T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ATÖLYE TİPİ ÇİZELGELEME PROBLEMLERİNDE EVRİMSEL ALGORİTMALAR İLE YAPAY ARI KOLONİSİ ALGORİTMASININ BÜTÜNLEŞİK BİR YAKLAŞIMI

DOKTORA TEZİ

Mümin ÖZCAN

Enstitü Anabilim Dalı : ENDÜSTRİ MÜHENDİSLİĞİ

Tez Danışmanı : Prof. Dr. İsmail Hakkı CEDİMOĞLU

Nisan 2016

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ATÖLYE TİPİ ÇİZELGELEME PROBLEMLERİNDE EVRİMSEL ALGORİTMALAR İLE YAPAY ARI KOLONİSİ ALGORİTMASININ BÜTÜNLEŞİK BİR YAKLAŞIMI

DOKTORA TEZİ

Mümin ÖZCAN

Enstitü Anabilim Dalı : ENDÜSTRİ MÜHENDİSLİĞİ

Bu tez 5 / 4 / 2016 tarihinde aşağıdaki jüri tarafından oybirliği/~~oyçokluğu~~ ile kabul edilmiştir.

Prof. Dr. MÜZEFFER KAPANOĞLU
Jüri Başkanı

Prof. Dr. Prof. Dr. İlhami ÇEDİMOĞLU
Üye

Prof. Dr. Ali Fuat Erdinç
Üye

Prof. Dr. Orhan TORKUL
Üye

Prof. Dr. Nejat YUMUŞAK
Üye

BEYAN

Tez içindeki tüm verilerin akademik kurallar çerçevesinde tarafımdan elde edildiğini, görsel ve yazılı tüm bilgi ve sonuçların akademik ve etik kurallara uygun şekilde sunulduğunu, kullanılan verilerde herhangi bir tahrifat yapılmadığını, başkalarının eserlerinden yararlanılması durumunda bilimsel normlara uygun olarak atıfta bulunulduğunu, tezde yer alan verilerin bu üniversite veya başka bir üniversitede herhangi bir tez çalışmasında kullanılmadığını beyan ederim.

Mümin ÖZCAN

05.04.2016

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. İsmail Hakkı Cedimoğlu'na ve tez izleme komitemde yer alan değerli hocalarım Prof. Dr. Orhan Torkul ve Prof. Dr. Nejat Yumuşak'a, teşekkürü bir borç bilirim.

Tez çalışmamın düzeltme kısmında değerli fikirleriyle destek olan hocalarım Prof. Dr. Alptekin Erkollar ve Prof. Dr. Muzaffer Kapanoğlu'na çok teşekkür ederim. Bu süreç boyunca maddi ve manevi her türlü desteğini benden hiç esirgemeyen çok kıymetli hocam Prof. Dr. Mustafa Güneş'e, çalışmamın yazılım kısmında desteğini esirgemeyen çok değerli çalışma arkadaşım Yrd. Doç. Dr. Yavuz İnce'ye, uzun ve zahmetli doktora süreci boyunca desteklerini ve güvenini benden esirgemeyen sevgili eşim ve aileme çok teşekkür ederim.

İÇİNDEKİLER

SİMGELER VE KISALTMALAR LİSTESİ

ABC : Yapay arı kolonisi (Artificial Bee Colony)

ACO : Karınca kolonisi optimizasyonu (Ant Colony Optimization)

ACROA : Yapay kimyasal reaksiyon algoritması (Artificial Chemical Reaction Optimization Algorithm)

AIS : Yapay bağışıklık sistemi (Artificial Immune System)

ANN : Yapay sinir ağları (Artificial Neural Networks)

ANNGaT : Evrimsel Yapay Sinir Ağı Üretimi ve Eğitim

ARPD : Ortalama bağıl sapma yüzdesi (Average Relative Percentage Deviation)

ARPE : Ortalama bağıl hata yüzdesi (Average Relative Percentage Error)

BB : Dal sınır algoritması (Branch and Bound Algorithms)

BFO : Bakteriyel beslenme optimizasyonu (Bacterial Foraging Optimization)

BP : Geriye yayılım (Back Propagation)

BR : Esneme oranı (Bulge Ratio)

C_{max} : İşlerin toplam tamamlanma zamanı (makespan)

COCOMO : Yapıcı maliyet modeli (Constructive Cost Model)

CS : Guguk kuşu arama (Cuckoo Search)

CSO : Kedi sürüsü optimizasyonu (Cat Swarm Optimization)

DE : Diferansiyel gelişim (Differential Evolution)

DP : Dinamik programlama (Dynamic Programming)

EA : Evrimsel algoritmalar (Evolutionary Algorithms)

EANN : Evrimsel yapay sinir ağları (Evolutionary Neural Networks)

ECPSO : Evrimsel kanonik parçacık sürüsü optimizasyonu (Evolutionary Canonic Particle Swarm Optimization)

EPSO	: Evrimsel parçacık sürüsü optimizasyonu (Evolutionary Particle Swarm Optimization
FA	: Ateşböceği algoritması (Firefly Algorithm)
FS	: Balık sürüsü (Fish Swarm)
FSS	: Akış tipi çizelgeleme (Flow Shop Scheduling)
GA	: Genetik algoritmalar (Genetic Algorithms)
GS	: Yerçekimi arama (Gravitational Search)
HS	: Armoni arama (Harmony Search)
ICA	: Emperyalist yarışmacı algoritma (Imperialist Competitive Algorithm)
IP	: Tamsayılı programlama (Integer Programming)
IWD	: Zeki su damlacıkları (Intelligent Water Drops)
KA	: Kanguru algoritması (Kangaroo Algorithms)
KH	: Kril sürüsü (Krill Herd)
JSSP	: Atölye tipi çizelgeleme problemleri (Job Shop Scheduling Problem)
LM	: Levenberg-Marquardt
MAPE	: Ortalama mutlak yüzde hata (Mean Absolute Percentage Error)
ME	: Memetik evrim (Memetic Evolutionary)
POA	: Parlamenter optimizasyon algoritması (Parliamentary Optimization Algorithm)
PSO	: Parçacık sürüsü optimizasyonu (Particle Swarm Optimization)
PSP	: Protein yapısı tahminlemesi (Protein Structure Prediction)
PSS	: Güç sistemi dengeleyicisi (Power System Stabilizer)
SA	: Tavlama benzetim (Simulated Annealing)
THP	: Boruda hidrolik presleme (Tube Hydroforming Process)
TR	: İncelme oranı (Thinning Ratio)
TS	: Tabu araştırma (Tabu Search)
VNS	: Değişken komşuluklu arama (Variable Neighborhood Search)
WFA	: Kurt Kolonisi Algoritması (Wolf Colony Algorithm)

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Anahtar kelimeler: Sezgisel algoritmalar, atölye tipi çizelgeleme problemleri, evrimsel algoritmalar, yapay arı kolonisi algoritması, ortalama bağıl hata yüzdesi, ortalama bağıl sapma yüzdesi.

Hayatımızın birçok alanında çok önemli bir yeri olan çizelgeleme problemlerinin çözümü ile ilgili olarak yıllardır çok ciddi çalışmalar yapılmıştır. Bu çalışmaların yapılmasında şüphesiz en büyük sebep, mevcut çizelgeye göre daha iyilerinin geliştirilmesini sağlamaya çalışmak ve daha büyük kazanç ve verimlilikler ortaya koymaktır. Bundan dolayı, doğru ve etkin bir çizelgeleme, hem insanlar hem de işletmeler için büyük önem arz etmektedir. Bu bağlamda özellikle son yıllarda çizelgeleme problemlerinin çözümünde sezgisel algoritmaların araştırmacılar tarafından yoğun bir biçimde kullanıldığı görülmektedir.

Bu tez çalışmasında, atölye tipi çizelgeleme problemlerinin çözümünün eniyilemesi için bütünleşik bir yaklaşım önerilmiştir. Bu bağlamda sürü zekâsına dayalı sezgisel algoritmalardan olan yapay arı kolonisi algoritması ile evrimsel algoritmalar bütünleşik yaklaşım için kullanılmıştır. Önerilen metot, atölye tipi çizelgeleme ile ilgili data setlerine uygulanmış ve elde edilen sonuçlar ortalama bağıl hata yüzdesi (ARPE) ile ortalama bağıl sapma yüzdesi (ARPD) kriterleri kullanılarak, karınca kolonisi optimizasyon (ACO) tekniği, kuş sürüsü algoritması (PSO) ve diferansiyel gelişim (DE) algoritması ile kıyaslanmıştır. Sonuçlar kıyaslanırken, parametrik ve parametrik olmayan testler kullanılarak metotlar arasında istatistiksel olarak anlamlı farklar olup olmadığı kurulan hipotezlerle araştırılmıştır. ARPE kriterine göre, önerilen yaklaşım ile ACO tekniği sonuçları arasında istatistiksel olarak anlamlı farklar gözlemlenirken, önerilen metot ile PSO ve DE algoritmalarının sonuçları arasında ise istatistiksel olarak anlamlı farklar olmadığı görülmüştür. Yapılan testler sonucunda, önerilen metot ile elde edilen ARPE değeri, ACO metodu ile elde edilen ARPE değerinden 4,3 puan (yüzdesel değişim olarak) daha düşük olduğundan daha etkin bir netice vermiştir. ARPD kriterine göre ise, önerilen yaklaşım ile diğer tüm algoritmaların sonuçları arasında istatistiksel olarak anlamlı farklar olduğu yapılan testlerle ortaya konmuştur. Yapılan testler sonucunda, önerilen metot ile elde edilen ARPD değeri, ACO metodu ile elde edilen ARPD değerinden 6,3 puan, PSO metodu ile elde edilen ARPD değerinden 0,6 puan, DE metodu ile elde edilen ARPD değerinden ise 0,7 puan daha düşük olduğundan daha kararlı ve etkin neticeler vermiştir. Yapılan testler sonucunda, çizelgelemesi yapılacak olan iş veya makine sayısının 20 ve 20'den az olduğu durumlarda önerilen metodun çok daha hızlı ve etkin sonuçlar verdiği gözlemlenmiştir.

AN INTEGRATED APPROACH OF EVOLUTIONARY ALGORITHMS WITH ARTIFICIAL BEE COLONY ALGORITHM FOR JOB SHOP SCHEDULING PROBLEMS

SUMMARY

Keywords: Heuristic algorithms, job shop scheduling problems, evolutionary algorithms, artificial bee colony algorithm, average relative percentage error, average relative percentage deviation.

There have been a lot of research made about solution of scheduling problems that have a very important place in many areas of our lives for years. The cause of these researches is to develop better than the current schedule and achieve greater profits. Therefore, there is great importance of efficient scheduling for both humans and businesses. In this context, heuristic algorithms are used extensively by researchers for solving scheduling problems in recent years.

In this dissertation study, an integrated approach has been developed for optimizing the solution of job shop scheduling problems. In this context, artificial bee colony algorithm and evolutionary algorithms are used for the integrated approach. The proposed hybrid method has been applied to data sets related to job shop scheduling. The obtained results have been compared with the results of different optimization techniques that these techniques are ant colony optimization (ACO), particle swarm optimization (PSO) and differential evolution algorithm (DE) using the average relative error percentage (ARPE) and average relative percentage deviation (ARPD) criteria. It has investigated whether statistically significant differences among methods using parametric and non-parametric tests with the founded hypotheses for the comparisons. According to the ARPE criterion, statistically significant differences have been obtained between the results of the recommended approach and ACO technique. According to the same criterion, statistically significant differences have not been observed between the result of the proposed method with PSO and DE algorithms. ARPE value of the recommended approach yielded 4.3 points (as percentage changes) more effective than ARPE value of the ACO technique according to the results of the tests. According to the ARPD criterion, statistically significant differences have been obtained between the results of the recommended approach and other all techniques. According to the results of the tests, ARPD value of the proposed method yielded more effective and stable of 6.3 points than ARPE value of the ACO technique, of 0.6 points than ARPE value of the PSO algorithm, of 0.7 points than ARPE value of the DE algorithm. According to the results of the tests, it observed that the proposed method has much faster and more effective results in conditions less than 20 number of machines or jobs which will be scheduling.

BÖLÜM 1. GİRİŞ

Günümüzde çizelgeleme problemleri hayatın birçok alanında etkin bir şekilde karşılaşılan en önemli problemlerden biridir. İşgücünün planlanması, araç rotalama, nakliye çizelgelemesi, uçakların iniş kalkışlarının çizelgelemesi bunlardan sadece birkaçıdır. Bu problemlerin çözümünde hedeflenen esas amaç, yapılması gerekli olan işlerin hangi zaman aralıklarında kimler tarafından yapılacağının belirlenmesidir. Yine aynı şekilde üretim yönetimi alanında da çizelgeleme büyük önem arz etmektedir. Üretimde çizelgeleme ile yapılmak istenen mevcut makinelerde işlevi olan işlerin en uygun olan sıralamasının bulunarak, işlerin bitirilme zamanını en küçüklemektir. Çizelgeleme, belirli işleri yapmak için hangi kaynakların nasıl ve ne zaman kullanılacaklarını belirler.

Günümüzde çizelgeleme problemleri hem imalat sektöründe hem de hizmet sektöründe çokça karşımıza çıkmaktadır. Hayatımızın birçok alanında çok önemli bir yeri olan çizelgeleme problemlerinin çözümü ile ilgili olarak araştırmacılar yıllardır çok ciddi çalışmalar yapmıştır ve yapmaya devam etmektedirler. Bu çalışmaların yapılmasında şüphesiz en büyük sebep, mevcut çizelgeye göre daha iyilerinin geliştirilmesini sağlamaya çalışmak ve daha büyük kazanç ve verimlilikler ortaya koymaktır. Örneğin; bir vardiya çizelgeleme probleminde yapılabilecek daha iyi bir çizelgeleme ile kazanılacak saatler işletme için ekstra kazanç demektir. Uçakların iniş kalkışları ile ilgili bir havalimanında yapılacak daha iyi ve doğru bir çizelgeleme, yakıt tasarruflarından şirketlere çok büyük meblağlarda kazançlar sağlayabilmektedir. İyi bir hava trafiği çizelgelemesi sonucunda çizelgenin doğru yapılması yakıt tasarrufunun yanında hava trafiğinde olası meydana gelebilecek kaza ve karışıklıkları da önleyerek çok ciddi yararlar sağlayabilmektedir. Yine üretim hatlarında yapılması gereken işlerin makinelere daha iyi bir çizelge ile atanması sonucu işletmenin kazanabileceği süreler çok büyük karlılıklar sağlayabilmektedir. Özellikle büyük ölçekli firmalarda kazanılacak 1 dakikanın, hatta 1 saniyenin bile çok büyük önemi vardır. Yukarıda kısa

örnekler ile giriş yapılarak önemi anlatılmaya çalışılan çizelgeleme problemleri ile ilgili olarak, bu tez çalışmasında imalat sektöründe önem arz eden Atölye Tipi Çizelgeleme Problemleri (Job Shop Scheduling Problem- JSSP) üzerinde çalışılmıştır.

1.1. Kullanılan Çözüm Yöntemleri

Literatür incelendiğinde, çizelgeleme ile ilgili problemleri çözmek için araştırmacılar birçok metot kullanmışlardır. Genel olarak çizelgeleme problemlerinin çözümü için kullanılan yöntemleri eniyileme yöntemleri ve Sezgisel (Heuristic) yöntemler olmak üzere ikiye ayırabiliriz [1].

1.1.1. Eniyileme yöntemleri

Öncelikle eniyileme kelimesinin anlamına bakarsak; eniyileme, 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 kısıtları yerine getirecek şekilde, bilinmeyen parametre değerlerinin bulunmaya çalışıldığı herhangi bir problem olarak tanımlanabilir [2]. 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 [3-4].

Bu tanımlamalardan sonra çizelgeleme problemlerinin çözümünde kullanılan eniyileme yöntemlerini incelersek, eniyileme yöntemlerinin 3 grupta toplandığını gözlemleriz [1]:

İlk metot Dinamik Programlama (Dynamic Programming-DP) tekniği olup, dinamik programlama tekniği ilk olarak 1954 yılında Richard Bellman [5] tarafından ortaya konulmuş bir eniyileme neticesini bulmak amacıyla uygulaması yapılan bir karar verme yöntemidir. Belirtilen metot, 1940'lı yıllarda karar teorisindeki ardışık ilişkiye dayanmaktadır. Daha sonraları bu teknik yöneylem araştırma ve çizelgeleme

alanlarında etkin bir şekilde kullanılmıştır [6]. Richard Bellman'dan sonra bu yöntemle ilgili olarak yapılan öncü araştırmalar, Held and Karp [7], Lawler [8] ile Lawler and Moore [9] tarafından icra edilmiştir. Bu yöntem belirli kısıtlama kurallarını akılcı bir biçimde uygulayarak birçok aday çözümünü elemektedir. Fakat bu yöntem çok büyük boyutlu problemlerde etkili değildir. Bu durumun nedeni ise değişkenlerin sayısı arttıkça problemi çözmek için gereken işlemler de artmaktadır. Bundan dolayı bu durum büyük boyutlu problemlerin çözümü için dinamik programlama metodunun kullanımını kısıtlamaktadır [10, 11, 12].

İkinci metot ise Dal Sınır Algoritması (Branch and Bound Algorithms-BB)'dır. Bu yöntem de kombinatoryal problemlerin çözümü için sık kullanılan metotlardandır. İlk olarak 1959 yılında gezgin satıcı problemine Eastman [13] tarafından uygulanmıştır. Bu metotta, çeşitli veri setlerine göre çözüm zamanları anlamlı derecede değişkenlik gösterir. Sınırlama yaklaşımının seçimi ile dallanan değişken algoritmanın performansını büyük ölçüde etkilemektedir. Problemlerin boyutu arttıkça, bu yöntem ile de çizelgeleme problemlerinin çözümü zorlaşmaktadır [11, 12]. Bundan dolayı üçüncü bir metot olarak Tamsayılı Programlama (Integer Programming-IP) geliştirilmiştir. Birçok araştırmacı, çizelgeleme problemlerinin çözümü için tamsayılı programlama modellerini kullanmışlardır. İlk olarak ise 1954 yılında George Dantzig [14] tarafından vardiya çizelgeleme konusunda kullanılmıştır. Tamsayılı programlama modeli olarak bir çizelgeleme problemi formüle edilebileceği için çözümü de bu algoritmalar ile yapılabilir. Fakat bu şekilde bir yaklaşım için sadece küçük ölçekli problemler ele alınabilir. Genellikle çizelgeleme ile ilgili problemlerin matematiksel olarak formülize edilebilmeleri için çok sayıda kısıt ve değişkene ihtiyaç duyulmaktadır. Bu tür problemleri etkin bir şekilde çözmek için mevcut tamsayılı programlama algoritmaları başarılı olamamaktadır. Fakat bu şekilde olan formülasyonların üstünlüğü birden çok kriteri tek bir hedef fonksiyonu altında toplayabilmesidir [10, 11, 12].

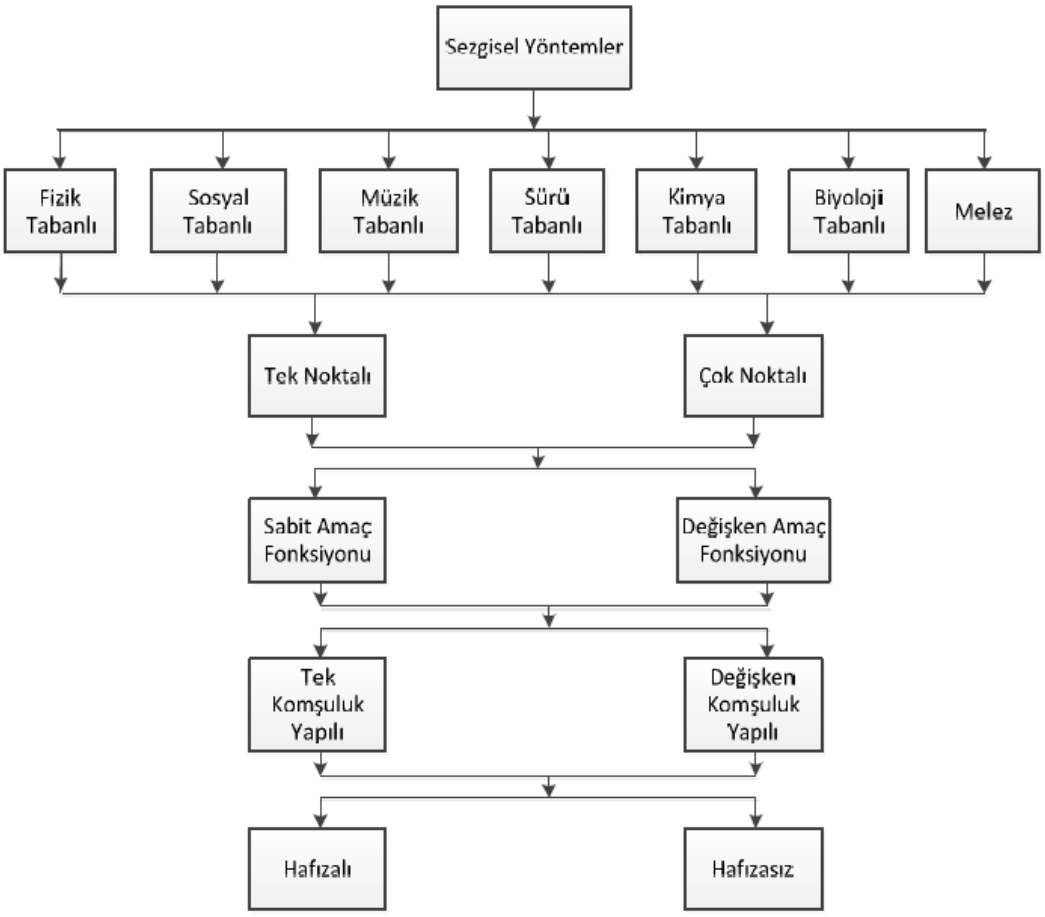
1.1.2. Sezgisel yöntemler

Çizelgeleme problemlerinin çözümünde yukarıda bahsedilen eniyileme yöntemleri kullanılabilmektedir. Ancak hesaplama zamanı ve maliyetlerinin yüksek olmasından dolayı ve çizelgeleme problemlerinin boyutundaki artışa paralel olarak bu yöntemlerin uygulanması zorlaşmaktadır [15]. 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 sezgisel yöntemlerin kullanımı daha makul bir çözüm olarak karşımıza çıkmaktadır. Son yıllarda birçok yeni sezgisel teknikler geliştirilmiştir. Bu sezgisel algoritmaların birçoğu özellikle doğadan ilham almaktadırlar [16].

Sezgisel yöntemler, herhangi bir amaç doğrultusunda hedefe ulaşmak için doğal fenomenlerden ilham alan algoritmalardır. Belirtilen algoritmaların, çözüm uzayındaki optimum olan bir çözüme yakınsama durumu ispat edilememektedir. Açıkçası, bu algoritmaların yakınsama özelliği vardır, fakat problemin kesin çözümünü garanti edemez. Buna ilave olarak kesin çözümün civarlarında olan bir çözüm garanti edebilmektedir. Karar verici açısından sezgisel algoritmaların anlaşılırlığının çok daha iyi olabilmesinden dolayı sezgisel algoritmalara ihtiyaç duyulmaktadır [17].

Sezgisel yöntemler genel olarak fizik tabanlı, sürü tabanlı, biyoloji tabanlı, müzik tabanlı, sosyal tabanlı ve kimya tabanlı olmak üzere altı değişik grupta incelenmektedir. Ayrıca bu algoritmaların hibrit olarak kullanımı melez yöntemler de mevcuttur. Bahsi geçen bu yöntemler Şekil 1.1'de sunulmaktadır. Genetik Algoritma (Genetic Algorithms-GA), Diferansiyel Gelişim (Differential Evolution-DE) algoritması, Karınca Kolonisi Optimizasyonu (Ant Colony Optimization-ACO), Yapay Sinir Ağları (Artificial Neural Networks-ANN), Yapay Arı Kolonisi (Artificial Bee Colony-ABC) algoritması ve Yapay Bağışıklık Sistemi (Artificial Immune System-AIS) algoritmaları biyolojik tabanlı; Emperyalist Yarışmacı Algoritma (Imperialist Competitive Algorithm-ICA), Parlamenter Optimizasyon Algoritması (Parliamentary Optimization Algorithm-POA) ve Tabu Arama (Tabu Search-TS) sosyal tabanlı; Yapay Kimyasal Reaksiyon Algoritması (Artificial Chemical Reaction

Optimization Algorithm-ACROA), kimya tabanlı; Armoni Arama (Harmony Search-HS) algoritması müzik tabanlı; Tavlama Benzetim (Simulated Annealing-SA), Yerçekimi Arama (Gravitational Search-GS), Zeki Su Damlacıkları (Intelligent Water Drops algorithm-IWD) algoritması fizik tabanlı ve Parçacık Sürü Optimizasyonu (Particle Swarm Optimization-PSO), Kedi Sürüsü Optimizasyonu (Cat Swarm Optimization-CSO) sürü tabanlı algoritma ve modellerdir [18].



Şekil 1.1. Sezgisel yöntemler [18]

Bu algoritmalardan bazılarını inceleyecek olursak karınca kolonisi optimizasyon algoritması bu sezgisel algoritmalardan biridir. Karınca kolonisi optimizasyon algoritması ilk defa Marco Dorigo [19, 20] tarafından ortaya çıkarılmıştır. Bu algoritmanın karıncaların doğal davranışlarından esinlenerek geliştirilmiş olması ana felsefesidir [21]. Karıncalar gittikleri yolların üzerinde "pheromone" olarak nitelendirilen bir element bırakmaktadırlar. Karıncalar birden fazla güzergâh içinden birisini seçmek durumunda kaldıklarında madde miktarının daha fazla olduğu

güzergahı seçmektedirler. İşte bu madde karıncaların birbiri arasında iyi iletişim kurarak yuvaları ile yiyecek maddeleri arasındaki en kısa mesafenin elde edilmesine yardım eder [22].

Diğer bir sezgisel algoritma ise parçacık sürüsü optimizasyon algoritmasıdır. PSO algoritması ilk olarak 1995'te Kennedy ve Eberhart [23] tarafından; kuş sürüsü hareketlerinden ilham alınarak geliştirilen populasyon tabanlı olasılıklı bir optimizasyon metodudur. Lineer olmayan problemlerin çözümünde kullanılmaktadır. Bu metot, çok parametre ve değişkene sahip optimizasyon problemlerinin çözümünde kullanılmaktadır. PSO tekniğinde sistem rasgele çözümlerin mevcut olduğu bir populasyon ile başlatılır. Daha sonra nesiller güncellenerek en optimum çözüm araştırılır. PSO'da parçacık denen olası çözümler, o andaki optimum olan parçacığı takip ederek problem uzayında çözüm ararlar. Bu yöntemin geleneksel optimizasyon metotlarından en büyük farkı, PSO'da türev bilgisine gerek olmamasıdır. PSO yöntemi, algoritmasında düzenlenmesi gerekli olan parametrelerin cüzi olmasından dolayı nispeten basittir ve bu yöntem bulanık sistem kontrolü, fonksiyonların optimizasyonu, yapay sinir ağı eğitimi gibi alanlarda başarıyla uygulanabilmektedir [24, 25, 26, 27].

