i)<Pi(j)
                A(i) = j;
                break;
            end
```

```
    end

    for j=1:pSize
        if PB(i)<Pi(j)
            B(i) = j;
            break;
        end
    end

end

% Çaprazlama işlemi
for i=1:pSize/2
    chromosomeA = x(A(i),:);
    chromosomeB = x(B(i),:);
    alpha = randi(dim);
    beta = rand;

    Pa_new = chromosomeA(alpha) - ...
        beta*(chromosomeA(alpha) - chromosomeB(alpha));

    Pb_new = chromosomeB(alpha) - ...
        beta*(chromosomeA(alpha) - chromosomeB(alpha));

    chromosomeA(alpha) = Pa_new;
    chromosomeB(alpha) = Pb_new;

    if alpha < dim
        tmp = chromosomeA(alpha:dim);
        chromosomeA(alpha:dim) = chromosomeB(alpha:dim);
        chromosomeB(alpha:dim) = tmp;
    end

    newGen(2*i-1, :) = chromosomeA;
    newGen(2*i  , :) = chromosomeB;

end

% Mutasyon işlemi
if t<maxCyc
    totPar = pSize * dim;
    totMut = round(totPar*Mu/100);

    for i=1:totMut
        rowIndex = randi(pSize);
        colIndex = randi(dim);
        mutVal = lim(1) + rand*(lim(2)-lim(1));
        newGen(rowIndex,colIndex) = mutVal;
    end
end



% Yeni jenerasyonun maliyet değerlerinin hesaplanması
[fNg, ~] = EvalSolution(newGen, bFunc, prm);

% Jenerasyonları birleştirilmesi
xx = [x;newGen];
```

```
    ff = [f;fNg];

    % Seleksiyon
    [ff, ix] = sort(ff);
    xx = xx(ix,:);
    x = xx(1:pSize,:);
    f = ff(1:pSize,:);

    fHistGA(t) = f(1);

end

fBest = f(1);
bestSolution = x(1,:);
```

EK F: Öğrenci Tercih Listesi

Öğrenci Adı	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20
St-001	●		●		●															
St-002		●	●	●																
St-003	●			●			●													
St-004		●											●			●				
St-005														●	●	●				
St-006															●		●	●		
St-007																●	●	●		
St-008				●	●	●														
St-009	●	●	●																	
St-010		●						●			●									
St-011			●						●			●								
St-012				●						●			●							
St-013		●		●					●											
St-014		●		●						●										
St-015	●						●	●												
St-016							●	●	●											
St-017								●	●				●							
St-018									●	●	●									
St-019					●	●									●					
St-020						●									●	●				
St-021															●			●	●	
St-022																		●	●	●
St-023	●		●		●															
St-024		●		●		●														
St-025			●		●										●					
St-026				●		●												●		
St-027									●						●				●	
St-028										●		●		●						
St-029											●								●	●
St-030	●																		●	●
St-031		●					●												●	
St-032	●						●		●											
St-033		●							●				●							
St-034	●			●					●											
St-035		●			●												●			
St-036						●			●								●			
St-037									●										●	●
St-038								●	●				●							
St-039				●	●	●														
St-040	●	●	●																	
St-041		●			●			●												
St-042			●			●			●											
St-043				●			●						●							
St-044		●		●		●														
St-045		●		●			●													
St-046	●	●	●																	
St-047							●	●	●											
St-048								●	●	●										
St-049	●								●	●										
St-050					●	●	●													
St-051	●									●		●								
St-052											●					●			●	
St-053																●			●	●
St-054	●		●		●															
St-055		●		●		●														
St-056											●		●		●					
St-057																●		●		●

Öğrenci Adı	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20
St-058													●		●		●			
St-059						●		●		●										
St-060																	●		●	●
St-061	●																		●	●
St-062		●	●																●	
St-063	●		●												●					
St-064		●	●	●																
St-065	●			●													●			
St-066		●													●			●		
St-067																●	●	●		
St-068							●		●	●										
St-069								●	●	●										
St-070				●	●	●														
St-071	●											●	●							
St-072												●			●			●		
St-073													●			●			●	
St-074														●			●			●
St-075												●		●		●				
St-076		●		●			●													
St-077	●	●	●																	
St-078		●							●	●										
St-079									●	●	●									
St-080													●	●	●					
St-081														●					●	●
St-082																	●	●	●	
St-083																		●	●	●
St-084								●	●	●										
St-085	●		●		●															
St-086		●		●									●							
St-087			●									●					●			
St-088				●									●					●		
St-089												●					●		●	
St-090													●					●		●
St-091							●		●	●										
St-092	●								●	●										
St-093		●	●						●											
St-094	●										●		●							
St-095		●									●	●								
St-096	●											●			●					
St-097		●													●			●		
St-098																●	●	●		
St-099														●		●				●
St-100															●	●				●
St-101				●	●	●														
St-102	●	●	●																	
St-103		●			●			●												
St-104													●			●			●	
St-105														●			●			●
St-106															●		●		●	
St-107												●		●			●			
St-108	●											●	●							
St-109		●	●	●																
St-110			●								●	●								
St-111											●	●	●							
St-112															●	●			●	
St-113																●			●	●
St-114														●	●	●				
St-115								●	●	●										
St-116	●		●		●															
St-117		●			●						●									

Öğrenci Adı	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20
St-118			●					●						●						
St-119					●						●						●			
St-120								●						●					●	
St-121											●						●			●
St-122														●					●	●
St-123	●																	●		●
St-124		●	●																●	
St-125	●		●					●												
St-126		●	●		●															
St-127	●				●									●						
St-128		●						●									●			
St-129											●			●			●			
St-130														●					●	●
St-131																	●		●	●
St-132				●	●			●												
St-133	●	●	●																	
St-134		●			●									●						
St-135	●		●			●														
St-136				●			●							●						
St-137		●		●		●														
St-138		●		●			●													
St-139		●		●	●															
St-140				●	●		●													
St-141	●				●		●													
St-142	●						●				●									
St-143	●										●		●							
St-144											●		●		●					
St-145													●					●	●	
St-146																		●	●	●
St-147		●			●					●										
St-148				●			●				●									
St-149					●					●			●							
St-150							●				●							●		
St-151										●			●						●	
St-152											●							●		●
St-153													●						●	●
St-154		●														●	●			
St-155				●	●											●				
St-156		●			●					●										
St-157				●	●		●													
St-158		●					●						●							
St-159				●						●					●					
St-160											●		●		●					
St-161													●			●	●			
St-162																		●	●	●
St-163	●						●			●										
St-164		●		●	●															
St-165				●						●								●		
St-166	●				●														●	
St-167							●									●				●