Sezgisel yöntemlerden olan Kanguru Algoritması (Kangaroo Algorithms-KA) ilk olarak 1978 yılında Pollard tarafından ortaya atılan bir yöntemdir [28]. KA yöntemi, isminden de anlaşıldığı üzere Kanguruların zıplayarak hareket etmesinden ilham alınarak geliştirilmiş bir algoritmadır. Kanguru algoritması, tavlama benzetim algoritması ile benzerlik gösteren, fakat daha değişik bir arama yöntemi kullanan rasgele bir yakınsama yöntemidir [29]. KA, bir f(u) fonksiyonunun yinelemeli prosesine yerleştirilerek minimizasyon amaçlı olarak uygulanır. Bu yöntemde çözüm prosesinde bir iyileşme olmuyorsa, kanguru gibi zıplama yapılarak yerel minimumdan uzaklaşmaya çalışılır. Kısa bir süre içerisinde karşılaşılabilme durumu olsa da mevcut çözümden daha iyi bir çözüme ulaşılması, bu aşamada beklenmemektedir. Bu yöntem için durma kriteri olarak, maksimum iterasyon sayısı veya amaç fonksiyonunun alt sınırı kullanılabilir [30].

Yakın zamanda ortaya atılan diğer bir sezgisel ise Kril Sürüsü (Krill Herd-KH) algoritmasıdır. Bu algoritma, antartik krillerinin beslenme ile ilgili davranışlarından ilham alınarak ilk defa 2012 yılında Gandomi ve Alavi [31] tarafından ortaya çıkmış sürü zekâsı tabanlı bir sezgisel algoritmadır. Kril sürüsü algoritmasında, "mevcut sürü diğer canlılar tarafından avlanır ve ortalama kril yoğunluğu azalarak besin kaynağından uzaklaşılır" durumu başlangıç durumu kabul edilir. Doğal bir kril sürüsü sisteminde, her bir bireyin uygunluk değeri hesaplanırken, mevcut sürüdeki en yüksek yoğunluk noktası ve besinin bulunduğu noktalara olan uzaklıkların birleşimi esas alınır. Bu şekilde uygunluk değeri, hedef fonksiyonunun değeri olmaktadır [32].

Tavlama benzetim algoritması, kombinatoryal olan optimizasyon problemlerinin çözümü için kullanılan ve iyi çözümler veren stokastik bir arama yöntemidir. Algoritmanın adı, katı maddelerin fiziksel tavlanma prosesi ile olan benzerliklerinden kaynaklanmaktadır. Bu algoritma, ilk olarak 1983 yılında Kirkpatrick ve arkadaşları [33] ve daha sonra da 1985 yılında Cherny [34] tarafından birbirlerinden bağımsız olarak ortaya konmuştur. Bu yöntem şimdiye kadar, çeşitli alanlarda birçok eniyileme problemlerine uygulanmıştır. Tavlama benzetim yöntemi, bir katı maddenin minimum enerji hâli elde edilesiye kadar yavaş yavaş soğumaya tabi tutulduğu fiziksel tavlama oluşumunu taklit eden stokastik bir arama metodudur. Bu teknik ile üretilen çözümler sırasının hedef fonksiyon değerleri genel olarak azalma eğilimi göstermektedir. Ancak bir kısım durumlarda hedef fonksiyon değerleri fazla olan çözümler de kabul görmektedir. Bu yöntemle, yerel bir minimum çevresinde yapılmakta olan arama terkedilip, daha iyi olabilecek bir yerel veya daha da ötesi global bir minimumu elde etmek için arama yapmaya devam edilir. [35, 36].

Yine başka bir sezgisel algoritma ise Ateşböceği Algoritmasıdır (Firefly Algorithm-FA). Xin-She Yang [37] tarafından geliştirilmiştir. Bu algoritma tropikal hava koşullarındaki ateşböceklerinin göstermiş oldukları sosyal davranışlardan ilham almaktadır. Yine bu yöntem diğer sürü zekâsı tabanlı sezgisel algoritmalar ile benzerlikleri olmasına rağmen kavram olarak ve uygulama yönünden daha basittir. Doğada bulunmakta olan ateş böceklerinin ışıkları sürekli yanıp sönmektedir. Bu durumun ilk hedefi, diğer başka ateş böceklerini yanına çekmek için bir sinyalizasyon

sistemi olarak davranış göstermektir. Yanıp sönmekte olan ışıkların üretimi için kompleks bir biyokimyasal sürecin ayrıntıları ve reel hedefi bilim camiasında hâlen münazara konusu olmaktadır. Birçok bilim insanı bu yanıp sönmekte olan ışıkların ateşböceklerine, kendi cinslerini bulmak için ve mümkün olabilecek avlarını çekmek için, kendilerini avlamak isteyenlerden kendilerini korumaya yardımcı olduklarını düşünmektedir. Bu algoritmada, kazançlı iyi sonuçlar elde edebilmek için, verilmiş olan bir eniyileme probleminin hedef fonksiyonu, ateşböceği sürüsüne parıltılı ve daha cazip konumlara gitmek için yardım etmekte olan yanıp sönmekte olan ışıklar ile ilişkisi olmaktadır. Tüm ateş böcekleri tek sınıf olarak kabul görmektedir. Birbirlerini çekmeleri bu algoritmanın temelini meydana getirmektedir. Bu algoritmada bir ateş böceğinin parlaklığının fazla olması diğer ateş böcekleri için daha çekici hale gelmesine sebep vermektedir. Bir ateşböceği kendisinin parlaklığından daha parlak bir ateşböceği gördüğünde ona doğru gitmek isteyecektir [38, 39, 40].

Tabu arama yöntemi de kombinatoryal eniyileme problemlerinin çözümü için geliştirilmiş sezgisel algoritmalardandır. Bu metot başka yöntemler ile beraber kullanılabilmekte ve bu şekildeki hibrit kullanımlar ile yerel optimuma takılıp kalmaktan korunabilmeyi sağlayan bir yöntemdir. Tabu arama yöntemini bugünkü kullanılan modern şeklini Glover [41, 42, 43] yaptığı çalışmalar ile ortaya koymuştur. Bu yöntem, başlangıç çözümü, aday liste stratejileri, hareket mekanizması, hafıza, durdurma kriterleri olarak isimlendirilen belli başlı elemanlara sahiptir. Tabu arama yönteminde başlangıç olan çözüm rasgele olarak seçilebileceği gibi, başlangıç çözümünün belirlenmesi için diğer başka bir optimizasyon tekniği de kullanılabilmektedir. Bu yöntemin aksiyon mekanizması, elde var olan çözümde yapacağı bir değişiklikle ulaşılabilecek yeni çözümleri belirler, olası aksiyonlar, eldeki çözümün tüm komşuluklarını oluşturur. Aksiyon mekanizmasının, problemin yapısına bağlı olması ve uygun bir şekilde belirlenmiş olması bu yöntemin performansı için çok önem arz etmektedir. Daha kötü gibi görünmekte olan bir çözüme arada sırada yönelme esnekliği göstermesi tabu arama yönteminin bir özelliğidir [1].

Genetik algoritma yöntemi tabu arama ile tavlama benzetim tekniklerinden daha da genel bir yöntemdir. Tabu arama ile tavlama benzetim, genetik algoritmaların daha

özel durumlarıdır. Holland [44], De Jong [45] ve Goldberg [46] isimli araştırmacılar genetik algoritmaların öncü çalışmalarını yapmışlardır. Genetik algoritmalardaki çözüm uzayının oluşumunu, kromozom veya dizi olarak gösterilen aday çözümler sağlamaktadır. Her kromozom bir hedef fonksiyon değerine sahiptir. Yani her kromozomun uygunluk değeri bulunmaktadır. Seçilmiş olan bir kromozomun kümesi ve uygunluk değerleri bir küme oluşturur. Genetik algoritmalarda her bir iterasyonda üretilmiş olan bir yığının hacmi, ilgili iterasyonun neslini oluşturmaktadır. Bu yöntem çizelgeleme problemleri için uygulandığı zaman çizelgeler bir kümenin bireyleri olarak dikkate alınmaktadır. Her bir bireyin bir uygunluğu mevcuttur. Bir bireyin uygunluğu ise onula ilgili olan problemin hedef fonksiyon değeri ile ölçülmektedir. Metot tekrarlamalı olarak çalışmakta ve her iterasyon ise bir nesile karşılık gelmektedir. Bir neslin büyüklüğü bir önceki nesilde yaşayan bireyler ile önceki nesilden gelen çocuklardan (veya yeni çizelgelerden) oluşmaktadır. Algoritmada bir nesile karşılık gelmekte olan yığının büyüklüğü bir iterasyondan diğerine kadar genel olarak sabit olmaktadır. Çocuklar ise önceki nesildeki bireylerin mutasyonları ve çaprazlamaları ile üretilmektedir. Bireyler, kromozom cinsinden ifade edilmektedir. İlgili kromozomların her birisi bir makinadaki işlerin sırasıyla alakalıdır. Bir aileyi temsil eden kromozomdaki olan mutasyon, ilgili sıradaki bulunan eşdeğer olan iş çiftlerinin yer değiştirmelerine eşdeğer olabilmektedir. Her nesil için uygunluk değeri fazla olan bireyler artarken değeri az olanlar ise ölmektedir. Yeni kuşağın teşkilini belirleyen doğum, üreme süreçleri, ölüm karışık olabilmekte ve çoğunlukla mevcut olan nesil bireylerinin uygunluk seviyelerine bağlı olmaktadır [1].

Yine başka bir sezgisel algoritma olan yapay balık sürüsü optimizasyonu, besin arama sürecinde balık sürülerinin sosyal davranışlarından esinlenen sezgisel bir eniyileme algoritmasıdır. Denizdeki bir balık, yiyeceğini ararken besin değeri yüksek olan alanları birer birer arayarak veya diğer balıkları izleme yoluyla bulabilmektedir. Fazla balık olan bir alanın besin değeri de çoğunlukla daha fazla olmaktadır. Bu yöntem için temel ana düşünce, global optimuma ulaşabilmek için balık bireyinin lokal arama yapmasıyla izleme ve toplanma gibi balıkların hareketlerini benzetebilmektir. Başlıca bir suni balığın hayatını sürdürdüğü ortam çözüm uzayı olmaktadır. Balığın bir sonraki hareketi ise balığın var olan durumuna ve lokal çevresel durumuna bağlıdır. Bu şekilde

yapay bir balık kendisinin ve arkadaşlarının faaliyetleri sayesinde çevresini de etkileyebilecektir [47].

Bakteriyel Beslenme Optimizasyon (Bacterial Foraging Optimization-BFO) algoritması, Escherichia Coli (E. coli) bakterisinin beslenme davranışlarından ilham alarak kompleks mühendislik problemlerinin çözümü için geliştirilmiş olan bir yöntemdir. Bakteriler, karmaşık yaşam formlarında bulunan diğer canlılara nispetle çok daha basit yapıda bulunmaktadırlar. Bakteriler kısıtlı olan algı ve hareket yeteneklerini kullanarak optimum seviyede enerji harcayarak beslenmeleri gerekmektedir. Bakterilerin farklı yaşam formlarına göre modellenebilmeleri çok daha kolay olmaktadır. E. coli bakterisi de bu belirtilen türden canlılardan birisidir. E. coli bakterisi, yapısı ve çalışma şekli en iyi anlaşılan mikroorganizmaya sahip canlılardan biridir. Bu bakteri bir yiyecek maddesine ulaştığı zaman diğer bakterileri uyaran bir etkiye sahip olan kimyasal olan bir madde salgılamaktadır. Bu salgılanan madde, diğer bakterilerin besini bulan bakterilerin olduğu yere doğru gitmesini sağlamaktadır. Eğer besin yoğunluğu çok fazla ise bakteriler kenetlenerek grup olarak hareket edebilmektedirler [48].

Diğer bir sezgisel algoritma olan Kurt Kolonisi Algoritması (Wolf Colony Algorithm-WFA) kurt kolonilerinin yoğun bir organize olabilme yeteneklerinden ilham alınarak geliştirilmiştir. Kurtlar görevlerini diğerleriyle bölüşmektedir ve bu hayvanlar avlandıktan sonra tutarlı adımlar atmaktadırlar. Az miktardaki yapay kurt aktif olduğu av bölgesinde arama için atanmaktadır. Atanan arama kurtları avı keşfedince, avın konum bilgisini diğer kurtlara ulumayla bildirir. Diğer yapay kurtlar da belirtilen ava yaklaşırlar ve avı kuşatırlar. Kurt kolonisindeki atanma kuralı ise, yiyeceğin ilk olarak güçlü olan kurda atanması ve daha sonra zayıf olan kurta atanması şeklindedir. Kurt kolonisi algoritması da belirtilen bu davranışların taklit edilmesiyle geliştirilmiş olan bir algoritmadır [49].

Kedi sürüsü optimizasyon algoritması da kedigillerin doğadaki davranışlarından esinlenerek ortaya çıkarılmış bir sezgisel algoritmadır. Kedilerin davranışlarının benzetimi yapılmış ve 2 alt model geliştirilmiştir. Bu yöntemde bulunan matematiksel

modeller, kedi hareketlerinin gözlenip, çözümlenmesi neticesinde oluşturulmuştur [50]. CSO metodu kullanılarak yapılan literatürdeki birtakım araştırmalar şunlardır: Santaso ve arkadaşları kümeleme problemlerinin çözümünde CSO'yu kullanmışlardır [51]. Wang ve arkadaşları ise en az duyarlı bitin yerine en iyisini getirmek için CSO stratejisini kullanmışlardır [52]. Hwang ve arkadaşları da müşterilerin isteklerine göre en uygun sözleşme kapasitesi probleminin çözümünde CSO ve PSO'yu hibrit olarak kullanmışlardır [53]. Lin ve Chen ise destek vektör makinesi için parametre optimizasyonu problemlerinde CSO'yu kullanmışlardır [54]. Kalaiselvan ve arkadaşları filigran performansının arttırılması için CSO algoritmasını kullanmışlardır [55]. Wang ve Wu, CSO'yu destek vektör makinesiyle birlikte e-öğrenmede duygu tanıma probleminin çözümü için kullanmışlardır [56]. CSO algoritmasının paralel bir değişik versiyonu da Tsai ve arkadaşları tarafından önerilmiştir [57].

Atashpaz-Gagari ve Lucas [58]'ın önerdiği sosyal tabanlı sezgisel algoritmalardan olan emperyalist yarışmacı algoritması da bir başlangıç populasyonu ile başlar (örneğin; dünyadaki ülkeler). Populasyonda bulunan iyi durumdaki bazı ülkeler emperyalist olması için seçilmektedir. Kalanlar ise bu emperyalistlerin kolonisi olmaktadırlar. Tüm koloni bu emperyalist devletlerin arasında dağıtılmaktadır. Kolonilerin dağıtılmasından sonra, ilgili koloniler uygun olan emperyalistlere doğru hareket etmeye başlarlar. Bir imparatorluğun toplam gücünün göstergesi emperyalistin ve bu emperyalistin kolonilerinin gücüdür. Daha sonra bu bütün imparatorlukların birbirleri arasında yayılımcı rekabet başlar. Bu rekabet içerisinde bir imparatorluk gücünü arttıramaz ise başarılı olamadığı için yarıştan elenmektedir. Bu rekabet içerisinde güçlü olan imparatorluklar daha da güçlenirken zayıf olan imparatorluklar ise daha da zayıflar ve sonunda zayıf olan imparatorluklar yıkılırlar. Bu şekildeki yarış en sonunda tek bir imparatorluk kalana değin devam etmektedir. Sonunda ise diğer bütün ülkeler belirtilen imparatorluğun bir kolonisi olmaktadırlar [58].

Diğer bir sosyal tabanlı sezgisel algoritma olan parlamenter optimizasyon algoritması da reel yaşamdaki parlamento seçimlerinin benzetimi sonucu geliştirilmiştir. Bu yöntemdeki optimizasyon işlemi, birey populasyonun oluşturulması ile başlamaktadır. Belirtilen bireyler parlamento üyesi olarak kabul edilirler. Daha sonra ise populasyon

politik bazı gruplar arasında dağıtılmakta ve yüksek uygunluğa sahip sabit sayıdaki üyeler grupların adayı olarak seçilmektedirler. Populasyonun bu şekilde bölüştürülmesi sonucu, grup içi rekabet başlamaktadır. Grup içi rekabette asil olan üyeler kendileri için uygun olan adaylara doğru yönelmektedir ve bu mevcut anlatılan durum asil olan adayların vektörlerinin ağırlıklı olarak ortalaması şeklinde modellenmektedir. Parti içi rekabet bittikten sonra birkaç tane yüksek uygunluğa sahip aday, grubun nihai adayları olarak kabul edilmektedir. Bir sonraki aşamadaysa seçilmiş olan bu adaylar diğer gruplardaki adaylar ile rekabet ederler. Bir grupta bulunan asil üyeler ve adaylar grubun toplam gücünün belirlenmesinde önem arz etmektedirler. Adayların temel gücünün lineer birleşimi ve asil üyelerin temel gücü, bir grubun toplam uygunluğu olarak isimlendirilir. Bahsi geçen grup içi rekabet bittikten sonra gruplar arasındaki rekabet başlamaktadır. Politik olan gruplar parlamentoda bulunan kendi adaylarını kabul ettirmek için diğer gruplar ile rekabet yaparlar. Uygunluğu göz ardı edilen grupların gücü seviye seviye azalmakta ve sonunda çökmektedir. Diğer taraftan, güçlü gruplar aşama aşama daha da güçlenir ve yarışı kazanmak için daha fazla şansa sahip olurlar. Güçlü olan gruplar bazen birleşmek için anlaşırlar ve kazanma şanslarını daha da arttırırlar. Gruptaki asil üyeler ile kendi gruplarındaki adaylar beraber olarak kendileriyle benzer özellikteki daha güçlü bir gruba dâhil olurlar. Bu anlatılan adımlar parlamentoda tek bir grupta birleşilene kadar sürer. Tüm gruplar birleştikten sonra da, asil olan üyeler lider olan adayla hemen hemen eşit güce sahip olurlar [59, 60].

Yukarıda bahsedilen sezgisel algoritmalar gibi literatürde daha birçok sezgisel algoritmalar araştırmacılar tarafından çalışılmaktadır. Bu algoritmaların çizelgeleme problemlerinde kullanılmasıyla ilgili literatür bilgilerine tezin 4. Bölümünde yer verilmiştir.

1.2. Tezin Amacı ve Düzenlenmesi

Günümüzde doğru ve etkin bir çizelgeleme hem insanlar hem de işletmeler için büyük önem arz etmektedir. Bu bağlamda çizelgeleme problemlerinin çözümünde özellikle son yıllarda sezgisel algoritmaların araştırmacılar tarafından yoğun bir biçimde

kullanıldığı görülmektedir. Bu tez çalışmasında da, atölye tipi çizelgeleme problemlerinin çözümünün eniyilemesi için etkili bir bütünleşik yaklaşım önerilmiştir. Bu bağlamda da son yıllarda etkin bir biçimde kullanılan [61, 62, 63] sürü zekâsına dayalı sezgisel algoritmalardan olan yapay arı kolonisi algoritması ile Evrimsel Algoritmalar (Evolutionary Algorithms-EA) bütünleşik olarak kullanılarak bir eniyileme yöntemi önerilmiştir.

Tezin bundan sonraki kısımlarında ise ilk olarak önerilen metodun yapısındaki evrimsel algoritmalardan bahsedilerek konu hakkında bir literatür çalışması yapılmıştır. Evrimsel algoritmaların farklı algoritmalar ile beraber kullanıldığında eniyileme konusunda etkili olduğunu göstermek amacıyla evrimsel algoritmaların yapay sinir ağları ile beraber kullanımı sonucu oluşan Evrimsel Yapay Sinir Ağları (Evolutionary Neural Networks-EANN) kısaca anlatılmış, daha sonra önerilen metot kullanılarak endüstriyel optimizasyon problemleri ile ilgili deneysel bir çalışma yapılmıştır. Literatürde daha önce farklı bir optimizasyon metodu ile çözülmüş olan bir çalışmaya EANN metodu uygulanmış ve sonuçlar kıyaslanarak evrimsel algoritmaların farklı algoritmalar ile beraber kullanımındaki etkinliği gösterilmeye çalışılmıştır. Sonraki bölümde ise önerilen metotta kullanılan diğer algoritma olan ABC algoritması detaylı olarak sunulmuştur. ABC ile ilgili literatürde yapılmış olan çalışmalar hakkında bilgi verilmiş, bu algoritmanın doğadan nasıl esinlendiği detaylı anlatılmış, algoritmanın çalışma prensipleri ve özellikleri sunulmuştur.

Uygulama bölümünde ise, öncelikle üzerinde çalışma yapılan atölye tipi çizelgeleme problemlerinden bahsedilmiş, bu konu hakkında literatürde yapılmış olan çalışmalar hakkında bilgi verildikten sonra önerilmiş olan metodun yapısı detaylı bir şekilde anlatılmıştır. Daha sonra önerilen metot atölye tipi çizelgeleme problemleri ile ilgili data setlerine uygulanarak sonuçlar tablo ve şekillerde gösterilmiştir. Sunulan yaklaşımın farklı iterasyonlarda çalıştırılmasıyla elde edilen sonuçlara parametrik testler uygulanarak, elde edilen sonuçların istatistiksel olarak anlamlı olup olmadığı araştırılmıştır. Sonuç kısmında önerilen metot belirli kıyaslama kriterleri kullanılarak farklı optimizasyon teknikleri ile karşılaştırılarak önerilen metodun etkinliği gösterilmeye çalışılmıştır. Parametrik ve parametrik olmayan testler kullanılarak

kurulan hipotezlere göre önerilen metot ile elde edilen sonuçlar ile kıyaslanan optimizayon tekniklerinin sonuçları arasında istatistiksel olarak anlamlı bir fark olup olmadığı araştırılmıştır.

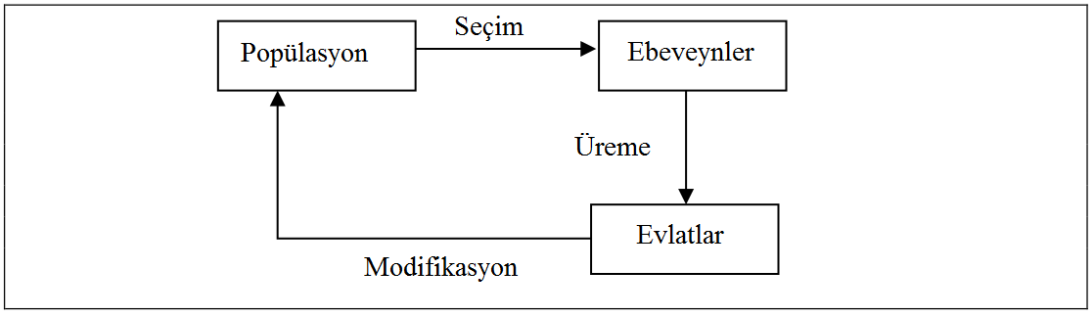
BÖLÜM 2. EVRİMSEL ALGORİTMALAR

2.1. Giriş

Evrimsel algoritmaların mühendislik, ekonomi, pazarlama, yöneylem araştırması, robotik, sosyal bilimler, fizik ve kimya gibi birçok alanda başarılı uygulamaları vardır [64]. Evrimsel Algoritmalar çok amaçlı optimizasyon problemlerini çözmek için güçlü bir alternatif olarak ortaya çıkmışlardır [65]. Evrimsel algoritmalar çok amaçlı optimizasyon problemlerine uygulanmaya başladığından itibaren, bu problemlere geleneksel optimizasyon teknikleri uygulanarak elde edilen sonuçlara göre daha etkin sonuçlar elde edilmiş ve önemli bir başarıya ulaşılmıştır [66]. Daha önceki yıllarda birçok araştırmacı tek hedefli optimizasyon problemleri ile uğraşırlardı. Son 15 yıldan beri ise çok amaçlı optimizasyon problemleri ile ilgili çalışmalar hız kazanmış ve çok amaçlı optimizasyon problemlerinde evrimsel metot uygulamaları iyi çözüm üreten metotlar olarak bilinmektedir [67]. Farklı optimizasyon teknikleri ile karşılaştırıldığında evrimsel algoritmaların avantajları çoktur. Evrimsel algoritmalar geniş bir uygulama alanına sahiptir ve karmaşık arama uzayları için oldukça uygundur. EA optimizasyon problemlerinde diğer klasik metotlara göre daha iyi sonuçlara ulaşabilmektedir [68].

Evrimsel algoritmalarda temel düşünce, arama uzayında bir çözüm ihtiva edebilecek bireylere sahip bir populasyon üretmektir. Daha sonra istenen sonucu elde etmek için populasyondaki bireylerin uygunluk önem derecelerine bakılarak evrimsel dönüşümler uygulanarak sonraki nesiller oluşturulur [69]. Evrimsel algoritmalar biyolojik evrimden esinlenir ve populasyonlarda bireylerin üretilmesinde çaprazlama, mutasyon ve turnuva seçimi gibi evrimsel stratejiler kullanılır [70]. Bir üreme süreci için evrimsel algoritmaların çalışma yapısı Şekil 2.1'deki gibidir. Optimizasyon

problemlerindeki her bir aday çözüm populasyondaki herhangi bir bireyi temsil etmektedir. Yeni bireyler yukarıda da bahsedilen üreme operatörleri kullanılarak üretilmektedir. Optimizasyon problemlerinin çözümü için kullanılan bu bireyler populasyonu oluştururlar. Daha sonraki nesillerin oluşturulmasında populasyondaki iyi bireyleri belirleyip seçebilmek için uygunluk fonksiyonları kullanılmaktadır. Bu uygunluk fonksiyonlarına göre iyi bireyler (optimizasyon problemlerinde istenilen sonuca göre) sonraki nesillere de aktarılabilir, kötü bireyler ise elemeye tabi tutulurlar. Süreç benzer şekilde belirlenmiş bir durdurma kriterine kadar sürekli bu şekilde tekrar edilerek amaç doğrultusunda en iyi çözüm arama uzayında bulunmaya çalışılır [71].



Şekil 2.1. Bir üreme süreci için evrimsel algoritmaların çalışma yapısı [71].

2.2. Literatür Taraması

Evrimsel algoritmalar ile ilgili literatür incelendiğinde araştırmacılar tarafından birçok çalışma yapıldığını görmekteyiz. Sedki ve arkadaşları [72] yaptıkları çalışmalarında, Fas'taki yarı-kurak bir iklim koşullarında bulunan bir su toplama havzasındaki yağış akışını tahmin etmek amacıyla Evrimsel Sinir Ağlarını ve yapay sinir ağlarını kullanmışlar ve geliştirdikleri algoritma sonucu elde ettikleri sonuçları aynı koşullar için yapay sinir ağlarını kullanarak elde ettikleri sonuçlarla kıyaslamışlardır. Geliştirdikleri algoritmada genetik algoritmalar ile geri yayılımlı yapay sinir ağlarını beraber kullanmışlardır. Elde ettikleri sonuçlar, yapay sinir ağlarına göre evrimsel sinir ağının daha üstün tahmin yeteneklerinin olduğunu göstermiştir.