St-168	●			●			●													
St-169				●			●									●				
St-170		●		●	●															
St-171				●	●		●													
St-172					●		●			●										
St-173	●						●			●										
St-174	●									●						●				
St-175											●					●		●		
St-176																●		●	●	
St-177															●				●	●

Öğrenci Adı	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20
St-178		●			●					●										
St-179				●			●				●									
St-181					●		●		●											
St-182						●		●		●										
St-183	●						●		●											
St-184	●							●		●										
St-185	●	●								●										
St-186			●	●						●										
St-187		●		●		●														
St-188	●									●		●								
St-189									●					●			●			
St-190										●					●			●		
St-191																●	●	●		
St-192																	●		●	●
St-193																●	●	●		
St-194												●	●	●						
St-195	●								●	●										
St-196			●			●			●											
St-197				●			●			●										
St-198	●				●			●												
St-199			●		●		●													
St-200			●		●			●												
St-201	●	●	●																	
St-202		●	●	●																
St-203									●	●	●									
St-204											●			●	●					
St-205														●	●	●				
St-206																	●	●	●	
St-207																		●	●	●
St-208																	●	●	●	
St-209	●							●						●						
St-210									●		●				●					
St-211	●													●		●				
St-212					●		●		●											
St-213						●							●		●					
St-214							●							●		●				
St-215																●		●	●	
St-216		●													●	●				
St-217	●		●	●																
St-218		●		●		●														
St-219			●	●	●															
St-220	●	●											●							
St-221			●													●			●	
St-222																	●		●	●
St-223	●						●		●											
St-224	●							●	●											
St-225							●	●	●											
St-226	●		●				●													
St-227			●						●			●								
St-228							●			●			●							
St-229								●			●			●						
St-230			●							●		●								
St-231			●							●			●							
St-232	●		●						●											
St-233	●		●							●										
St-234	●			●	●															
St-235					●				●	●										
St-236													●	●	●					
St-237														●	●	●				
St-238															●	●			●	

Öğrenci Adı	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20
St-239																●			●	●
St-241			●		●									●						
St-242				●									●		●					
St-243					●									●		●				
St-244													●		●				●	
St-245										●		●				●				
St-246											●				●	●				
St-247	●														●	●				
St-248			●	●											●					
St-249			●	●											●					
St-250	●			●					●											
St-251			●	●	●															
St-252	●				●						●									
St-253			●						●						●					
St-254										●					●	●				
St-255																	●		●	●
St-256												●	●							●
St-257										●					●	●				
St-258	●		●						●											
St-259			●											●			●			
St-260									●			●			●					
St-261										●			●			●				
St-262			●							●		●								
St-263			●					●			●									
St-264	●		●				●													
St-265			●				●	●												
St-266							●	●	●											
St-267						●	●			●										
St-268							●			●	●									
St-269												●	●	●						
St-270													●				●	●		
St-271																	●		●	●
St-272	●				●		●													
St-273		●				●						●								
St-274					●		●						●							
St-275						●						●					●			
St-276							●						●					●		
St-277												●		●		●				
St-278											●		●	●						
St-279	●										●	●								
St-280		●			●						●									
St-281	●				●		●													
St-282		●			●	●														
St-283	●					●			●											
St-284		●					●			●										
St-285	●							●	●											
St-286									●		●									●
St-287	●										●									●
St-288						●	●	●												
St-289			●	●	●															
St-290	●			●			●													
St-291					●			●			●									
St-292						●			●											●
St-293				●		●		●												
St-294				●		●			●											
St-295	●	●	●																	
St-296		●					●	●												
St-297							●	●	●											
St-298	●							●	●											
St-299											●	●	●							

Öğrenci Adı	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20
St-300												●	●	●						
St-301													●	●	●					
St-302														●	●					●
St-303							●		●		●									
St-304	●							●				●								
St-305									●		●		●							
St-306	●											●		●						
St-307											●		●		●					
St-308	●							●												●
St-309									●		●									●
St-310			●															●		●
St-311				●								●						●		
St-312			●									●		●						
St-313				●								●	●							
St-314			●				●			●										
St-315	●			●				●												
St-316	●								●	●										
St-317										●								●		●
St-318	●																	●		●
St-319							●	●	●											
St-320			●	●		●														
St-321	●			●				●												
St-322						●			●									●		
St-323							●			●										●
St-324				●									●		●					
St-325				●									●			●				
St-326			●	●	●															
St-327							●	●	●											
St-328	●	●						●												
St-329	●	●									●									
St-330		●									●	●								
St-331											●	●						●		
St-332												●						●	●	
St-333																		●	●	●
St-334	●					●				●										
St-335							●		●		●									
St-336	●									●		●								
St-337									●		●							●		
St-338										●		●							●	