Kiranyaz ve arkadaşları [73] mimari bir alan içinde optimum ağ yapılandırmasının evrimi için yapay sinir ağlarının otomatik tasarımı için yeni bir teknik önermişlerdir. Bu yöntem sezgisel bir optimizasyon tekniği olan çok boyutlu parçacık sürüsü

optimizasyon tekniği esaslı bir tekniktir. Bu yöntemde optimuma ulaşmada, optimum boyutu bilinmeyen çok boyutlu bir arama uzayında, sürücük parçacıkları pozisyonel ve boyutsal olarak arama yapabilmektedir. Parçacık sürüsü optimizasyon tekniğinin evrimsel sinir ağı yapısında kullanılması sonucu elde edilen deneysel sonuçlar optimuma ulaşmada önerilen yapının iyi sonuçlar verdiğini ortaya koymuştur.

Hacıoğlu [74] yaptığı çalışmasında, önceki çalışmalarda tersinden dizayn etme problemleri için önerilen yapay sinir ağı kullanılarak güçlendirilen genetik algoritmaların hızlı evrimsel eniyileme için kendi problemlerine uyarlamasını yapmıştır. Bu çalışmada gerçek kodlu bir genetik algoritma, bir yapay sinir ağı ile hibrit olarak belirli bir yapı içerisinde beraber kullanılmışlardır. Yapay sinir ağları ilgili hibrit yapıda populasyonun kuvvetlenmesini sağlamıştır. Bundan dolayı yapay sinir ağının eğitiminde, genetik algoritmanın her aşamasında, o aşamada kullanılan populasyondaki bireyler ve bireylerin uygunluk değerleri kullanılmıştır. Ağın eğitilmesinde, yapay sinir ağının girdisi olarak bireyleri ifade eden parametreler kullanılmış, ağın çıktısı olarak bireylerin uygunluk değerleri kullanılmıştır. Eğitimi yapılan bu ağa, benzetimli tavlama metodu yardımı ile bir optimizasyon süreci uygulanmış, var olan populasyon içerisindeki bireylerden daha iyi uygunluk değerine sahip bir birey üretilmeye çalışılmıştır. Bu şekilde uygun bir birey elde edilmiş ise, birey GA tarafından üretimi yapılan yeni populasyona ilave edilmektedir. Genetik algoritmanın her aşamasında tekrarı yapılan bu işlemlerin neticesinde populasyonun gelişmesi daha hızlı sağlandığından, daha az hedef fonksiyonu hesabı ile daha iyi uygunluk değerlerine ulaşılmıştır. Yönteminin ne kadar etkin olduğunu göstermek amacıyla, ilgili metot deneme fonksiyonlarına uygulanmıştır.

Zhang ve Ishikawa [75] çalışmalarında, Evrimsel Kanonik Parçacık Sürüsü Optimizasyonu (Evolutionary Canonic Particle Swarm Optimization-ECPSO) ismiyle yeni bir optimizasyon algoritması geliştirmişlerdir. Araştırmacılar daha önce yaptıkları bir çalışmada Evrimsel Parçacık Sürüsü Optimizasyonu (Evolutionary Particle Swarm Optimization-EPSO) ile ilgili bir araştırma yapmışlar ve PSO metoduna göre Evrimsel algoritmaların daha iyi sonuç verdiğini gözlemlemişlerdir. Yazarlar bu çalışmalarında ise PSO metoduna göre optimuma daha iyi yakınsama özelliğine sahip kanonik PSO

metoduna evrimsel yapıyı uygulamışlar ve ECPSO metodunu geliştirmişlerdir. Bahsi geçen metotları kıyaslamalı olarak uygulamaya tabi tutmuşlar ve deneysel sonuçlara göre ECPSO, diğer metotlara göre daha etkili sonuçlar vermiştir.

Ashena and Moghadasi [76] çalışmalarında delik dibi basıncını tahmin etmek için 3 farklı metot önerdiler. İlk olarak, gizli katmanında 7 nöronlu Geriye Yayılımlı (Back Propagation-BP) yapay sinir ağı (BP-YSA) kullanılmışlardır. Sonraki yöntemlerde yapının evrimi veya optimize edilmesi yapılmıştır. Sinir ağlarının ağırlıklarını ve eşik değerlerini optimize etmek için son derece etkili bir araç karınca kolonisi optimizasyonu ikinci bir yöntem olarak kullanılmışlardır. Bu yöntem, ACO-BP olarak adlandırılır. GA-BP olarak adlandırılan üçüncü yöntemde ise, son derece etkili bir optimizasyon tekniği olan Genetik Algoritma ağın eğitilmesinde kullanılmıştır. Elde edilen sonuçlara göre GA-BP metodu, BP-YSA yöntemine göre daha iyi performans sergilemiştir. Sonuç olarak, GA ve ACO metotlarının delik dibi basıncı tahmininde yapay sinir ağlarının performansını optimize etmek için son derece etkili olduğu gösterilmiştir.

Castellani ve Rowlands [77] sinir ağı sistemi oluşturmak için Evrimsel Yapay Sinir Ağı Üretimi ve Eğitim (ANNGaT) adlı bir algoritma geliştirmişlerdir. Sunulan algoritma sinir ağı topolojisi ve ağırlıklarını eş zamanlı olarak geliştirir. ANNGaT bireyin genetik mutasyonlar yoluyla sinir ağ yapısının gizli katmanlarının büyüklüğünü optimize eder. Gizli katman sayısı bir sistem parametresidir. Deney sonuçları ANNGaT metodunun doğru ve sağlam öğrenme yeteneğine sahip son derece kompakt sinir ağ yapıları ürettiğini göstermektedir. Evrimsel algoritmalar daha küçük yapılar kullanarak güzel performanslı çözümler üretmiştir. Ayrıca süreç tamamı ile otomatik olduğundan önerilen algoritma daha az tasarım çabası gerektirmektedir.

Rocha ve arkadaşları [78] çalışmalarında Yapay sinir ağlarının yerel minimuma sıkışıp kalma dezavantajını gideren Evrimsel algoritmalar ile yapay sinir ağlarının hibrit kullanımını içeren iki aşamalı bir algoritma geliştirmişlerdir. İlki sinir ağ topolojisini geliştirirken, diğeri yapıyı ve ağırlıkları eş zamanlı optimize etmeye çalışmaktadır. Bu stratejileri test etmek için sınıflandırma ve regresyon ile ilgili gerçek hayattan 16 veri

seti kullanılmıştır. Elde edilen sonuçlar diğer veri madenciliği algoritmaları ile karşılaştırılmış ve rekabetçi sonuçlar elde edilmiştir.

Tian ve Noore [79] yaptıkları çalışmada, yazılım ile ilgili kümülatif başarısızlık zaman tahmini için evrimsel bir sinir ağı modelleme yaklaşımı önermişlerdir. Yazarlara göre, genetik algoritma, giriş nöronları ve gizli tabaka nöronların sayısının hesaplanması ile ilgili sinir ağı yapısının global olarak optimizasyonunda kullanılır. Sinir ağı mimarisi optimize edildikten sonra, Bayes teoremi ve Levenberg-Marquardt (LM) algoritması modifikasyonu, bilinmeyen verilerin öngörülebilirlik performansını artırmak için uygulanır. Önerilen algoritmanın deneysel sonuçları göstermiştir ki, farklı yazılım projeleri arasında iyi uyum ve kümülatif başarısızlık zaman tahmini için mevcut sinir ağı modelleri ile önerilen metot karşılaştırıldığında kuramsal bir model ile analiz sonrası gerçek veriler arasındaki uygunluk derecesi ve bir sonraki adım-öngörülebilirliği bakımından, daha iyi bir performansa sahip olduğunu göstermiştir.

2.3. Evrimsel Yapay Sinir Ağları

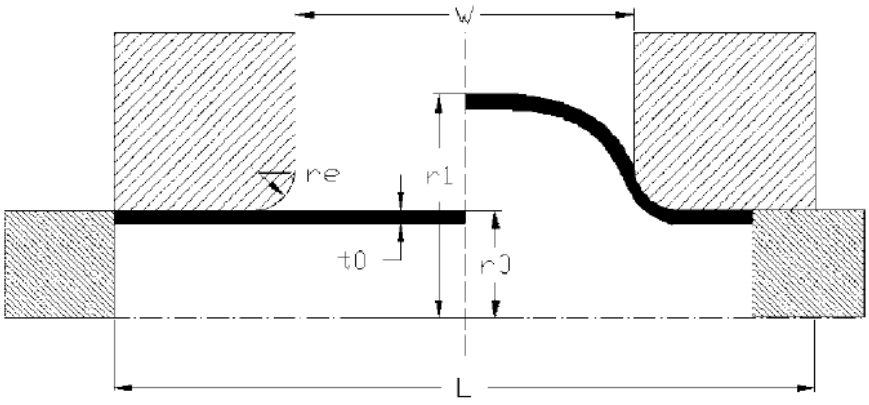
Evrimsel yapay sinir ağları, Yapay Sinir Ağları ve Evrimsel Algoritmaların birlikte kullanımı sonucu oluşan bir optimizasyon tekniğidir. Her iki yöntemin de birbirlerinin eksiklerini tamamlayıcı özellikleri vardır. Yapay Sinir Ağları doğrusal olmayan fonksiyonlar için esneklik sağlarken, evrimsel algoritmalar arama uzayındaki türevin alınamadığı yerlerde bile çok iyi sonuçlar verebilmektedirler [80]. Evrimsel Algoritmalar normalde yavaş olduğu gibi hesaplama işlemleri de pahalı olabilmektedir. Bu algoritmalar YSA metodu ile beraber kullanıldığında, YSA'daki geri yayılım algoritması gibi teknikler arama uzayındaki hızı arttırabilmektedir. Geri yayılım algoritması gibi geleneksel öğrenme algoritmaları üzerinde evrimsel yaklaşımın önemli bir avantajı da yerel bir optimumdan kaçabilme yeteneğidir. Aynı zamanda bu algoritmalar değişen koşullara çabuk adaptasyon gösterebilme yeteneğine de sahiptirler. Literatürde, EANN metodu ile ilgili araştırmalarda üç yaklaşımdan biri kullanılmıştır; ağın ağırlıkları evrimi, yapının evrimi ya da her ikisinin aynı anda evrimi [81]. Yao [82] EANN alanında 300'den fazla referans göstererek literatürde ayrıntılı bir inceleme sunmuştur.

Çalışmanın bu kısmında evrimsel algoritmaların farklı algoritmalar ile beraber kullanımındaki etkinliğini göstermek amacıyla EANN metodu kullanılarak endüstriyel optimizasyon problemleri ile ilgili deneysel bir çalışma yapılmıştır. Literatürde daha önce farklı bir optimizasyon metodu ile çözülmüş olan bir çalışmanın sonuçları ile EANN metodunun sonuçları kıyaslanarak evrimsel algoritmaların hibrit kullanımındaki etkinliği gösterilmeye çalışılmıştır. Bu çalışmanın seçimi yapılırken problemin geleneksel optimizasyon yöntemleri ile çözülmüş olması ve iş endüstrisinde yaygın olarak kullanılması kriterleri göz önünde bulundurulmuştur.

2.4. Boruda Hidrolik Presleme İşleminin Performansını Etkileyen Parametrelerin Eniyilenmesi için EANN Metodunun Uygulanması

2.4.1. Problemin tanımı

Seçilen çalışmada, problemin amacı, Şekil 2.2'de görülen, Boruda Hidrolik Presleme (Tube Hydroforming Process-THP) işleminin performansını etkileyen parametreleri, presleme sonucunda oluşacak düşük İncelme Oranı (Thinning Ratio-TR) ve yüksek Esneme Oranı (Bulge Ratio-BR)'na göre çok amaçlı bir şekilde eniyilemektir. Metzger ve arkadaşları [83] boruda hidrolik presleme ile ilgili olarak Taguchi Metodu kullanarak çözmüş oldukları çok amaçlı optimizasyon problemi aynı koşullar altında evrimsel sinir ağları kullanılarak çözülmüştür. Bu çalışma için sistemin performansını etkileyen parametreler ve seviyeleri Tablo 2.1'de sunulmuştur.



Şekil 2.2. Boruda hidrolik presleme işlemi [83].

Tablo 2.1. Sistemin performansını etkileyen parametreler ve seviyeleri [83].

İsim	Şekillendirme Parametreleri	Seviye 1	Seviye 2	Seviye 3
A	Tüpün uzunluğu (mm)	180	200	220
B	Tüpün kalınlığı (mm)	1.35	1.5	1.65
C	Kalıp giriş yarıçapı (mm)	8	10	12
D	Esneme genişliği (mm)	90	100	110
E	Sertleşme üst değeri	0.207	0.227	0.247
F	İç basınç (MPa)	36	40	44
G	Nominal gerilme oranı	0.2	0.4	0.6
H	Sürtünme katsayısı	0.02	0.06	0.1

Bu problemdeki çok amaçlı kriterlerimizden incelme oranı için formülasyonu Denklem (2.1)'de, esneme oranı için formülasyonu Denklem (2.2)'de verilmiştir.

$$\text{İncelme oranı } (\%) = \frac{t0-t1}{t0} \times 100 \tag{2.1}$$

Bu denklemde t_0, borunun orjinal incelik ölçüsünü, t_1 ise hidrolik presleme sonucu borunun incelik ölçüsünü simgeler.

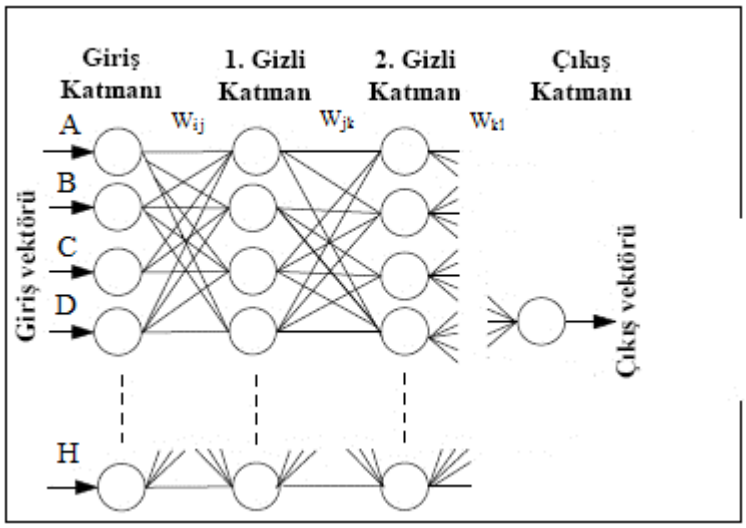
$$\text{Esneme oranı } (\%) = \frac{r1}{r0} \times 100 \tag{2.2}$$

Bu denklemde r_0 hidrolik presleme sonucu borunun yarıçapını, r_1 ise borunun orjinal yarıçapını simgeler.

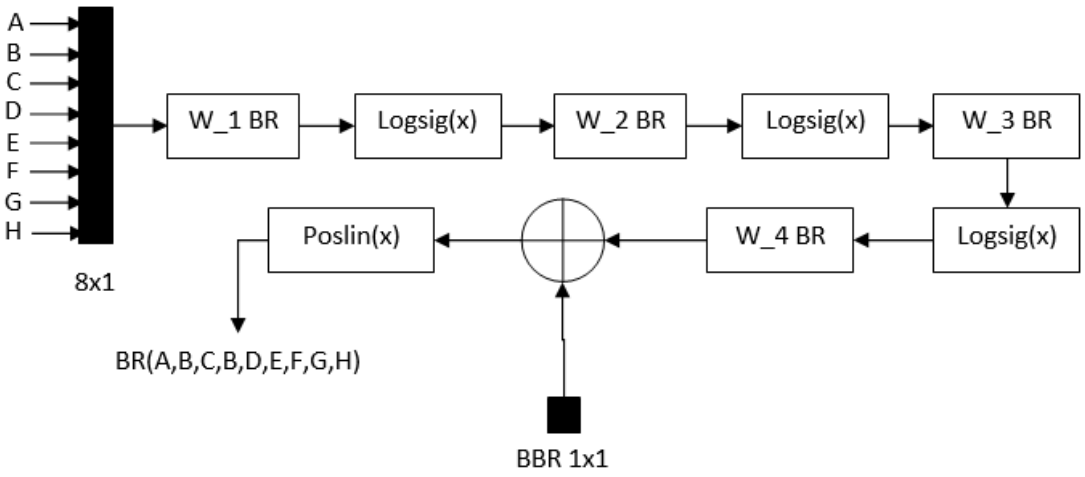
2.4.2. Çözüm metodolojisi

Problemin iki amacı olduğu için, bu iki amacı simüle edecek iki farklı evrimsel yapı inşa edilmiştir. Bu yapıların evrimselliği, genetik algoritma tekniği ile eğitilmesindendir. Sözü edilen yapılar Şekil 2.3'te de yapısı kısaca özetlenen şekilde

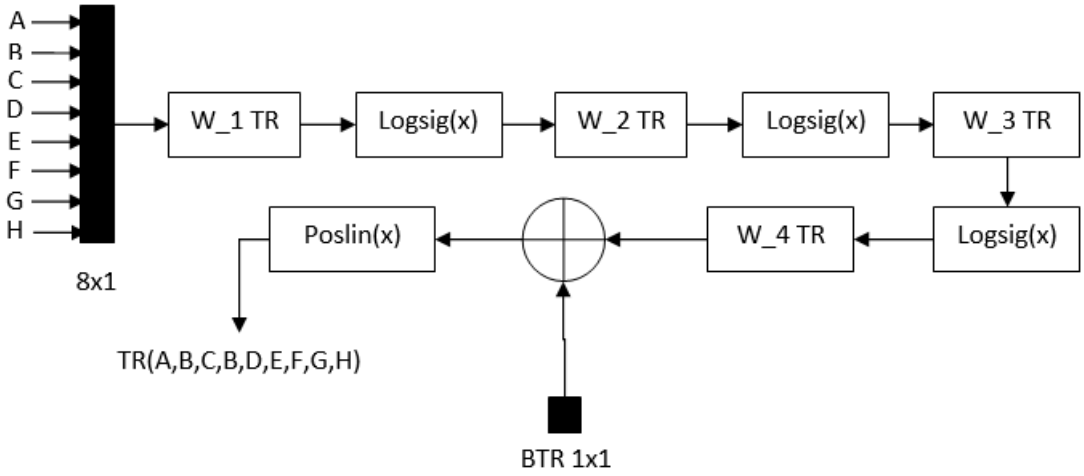
ileri beslemeli ve iki saklı katmana sahip dört katmanlı yapılar şeklindedir. Yeterli eğriliği ve doğrusal olmayan uyumluluğu sağlaması ve aynı zamanda hesaplama karmaşıklığının az olması için dört katmanlı yapı seçilmiştir. Şekil 2.4 esneme oranına yaklaşım yapacak olan yapıyı, Şekil 2.5 ise incelme oranına yaklaşım yapacak olan yapıyı göstermektedir.



Şekil 2.3. İleri beslemeli giriş, gizli ve çıkış katmanlarından oluşan yapı.



Şekil 2.4. Esneme oranı için sunulan yapı.



Şekil 2.5. İncelme oranı için sunulan yapı.

Şekil 2.4 ve 2.5'te:

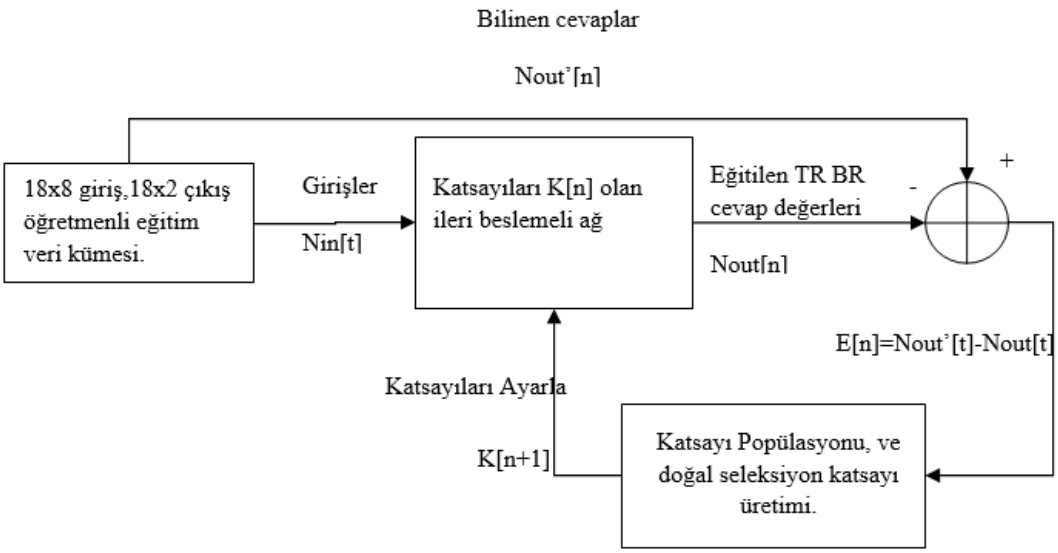
- (A, B, C, D, E, F, G, H): Sunulan yapının giriş vektörünü oluşturan şekillendirme parametreleri.
- W_iBR and W_iTR, i=1,2,3,4: Katman geçiş ağırlıkları. Ağırlıklar, ağın eğitimi sırasında ayarlanır.
- Logsig(x) and Poslin(x): Aktivasyon fonksiyonları. Aktivasyon fonksiyonu hücreye gelen net girdiyi alır ve bu girdiye karşılık üretilecek çıktıyı belirler.
- BBR and BTR: Bias değerleri. Bias terimi genellikle ağın yapısında toplam ağırlığın sıfır olmaması için kullanılan bir terimdir.
- BR(A,B,…….,G,H) and TR(A,B,…….,G,H): Çıktı cevap değerlerini simgeler.

"BR" ve "TR" terimlerinin analitik olarak tanımları aşağıdaki Denklemler (2.3) ve (2.4)'te verilmiştir.

$$BR = (W\_4BR \times \frac{1}{1+\exp\left(-\left(W\_3BR \times \frac{1}{1+\exp\left(-\left(W\_2BR \times \frac{1}{1+\exp(-(W\_1BR \times V))}\right)\right)}\right)\right)}) + BBR \qquad (2.3)$$

$$TR = (W\_4TR \times \frac{1}{1+\exp\left(-\left(W\_3TR \times \frac{1}{1+\exp\left(-\left(W\_2TR \times \frac{1}{(1+\exp(-(W\_1TR \times V)))}\right)\right)}\right)\right)}) + BTR \qquad (2.4)$$

Yukarıdaki fonksiyonlar belirlendikten sonra genetik algoritmalar kullanılarak Şekil 2.6'daki yapı eğitilip ağ ağırlıkları bulunmuştur.



Şekil 2.6. Evrimsel sinir ağının katsayılarının belirlenmesi.

Şekil 2.6'da:

- Nin[t] : Giren parametre değerleri.
- Nout[t]: Çıkan cevap değerleri.
- K[n]: Mevcut eğitilen network'un katsayılarını simgeler.

Populasyondaki her bireyin eğitim performansını belirlemek ve hataları azaltmak için aşağıdaki Denklem (2.5)'teki f_1 aktivasyon fonksiyonu kullanılmıştır.

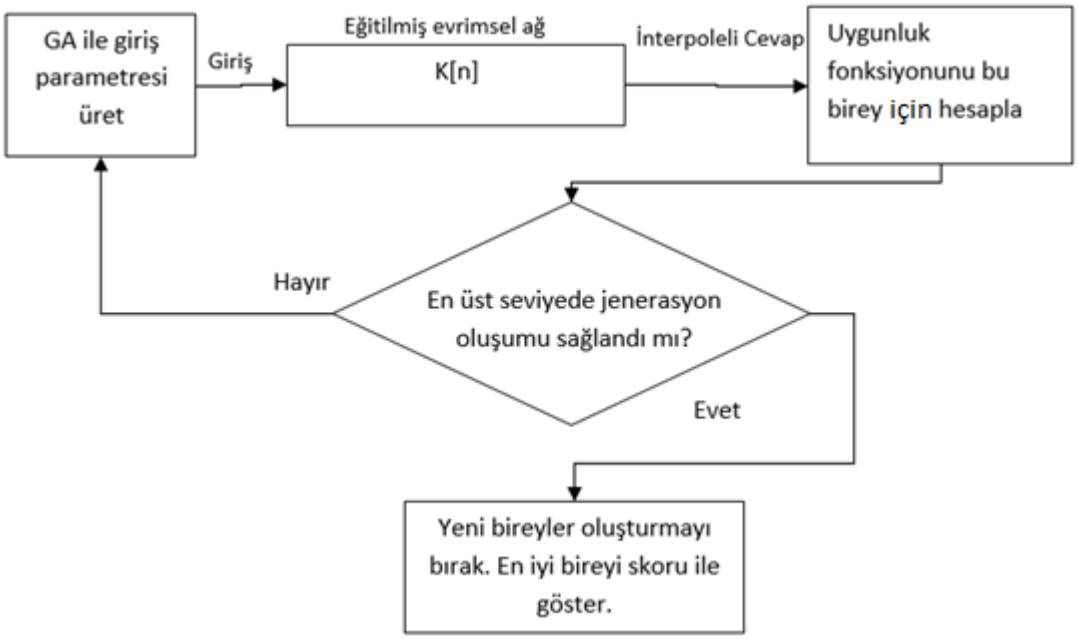
$$f_1 = \frac{\sqrt{\text{abs } E^2[n]}}{S} \qquad (2.5)$$

$E^2[n]$, hata terimlerinin karesinin toplamını, S ise eğitilecek veri kümesinin uzayını simgeler. İleri beslemeli evrimsel yapı eğitildikten sonra, giriş parametreleri esneme oranını arttırıp, incelme oranını azaltacak şekilde eniyilemesi yapılmıştır (Şekil 2.7).

Eniyileme süreci genetik algoritma içerdiğinden dolayı uygunluk fonksiyonu aşağıdaki Denklem (2.6)'daki gibidir.

$$f_2 = \alpha TR(A, B, C, D, E, F, G, H)^2 - (1 - \alpha)BR(A, B, C, D, E, F, G, H)^2 \quad (2.6)$$

- $TR(A, B, C, D, E, F, G, H)$: Eğitilen evrimsel yapının incelme oranını modelleyen fonksiyondur.
- $BR(A, B, C, D, E, F, G, H)$: Eğitilen evrimsel yapının esneme oranını modelleyen fonksiyondur.
- α : Çok amaçlı optimizasyon için önem katsayısı ($0 \leq \alpha \leq 1$).