St-339											●							●		●
St-340												●							●	●
St-341											●								●	●
St-342												●	●						●	
St-343											●		●		●					
St-344												●	●	●						
St-345											●			●			●			
St-346	●			●														●		
St-347					●	●												●		

EK G: DMBO Algoritmasının Katılımcı Listesi

	P1			P2			P3			P4			P5		
#	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3
1	St-001	St-040	St-003	St-002	St-004	St-010	St-011	St-042	St-001	St-003	St-002	St-039	St-008	St-001	St-019
2	St-009	St-046	St-034	St-031	St-009	St-013	St-025	St-062	St-002	St-013	St-008	St-043	St-039	St-023	St-035
3	St-015	St-092	St-049	St-035	St-041	St-014	St-046	St-087	St-009	St-024	St-012	St-055	St-070	St-025	St-041
4	St-023	St-096	St-061	St-040	St-055	St-024	St-077	St-109	St-023	St-026	St-014	St-070	St-103	St-050	St-117
5	St-030	St-123	St-065	St-062	St-077	St-033	St-102	St-110	St-040	St-044	St-034	St-086	St-119	St-054	St-149
6	St-032	St-125	St-077	St-064	St-093	St-044	St-116	St-186	St-054	St-065	St-045	St-088	St-134	St-085	St-156
7	St-051	St-133	St-094	St-066	St-095	St-045	St-118	St-200	St-063	St-076	St-064	St-109	St-139	St-101	St-157
8	St-054	St-135	St-102	St-086	St-097	St-046	St-125	St-201	St-064	St-101	St-155	St-137	St-147	St-116	St-166
9	St-063	St-142	St-116	St-109	St-102	St-076	St-133	St-221	St-085	St-132	St-157	St-139	St-155	St-126	St-178
10	St-071	St-163	St-141	St-124	St-103	St-078	St-199	St-227	St-093	St-136	St-168	St-159	St-164	St-127	St-198
11	St-085	St-166	St-143	St-126	St-117	St-128	St-219	St-230	St-124	St-138	St-170	St-164	St-170	St-132	St-199
12	St-108	St-174	St-168	St-178	St-134	St-133	St-226	St-231	St-126	St-140	St-202	St-165	St-171	St-140	St-219
13	St-127	St-183	St-173	St-187	St-137	St-147	St-232	St-251	St-135	St-148	St-217	St-171	St-172	St-141	St-234
14	St-185	St-184	St-209	St-202	St-138	St-158	St-233	St-253	St-196	St-169	St-219	St-179	St-200	St-180	St-243
15	St-188	St-198	St-226	St-218	St-139	St-170	St-241	St-259	St-202	St-197	St-234	St-186	St-212	St-181	St-252
16	St-195	St-211	St-232	St-280	St-154	St-185	St-248	St-264	St-217	St-240	St-248	St-187	St-235	St-241	St-274
17	St-201	St-220	St-233	St-282	St-156	St-201	St-249	St-289	St-258	St-242	St-249	St-218	St-251	St-272	St-282
18	St-217	St-240	St-247	St-295	St-164	St-216	St-262	St-312	St-295	St-289	St-290	St-250	St-281	St-280	St-289
19	St-223	St-252	St-264	St-328	St-273	St-220	St-263	St-314	St-310	St-293	St-321	St-251	St-326	St-347	St-291
20	St-224	St-258	St-290	St-330	St-329	St-284	St-265	St-320	St-326	St-311	St-324	St-294			
21	St-234	St-279	St-298			St-296				St-320	St-326	St-313			
22	St-250	St-281	St-306							St-325	St-346	St-315			
23	St-272	St-283	St-308												
24	St-285	St-295	St-316												
25	St-287	St-304	St-321												
26	St-315	St-334	St-328												
27	St-318	St-336	St-346												
28	St-329														

	P6			P7			P8			P9			P10		
#	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3
1	St-019	St-024	St-008	St-016	St-003	St-015	St-010	St-015	St-048	St-033	St-013	St-011	St-014	St-048	St-012
2	St-042	St-036	St-020	St-045	St-043	St-031	St-041	St-016	St-069	St-034	St-018	St-016	St-028	St-051	St-018
3	St-055	St-039	St-026	St-050	St-076	St-032	St-047	St-017	St-103	St-036	St-032	St-017	St-049	St-078	St-079
4	St-135	St-044	St-050	St-091	St-136	St-047	St-084	St-038	St-115	St-048	St-047	St-027	St-059	St-084	St-092
5	St-137	St-070	St-059	St-141	St-148	St-068	St-182	St-059	St-118	St-078	St-049	St-037	St-068	St-091	St-151
6	St-267	St-182	St-101	St-150	St-158	St-138	St-198	St-120	St-125	St-092	St-068	St-038	St-069	St-115	St-180
7	St-273	St-187	St-213	St-157	St-171	St-140	St-229	St-128	St-132	St-093	St-069	St-042	St-149	St-147	St-182
8	St-275	St-196	St-283	St-168	St-197	St-142	St-266	St-209	St-184	St-115	St-079	St-084	St-156	St-165	St-188
9	St-288	St-218	St-292	St-173	St-199	St-163	St-296	St-224	St-200	St-196	St-210	St-091	St-159	St-172	St-190
10	St-294	St-282	St-293	St-179	St-212	St-167	St-297	St-225	St-263	St-203	St-223	St-181	St-163	St-173	St-197
11	St-334	St-322	St-320	St-181	St-214	St-169	St-298	St-265	St-285	St-227	St-232	St-183	St-174	St-178	St-203
12	St-347			St-183	St-226	St-172	St-304	St-291	St-288	St-258	St-235	St-189	St-184	St-185	St-231
13				St-225	St-228	St-223	St-308	St-293	St-319	St-283	St-250	St-195	St-186	St-195	St-235
14				St-264	St-266	St-265	St-321	St-315	St-327	St-286	St-285	St-212	St-228	St-233	St-245
15				St-274	St-267	St-268		St-328		St-292	St-294	St-224	St-230	St-257	St-261
16				St-276	St-288	St-272				St-303	St-297	St-225	St-254	St-268	St-262
17				St-284	St-296	St-281				St-305	St-298	St-253	St-314	St-284	St-267
18				St-290	St-303	St-297				St-309	St-319	St-260	St-317	St-316	St-334
19				St-319	St-323	St-314				St-316	St-335	St-266	St-323	St-338	St-336
20					St-327	St-335				St-327	St-337	St-322			

	P11			P12			P13			P14			P15		
#	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3
1	St-018	St-010	St-056	St-072	St-011	St-028	St-012	St-033	St-004	St-099	St-005	St-081	St-027	St-019	St-005
2	St-079	St-029	St-110	St-087	St-071	St-051	St-017	St-056	St-071	St-105	St-028	St-107	St-056	St-020	St-006
3	St-094	St-052	St-144	St-110	St-107	St-075	St-038	St-058	St-073	St-114	St-074	St-127	St-058	St-063	St-021
4	St-095	St-119	St-148	St-269	St-111	St-089	St-043	St-086	St-080	St-120	St-075	St-129	St-080	St-072	St-025
5	St-111	St-129	St-152	St-277	St-188	St-095	St-088	St-094	St-104	St-122	St-080	St-134	St-096	St-100	St-066
6	St-117	St-150	St-210	St-299	St-245	St-096	St-090	St-108	St-111	St-130	St-118	St-136	St-160	St-106	St-097
7	St-121	St-160	St-229	St-306	St-256	St-108	St-144	St-143	St-160	St-209	St-189	St-204	St-177	St-144	St-112
8	St-142	St-179	St-246	St-312	St-260	St-194	St-145	St-149	St-228	St-214	St-194	St-211	St-205	St-159	St-114
9	St-143	St-203	St-280	St-331	St-262	St-227	St-153	St-151	St-240	St-236	St-205	St-241	St-210	St-190	St-236
10	St-175	St-263	St-287	St-336	St-300	St-230	St-158	St-161	St-242	St-237	St-229	St-269	St-213	St-204	St-237
11	St-204	St-278	St-303	St-338	St-311	St-273	St-180	St-213	St-256	St-243	St-277	St-278	St-216	St-242	St-238
12	St-252	St-286	St-307	St-340	St-313	St-275	St-194	St-236	St-299	St-259	St-301	St-302	St-246	St-247	St-244
13	St-268	St-299	St-309	St-342	St-330	St-279	St-220	St-244	St-300	St-300	St-306	St-312	St-253	St-254	St-248
14	St-279	St-305	St-329	St-344	St-332	St-304	St-231	St-261	St-305	St-345		St-344	St-257	St-302	St-249
15	St-291	St-339	St-330				St-278	St-269	St-324				St-260	St-307	St-301
16	St-335	St-341	St-331				St-301	St-270	St-325				St-324	St-343	
17	St-343	St-345	St-337				St-307	St-274	St-342						
18							St-313	St-276	St-343						
19								St-344							

	P16			P17			P18			P19			P20		
#	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3
1	St-004	St-053	St-100	St-006	St-007	St-036	St-021	St-006	St-007	St-029	St-021	St-052	St-057	St-037	St-022
2	St-005	St-057	St-154	St-067	St-035	St-058	St-022	St-026	St-057	St-037	St-022	St-062	St-081	St-060	St-029
3	St-007	St-067	St-155	St-074	St-065	St-060	St-082	St-066	St-067	St-053	St-027	St-082	St-100	St-061	St-030
4	St-020	St-098	St-161	St-106	St-082	St-087	St-083	St-088	St-072	St-060	St-030	St-083	St-113	St-083	St-053
5	St-052	St-099	St-174	St-107	St-089	St-098	St-097	St-090	St-123	St-061	St-031	St-106	St-123	St-105	St-074
6	St-073	St-113	St-175	St-128	St-131	St-105	St-098	St-145	St-146	St-089	St-073	St-113	St-146	St-121	St-090
7	St-075	St-114	St-191	St-129	St-191	St-119	St-152	St-162	St-150	St-131	St-081	St-120	St-162	St-122	St-099
8	St-104	St-169	St-193	St-154	St-192	St-121	St-165	St-175	St-176	St-151	St-104	St-122	St-192	St-152	St-130
9	St-112	St-176	St-205	St-161	St-193	St-206	St-190	St-206	St-270	St-166	St-112	St-145	St-222	St-167	St-131
10	St-167	St-216	St-214	St-189	St-222	St-255	St-191	St-207	St-276	St-176	St-124	St-162	St-239	St-177	St-153
11	St-211	St-237	St-221	St-208	St-271	St-259	St-193	St-208	St-311	St-206	St-130	St-177	St-255	St-287	St-207
12	St-215	St-238	St-239	St-270	St-275	St-345	St-310	St-215	St-332	St-207	St-146	St-192	St-256	St-292	St-286
13	St-245	St-243	St-254				St-322	St-317	St-333	St-221	St-153	St-208	St-271	St-308	St-317
14	St-247	St-246	St-257				St-337	St-318	St-339	St-238	St-239	St-215	St-302	St-309	St-318
15	St-261	St-325	St-277				St-346	St-331	St-347	St-244	St-255	St-222	St-333	St-310	St-323
16										St-332	St-333	St-271	St-339	St-340	St-341
17										St-341	St-342	St-338			
18												St-340			

EK H: DABC Algoritmasının Katılımcı Listesi

	P1			P2			P3			P4			P5		
#	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3
1	St-003	St-001	St-030	St-009	St-002	St-010	St-001	St-040	St-002	St-002	St-003	St-008	St-023	St-008	St-001
2	St-015	St-009	St-034	St-024	St-004	St-013	St-011	St-063	St-009	St-026	St-013	St-012	St-035	St-050	St-019
3	St-046	St-023	St-040	St-033	St-031	St-014	St-025	St-064	St-023	St-088	St-014	St-039	St-039	St-085	St-025
4	St-049	St-032	St-071	St-040	St-035	St-062	St-042	St-087	St-046	St-137	St-024	St-055	St-116	St-101	St-041
5	St-061	St-051	St-077	St-041	St-046	St-066	St-054	St-102	St-085	St-157	St-034	St-064	St-117	St-103	St-054
6	St-063	St-054	St-094	St-044	St-076	St-095	St-062	St-116	St-093	St-164	St-043	St-065	St-119	St-127	St-070
7	St-065	St-096	St-102	St-045	St-077	St-117	St-077	St-126	St-109	St-169	St-044	St-076	St-126	St-149	St-141
8	St-085	St-125	St-108	St-055	St-078	St-126	St-110	St-135	St-124	St-171	St-045	St-086	St-132	St-155	St-156
9	St-092	St-133	St-116	St-064	St-086	St-128	St-118	St-186	St-196	St-187	St-070	St-101	St-134	St-164	St-157
10	St-127	St-142	St-123	St-093	St-097	St-133	St-125	St-200	St-199	St-197	St-109	St-132	St-139	St-171	St-166
11	St-141	St-166	St-135	St-102	St-124	St-134	St-133	St-201	St-202	St-202	St-136	St-138	St-140	St-180	St-172
12	St-168	St-183	St-143	St-103	St-138	St-137	St-221	St-219	St-217	St-217	St-139	St-148	St-147	St-199	St-181
13	St-173	St-184	St-163	St-109	St-170	St-139	St-230	St-249	St-226	St-219	St-140	St-155	St-170	St-212	St-198
14	St-185	St-195	St-174	St-156	St-178	St-147	St-232	St-253	St-227	St-240	St-165	St-159	St-178	St-234	St-200
15	St-198	St-211	St-188	St-158	St-185	St-154	St-241	St-259	St-231	St-249	St-218	St-168	St-235	St-241	St-219
16	St-220	St-217	St-201	St-201	St-187	St-164	St-258	St-262	St-233	St-250	St-242	St-170	St-243	St-251	St-272
17	St-223	St-224	St-209	St-273	St-202	St-216	St-263	St-310	St-248	St-251	St-248	St-179	St-252	St-274	St-282
18	St-226	St-233	St-232	St-328	St-282	St-218	St-265	St-312	St-251	St-294	St-289	St-186	St-281	St-280	St-291
19	St-234	St-250	St-240	St-329	St-295	St-220	St-295	St-320	St-264	St-320	St-290	St-234	St-289	St-347	St-326