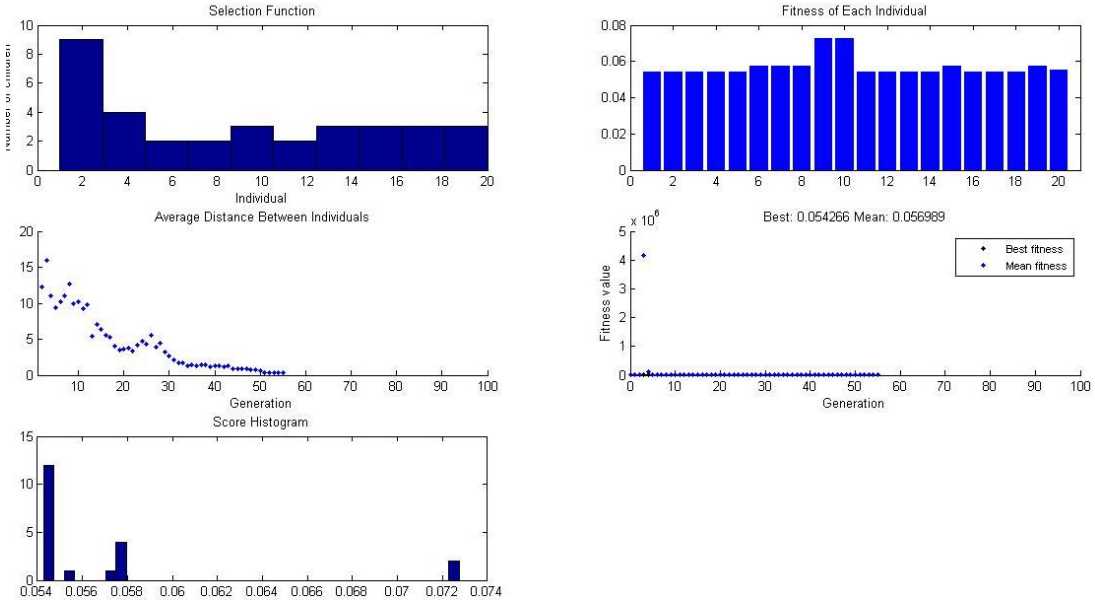
Şekil 2.7. Eğitilen modelin incelme ve esneme oranlarına göre olası optimum giriş parametrelerinin belirlenmesi.

2.4.3. Sonuçlar

Evrimsel sinir ağı yapısının kurulup, çalıştırılması için MATLAB 7.4.0 paket programının Genetik Algoritma tool box'ı kullanılmıştır. Eğitilen yapının katsayı populasyonu 2000 bireyden oluşmaktadır. Şekil 2.8'deki grafik, evrimsel sinir ağının

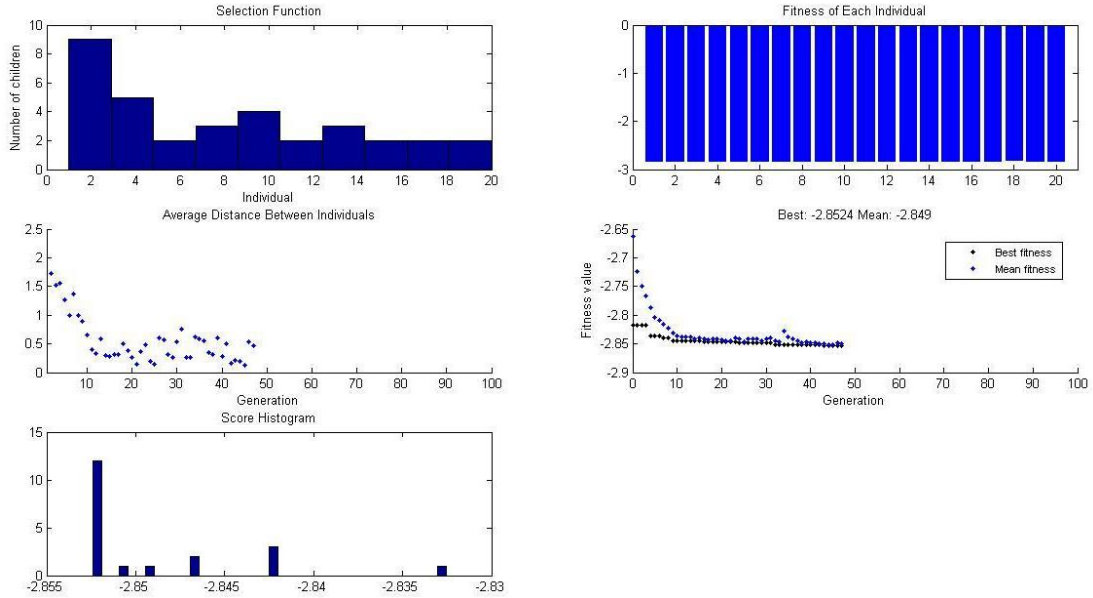
katsayılarının belirlenmesi sürecindeki populasyon ile ilgili genel bilgileri içermektedir.

Şekil 2.8'de "Selection Function" grafiği ilgili iterasyondaki bireylerin çocuk sayılarının dağılımını göstermektedir. "Fitness of Each Individual" grafiği her bir birey için uygunluk fonksiyonunun skorunu göstermektedir. "Average Distance Between Individuals" grafiği bireyler arasındaki öklit uzaklıklarının ortalamasını göstermektedir. "Fitness Value" grafiği her iterasyon için ortalama ve en iyi uygunluk değerlerini gösterir."Score Histogram" grafiğinde ise aynı skor aralığına düşen bireylerin sayıları verilmektedir.



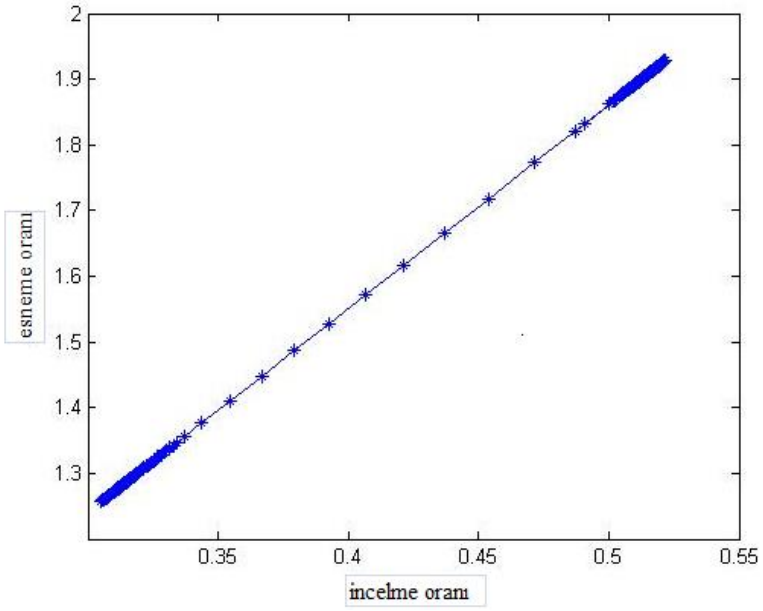
Şekil 2.8.Evrimsel sinir ağının katsayılarının belirlenmesi sürecindeki populasyon ile ilgili genel bilgileri içeren grafikler.

Şekil 2.9'daki grafik ise eğitilen modelin incelme ve esneme oranlarına göre olası optimum giriş parametrelerinin belirlenmesi sürecindeki populasyon ile ilgili genel bilgileri içermektedir. Yine bu şekildeki grafikler de Şekil 2.8'deki grafikler ile aynı olup buradaki arama uzayı giriş parametre değerleridir.



Şekil 2.9. Eğitilen modelin incelme ve esneme oranlarına göre olası optimum giriş parametrelerinin belirlenmesi sürecindeki populasyon ile ilgili genel bilgileri içeren grafikler.

Şekil 2.10 ise olası optimum incelme ve esneme oranlarının grafiğini vermektedir.



Şekil 2.10. Olası optimum incelme ve esneme oranları.

Evrimsel sinir ağları kullanılarak uygulanan deneysel yöntemin ve literatürde Metzger ve arkadaşları [83] tarafından L_9 ortogonal seri için Taguchi metot kullanılarak elde edilen olası optimum incelme ve esneme oranları sırasıyla Tablo 2.2 ve 2.3'te verilmiştir. Modellerin öngörü doğruluğunun ölçümünde kullanılan istatistiklerden

olan [84-85-86-87] ve formülü Denklem (2.7)'de verilen Ortalama Mutlak Yüzde Hata (Mean Absolute Percentage Error-MAPE) kriteri kullanılarak her iki metod ile elde edilen değerler kıyaslanmıştır.

$$\text{Ort. Mutlak hata } \% = \left| \frac{(\text{Deneysel Sonuç} - \text{Tahminleme sonucu}) \times 100}{(\text{Deneysel sonuç})} \right| \quad (2.7)$$

Tablo 2.2 incelendiğinde kurulan EANN model ile esneme oranı için ortalama mutlak hata değeri %2,044, incelme oranı için ise % 5,278 olarak bulunmuştur. Buna karşılık Tablo 2.3 incelendiğinde Taguchi metot kullanılarak elde edilmiş sonuçlara bakarsak esneme oranı ortalama mutlak hata değeri %2,27, incelme oranı için ise % 6,91 olarak bulunmuştur. Kıyaslama yaptığımızda EANN metodu ile elde edilen MAPE değerinin diğer metoda göre incelme oranı için % 0,226, esneme oranı için ise % 1,632 daha az çıktığı görülmektedir. Bunun sonucu olarak iki metot için yapılan kıyaslama sonucu olarak EANN metot ile elde edilen sonuçlar hata miktarı açısından daha az olduğu için Taguchi metoda göre daha etkin neticeler vermiştir.

Tablo 2.2. Kurulan EANN model ile elde edilen sonuçlar.

Test No	Esneme oranı için deneysel sonuçlar	Esneme oranı için EANN sonuçları	İncelme oranı için deneysel sonuçlar	İncelme oranı için EANN sonuçları	Esneme oranı için % hata miktarı	İncelme oranı için % hata miktarı
1	1,448	1,3419	0,284	0,281	7,327	1,056
2	1,449	1,362	0,304	0,307	6,004	0,987
3	1,492	1,4695	0,384	0,372	1,508	3,125
4	1,597	1,574	0,416	0,392	1,440	5,769
5	1,59	1,591	0,429	0,431	0,063	0,466
6	1,639	1,645	0,417	0,441	0,366	5,755
7	1,744	1,737	0,493	0,483	0,401	2,028
8	1,719	1,741	0,423	0,494	1,280	16,785
9	1,923	1,9231	0,477	0,532	0,005	11,530
% Ortalama Mutlak Hata					**2,044**	**5,278**

Tablo 2.3. Taguchi metot ile elde edilmiş sonuçlar.

Test No	Esneme oranı için deneysel sonuçlar	Esneme oranı için EANN sonuçları	İncelme oranı için deneysel sonuçlar	İncelme oranı için EANN sonuçları	Esneme oranı için % hata miktarı	İncelme oranı için % hata miktarı
1	1,448	1,412	0,284	0,293	2,468	3,169
2	1,449	1,419	0,304	0,303	2,070	0,329
3	1,492	1,424	0,384	0,311	4,558	19,010
4	1,597	1,591	0,416	0,399	0,376	4,087
5	1,59	1,603	0,429	0,411	0,818	4,196
6	1,639	1,638	0,417	0,433	0,061	3,837
7	1,744	1,767	0,493	0,479	1,319	2,840
8	1,719	1,787	0,423	0,493	3,956	16,548
9	1,923	1,831	0,477	0,516	4,784	8,176
% Ortalama Mutlak Hata					**2,27**	**6,91**

BÖLÜM 3. YAPAY ARI KOLONİSİ OPTİMİZASYON TEKNİĞİ

3.1. Giriş

Yapay arı kolonisi algoritması ilk olarak Karaboğa [88] tarafından geliştirilmiştir. Bu algoritma doğadaki arı kolonilerinin besin arama davranışlarından esinlenerek geliştirilmiştir. Yapay arı kolonisi algoritması sürü zekasını temel almaktadır ve doğada sürüler halinde aksiyon gösteren arıların yiyecek aramada göstermiş oldukları davranışları temel alarak optimizasyon problemlerini çözmekte kullanılmaktadırlar [40].

ABC algoritması kullanılarak birçok alanda çalışmalar yapılmıştır [89]. Banharnsakun ve arkadaşları [90] çalışmalarında büyük boyutlu problemler için ABC'nin dağıtık bir sürümünü önermişlerdir. Alam ve arkadaşları [91] sürekli fonksiyon optimizasyonu için ABC algoritmasını kullanmışlardır. Karaboğa ve Görkemli [92] ile Pandey ve Kumar [93] yaptıkları çalışmalarda gezgin satıcı problemlerine ABC algoritmasını uygulamışlardır. Guo ve arkadaşları [94] sayısal optimizasyon algoritmasında ABC'yi kullanmışlardır. Ning ve Zhang [95] bulanık sinir ağına dayalı bir konuşma tanıma sistemine ABC algoritmasını uygulamışlardır. Jadhav ve arkadaşları [96] yapay arı kolonisi algoritmasını rüzgar enerjisi geçişimli kombine emisyon görev dağıtımı problemi için kullanmışlardır. Çelik ve arkadaşları [97] kural tabanlı sınıflandırma modelinin optimizasyonu için ABC'yi önermişlerdir. Zakaria ve Rosni [98] biyoinformatik problemlerinde kullanılan Protein Yapısı Tahminlemesi (Protein Structure Prediction-PSP) için ABC'nin hibrit bir metodolojisini önermişlerdir. Khaze ve arkadaşları [99] ise ABC ile Ateşböceği Algoritmasını hibrit olarak kullanarak farklı sürekli optimizasyon problemlerinde bu metodolojinin ne kadar etkili olduğunu göstermeye çalışmışlardır. Brajevic [100], Maheshwari ve Dutta [101], Hedayatzadeh

ve arkadaşları [102] ile Sulaiman ve arkadaşları [103] çalışmalarında ABC algoritmasına farklı modifikasyonlar uygulayarak geliştirdikleri metodolojileri değişik kısıtlar içeren optimizasyon problemlerinin çözümü için uygulamışlar ve daha etkin sonuçlar elde ettiklerini ortaya koymuşlardır.

Ayat ve Sangar [104] ise ABC algoritması ile sınıflandırma algoritmalarından en çok kullanılan algoritmalardan olan k-Ortalama (k-Means) algoritmasını hibrit olarak kullanarak doğru maliyet tahmini yapmaya çalışmışlar ve kıyaslama kriteri olarak Ortalama Mutlak Göreceli Hatayı (Mean Absolute Relative Error) kullanmışlardır. Araştırmacılar geliştirdikleri metodun Yapıcı Maliyet Modeli (Constructive Cost Model-COCOMO)'ne göre %13 daha doğru maliyet tahminlemesi yaptığını ortaya koymuşlardır. Eke ve arkadaşları [105] çalışmalarında yapay arı kolonisi tabanlı Güç Sistemi Dengeleyicisi (Power System Stabilizer-PSS) kullanarak geniş alan çalışmalarında güç sistemlerinin kararlılığını artırmak için güç sistem sönümleme verimliliğini iyileştiren optimal PSS parametrelerinin elde edilmesini açıklamaya çalışmışlardır. Elde ettikleri sonuçlara göre tasarımı yapılan yapay arı kolonisi tabanlı PSS'in klasik ve PSO algoritması tabanlı PSS'lere göre daha hızlı cevap ile daha iyi sönümleme verdiğini göstermişlerdir.

Yıldız [106] çalışmasında taşıtlarda kullanılan salıncak kolunun topoloji optimizasyonu ve yapay arı koloni algoritması kullanılarak şekil optimizasyonunu yapmıştır. Araştırmacının elde ettiği sonuçlara göre, topoloji optimizasyonu ve yapay arı koloni algoritması taşıt elemanlarının optimum tasarımında etkin bir şekilde kullanılabilmektedir. Özyön ve arkadaşları [107] çalışmalarında çevresel ekonomik güç dağıtım probleminin çözümü için termik birimlerden oluşan kayıplı bir sisteme ABC algoritmasını uygulamışlardır. Küçüksille ve Tokmak [89] araştırmalarında bir üniversitedeki fakülteye ait ders çizelgeleme problemini ele almış ve yapay arı kolonisi algoritmasını kullanılarak ders çizelgelerini oluşturmuşlardır. Araştırmacılar programın sert kısıtlama tanımlarına tamamen uygun ders çizelgelerini oluşturmayı başardıklarını vurgulamışlardır. Sevim ve arkadaşları [108] yapay arı kolonisi algoritmasını kullanarak uzay çelik çerçevelerin ayrık optimizasyonunu üzerine çalışma yapmışlardır. Bu bağlamda araştırmacılar üç boyutlu çerçevenin optimum

tasarımı için nasıl kullanılacağı üzerinde durmuşlar ve optimum tasarımı için modellenmesi zor olan uzay çelik çerçeve için ABC algoritması kullanmışlardır. Araştırmacılar tasarım kısıtları göz önüne alınarak yaptıkları optimum tasarım sonuçları ile ABC algoritmasının başarılı bir optimizasyon yöntemi olduğunu gözlemlediklerini belirtmişlerdir.

3.2. Sürü Zekâsı

Karıncalar, arılar, termitler, balık sürüleri, kuşlar gibi aralarında belirli bir etkileşim bulunmakta olan hayvanların topluluklar şeklindeki hareketlerini örnek alan ve bunun sonucunda problemleri çözmeyi amaçlayan yapay zekâ tekniklerinden biri de sürü zekâsıdır. Arıların kovan etrafında dolaşmaları ve birbirlerine bilgi aktarımında bulunmaları, karıncaların ise gittikleri yollara "pheromone" denen kimyasal maddeleri salgılayarak diğerlerine bilgi aktarımında bulunmaları, balık ve kuş sürülerinin mevki ve süratlerini ayarlayarak ilerlemeleri de sürü zekâsını gösteren zeki davranışlardır.

Bir sürüde iki adet hayati vazife bulunur:

1. Self-organization da denilen kendi kendine organize olabilme
2. Mesai tanzimi.

Self-organization, bir sistemi oluşturan birimlerin diğer birimlerle etkileşimi sonucu elde ettikleri bilgileri kullanarak, kendi kendilerine faaliyet göstererek sistemin tümünü etkilemeleridir. Sistemde birimlerin birbirlerini etkilemelerinde komşuluk yapılarından veya bilgilerinden faydalanılır [109]. Bonabeau ile arkadaşları kendi kendine organize olabilme durumunu, negatif geri besleme (negative feedback), pozitif geri besleme (positive feedback), çoklu etkileşim ve salınım (fluctuation) olmak üzere toplam 4 adet özellik ile karakterize etmişlerdir [110]. Positive feedback daha elverişli yapıların bulunmasını sağlayan davranışlardır.

Pozitif geri beslemeye örnek olarak; karıncaların gittikleri yollara "pheromone" denen kimyasal maddeleri salgılaması ve diğerlerinin bu salgılanan maddeleri takip ederek

en elverişli yolları bulmaları örneği ile arıların kovanın değişik yerlerinde dans ederek nektar açısından zengin olan kaynaklar hakkında bilgileri diğerlerine aktarmaları verilebilir. Negatif geri beslemede toplanmış olan bilgilerin kararlı hale gelmesi için çalışma yapılır ve sürünün doyuma ulaşması engellenir. Çoklu etkileşim özelliği ile herhangi bir bireyin diğer başka bir bireye ait bilgiyi kullanabilmesi söz konusudur. Salınımlar ise yeni yiyecek kaynaklarının keşfedilebilmesi için rastgele dolaşımlardır.

Mesai tanzimi, toplulukta bulunan bütün bireylerin senkronize olarak çeşitli işleri gerçekleştirmeleridir. Topluluktaki bireylerin beraberce çalışarak gösterdikleri performans, herhangi bir mesai tanziminin söz konusu olmadığı bireylerin göstermiş oldukları performanstan daha müessir olmaktadır ve bu hususiyet çözüm uzayındaki değişmelere cevap verebilmeyi sağlamaktadır [109].

3.2.1. Doğadaki arı çeşitleri

Doğadaki arı çeşitlerine bakacak olursak, bal arıları içgüdüleri ile bir sosyal düzenin içinde yaşadıklarını iyi bilirler ve hayatlarını bu sosyal düzen içerisindeki kurallara uyarak devam ettirirler. Kovanda bulunan her bir arının yapması gereken görev bellidir. Arılar vazifeli oldukları bu sorumlulukların dışına asla çıkmazlar. Erzakların depo edilmesi, yiyecek arama, nektar getirme ve iletişim v.b. görevler sosyal düzen içerisindeki kendilerine ait sorumluluklardandır. Bu müthiş işleyiş ve denge araştırmacıların ilgisini çekmiş ve arıların bu davranışlarını modellemeye sevk etmiştir. Kolonilerde yaşamakta olan arılar üç farklı sınıf olarak nitelendirilmişlerdir. Bunlar:

- Kraliçe Arı (Queen Bee): Kraliçe arı her kovanda yalnızca bir tane bulunmaktadır. Kraliçe arı diğer dişilerden oldukça büyüktür. Kraliçe arının temel vazifesi yumurtlama işlevidir. Üremede asıl kaynak kraliçe arıdır, kraliçe arı dışında hiçbir dişi arı erkek arılar ile çiftleşme yapamazlar. Kraliçe arı çiftleşme sonucu depolamış olduğu spermleri kullanarak iki yıl süresince yumurtlayabilir. Yumurtlama işlevinden hariç, kovandaki sistemin işleyişini ve koloninin bütünlüğünü sağlayan mühim maddeler de salgılar. Salgılanan bu

maddeyi diğer bütün arılar tanımaktadır. Kraliçe arının salgıladığı bu koku kovandaki bütün arılara sirayet etmektedir. Yabancı olan herhangi bir arı bu salgılanan maddeler sayesinde fark edilir. Bu şekilde kraliçe arı kolonisinin bütünlüğünü sağlamış olur.

- Erkek Arılar (Drones): Bu arılar kolonide kraliçe arının döllenmesi görevini ifa ederler. Fiziksel olarak dişi arılardan büyüktürler. Fakat bu arıların iğneleri ve yiyecek toplayabilecek organları mevcut değildir.
- İşçi Arılar (Workers): İşçi arılar bir arı kolonisinde sayısal olarak çoğunluğu teşkil ederler. Bu arılar döllenmiş olan yumurtalardan çıkarlar ve üreme yetenekleri olmayan dişi arılardır. Bu arılar kolonin devamının sağlanması için gerekli olan her türlü yapısal ve içgüdüsel yeteneklere haizdirler. İşçi arılar aralarında iş bölümü yaparak farklı işleri düzenli bir şekilde yaparlar. Yine işçi arılar yiyecek toplamak ve bunları depolamak, kovanın her türlü temizliğini sağlamak, larvalar ve diğer arıların beslenmesi gibi birçok görevi ifa ederler.

Çiftleşme Uçuşu (Mating Flight), genellikle kraliçe arı bir grup işçi arı tarafından çevrelenip özel olarak beslenir ve temizliği yapılır. Kraliçe arı gelişimi tamamlandıktan sonra takribî bir hafta süresinde kovandan uzaklaşarak çiftleşmek hedefiyle uçuş gerçekleştirir. Bu uçuşlar sayesinde çiftleşme gerçekleştirilmiş olur ve spermler kraliçe arının spermatica isimli keseciğinde toplanır. Bu şekilde arı kolonisinin devamı sağlanmış olur [111].

3.2.2. Görev ve bilgi paylaşımı

Arı kolonisinde çok sayıdaki çeşitli birçok iş için olurlu sayıda bireylerin belirtilen bu işlere atanması gereklidir [112]. Bir arı için büyüme süreci, kraliçe arının yumurtlaması ile başlamaktadır. Yumurta tedricen larva ve pupaya dönüşür ve bu süreçteki arıların beslenmesi, beslenme alanlarından çıktıktan sonraki görevleri sürecinde de etkilidir. Bu sürecin akabinde yumurtadan çıkmış olan arı kendisi için atanan görevleri ifa etmek için hayatına başlayacaktır. Yumurtadan çıkarak yetişkin bir birey olana kadar olan süreçte herbir arının depolama, besleme, besin arama, dağıtım ve iletişim gibi birçok görevi vardır. Bir arının herhangi bir anda hangi işi

gerçekleştireceği o andaki aksiyonel görevine, çevresinden toplamış olduğu algılara ve bu algılara göstermiş olduğu tepkiye bağlıdır.

Yapılacak işlere göre arılar arasında bir iş bölümü vardır ve herhangi bir merkezi otorite olmadan bu iş dağılımını gerçekleştirdikleri için kendi kendine organize olabilmektedirler. İş bölümü yapabilme ve kendi kendilerine organize olabilme özelliği sürü zekâsındaki iki hayati bileşendir. Kolektif zekânın ortaya çıkmasını sağlamakta olan minimal besin arama modelinde başlıca üç bileşen vardır: yiyecek kaynakları, görevi belirli işçi arılar ve görevi belirsiz işçi arılar. Bu minimal model besin kaynağına yönlenme ve kaynağı bırakma olarak iki modda çalışmaktadır. Bileşenleri şu şekilde açıklayabiliriz:

a) Yiyecek Kaynakları: Arıların nektar, polen veya bal elde etmek için gittikleri yerlerdir. Bir besin kaynağının değeri, türü, yuvaya olan yakınlığı, besin konsantrasyonu veya besinin bulunmasının kolaylığı tek bir kriter olarak alınabilir.
b) Görevi Belirli İşçiler: İşçi arılar, daha önceden keşfi yapılan belirli kaynaklara ait nektarın kovana getirilmesinden sorumludurlar ve aynı anda ziyaret etmiş oldukları kaynakların kalitesi ve yeriyle ilgili bilgiyi kovanda bekleyen diğer arılarla paylaşırlar. Bundan sonra görevi belirli işçi arılar için görevli arılar ifadesi kullanılacaktır.
c) Görevi Belirsiz İşçi Arılar: Bu arılar nektarını toplayabilecekleri kaynak arayışı içerisindedirler. Görevi belirsiz iki çeşit işçi arı bulunmaktadır. Dâhilî bir dürtüyü veya bir hârici etmene bağlı rastgele kaynak arayışı içerisinde olan kâşif arılar ve kovanda bekleyen ve görevli olan arıları gözetleyerek bu arıların paylaşmış oldukları bilgileri kullanarak yeni bir nektar kaynağına yönelen gözcü arılar. Kâşif arıların sayısının tüm koloniye oranı ortalama %5-10 arasındadır.

Arılar arasında bilgi paylaşımı kolektif bilginin oluşumunda en önemli husustur. Bir kovan göz önüne alındığında tüm kovanlarda ortak olan bazı parçalara ayırmak mümkündür ve bu parçalardan en önemlisi is dans alanıdır. Yiyecek kaynağının

kalitesi ve yeri ile ilgili bilgi paylaşımı dans alanında olmaktadır. Bir arı dans ederken diğer arılarda ona antenleri ile dokunurlar ve bulduğu kaynağın tadı ve kokusu ile ilgili de bilgi alırlar. Ziyaret ettikleri kaynağa daha fazla arı yönlendirebilmek için kovandaki çeşitli alanlarda bu dansı gerçekleştirir ve kaynağına geri döner.

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

Yiyecek getirenlerin diğer arılara, uzak noktadaki bir kaynağa yönlendirmek için hedefe ait yön bilgisini vermeleri gerekir. Yön bilgisi alındıktan sonra hedefe ulaşmada güneşten faydalanılır. Bileşik gözleri ile arılar kendi yörüngeleri ile güneş arasında açıyı hesaplayabilmektedir. Arılar, güneş kapanmış olsa dahi polarize gün ışığından yine güneşin konumunu tayin edebilmektedir. Arılar, bir noktanın uzaklığını enerji tüketimine bağlı olarak hesaplamakta ve yüklerine göre farklı yükseklikte uçarak enerji tüketimlerini ayarlamaktadır [3].

3.2.3. Arılarda dans

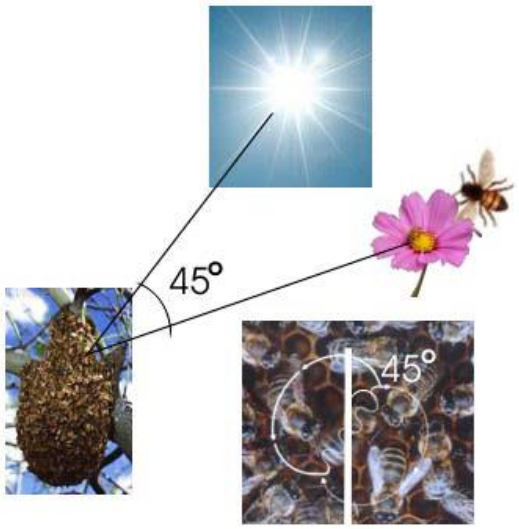
Kaynağın kovana olan mesafesine göre çeşitli danslar mevcuttur: dairesel dans (round dans), kuyruk dansı (waggle dans) ve titreme dansı (tremble dance). Daire ve kuyruk dansları yiyecek getiricilerin yeniden aktivasyonunda etkilidir. Bu iki dans farklı uzaklıktaki bölgelerin ayırımında kullanılır. Daire dansı ile belirtilen yiyecek kaynağının kovana olan uzaklığı maksimum 50-100 metre civarında olduğundan bu dans yön ve uzaklık bilgisi vermemektedir.

Titreme dansında, arıların petek üzerinde düzensiz tarzda ve yavaş tempoda bacaklarını titreterek ileri, geri, sağa ve sola hareketleri söz konusudur. Arı zengin bir

nektar kaynağı bulduğunu, ancak kovana işlenebileceğinden fazla nektar geldiğini ve bundan dolayı nektarı işleme görevini geçmek istediğini belirtmektedir. Sadece dans alanında değil kovanın başka bölümlerinde de bu dans gerçekleştirilmektedir. Bu dansın amacı kovan kapasitesi ve yiyecek getirme aktivitesi arasındaki dengeyi sağlamaktadır.

Arılar 100 metre-10 kilometre arası mesafe kadar olan büyük bir alanda bulunan kaynaklar için ilgili bilgi aktarımında kuyruk dansını kullanmaktadırlar. Bu dans türü sekiz rakamına benzerlik gösteren bir dans türüdür. Bu dansı izlemekte olan arıların bir titreşim oluşturulması sonucunda bu dansı yapmakta olan arı, bu dansına son verir. Yapılan bu dans ile ilgili olarak 15 saniyede bir bu dansın tekrarlanma sayısı, nektar kaynağının uzaklığı ile ilgili bilgi verir. Daha az olan tekrarlanma sayıları besin bölgelerinin daha uzak bölgeler olduğunu belirtir. Arılarda yön bilgisi Şekil 3.1'deki gibi 8 rakamı şeklindeki dansın açı bilgisi ile elde edilir. Şekilde verilen örnekte ise dansı izleyen arılar, izledikleri danstan güneş ile nektar arasındaki açının 45 derece olduğunu anlarlar.

Tüm zengin kaynaklarla ilgili bilgiler dans alanında gözcü arılara iletildiğinden gözcü arılar birkaç dansı izledikten sonra hangisini tercih edeceğine karar verir. Zengin kaynaklarla ilgili daha fazla bilgi aktarımı olduğundan bu kaynakların seçilme olasılığı daha fazladır [3].



Şekil 3.1. Arılarda dans [3].

3.3. Arıların Nektar Bulma Davranışları

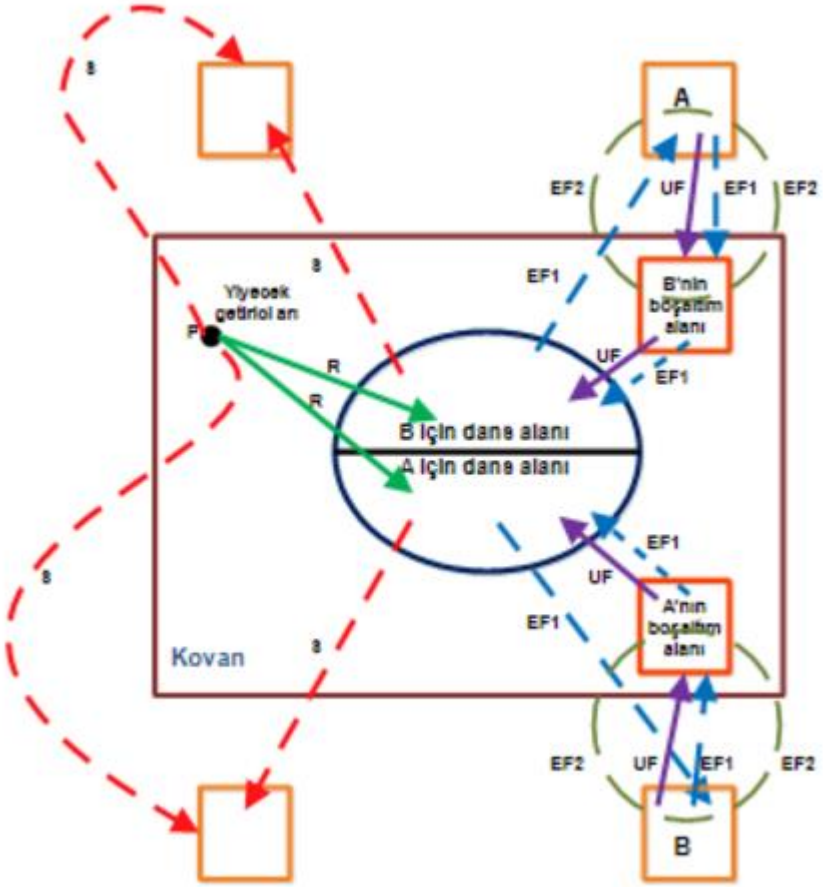
Besin arama faaliyeti, arı kolonisinde hayatın devam edebilmesi için en önemli işlevlerden biridir. Kovanın içerisindeki toplanan kaynaklar ile tabiattan keşfedilecek yeni nektar membaları ve arıların birbirleri ile etkileşimleri süreçteki en mühim etkenlerdir. Besini arama süreci, arının kovandan ayrılmasıyla başlar. İlk etapta besin aramaları rasgele yapılır. Daha sonra keşfedilen kaynaktaki besin miktarının azalması sonucunda arılar başka nektar araştırmaya başlarlar veya edindikleri bilgilere göre diğer kaynaklara yönelirler. Bulunmuş olan su, polen gibi membaların kovana getirilmesi ve yine bulunmuş olan kaynakların bilgilerinin arılarca birbirine iletilmesi süreç içerisindeki yapılan faaliyetlerdendir [111].

Müşterek bilgi ve kollektif yapının teşkilindeki en hayati konu arıların kendi aralarında bilgiyi iletebilmesidir. Arıların yaşamlarını sürdürdükleri yer olan kovan çeşitli kısımlara ayrılır. Bu kısımlardan biri de arılar arasındaki bilgi paylaşımının gerçekleştirildiği "dancing area" olarak adlandırılan dans alanıdır. Arılar "waggle dance" olarak adlandırılan dans sayesinde birbirleri arasındaki bilginin paylaşımını veya iletimini gerçekleştirirler. Paylaşılan bu bilgiler ile daha yeni ve kaliteli besin kaynakları keşfedilmektedir [113].

Arıların besin keşfetme ve kovana getirme davranışları Şekil 3.2'de gösterilmiştir. İlgili davranışların şekil üzerindeki açıklama ve incelemeleri Akay tarafından şu şekilde yapılmıştır: A ile B bölgesi keşfedilmiş kaynak bölgeleri olarak varsayılmaktadır. İlk olarak kaynak bilgisi kendisinde olmayan ve görevi daha belli olmayan arı arama işlevine başlayacaktır. Bahsi geçen arı için iki durum söz konusudur: İlk durumda; S rotası ile tarif edilen bu arı kâşif arı olabilmekte ve nektar aramaya başlayabilmektedir. İkinci durum ise; Kovanda dans hareketini gerçekleştiren arıları izleyerek bilgisi aktarılan besin kaynaklarına giderek gözcü arı olabilir. Bu arı şekilde R ile nitelenmiştir. İlgili besin kaynaklarına giden arılar bu besin kaynaklarından nektar getirmeye başlarlar [3].

Anlatılan süreçten sonra ilgili arılar görevli arılar haline gelmiştirler. Besini kovana getirmiş olan arı için bundan sonraki süreçte üç alternatif durum vardır:

1) Şekilde EF2 ile gösterilen durumda ilgili arı herhangi bir bilgi paylaşmadan kaynaktan kovana yiyecek getirmeye devam edebilir.
2) EF1 ile şekilde gösterilmiş olan diğer bir durumda ise arı kaynağa dönmeden önce dans hareketini yaparak diğer arılara nektar kaynağının konumu ve miktarı hakkında bilgi vererek ilgili besin kaynağına yönlendirmede bulunabilir.
3) Şekilde UF ile gösterilmiş son durumda ise besin kaynağını terk eden arı dans alanında gözcü arı olabilir [3].



Şekil 3.2. Arıların besin arama davranışları [3].

3.4. Yapay Arı Kolonisi Algoritması

Karaboğa [88] arıların koloniler olarak zeki davranış göstermelerini ve yiyecek keşif sürecindeki davranışlarını modelleyerek Yapay Arı Kolonisi ismiyle bir yeni optimizasyon algoritması geliştirmiştir. Bu algoritma global ve lokal uzayı komşuluk prensibine göre araştırır. Kolonideki bulunan arılar üç farklı grup olarak nitelenir [3]:

1. İşçi Arılar: İşçi arılar komşuluk prensibine dayanarak daha fazla nektarın olduğu besin kaynaklarını araştırırlar. Her bir besin kaynağında bir işçi vardır. Dolayısıyla işçi arı sayısı besin kaynağı sayısına eşittir.
2. Gözcü Arılar: Gözcü arılar kovanda bekler ve diğer arıların besin kaynakları ile bilgileri dansla kendileriyle paylaştıktan sonra nektarın fazla olduğu besin kaynağına yönelirler.
3. Kâşif Arılar: Yiyecek arama sürecinin başlangıcında kâşif arılar rastgele dağılarak yiyecek aramaya başlarlar.

Geliştirilmiş olan bu algoritmada, toplam besin kaynağı sayısı toplam görevli arı sayısına eşit olarak kabul edilir. Ayrıca gözcü arıların sayısı da işçi arıların sayısına eşittir. Besin kaynakları için görevli olan arılar kaynaktaki besin miktarı bittiğinde kâşif arı olmaktadırlar. Doğadaki besin kaynaklarının konumları, çözümü yapılması planlanan problemin olası çözümlerine, yiyecek miktarı da çözümün ne derece kaliteli olduğunu belirtir. Algoritma problemin minimum veya maksimum noktasını bulabilmek için çözüm uzayında en çok nektara sahip kaynağın konumunu bulmaya çalışır.

Nektar arama sürecinde ilk olarak kâşif arılar rasgele besinleri aramaya başlar. Nektar kaynağını bulan kâşif arı görevli arı haline gelmekte ve kovana yiyecek götürmeye başlamaktadır. Görevli arı kovanına besinleri taşır. Daha sonra kovanda bekleyen gözcü arılara dans ederek yiyecek getirdiği kaynak ile ilgili bilgiyi iletir. Eğer kaynaktaki bulunan yiyecek miktarı bitmişse kendisi de gözcü arı haline gelmektedir. Gözcü arılar ise kendilerine yapılan dansları izleyip yiyecek kaynaklarının kalitesine göre herhangi bir kaynağı tercih ederek o kaynağa doğru yönelirler [112].

Algoritmanın temel adımları ise şu şekildedir [3]:

1. Başlangıç yiyecek kaynağı bölgelerinin üretilmesi.
2. İşçi arıların yiyecek kaynağı bölgelerine gönderilmesi.
3. Olasılıksal seleksiyonda kullanılacak olasılık değerlerinin görevli arılardan gelen bilgiye göre hesaplanması.
4. Gözcü olan arıların olasılık değerleri doğrultusunda besin kaynağı bölgelerini seçmeleri.
5. Kaynağı bırakma kriteri: limit ve kâşif arı üretimi.
6. Çevrim sayısı=Maksimum çevrim sayısı.

Algoritmada her bir döngü ise üç temel adım içermektedir:

a) Gözcü ve işçi arıların nektar kaynaklarına gönderilmeleri,
b) İlgili kaynaklardaki besin miktarlarının hesaplanması,
c) Kâşif arının belirlenmesi ve yeni bir nektar kaynağına rastgele konumlanması.

Optimizesi yapılmaya çalışılan problemin muhtemel çözümleri, her bir nektar kaynağına karşılık gelmektedir. Bir nektar kaynağındaki yiyecek miktarı, ilgili kaynak ile ifade edilen çözümün kalite değerini belirtmektedir. Her kolonideki rasgele araştırma yapan kâşif arılar yiyecek ararlarken herhangi bir ön bilgi kullanmazlar ve tamamıyla rastgele araştırma yaparlar. Bu arıların çok zengin besin kaynakları bulmaları da muhtemeldir. Ayrıca belli sayıda deneme sonucu geliştirilemeyen bir nektar kaynağı terk edilir ve bu kaynağa gidip gelmek ile işlevi olan işçi arılar kâşif arıya dönüşür [109].

3.4.1. Başlangıç yiyecek kaynağı bölgelerinin üretilmesi

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

$$x_{ij} = x_{\min j} + rand(0,1) * (x_{\max j} - x_{\min j}) \quad (3.1)$$

Bu denklemde i=1 …SN, j=1…D ve SN yiyecek kaynağı sayısı ve D ise optimize edilecek parametre sayısıdır. X_{min} j, ise parametrelerin alt sınırı, X_{max} j, parametrelerin üst sınırıdır [3].

3.4.2. İşçi arıların yiyecek kaynağı bölgelerine gönderilmesi

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

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) \quad (3.2)$$

x_i ile gösterilen her bir kaynak için bu kaynağın yani çözümünün tek bir parametresi (rastgele seçilen parametresi, j) değiştirilerek x_i komşuluğunda v_i kaynağı bulunur. ϕ_{ij} ise [-1 1] arasında rastgele değer alan ağırlık faktörüdür.

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

Bu işlem sonucunda üretilen $v_{i,j}$ 'nin daha önceden belli olan parametre sınırları aşması durumunda j parametresine ait olan alt veya üst sınır değerlerine ötelenmektedir (Denklem 3.3):

$$v_{ij} = \begin{cases} x_j^{\min}, & v_{ij} < x_j^{\min} \\ v_{ij}, & x_j^{\min} \le v_{ij} \le x_j^{\max} \\ x_j^{\max}, & v_{ij} > x_j^{\max} \end{cases} \quad (3.3)$$

Sınırlar dahilinde üretilen v_i parametre vektörü yeni bir kaynağı temsil etmekte ve bunun kalitesi hesaplanarak bir uygunluk değeri atanmaktadır (Denklem 3.4).

$$fitness_i = \begin{cases} 1/(1+f_i) & f_i \ge 0 \\ 1/abs(f_i) & f_i < 0 \end{cases} \quad (3.4)$$

Burada f_i, v_i kaynağının yani çözümünün maliyet değeridir. x_i ile v_i arasında nektar miktarlarına yani uygunluk değerlerine göre bir açgözlü seçme işlemi uygulanır. Yeni bulunan v_i çözümü daha iyi ise görevli arı hafızasından eski kaynağın yerini silerek v_i kaynağının yerini hafızaya alır. Aksi takdirde görevli arı x_i kaynağına gitmeye devam eder ve x_i çözümü geliştirilemediği için x_i kaynağı ile ilgili geliştirememe sayacı ($failure_i$) bir artar, geliştirdiği durumda ise sayaç sıfırlanır [3].

3.4.3. Gözcü arıların seleksiyonda kullanacakları olasılık değerlerinin hesaplanması

Tüm görevli arılar bir çevrimde araştırmalarını tamamladıktan sonra kovana dönüp buldukları kaynakların nektar miktarları ile ilgili gözcü arılara bilgi aktarırlar. Bir gözcü arı dans aracılıyla paylaşılan bilgiden faydalanılarak yiyecek kaynaklarının nektar miktarları ile orantılı bir olasılıkla bir bölge(kaynak) seçer. Bu durum ABC 'nin altında çoklu etkileşim sergilendiğinin bir örneğidir. Olasılıksal seçme işlemi, algoritmada nektar miktarlarına karşılık gelen uygunluk değerleri uygulanarak yapılmaktadır. Uygunluk değerine bağlı seçme işlemi rulet tekerleği, sıralamaya dayalı, stokastik örnekleme, turnuva yöntemi ya da diğer seleksiyon şemalarından herhangi biri ile gerçeklenir. Temel ABC algoritmasında bu seleksiyon işlemi rulet tekerliği kullanılarak yapılmıştır. Tekerlekteki her bir dilimin açısı uygunluk değeri toplamına oranı o kaynağın diğer kaynaklara göre nispi seçilme olasılığı olduğunu vermektedir (Denklem 3.5).

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

Burada kaynağın kalitesini gösteren SN, ayın zamanda görevli arı sayısını da göstermektedir. Bu olasılık hesaplama işlemine göre bir kaynağın nektar miktarı arttıkça (uygunluk değeri arttıkça) bu kaynak bölgesini seçecek gözcü arı sayısı da artacaktır. Bu özellik ABC' nin pozitif geri besleme özelliğine karşılık gelmektedir [3].

3.4.4. Gözcü arıların yiyecek kaynağı bölgesi seçmeleri

Algoritmada olasılık değerleri hesaplandıktan sonra rulet tekerliğine göre seçim yapılarak her bir kaynak için [0-1] aralığında rastgele sayı üretilir ve p_i değeri bu üretilen sayıdan büyük ise görevli arılar gibi gözcü arı da (Denklem 3.2)' yi kullanarak bu kaynak bölgesinde yeni bir çözüm üretir. Yeni çözüm değerlendirilir ve kalitesi hesaplanır. Sonra yeni çözümle eski çözümün uygunluklarının karşılaştırıldığı en iyi olanın seçildiği açgözlü seleksiyon işlemine tabi tutulur. Yeni çözüm daha iyi ise eski çözüm yerine bu çözüm alınır ve çözüm geliştirememe sayacı (failure) sıfırlanır. Eski çözümün uygunluğu daha iyi ise bu çözüm muhafaza edilir ve geliştirememe sayacı (failure) bir artırılır. Bu süreç tüm gözcü arılar yiyecek kaynağı bölgelerine dağılama kadar devam eder [3].

3.4.5. Kaynağı bırakma kriteri: limit ve kâşif arı üretimi

Bir çevrim sonunda tüm görevli ve gözcü arılar arama süreçlerini tamamladıktan sonra çözüm geliştirememe sayaçları (failure) kontrol edilir. Bir arının bir kaynaktan faydalanıp faydalanmadığı, yani gidip geldiği kaynağın nektarının tükenip tükenmediği çözüm geliştirememe sayaçları aracılığıyla bilinir. Bir kaynak için çözüm geliştirememe sayacı belli bir eşik değerinin üzerindeyse artık bu kaynağın görevli arısının tükenmiş olan o çözümü bırakıp kendisi için başka bir çözüm araması gerekir. Bu da biten kaynakla ilişkili olan görevli arının kâşif arı olması anlamına gelmektedir. Kâşif arı haline geldikten sonra, bu arı için rastgele çözüm arama süreci başlar (Denklem 3.1). Kaynağın terk ettiğinin belirlenmesi için kullanılan eşik değeri ABC

algoritmasının önemli bir kontrol parametresidir ve “limit” olarak adlandırılmaktadır. ABC algoritmasında her çevrim için yalnızca kâşif arının çıkmasına izin verilir. Tüm bu birimler arasındaki ilişki ve döngü, Şekil 3.3’ teki gibi bir akış diyagramı ile gösterilebilir [3].

3.4.6. Seleksiyon mekanizmaları

ABC algoritması 4 farklı seleksiyon işlemi kullanmaktadır. Bunlar ;

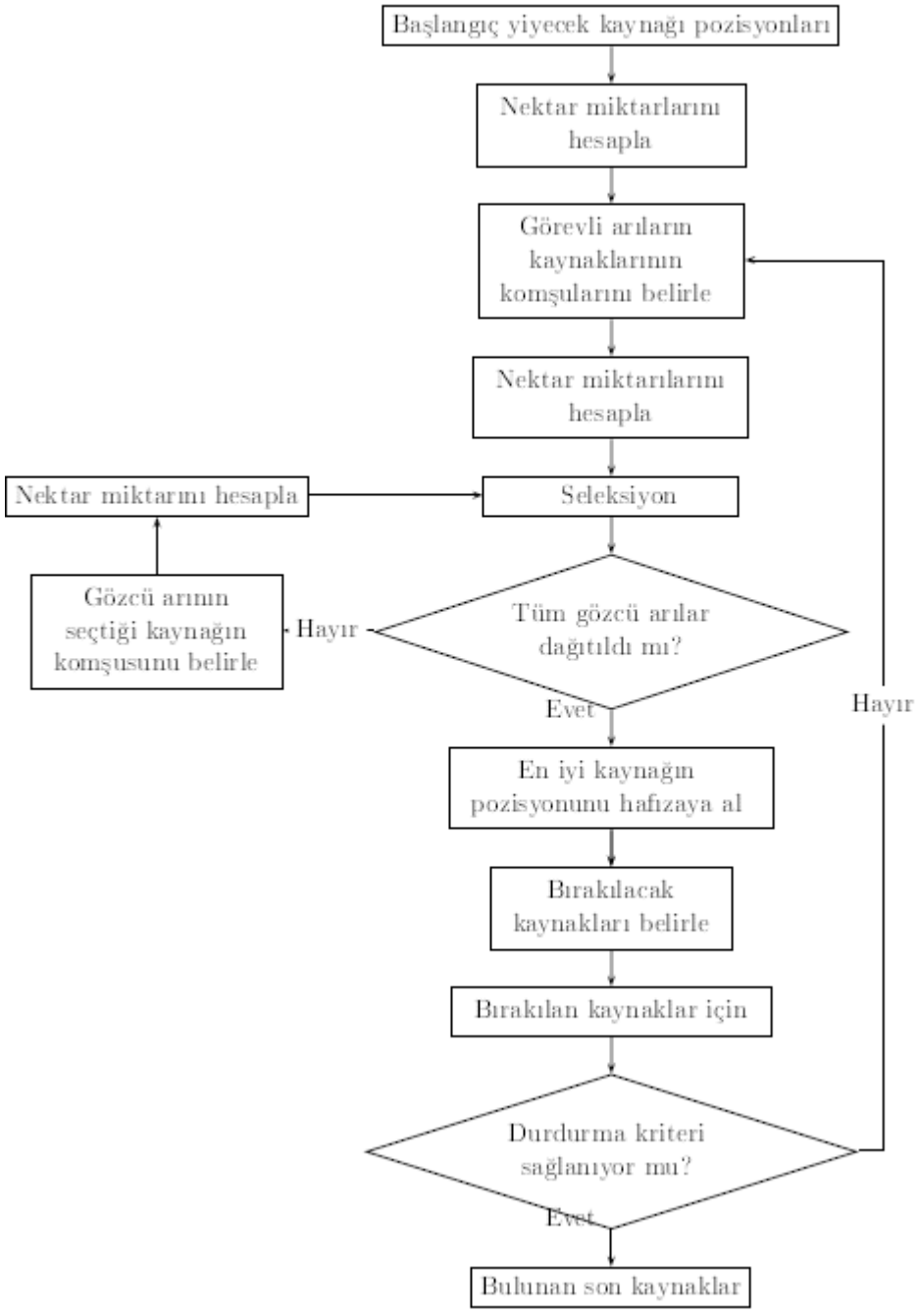
1. Potansiyel iyi kaynaklarının belirlenmesine yönelik Denklem 3.5 olasılık değerlerinin hesaplandığı global olasılık temelli seleksiyon süreci.
2. Görevli ve gözcü arıların renk, şekil, koku gibi nektar kaynağının turunu belirlenmesi sağlayan görsel bilgiyi kullanarak bir bölgede kaynağın bulunmasına vesile olan bölgesel olasılık tabanlı seleksiyon işlemi (Denklem 3.2).
3. İşçi ve gözcü arıların daha iyi olan kaynağı belirlemek amacıyla kullandıkları açgözlü seleksiyon.
4. Kâşif arılar tarafından (Denklem 3.1)’deki gerçekleştirilen rastgele seleksiyon.

Bütün bu seleksiyon metotların bir arada kullanılmasıyla ABC algoritması hem iyi bir global araştırma hem de bölgesel araştırma yapabilmektedir [3].

3.4.7. ABC’nin temel özellikleri

ABC algoritması ;

- Algoritma esnek ve basittir.
- Doğadaki besin arayan arıların hareketlerini oldukça yakın olarak simule eder.
- Sürü zekâsı tabanlı bir algoritmadır.
- Nümerik şekildeki problemler için geliştirilmiş olan bir algoritmadır. Fakat ayrık olan problemler için de kullanılabilmektedir.
- Algoritma nispeten daha az kontrol parametresine sahiptir.



Şekil 3.3. ABC Algoritması [3].

BÖLÜM 4. YAPAY ARI KOLONİSİ ALGORİTMASI İLE EVRİMSEL ALGORİTMALARIN (ABC-EA) BÜTÜNLEŞİK YAKLAŞIMI

4.1. Atölye Tipi Çizelgeleme Problemleri

Çizelgeleme problemleri üretim, ulaştırma, dağıtım, mühendislik, inşaat, yönetim gibi birçok alanda yoğun bir şekilde rastlanan ve endüstri mühendisliğinin yoğun bir şekilde çalıştığı alanlardandır. Üretim alanında atölye tipi çizelgeleme problemleri ve Akış Tipi Çizelgeleme (Flow Shop Scheduling-FSS) problemleri üzerinde en çok çalışma yapılmış ve yapılmakta olan çizelgeleme problemleridir. Akış tipi çizelgeleme problemlerinde her parça için işlerin gideceği makinelerin sırası aynı olmaktadır. Atölye tipi çizelgeleme problemlerinde ise, her iş için değişik makine rotaları mevcuttur. Özellikle, imalat ve üretim sektöründeki çizelgeleme ile ilgili problemler karışık yapıda olmakta ve klasik eniyileme metotları ilgili problemleri çözmek için kifayetsiz kalmaktadır [114].

Çizelgeleme problemleri için işlerin ve makinelerin sayısı çoğaldıkça, yapılan birçok çizelgeleme arasından en iyisinin seçilebilmesi epeyce karışık bir durum olabilmektedir [115]. Bu durumun sebebi ise üstel biçimde çözüm uzayının genişlemesi ve hesaplama zamanlarındaki ciddi artışlardır. Örnek vermek gerekirse; n tane iş ve m adet makineden teşkil bir atölye çizelgeleme probleminde tüm mümkün çizelgelemerin sayısı $(n!)^m$ olarak hesaplanabilmektedir [116].

Küçük boyutlu olan çizelgeleme problemlerinde çeşitli teknikler ile optimal çizelgelemeleri bulabilmek mümkün olabilmektedir. Ancak problemin boyutu arttıkça, optimal çizelgelemelerin hızlı bir şekilde elde edilebilmesi çok zor hale gelebilmektedir [117].