20	St-272	St-252	St-247	St-330	St-296	St-280	St-314	St-326	St-289	St-321	St-293	St-313			
21	St-283	St-258	St-279			St-284				St-325	St-311	St-315			
22	St-306	St-264	St-290							St-326	St-346	St-324			
23	St-308	St-281	St-295												
24	St-315	St-285	St-318												
25	St-316	St-287	St-321												
26	St-334	St-298	St-329												
27	St-336	St-304	St-346												
28		St-328													
	P6			P7			P8			P9			P10		
#	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3
1	St-008	St-019	St-024	St-031	St-015	St-003	St-010	St-041	St-015	St-013	St-018	St-011	St-014	St-012	St-018
2	St-020	St-026	St-042	St-043	St-016	St-032	St-016	St-084	St-038	St-032	St-027	St-016	St-028	St-059	St-048
3	St-036	St-039	St-044	St-068	St-141	St-045	St-017	St-115	St-103	St-034	St-033	St-017	St-051	St-068	St-049
4	St-050	St-055	St-059	St-076	St-148	St-047	St-047	St-120	St-118	St-038	St-036	St-068	St-151	St-069	St-078
5	St-070	St-137	St-187	St-091	St-150	St-050	St-048	St-128	St-125	St-078	St-037	St-069	St-159	St-079	St-084
6	St-101	St-182	St-288	St-138	St-157	St-136	St-059	St-132	St-182	St-079	St-042	St-092	St-163	St-092	St-091
7	St-135	St-213	St-292	St-167	St-158	St-140	St-069	St-198	St-184	St-084	St-047	St-195	St-165	St-147	St-115
8	St-196	St-267	St-293	St-172	St-163	St-142	St-200	St-225	St-224	St-115	St-048	St-212	St-182	St-156	St-149
9	St-218	St-273	St-320	St-179	St-168	St-171	St-209	St-263	St-229	St-183	St-049	St-225	St-184	St-172	St-173
10	St-275	St-283	St-322	St-181	St-169	St-183	St-285	St-265	St-266	St-210	St-091	St-235	St-186	St-174	St-178
11	St-282	St-294	St-347	St-199	St-173	St-197	St-288	St-291	St-296	St-224	St-093	St-250	St-190	St-188	St-180
12		St-334		St-212	St-214	St-223	St-293	St-308	St-297	St-227	St-181	St-258	St-195	St-197	St-185
13				St-225	St-226	St-228	St-304	St-315	St-298	St-253	St-189	St-260	St-203	St-231	St-230
14				St-264	St-266	St-265	St-319	St-321	St-328	St-266	St-196	St-283	St-228	St-235	St-261
15				St-267	St-272	St-268		St-327		St-297	St-203	St-285	St-233	St-245	St-267
16				St-274	St-284	St-276				St-298	St-223	St-286	St-262	St-254	St-314
17				St-290	St-288	St-281				St-309	St-232	St-294	St-268	St-257	St-317
18				St-296	St-297	St-319				St-322	St-292	St-303	St-284	St-316	St-334
19				St-303	St-314	St-327				St-327	St-305	St-316	St-323	St-336	St-338
20				St-335	St-323					St-337	St-319	St-335			

	P11			P12			P13			P14			P15		
#	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3
1	St-018	St-010	St-079	St-087	St-011	St-051	St-012	St-017	St-004	St-075	St-105	St-005	St-005	St-006	St-020
2	St-056	St-029	St-111	St-108	St-028	St-075	St-071	St-038	St-033	St-080	St-118	St-028	St-019	St-025	St-027
3	St-094	St-052	St-119	St-111	St-071	St-089	St-073	St-080	St-043	St-081	St-129	St-074	St-021	St-056	St-063
4	St-095	St-110	St-144	St-188	St-072	St-096	St-086	St-094	St-056	St-099	St-130	St-114	St-072	St-058	St-080
5	St-121	St-117	St-150	St-245	St-095	St-107	St-143	St-108	St-058	St-107	St-134	St-122	St-096	St-066	St-097
6	St-129	St-143	St-175	St-260	St-194	St-110	St-149	St-111	St-088	St-120	St-204	St-127	St-106	St-100	St-177
7	St-142	St-152	St-203	St-277	St-227	St-262	St-153	St-144	St-090	St-136	St-209	St-194	St-112	St-114	St-190
8	St-148	St-160	St-210	St-279	St-230	St-273	St-180	St-151	St-104	St-189	St-229	St-211	St-144	St-159	St-204
9	St-204	St-179	St-246	St-299	St-256	St-275	St-194	St-220	St-145	St-205	St-237	St-241	St-160	St-205	St-236
10	St-229	St-268	St-252	St-311	St-269	St-300	St-231	St-228	St-158	St-214	St-277	St-243	St-213	St-210	St-242
11	St-280	St-279	St-263	St-312	St-306	St-304	St-242	St-236	St-160	St-236	St-278	St-259	St-237	St-216	St-249
12	St-291	St-286	St-278	St-313	St-330	St-332	St-244	St-240	St-161	St-269	St-301	St-306	St-247	St-238	St-253
13	St-305	St-299	St-287	St-338	St-331	St-336	St-256	St-300	St-213	St-300	St-302	St-312	St-248	St-244	St-257
14	St-307	St-303	St-309	St-342	St-344	St-340	St-261	St-307	St-269	St-344		St-345	St-254	St-246	St-301
15	St-339	St-329	St-330				St-270	St-313	St-274				St-302	St-260	St-307
16	St-343	St-335	St-331				St-276	St-324	St-299				St-324		St-343
17	St-345	St-337	St-341				St-278	St-342	St-305						
18							St-301	St-343	St-325						
19									St-344						

	P16			P17			P18			P19			P20		
#	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3
1	St-004	St-005	St-052	St-006	St-065	St-007	St-007	St-022	St-006	St-022	St-021	St-029	St-029	St-030	St-022
2	St-053	St-007	St-067	St-058	St-067	St-035	St-057	St-088	St-021	St-027	St-061	St-031	St-083	St-053	St-037
3	St-100	St-020	St-098	St-060	St-082	St-036	St-066	St-090	St-026	St-030	St-062	St-053	St-090	St-060	St-057
4	St-113	St-057	St-099	St-074	St-107	St-087	St-067	St-098	St-072	St-037	St-083	St-060	St-122	St-074	St-061
5	St-114	St-073	St-112	St-089	St-119	St-106	St-097	St-145	St-082	St-052	St-089	St-073	St-130	St-081	St-100
6	St-154	St-075	St-167	St-098	St-154	St-121	St-123	St-162	St-083	St-082	St-106	St-081	St-152	St-099	St-105
7	St-155	St-104	St-169	St-105	St-192	St-129	St-150	St-175	St-146	St-104	St-112	St-120	St-162	St-121	St-113
8	St-174	St-161	St-205	St-128	St-255	St-189	St-193	St-190	St-152	St-124	St-113	St-130	St-177	St-123	St-153
9	St-175	St-193	St-214	St-131	St-270	St-191	St-208	St-191	St-165	St-145	St-122	St-131	St-222	St-131	St-192
10	St-176	St-215	St-221	St-161	St-271	St-193	St-317	St-206	St-176	St-146	St-153	St-151	St-239	St-146	St-207