Bundan dolayı atölye çizelgeleme problemlerinin çözümü için birçok sezgisel yöntemler önerilmiştir. Tavlama benzetimi algoritması [118, 119], tabu araştırma algoritması [120, 121], karınca kolonisi optimizasyon algoritması [122, 123, 124], Memetik Evrim (Memetic Evolutionary-ME) algoritması [125], parçacık sürüsü optimizasyon algoritması [126, 127, 128], yapay arı kolonisi algoritması [129, 130, 131], paralel kanguru algoritması [132], kril sürüsü [114], Guguk Kuşu Arama (Cuckoo Search-CS) algoritması [133, 134, 135], Ateşböceği Algoritması [136, 137], Balık Sürüsü (Fish Swarm-FS) algoritması [138], kedi sürüsü optimizasyon algoritması [139], Genetik Algoritma [140, 141, 142], diferansiyel gelişim algoritması [143, 144], yerçekimi arama algoritması [145, 146], armoni arama algoritması [147, 148, 149] gibi birçok yöntem geliştirilmiştir.

Sezgisel algoritmalar, atölye çizelgeleme problemindeki olduğu gibi, gerçek hayat problemlerinde de oldukça başarılı sonuçlar verdiği için birçok araştırmacının ilgisini çekmiştir. Araştırmacılar genellikle doğadaki canlıların davranışlarından esinlenerek geliştirilmiş bu algoritmaları atölye tipi çizelgeleme problemlerinde hibrit olarak kullanarak daha da iyi neticeler elde etmeye çalışmışlardır. Bundan dolayı araştırmacılar atölye çizelgeleme problemlerinin çözümü için birçok hibrit yöntemler önermişlerdir. Literatürde kullanılmış olan sezgisel algoritmaların hibrit olarak kullanılması sonucu geliştirilen algoritmalardan bazıları şunlardır: Genetik Algoritma-Karınca Kolonisi Optimizasyon Tekniği (GA-ACO) hibrit algoritması [150, 151, 152], Genetik Algoritma-Parçacık Sürüsü Optimizasyon Tekniği (GA-PSO) hibrit algoritması [153, 154], Genetik Algoritma-Tavlama Benzetimi (GA-SA) hibrit algoritması [155, 156, 157], Karınca Kolonisi Optimizasyon Tekniği-Parçacık Sürüsü Optimizasyon Tekniği (ACO-PSO) hibrit algoritması [158, 159], Parçacık Sürüsü Optimizasyon Tekniği-Tabu Search (PSO-TS) hibrit algoritması [160, 161], Karınca Kolonisi Optimizasyon Tekniği-Guguk Kuşu Arama (ACO-CS) hibrit algoritması [162, 163], Bakteriyel Beslenme Optimizasyon tekniği–Karınca Kolonisi Optimizasyon Tekniği (BFO-ACO) hibrit algoritması [164, 165], Bakteriyel Beslenme Optimizasyon Tekniği–Armoni Arama (BFO-HS) hibrit algoritması [166, 167], Paralel Kanguru Algoritması-Karınca Kolonisi Optimizasyon Tekniği (BFO-ACO) hibrit algoritması [168], Tabu Arama-Ateşböceği Algoritması (TA-FA) hibrit

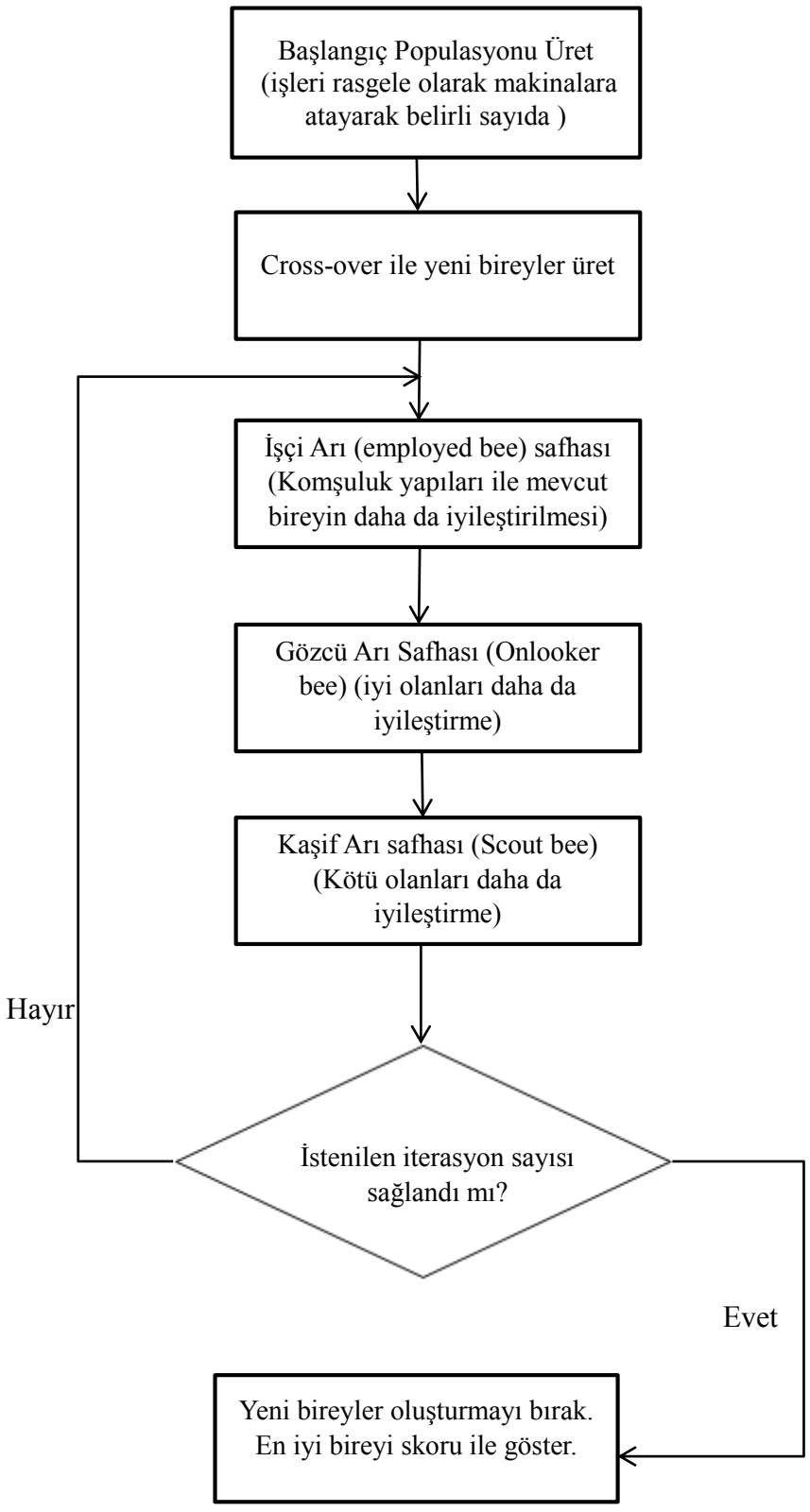
algoritması [169], Parçacık Sürüsü Optimizasyonu-Yerçekimi Arama (PSO-GS) hibrit algoritması [170], Yapay Bağışıklık Sistemi-Parçacık Sürüsü Optimizasyon Tekniği (AIS-PSO) hibrit algoritması [171, 172].

Bu çalışmada da atölye tipi çizelgeleme problemleri için yapay arı kolonisi optimizasyon metodu ile evrimsel algoritmalar bütünleşik olarak kullanılarak yeni bir metodoloji önerilmiştir.

4.2. ABC-EA Bütünleşik Yaklaşımının Yapısı

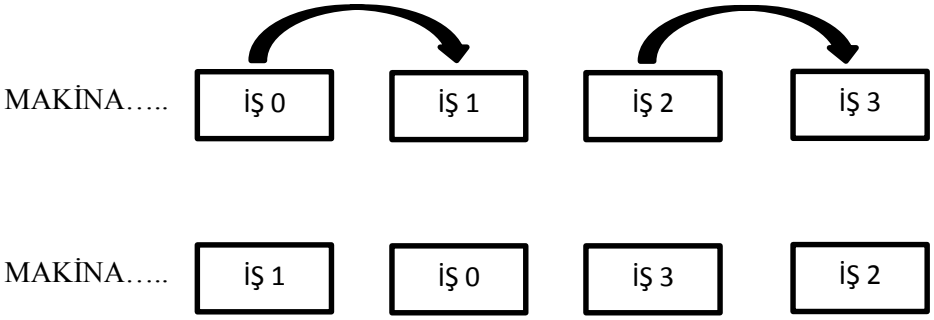
Yapay arı kolonisi optimizasyon metodu ile evrimsel algoritmalar bütünleşik olarak kullanılarak yeni bir metodoloji önerilmiştir. Önerilen metodun yapısı Şekil 4.1'de gösterilmiştir ve ilgili yapı şu şekildedir. İlk olarak çözümü yapılacak olan data setinde kaç adet iş varsa, sırasıyla ilk işten başlayarak işler makinalara atanır. Bu işlemden sonra belirlenen sayı kadar (yazılımda "Roll" diye tanımlandı) mevcut atanmış işlerin yerleri değiştirilerek bir sonraki birey oluşturulur. Örneğin 4×4'lük bir data seti için, Roll=1 alınırsa Şekil 4.2'de de görüldüğü gibi ilk olarak makinalara işler iş-0'dan iş-3'e kadar sırasıyla atanarak ilk birey (çözüm) Şekil 4.3'teki gibi oluşturulur. Sonra işler arasında 1 değişiklik yapılarak Şekil 4.2'deki yeni iş sırasına göre işler, atanarak bir sonraki birey Şekil 4.4'teki gibi oluşturulur. Bu şekilde kaç bireyli populasyon oluşturulmak isteniyorsa o sayı kadar bu işlemler aynı şekilde devam eder. Burada elde edilen bütün bireyler feasible sonuçlardır.

İkinci aşamada ise başlangıç populasyonundaki bireyler, evrimsel algoritmalarda kullanılan çaprazlama (cross-over) operatörü ile yine belirlenen sayıda yeni bireyler üretilir. Çaprazlama şu şekilde yapılmaktadır: Elimizdeki populasyondaki her bir birey, bir adet çizelgeleme sonucuna sahip farklı bir çözümdür. Her bir çözümde, her bir makinada işler farklı sıra ile yapılmakta ve bitiş süreleri de rasgele farklı olmaktadır. Çaprazlama işleminde herhangi iki birey alınır ve makinalardaki iş sıralarına göre rasgele çaprazlama yapılır ve yeni bireyler elde edilmiş olur. Bu yeni elde edilen bireyler, başlangıç populasyonuna eklenerek yapay arı kolonisi optimizasyon safhasına geçilmiştir. Bu safhada elde edilen bireyler feasible

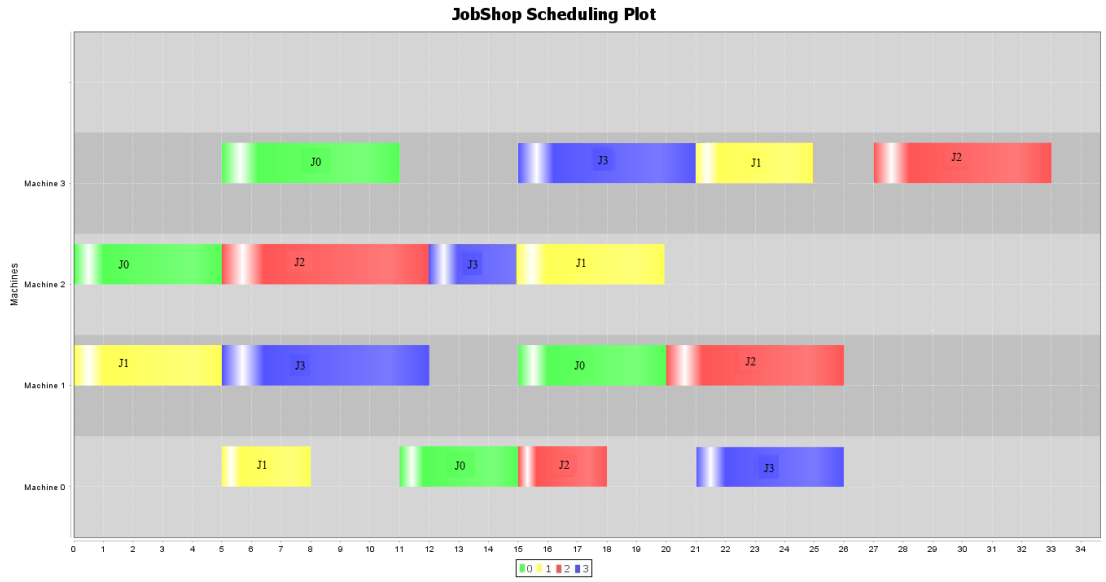


Şekil 4.1. Önerilen ABC-EA bütünleşik yaklaşımı için akış şeması.

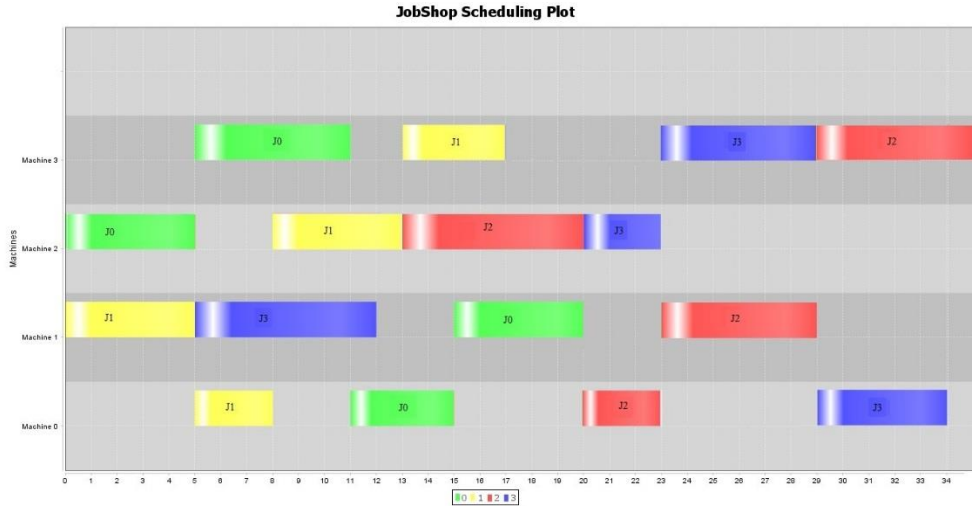
olmayabilir. Böyle bireylere bir sonraki safhada literatürdeki swap (ikili yer değiştirme) ve insertion (ekleme) metotları [173-174-175] uygulanıp bireyler feasible hale gelebilmektedir. Eğer bu işlemlere rağmen feasible olmuyorsa bu birey populasyondan atılır. Örnek olarak Şekil 4.3 ve Şekil 4.5'teki bireyleri ele aldığımızı varsayarsak, ilk bireydeki ilk iki makinadaki işler ile 2. bireydeki son iki makinadaki işleri çaprazlayacak olursak Şekil 4.6'daki bireyi elde ederiz. Bundan sonraki arı kolonisi optimizasyon safhası üç aşamadan oluşmaktadır.



Şekil 4.2. Bireyleri oluştururken iş sıralarına göre yapılan atamalar.

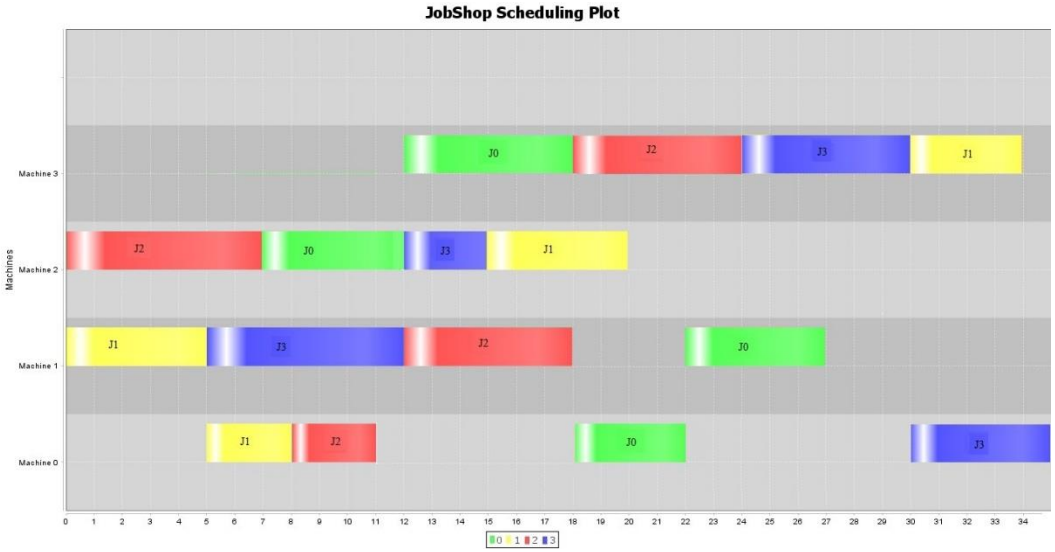


Şekil 4.3. İlgili data seti için ilk iş sırasına göre oluşturulan birey.

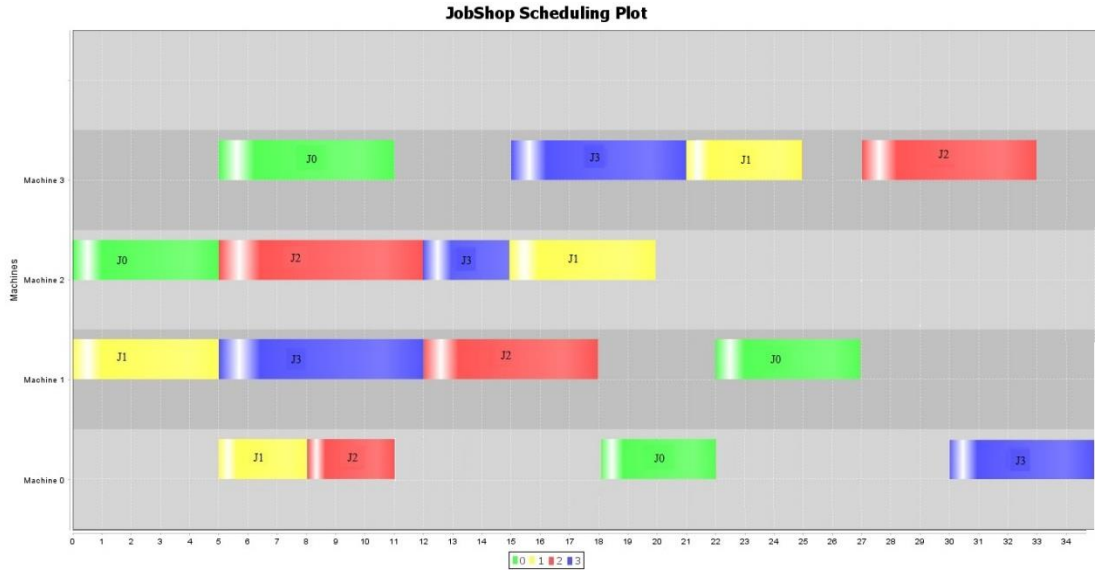


Şekil 4.4. İlgili data seti için değişiklik sonrası olan iş sırasına göre oluşturulan yeni birey.

İlk olarak işçi arı (employed bee) safhası uygulanmıştır. Bu safhada, populasyonu oluşturan bireyler yine swap ve insertion metotları ile Değişken Komşuluklu Arama (Variable Neighborhood Search-VNS) metodu [176, 177] ile populasyon sayısı aynı kalacak şekilde yeni bireyler üretilmiştir. Her bir birey için VNS uygulandığında elde edilen sonuca bakılır. Daha iyi ise iyi olan alınıp diğer sonuç atılır, eğer daha kötü bir sonuç çıkarsa mevcut olan ile devam edilir. Örnek olarak bir önceki Şekil 4.3'te verilen bireyi (C_{max}=33 sn.) ele alırsak ve J0 ile J2 işlerine VNS uygularsak Şekil 4.5'te gösterilen bireyi (C_{max}=35 sn.) elde ederiz. Bu çizelgeleme için sonuç ilk bireye göre daha kötü olduğundan dolayı populasyondan atılır.



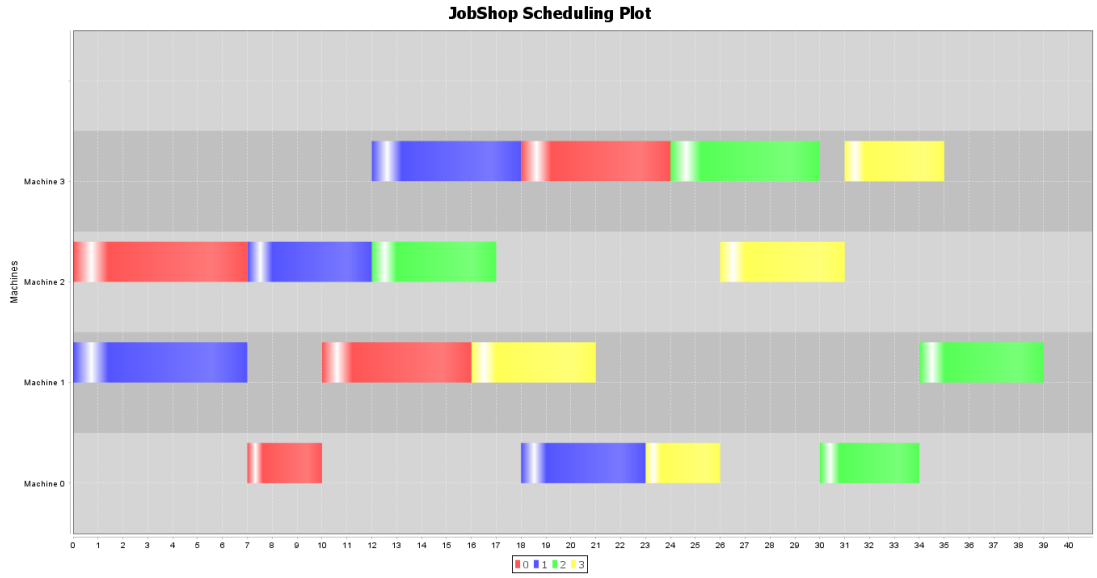
Şekil 4.5. İlgili data setine VNS uygulanması sonucu elde edilen yeni birey.



Şekil 4.6. Çaprazlama sonucu elde edilen yeni birey.

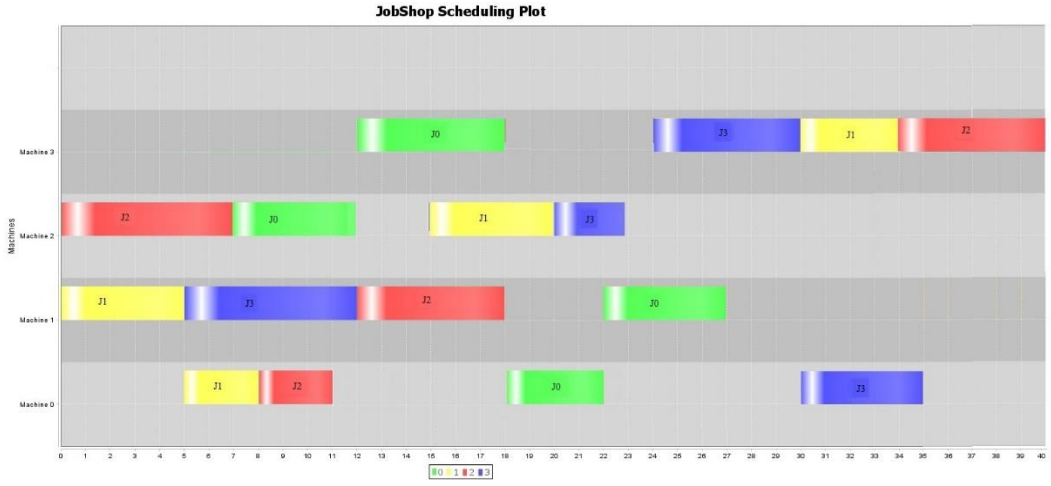
Diğer safha ise, gözcü arı (onlooker bee) safhasıdır. Bu aşamada, populasyondan herhangi iki birey alınır ve bunlardan hangisi iyi çözüme sahip ise bu bireye VNS uygulanarak bir çözüm elde edilir. Eğer bu birey için bir öncekine göre daha iyi bir çözüm üretilmişse iyi çözüm alınır, daha kötü ise ilk çözüm aynen kalmaktadır. Populasyondaki birey sayısı kadar bu işlem uygulanır ve diğer safhaya geçilir. Bu safhaya örnek olarak Şekil 4.5'te verilen bireyi (C_{max}=35 sn.) ve farklı bir birey olarak Şekil 4.7'de verilen bireyi (C_{max}=39sn.) populasyondan rasgele ele aldığımızı varsayarsak, bunlardan iyi sonuca sahip olan ilk bireye VNS uygularsak Şekil 4.8'deki bireyi (C_{max}=40 sn.) elde ederiz. Elde edilen ilgili bireyin sonucu ilk bireyin sonucuna göre daha kötü olduğundan bireyin VNS uygulanmamış ilk hali ile yola devam edilir.

Son aşama ise kâşif (scout) arı aşamasıdır. Bu aşamada ise yine populasyondan herhangi iki birey alınır. Bu sefer kötü olan birey ele alınır. Bunun yanında şu ana kadar mevcut sonuçlar içinden en iyisi alınarak şu işlem uygulanmıştır: En iyi bireye VNS uygulanır ve bir çözüm elde edilir. Eğer bu çözüm ele alınan diğer kötü çözümden iyi ise kötü çözüm atılarak diğer daha iyi olan çözüm populasyona dâhil edilir. Eğer en iyi çözüme VNS uygulanıp kötü olan çözümden daha iyi bir sonuç elde edilemezse, kötü olan çözüm aynen kalmaktadır.



Şekil 4.7. İlgili data seti ile elde edilmiş yeni birey.

Örnek olarak Şekil 4.7'deki birey (C_{max}=39 sn.) ile şimdiye kadar ki en iyi sonuca sahip birey olan Şekil 4.3'teki bireyi (C_{max}=33 sn.) ele aldığımızı varsayarsak, en iyi bireye VNS uyguladığımızda Şekil 4.5'teki bireyi (C_{max}=35 sn.) elde ederiz. Bu durumda VNS sonucu elde edilen birey kötü çözüme göre daha iyi olduğu için kötü çözüm populasyondan atılır ve yerine VNS sonucu elde edilen birey eklenir. Bu işlem yine populasyon sayısı kadar uygulanıp aşamalar 1 iterasyon için tamamlanmış olur. Son olarak sistemi durdurma kriteri olarak kaç iterasyon sisteme girilmişse, sistem o kadar çalıştırılır ve durdurma kriteri sağlandıktan sonra sistemin çalışması durdurulur. En son olarak ta en iyi birey (çözüm) o data seti için en iyi çizelgeleme sonucunu bize vermektedir.



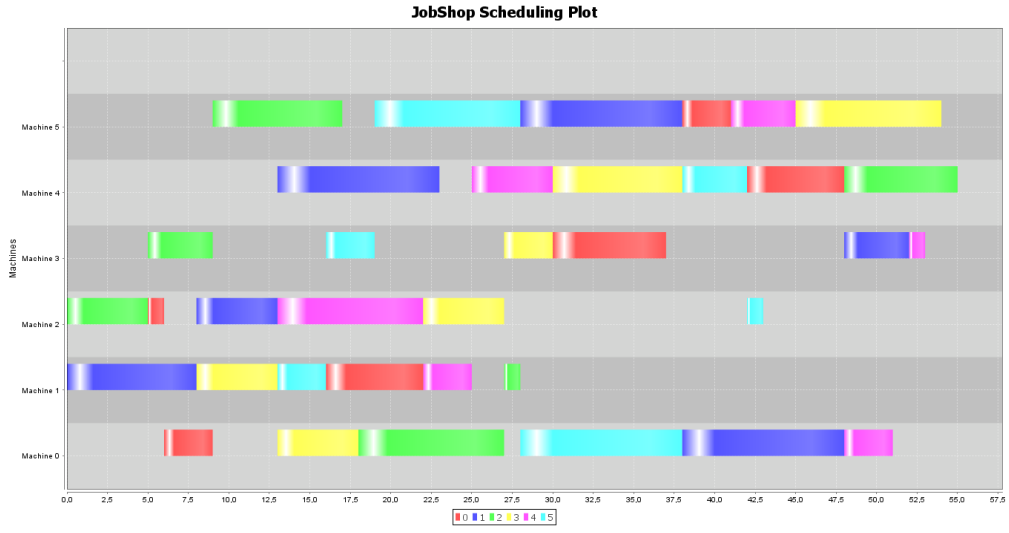
Şekil 4.8. Gözcü arı safhasındaVNS uygulanması sonucu elde edilmiş yeni birey.

4.3. Önerilen Metodun Data Setlerine Uygulanması

Yapay arı kolonisi optimizasyon metodu ile evrimsel algoritmaların ortak kullanımı sonucu önerilen metot, atölye tipi çizelgeleme problemleriyle ilgili olarak literatürdeki test datalarının [178] çözümünde, İşlerin Toplam Tamamlanma Zamanını gösteren C_{max}'ı (makespan) minimize etmek için kullanılmıştır. Önerilen metodun C_{max}'ı minimize etmek için atölye tipi çizelgeleme problemleriyle ilgili olarak çözümü için Eclipse geliştirme ortamında Java yazılım dili (Ek-A) kullanılmıştır. Literatürde kullanılan birçok test datası vardır. Bunlardan ft06, ft10, la01, la02, la03, la04, la05, la06, la07, la08, la09, la10, la11, la12, la13, la14, la15, la16, la17, la18, la19, la20, la30, la35, abz05, abz06, orb01, orb02, orb03, orb04, orb05, orb06, orb07, orb08, orb09, orb10 test dataları [178] alınmış ve önerilen metot ile C_{max}'ı minimize etmek için ilgili metot kullanılarak C_{max} minimize edilmeye çalışılmıştır. Herbir data seti farklı iş×makine sayısına sahiptir. Bu data setleri seçilirken değişik büyüklerdeki iş×makine sayılarına sahip datalar seçilerek önerilen metodun değişik kombinasyonlarda nasıl çalıştığı gözlemlenmiştir.

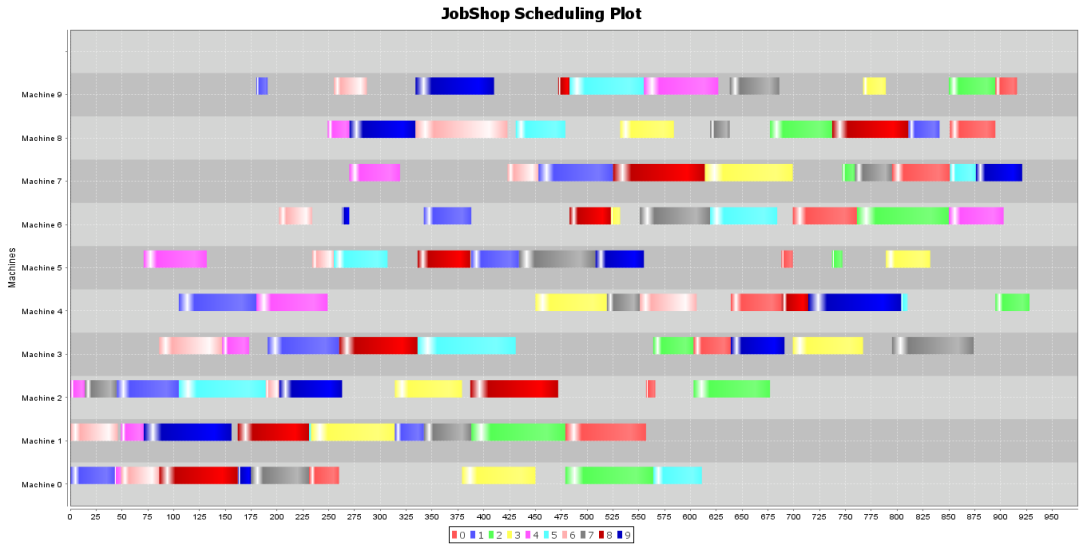
4.3.1. Farklı data setleri için elde edilen en iyi çizelgelemeler

Önerilen metot ile 6×6 (iş×makine sayısı) ft06 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.9'da gösterilmiştir (C_{max}=55 sn.).



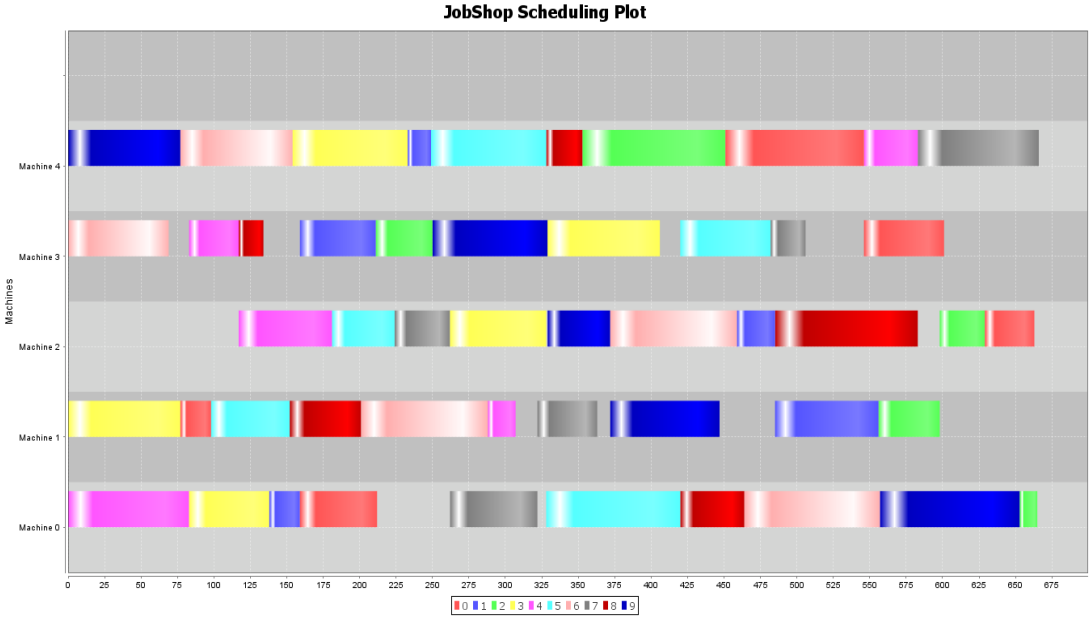
Şekil 4.9. ft06 data set için en iyi çizelgeleme sonucu.

(10×10)'luk ft10 data seti için elde edilen en iyi çözüm Şekil 4.10'da gösterilmiştir (C_{max}=930 sn.).



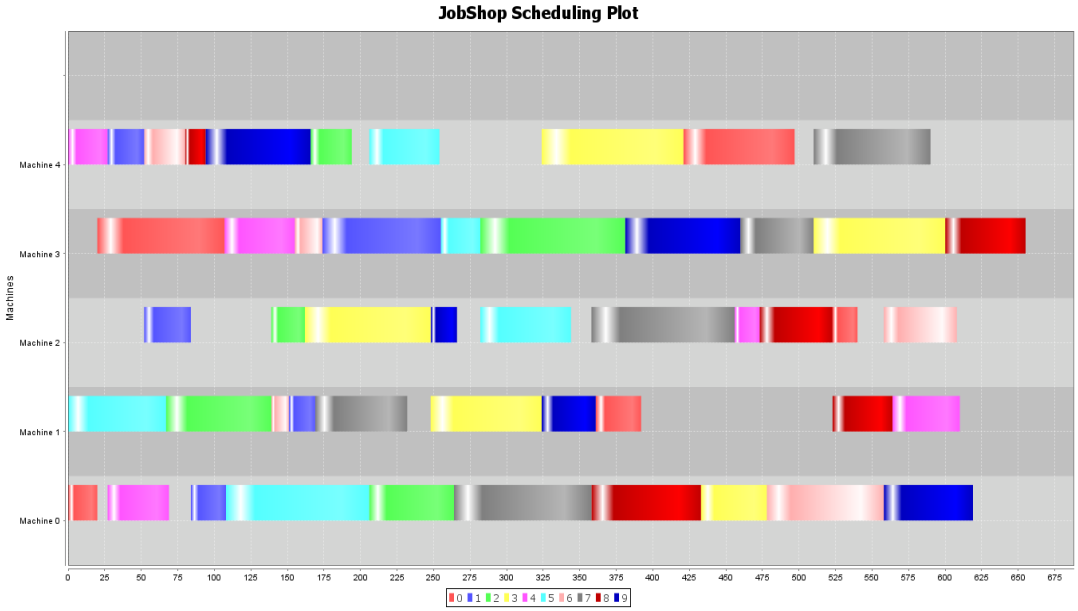
Şekil 4.10. ft10 data set için en iyi çizelgeleme sonucu.

(10×5)'lik la01 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.11'de gösterilmiştir (C_{max}=666 sn.).



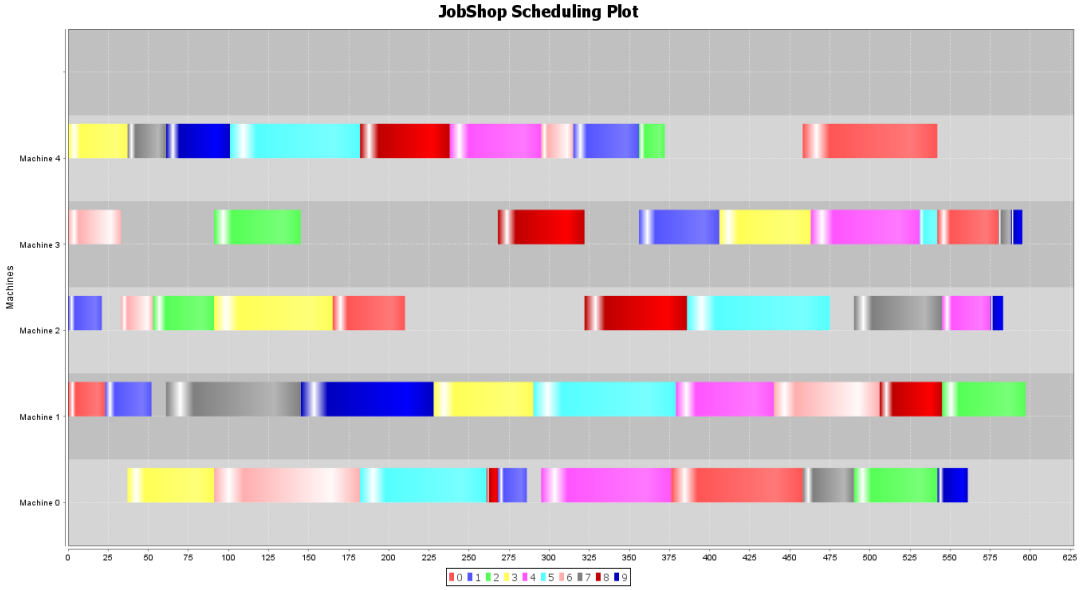
Şekil 4.11. la01 data set için en iyi çizelgeleme sonucu.

(10×5)'lik la02 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.12'de gösterilmiştir (C_{max}=655 sn.).



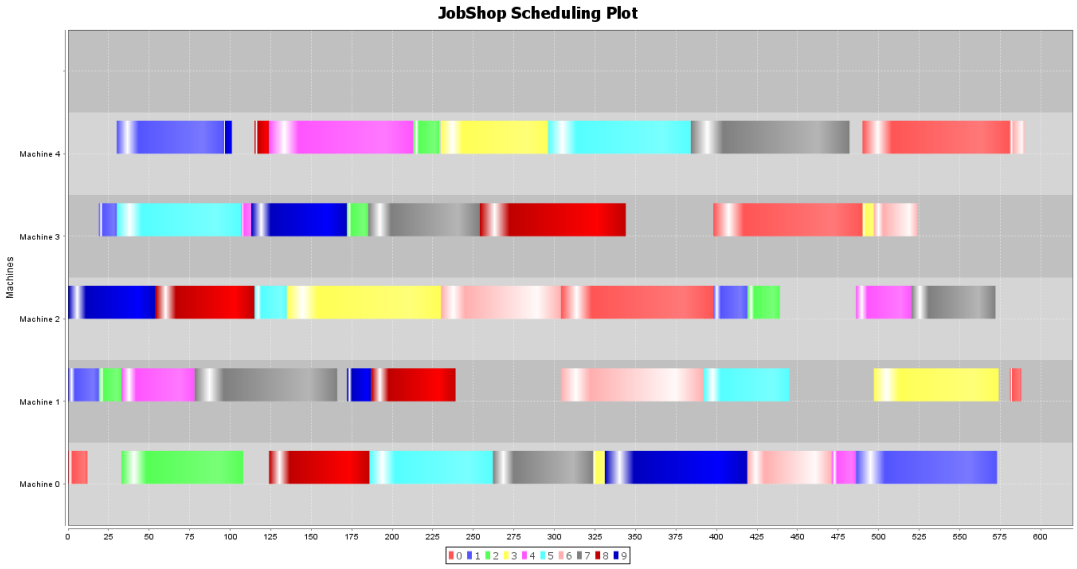
Şekil 4.12. la02 data set için en iyi çizelgeleme sonucu.

(10×5)'lik la03 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.13'te gösterilmiştir (C_{max}=597 sn.).



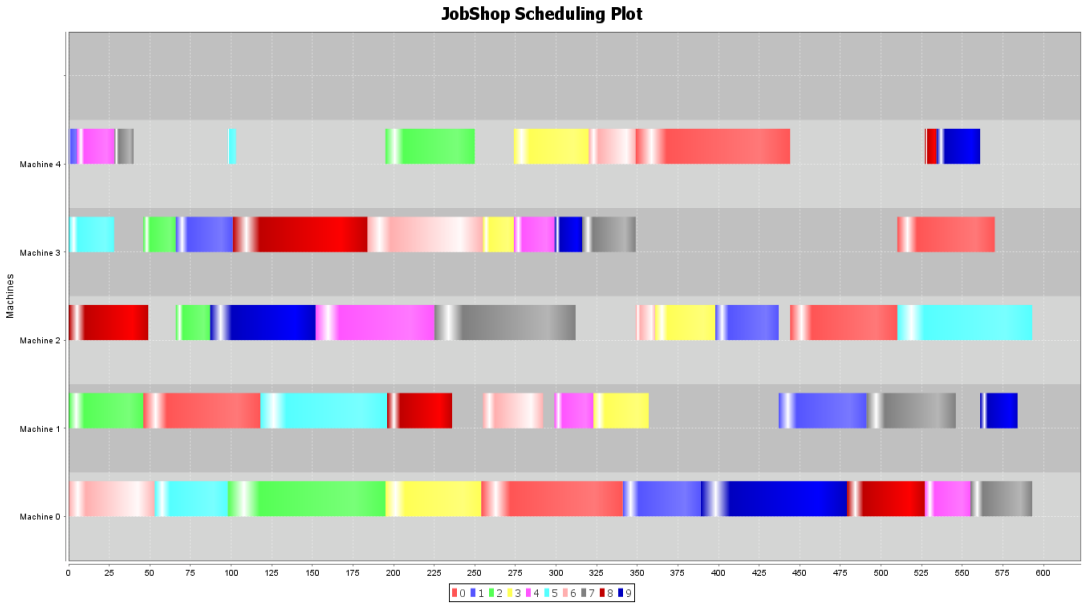
Şekil 4.13. la03 data set için en iyi çizelgeleme sonucu.

(10×5)'lik la04 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.14'te gösterilmiştir (C_{max}=590 sn.).



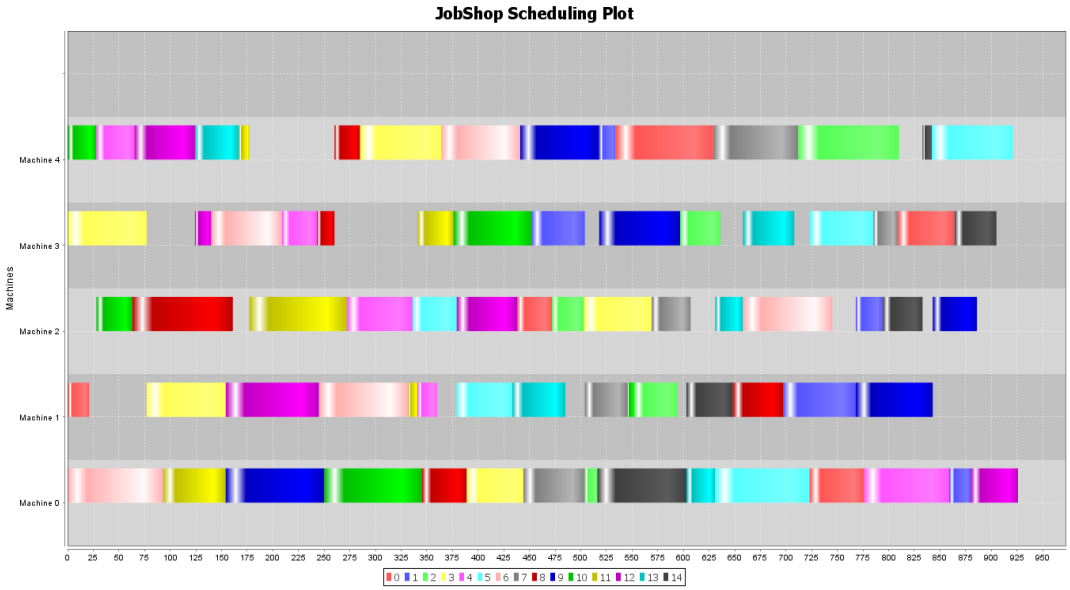
Şekil 4.14. la04 data set için en iyi çizelgeleme sonucu.

(10×5)'lik la05 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.15'te gösterilmiştir (C_{max}=593 sn.).



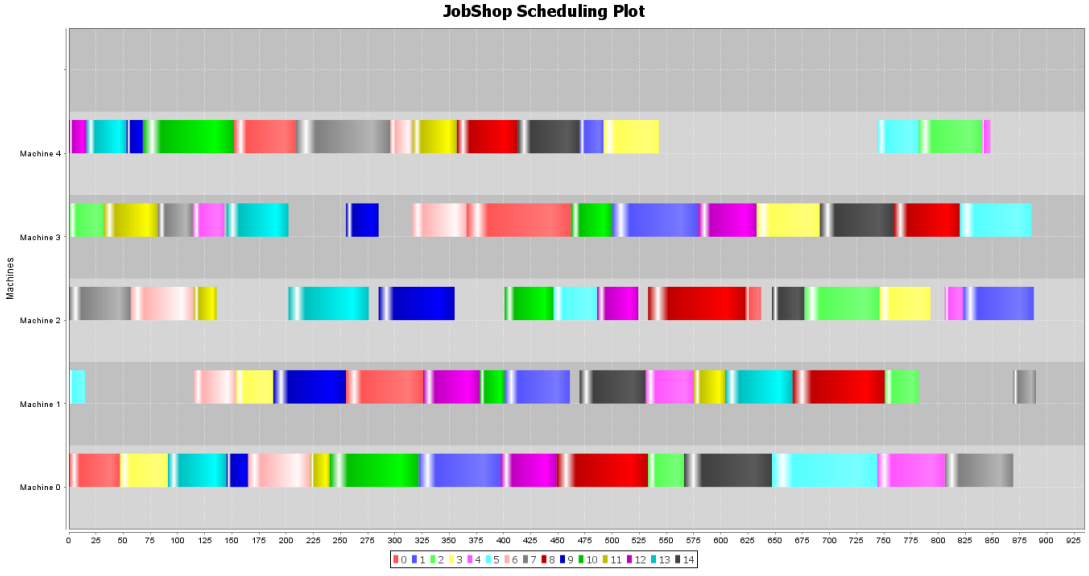
Şekil 4.15. la05 data set için en iyi çizelgeleme sonucu.

(15×5)'lik la06 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.16'da gösterilmiştir (C_{max}=926 sn.).



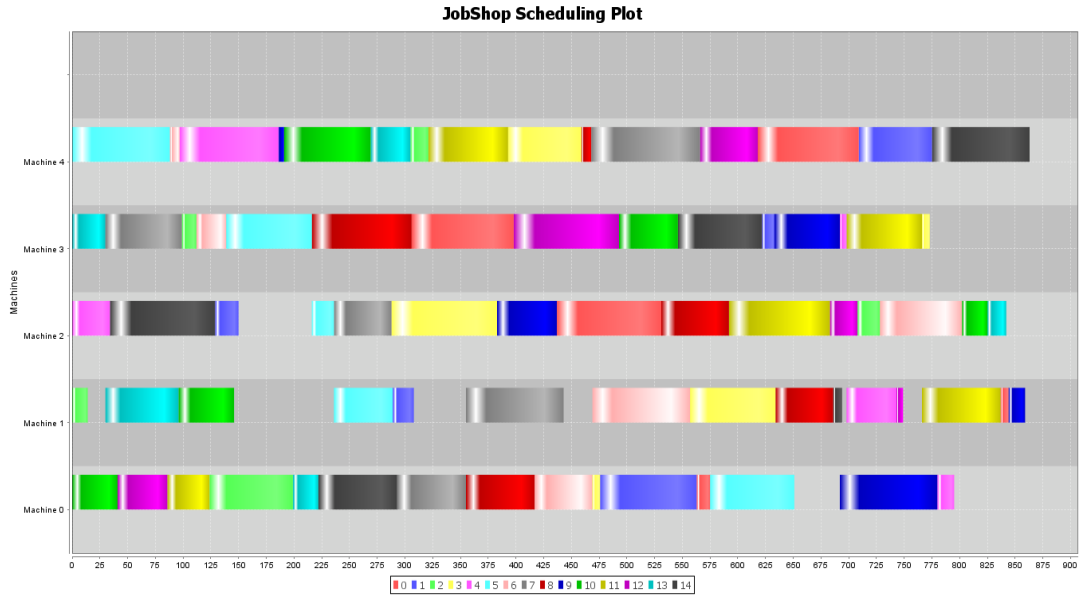
Şekil 4.16. la06 data set için en iyi çizelgeleme sonucu.

(15×5)'lik la07 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.17'de gösterilmiştir (C_{max}=890 sn.).



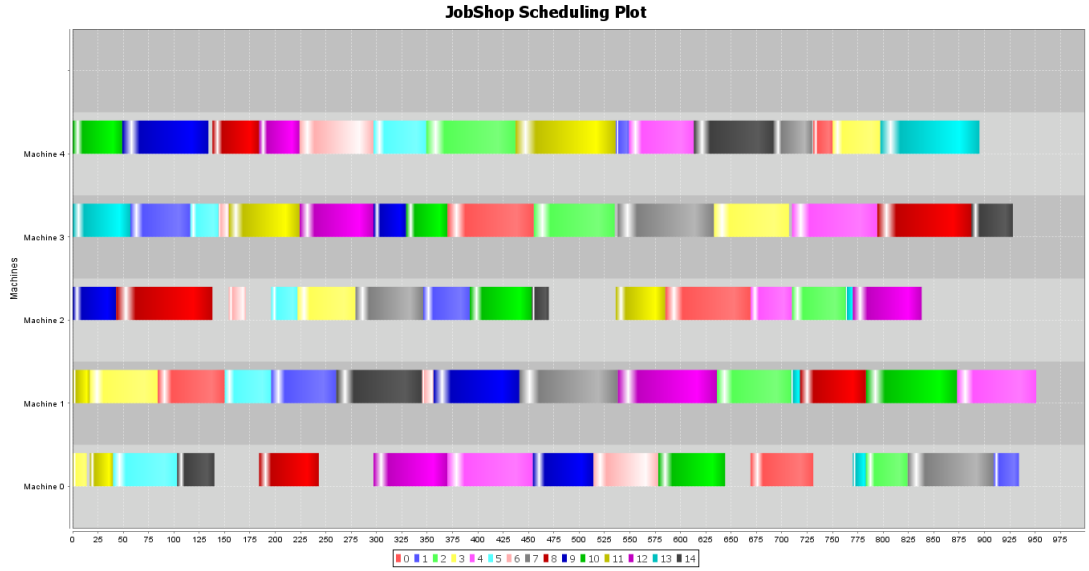
Şekil 4.17. la07 data set için en iyi çizelgeleme sonucu.

(15×5)'lik la08 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.18'de gösterilmiştir (C_{max}=863 sn.).



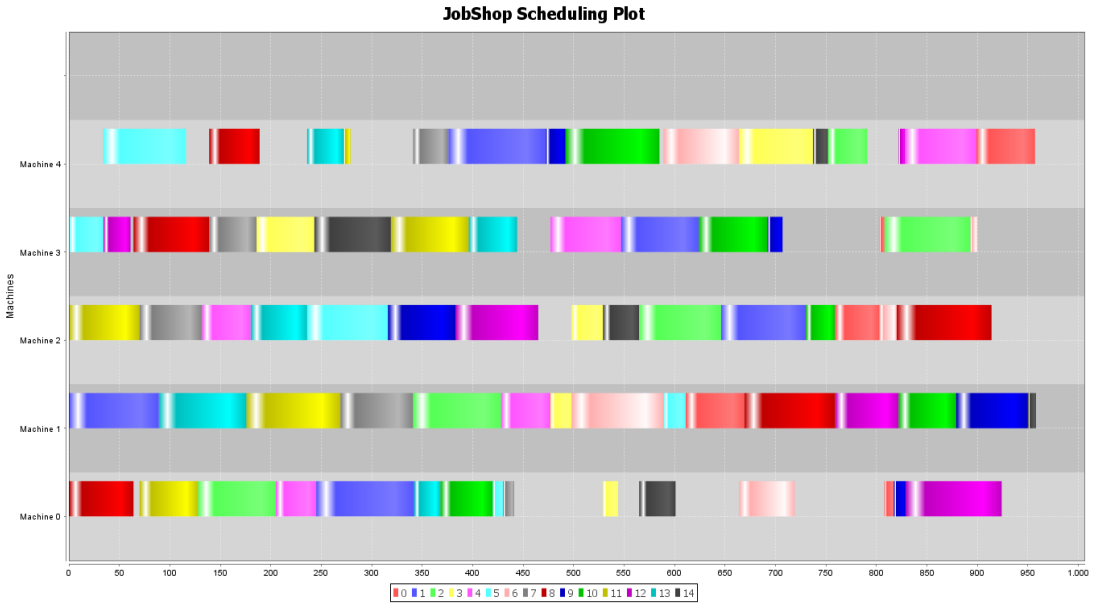
Şekil 4.18. la08 data set için en iyi çizelgeleme sonucu.