11	St-191	St-239	St-237	St-206	St-275	St-208	St-318	St-207	St-215	St-166	St-176	St-162	St-271	St-167	St-255
12	St-211	St-243	St-238	St-259	St-345	St-222	St-331	St-276	St-270	St-192	St-177	St-206	St-286	St-309	St-256
13	St-216	St-247	St-245				St-332	St-322	St-310	St-207	St-208	St-239	St-287	St-317	St-302
14	St-246	St-261	St-254				St-346	St-333	St-311	St-215	St-221	St-244	St-292	St-318	St-308
15	St-257	St-325	St-277				St-347	St-339	St-337	St-238	St-222	St-271	St-310	St-340	St-323
16										St-255	St-332	St-333	St-333	St-341	St-339
17										St-340	St-338	St-342			
18										St-341					

EK I: DGA Algoritmasının Katılımcı Listesi

	P1			P2			P3			P4			P5		
#	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3
1	St-001	St-003	St-009	St-002	St-009	St-004	St-009	St-002	St-001	St-012	St-024	St-002	St-008	St-001	St-025
2	St-015	St-032	St-023	St-024	St-010	St-013	St-023	St-042	St-011	St-013	St-045	St-003	St-035	St-019	St-041
3	St-049	St-034	St-030	St-040	St-062	St-014	St-025	St-046	St-040	St-014	St-055	St-008	St-039	St-023	St-050
4	St-061	St-040	St-046	St-041	St-064	St-031	St-054	St-063	St-064	St-034	St-070	St-026	St-116	St-054	St-070
5	St-063	St-065	St-051	St-045	St-078	St-033	St-062	St-087	St-093	St-043	St-076	St-039	St-127	St-117	St-085
6	St-071	St-077	St-054	St-046	St-086	St-035	St-077	St-116	St-102	St-044	St-109	St-065	St-140	St-134	St-101
7	St-102	St-085	St-094	St-066	St-102	St-044	St-085	St-118	St-110	St-064	St-136	St-086	St-155	St-139	St-103
8	St-123	St-092	St-116	St-093	St-124	St-055	St-109	St-125	St-124	St-101	St-140	St-088	St-156	St-141	St-119
9	St-125	St-096	St-141	St-095	St-133	St-076	St-186	St-126	St-133	St-132	St-164	St-137	St-157	St-166	St-126
10	St-133	St-108	St-163	St-097	St-137	St-077	St-200	St-135	St-227	St-148	St-165	St-138	St-164	St-180	St-132
11	St-135	St-127	St-166	St-103	St-138	St-109	St-217	St-196	St-231	St-159	St-169	St-139	St-171	St-181	St-147
12	St-143	St-142	St-173	St-126	St-147	St-117	St-219	St-199	St-232	St-170	St-171	St-155	St-172	St-200	St-149
13	St-174	St-168	St-184	St-139	St-154	St-128	St-221	St-201	St-233	St-179	St-186	St-157	St-178	St-212	St-170
14	St-188	St-183	St-185	St-158	St-156	St-134	St-248	St-202	St-251	St-187	St-217	St-168	St-198	St-272	St-219
15	St-201	St-209	St-195	St-185	St-170	St-164	St-249	St-226	St-253	St-242	St-218	St-197	St-199	St-280	St-234
16	St-223	St-224	St-198	St-202	St-178	St-201	St-258	St-230	St-259	St-250	St-219	St-202	St-235	St-281	St-241
17	St-226	St-232	St-211	St-216	St-187	St-218	St-262	St-241	St-264	St-289	St-234	St-248	St-243	St-282	St-252
18	St-233	St-247	St-217	St-220	St-284	St-282	St-263	St-265	St-289	St-290	St-240	St-249	St-251	St-289	St-274
19	St-234	St-258	St-220	St-273	St-296	St-295	St-312	St-295	St-310	St-313	St-251	St-293	St-347	St-291	St-326
20	St-252	St-264	St-240	St-280	St-330	St-329	St-326	St-320	St-314	St-315	St-311	St-294			
21	St-272	St-287	St-250	St-328						St-320	St-325	St-324			
22	St-281	St-298	St-279							St-321	St-326	St-346			
23	St-283	St-315	St-290												
24	St-285	St-316	St-304												
25	St-295	St-321	St-318												
26	St-306	St-329	St-328												
27	St-308	St-334	St-336												
28		St-346													

	P6			P7			P8			P9			P10		
#	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3
1	St-026	St-008	St-019	St-003	St-015	St-016	St-016	St-017	St-010	St-011	St-013	St-017	St-068	St-014	St-012
2	St-036	St-039	St-020	St-031	St-043	St-045	St-059	St-038	St-015	St-018	St-016	St-032	St-078	St-018	St-048
3	St-050	St-044	St-024	St-032	St-047	St-091	St-069	St-041	St-047	St-033	St-027	St-034	St-091	St-028	St-059
4	St-055	St-059	St-042	St-076	St-050	St-136	St-084	St-048	St-125	St-042	St-036	St-038	St-115	St-049	St-079
5	St-070	St-101	St-135	St-138	St-068	St-140	St-118	St-103	St-200	St-047	St-037	St-049	St-147	St-051	St-084
6	St-137	St-182	St-187	St-141	St-148	St-167	St-120	St-115	St-224	St-048	St-084	St-068	St-163	St-069	St-092
7	St-213	St-273	St-196	St-142	St-157	St-169	St-128	St-132	St-265	St-079	St-091	St-069	St-165	St-149	St-151
8	St-218	St-283	St-292	St-150	St-158	St-171	St-182	St-198	St-266	St-092	St-093	St-078	St-180	St-173	St-156
9	St-267	St-293	St-320	St-168	St-163	St-181	St-184	St-225	St-285	St-181	St-195	St-115	St-190	St-174	St-159
10	St-275	St-294	St-334	St-173	St-172	St-183	St-209	St-263	St-288	St-183	St-210	St-189	St-195	St-184	St-172
11	St-282	St-322	St-347	St-212	St-179	St-199	St-229	St-308	St-291	St-196	St-235	St-203	St-197	St-185	St-178
12	St-288			St-225	St-197	St-226	St-293	St-319	St-296	St-224	St-250	St-212	St-203	St-228	St-182
13				St-264	St-214	St-228	St-297	St-327	St-315	St-227	St-266	St-223	St-231	St-233	St-186
14				St-265	St-223	St-267	St-298	St-328	St-321	St-232	St-285	St-225	St-261	St-254	St-188
15				St-266	St-268	St-272	St-304			St-253	St-286	St-258	St-268	St-257	St-230
16				St-274	St-288	St-281				St-294	St-292	St-260	St-284	St-262	St-235
17				St-276	St-290	St-284				St-309	St-297	St-283	St-317	St-267	St-245
18				St-296	St-323	St-297				St-316	St-303	St-298	St-323	St-314	St-316
19				St-303	St-335	St-319				St-319	St-305	St-322	St-334	St-336	St-338
20				St-314		St-327				St-327	St-337	St-335			

	P11			P12			P13			P14			P15		
#	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3
1	St-010	St-052	St-018	St-051	St-011	St-028	St-004	St-012	St-043	St-028	St-005	St-075	St-019	St-006	St-005
2	St-029	St-056	St-142	St-075	St-072	St-071	St-017	St-033	St-056	St-081	St-074	St-099	St-020	St-025	St-027
3	St-094	St-079	St-143	St-108	St-110	St-087	St-038	St-071	St-073	St-122	St-080	St-118	St-021	St-058	St-063
4	St-110	St-095	St-148	St-230	St-111	St-089	St-058	St-094	St-090	St-130	St-105	St-127	St-056	St-066	St-072
5	St-111	St-119	St-152	St-256	St-188	St-095	St-080	St-143	St-108	St-134	St-107	St-129	St-096	St-097	St-080
6	St-117	St-144	St-179	St-260	St-194	St-096	St-086	St-161	St-111	St-136	St-114	St-194	St-112	St-106	St-100
7	St-121	St-150	St-263	St-279	St-227	St-107	St-088	St-220	St-144	St-189	St-120	St-204	St-114	St-159	St-160
8	St-129	St-160	St-268	St-299	St-245	St-262	St-104	St-231	St-158	St-205	St-211	St-209	St-144	St-204	St-177
9	St-204	St-175	St-280	St-300	St-275	St-269	St-145	St-236	St-180	St-214	St-259	St-229	St-237	St-213	St-190
10	St-210	St-203	St-286	St-311	St-277	St-273	St-149	St-261	St-213	St-236	St-278	St-237	St-244	St-238	St-205
11	St-278	St-229	St-299	St-330	St-304	St-312	St-151	St-269	St-242	St-241	St-300	St-243	St-247	St-242	St-210
12	St-287	St-246	St-303	St-331	St-306	St-313	St-153	St-270	St-244	St-269	St-302	St-301	St-254	St-248	St-216
13	St-291	St-252	St-330	St-336	St-332	St-340	St-160	St-274	St-256	St-277	St-312	St-306	St-307	St-249	St-236
14	St-305	St-279	St-331	St-338	St-344	St-342	St-194	St-276	St-278		St-345	St-344	St-324	St-253	St-246
15	St-329	St-307	St-339				St-228	St-299	St-300				St-343	St-260	St-257
16	St-335	St-309	St-341				St-240	St-313	St-305					St-301	St-302
17	St-337	St-343	St-345				St-301	St-324	St-307						
18							St-344	St-342	St-325						
19									St-343						

	P16			P17			P18			P19			P20		
#	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3	S 1	S 2	S 3
1	St-005	St-004	St-053	St-065	St-035	St-006	St-006	St-026	St-021	St-027	St-021	St-052	St-030	St-053	St-022
2	St-052	St-007	St-067	St-067	St-089	St-007	St-007	St-067	St-057	St-037	St-022	St-062	St-057	St-100	St-029
3	St-073	St-020	St-104	St-087	St-121	St-036	St-022	St-083	St-066	St-053	St-029	St-112	St-060	St-113	St-037
4	St-100	St-057	St-113	St-098	St-128	St-058	St-072	St-088	St-097	St-083	St-030	St-120	St-074	St-122	St-061
5	St-161	St-075	St-114	St-105	St-129	St-060	St-082	St-090	St-098	St-089	St-031	St-122	St-090	St-130	St-081
6	St-167	St-098	St-154	St-107	St-131	St-074	St-152	St-123	St-150	St-106	St-060	St-130	St-099	St-152	St-083
7	St-169	St-099	St-174	St-119	St-189	St-082	St-176	St-145	St-165	St-113	St-061	St-131	St-131	St-153	St-105
8	St-175	St-112	St-176	St-154	St-191	St-106	St-191	St-146	St-175	St-124	St-073	St-145	St-162	St-167	St-121
9	St-193	St-155	St-191	St-208	St-193	St-161	St-206	St-162	St-193	St-146	St-081	St-153	St-177	St-192	St-123
10	St-211	St-205	St-214	St-255	St-206	St-192	St-270	St-190	St-208	St-166	St-082	St-162	St-222	St-255	St-146
11	St-238	St-216	St-215	St-259	St-222	St-270	St-310	St-207	St-276	St-192	St-104	St-206	St-271	St-256	St-207
12	St-245	St-221	St-247	St-345	St-271	St-275	St-322	St-215	St-311	St-207	St-151	St-221	St-286	St-310	St-239
13	St-246	St-237	St-254				St-333	St-318	St-317	St-215	St-176	St-222	St-292	St-317	St-287
14	St-257	St-239	St-261				St-339	St-331	St-332	St-239	St-177	St-238	St-302	St-333	St-308
15	St-325	St-243	St-277				St-346	St-347	St-337	St-332	St-208	St-255	St-318	St-339	St-309
16										St-340	St-244	St-271	St-341	St-340	St-323
17										St-342	St-338	St-333			
18											St-341				

ÖZGEÇMİŞ

1972 yılında Sivas'ta doğdu. 1983 yılında Sivas Süleyman Sami Kepenek İlkokulu'ndan, 1986 yılında Sivas Danişmentgazi Ortaokulu'ndan ve 1989 yılında Sivas Lisesi'nden mezun oldu. Aynı yıl Hacettepe Üniversitesi Mühendislik Fakültesi Elektrik ve Elektronik Mühendisliği Bölümü'nü kazandı. Buradan 1994 yılında elektronik mühendisi olarak mezun oldu. Aynı yıl Cumhuriyet Üniversitesi Sivas Meslek Yüksek Okulu Kontrol Sistemleri Teknolojisi Bölümü'nde öğretim görevlisi olarak çalışmaya başladı. 1995 yılında Erciyes Üniversitesi Fen Bilimleri Enstitüsü Elektronik Mühendisliği Anabilim Dalı'nda yüksek lisans eğitimine başladı. 1998 yılında buradan mezun oldu. 1998-2000 yılları arasında askerlik görevini Ankara Elektro Optik Sistemler Bakım Merkezi Müdürlüğü'nde yedek subay olarak yerine getirdi. 2000 yılında Ereğli Demir ve Çelik Fabrikaları T.A.Ş.'de (Erdemir) ileri seviye otomasyon sistemleri (seviye 2) yazılım mühendisi olarak göreve başladı. 2005 yılından aynı şirkette başmühendis pozisyonuna atandı. Halen bu görevine devam etmektedir. Erdemir'deki görev süresi içerisinde çelik üretim sürecinde yer alan kükürt giderme tesisleri, çelikhane konvertörleri, ikincil metalürji istasyonları, sürekli döküm tesisleri ve slab fırınları gibi çeşitli tesislerin seviye 2 sistemlerinin modernizasyon ve devreye alma projelerinde mühendis/yönetici görevleri ile çalıştı. 2011 yılında Karabük Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Bölümü'nde doktora eğitimine başladı. Aynı yıl Sakarya Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar ve Bilişim Mühendisliği Bölümü'ne yatay geçiş yaptı ve doktora eğitimine burada devam etti. Doktora eğitimi süresince üzerinde çalıştığı tez konusu ile ilgili bilimsel makaleler ve sempozyum bildirileri hazırlayıp sundu.