(15×5)'lik la09 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.19'da gösterilmiştir (C_{max}=951 sn.).



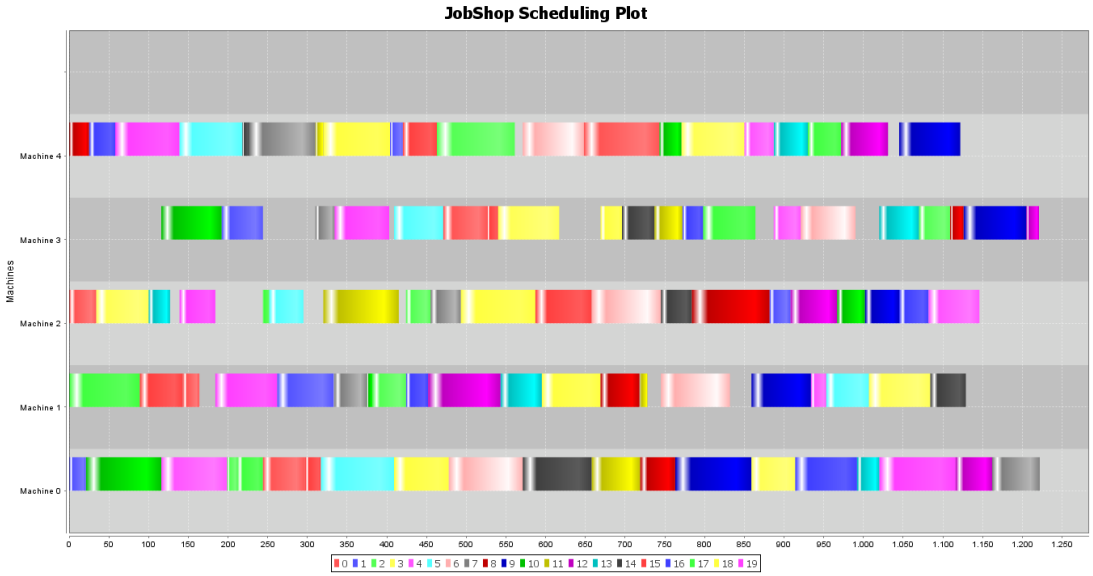
Şekil 4.19. la09 data set için en iyi çizelgeleme sonucu.

(15×5)'lik la10 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.20'de gösterilmiştir (C_{max}=958 sn.).



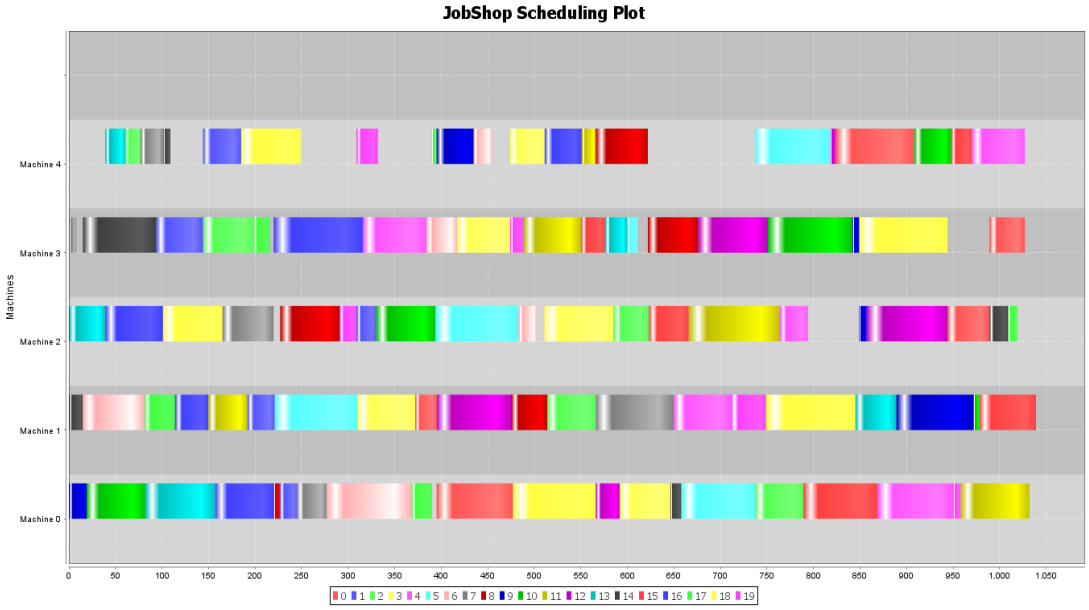
Şekil 4.20. la10 data set için en iyi çizelgeleme sonucu.

(20×5)'lik la11 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.21'de gösterilmiştir (C_{max}=1222 sn.).



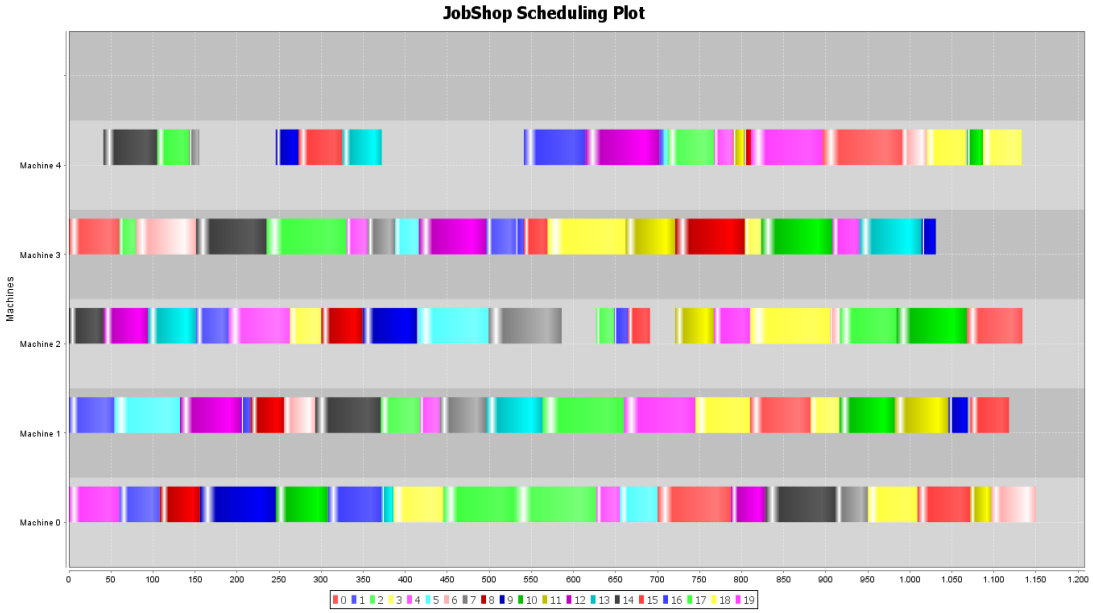
Şekil 4.21. la11 data set için en iyi çizelgeleme sonucu.

(20×5)'lik la12 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.22'de gösterilmiştir (C_{max}=1039 sn.).



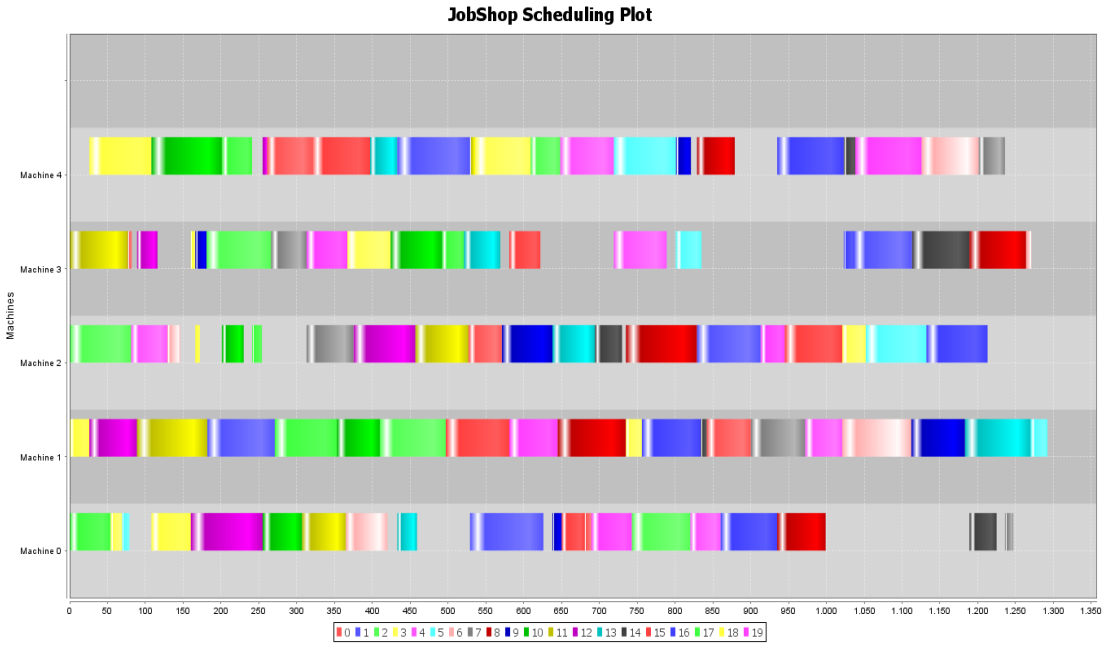
Şekil 4.22. la12 data set için en iyi çizelgeleme sonucu.

(20×5)'lik la13 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.23'te gösterilmiştir (C_{max}=1150 sn.).



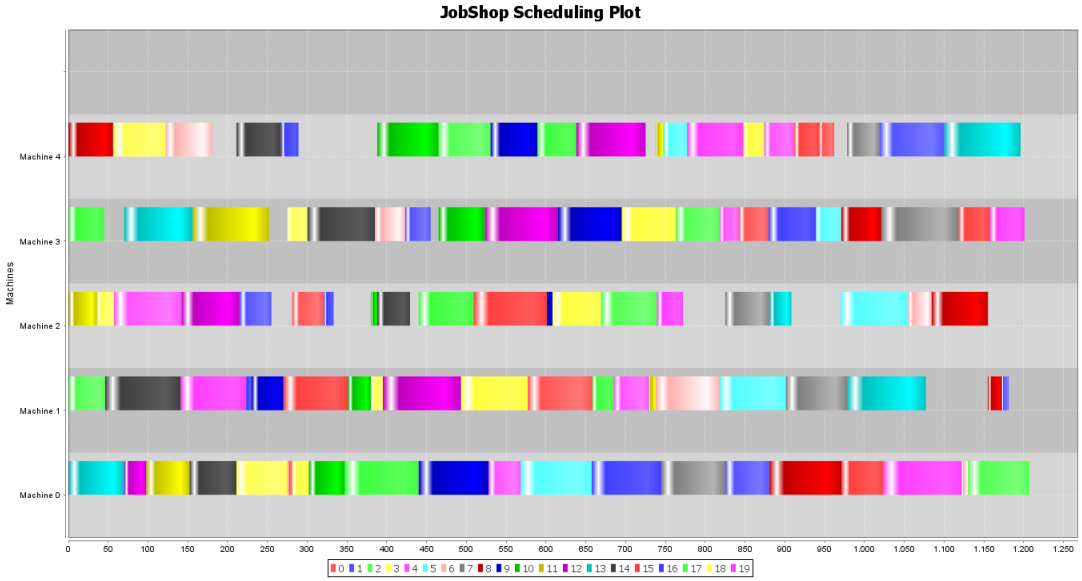
Şekil 4.23. la13 data set için en iyi çizelgeleme sonucu.

(20×5)'lik la14 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.24'te gösterilmiştir (C_{max}=1292 sn.).



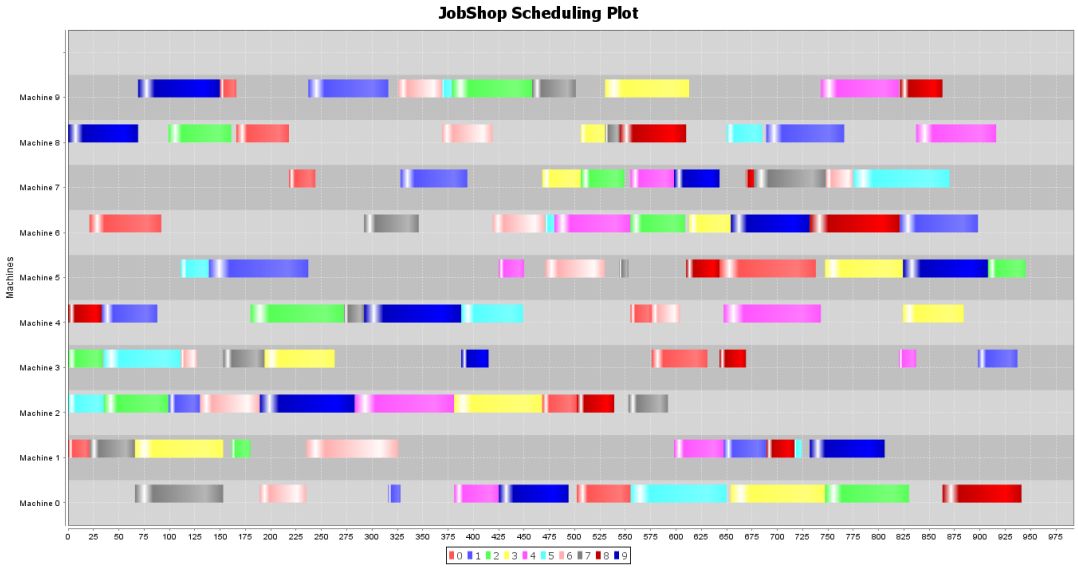
Şekil 4.24. la14 data set için en iyi çizelgeleme sonucu.

(20×5)'lik la15 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.25'te gösterilmiştir (C_{max}=1207 sn.).



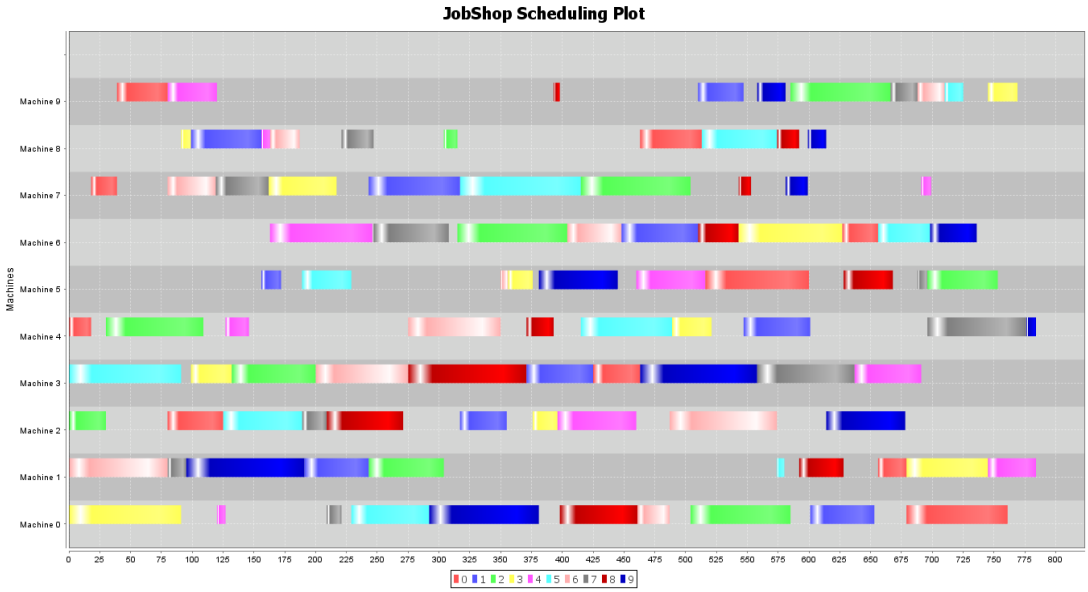
Şekil 4.25. la15 data set için en iyi çizelgeleme sonucu.

(10×10)'luk la16 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.26'da gösterilmiştir (C_{max}=945 sn.).



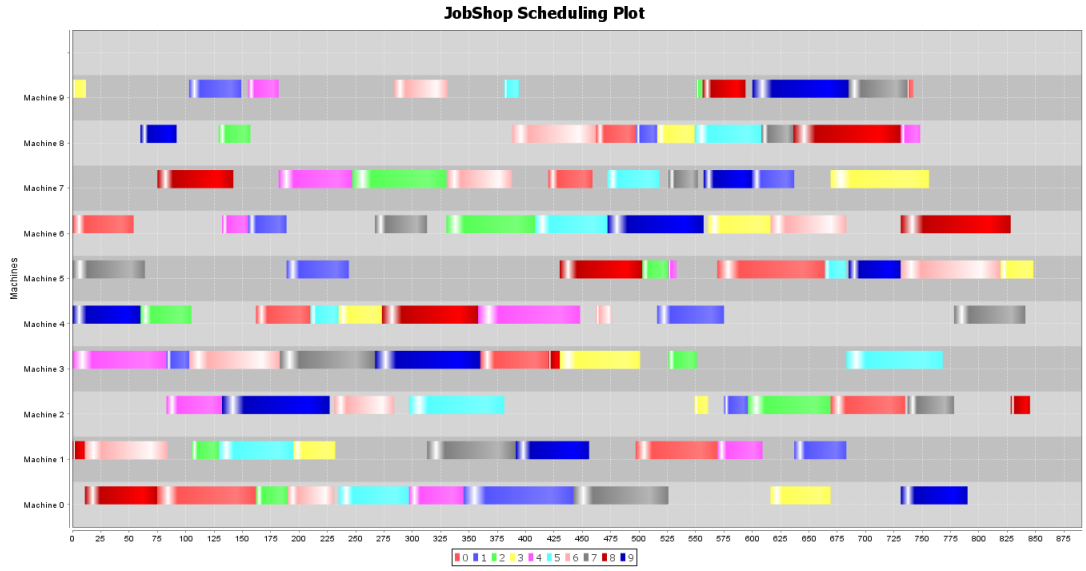
Şekil 4.26. la16 data set için en iyi çizelgeleme sonucu.

(10×10)'luk la17 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.27'de gösterilmiştir (C_{max}=784 sn.).



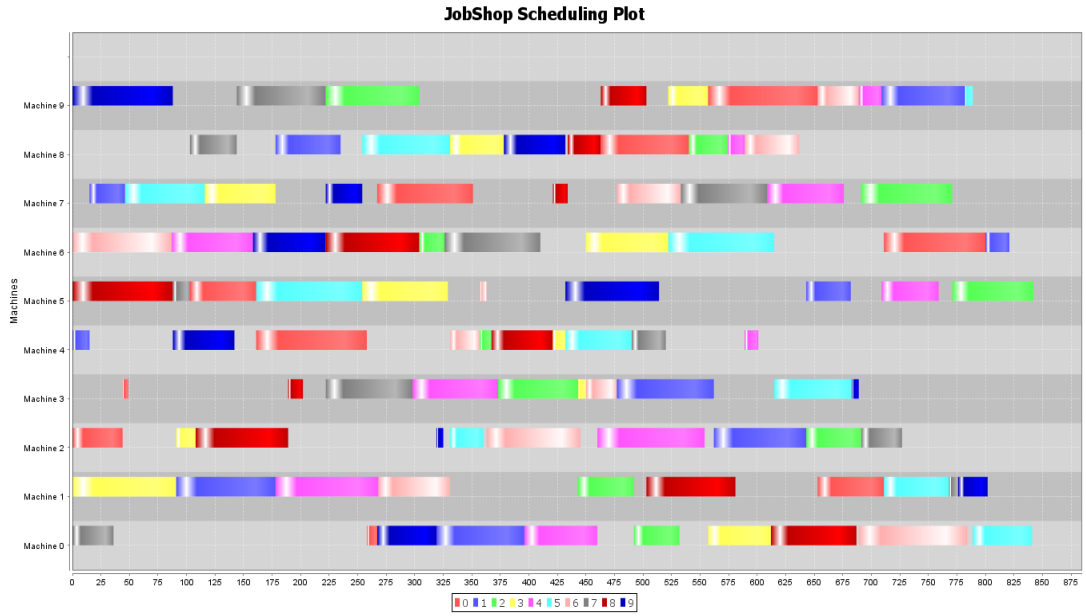
Şekil 4.27. la17 data set için en iyi çizelgeleme sonucu.

(10×10)'luk la18 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.28'de gösterilmiştir (C_{max}=848 sn.).



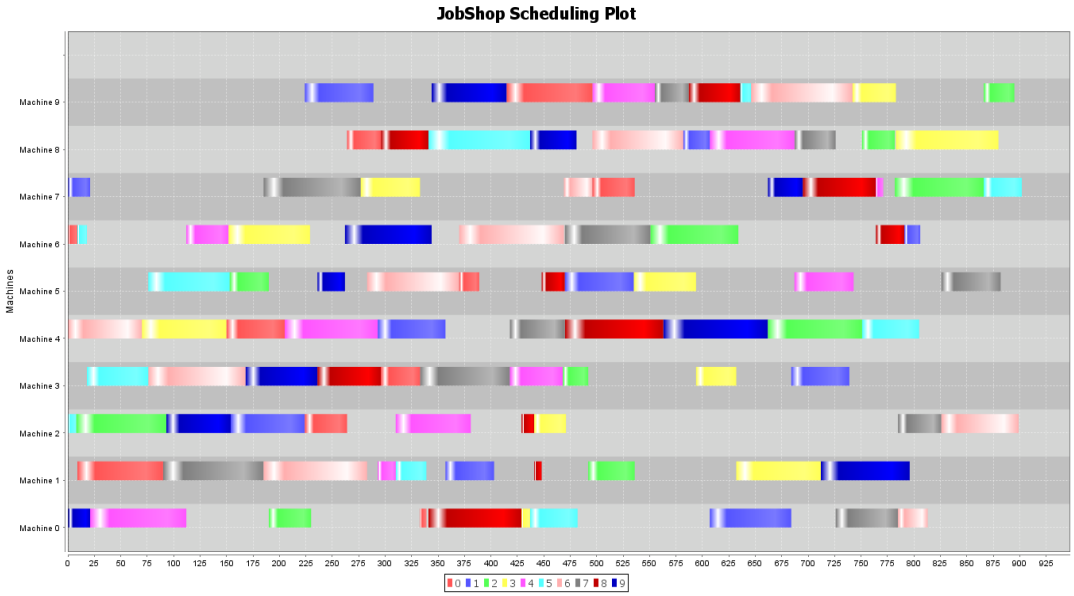
Şekil 4.28. la18 data set için en iyi çizelgeleme sonucu.

(10×10)'luk la19 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.29'da gösterilmiştir (C_{max}=842 sn.).



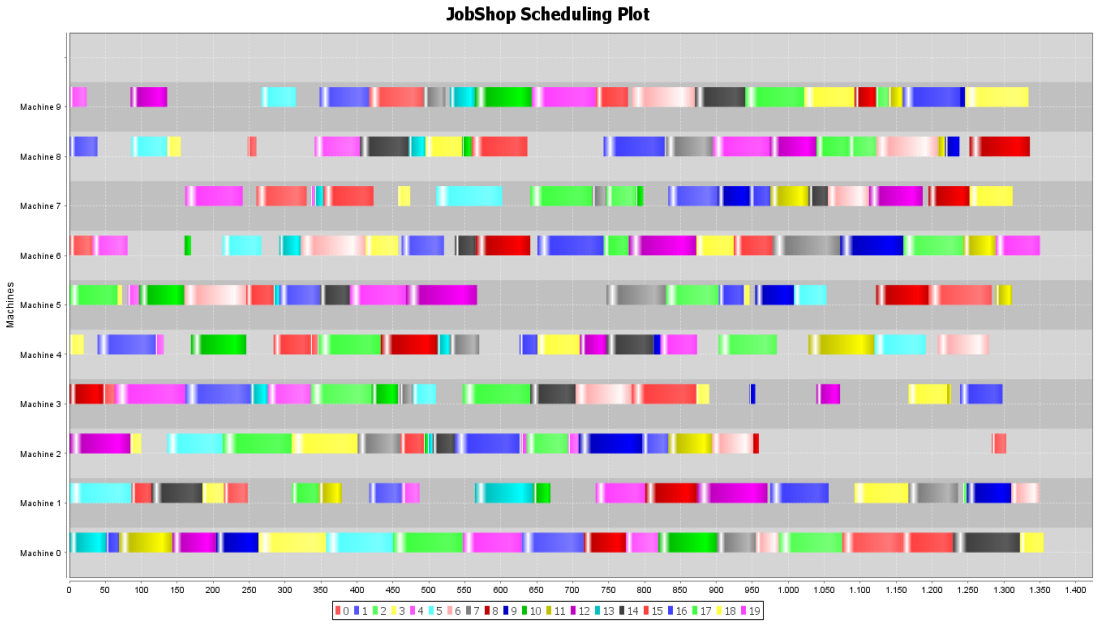
Şekil 4.29. la19 data set için en iyi çizelgeleme sonucu.

(10×10)'luk la20 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.30'da gösterilmiştir (C_{max}=902 sn.).



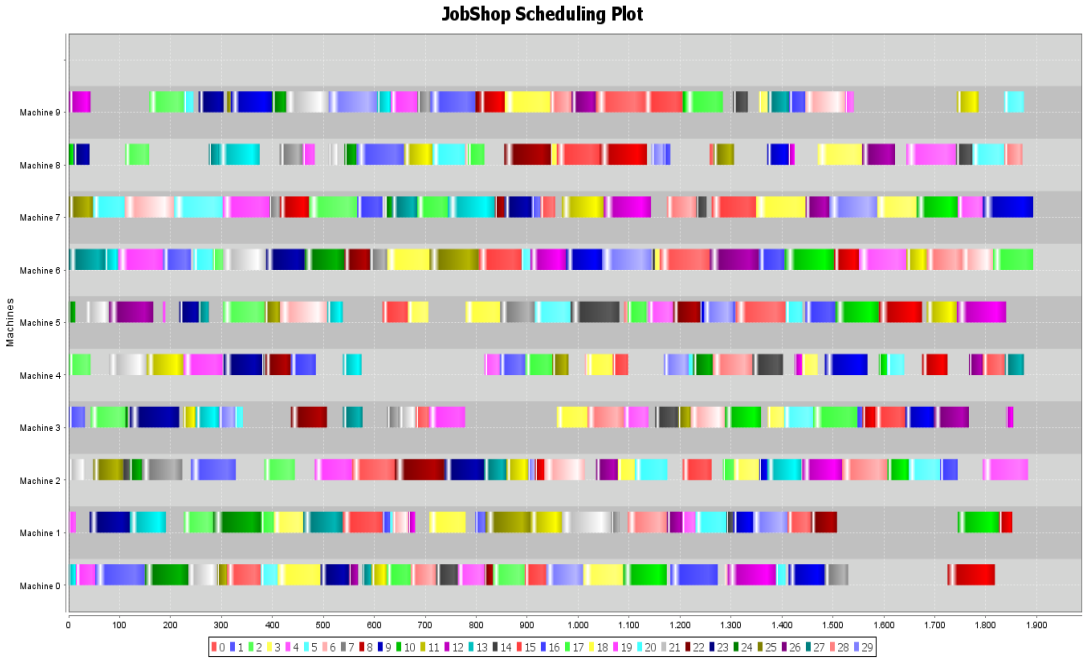
Şekil 4.30. la20 data set için en iyi çizelgeleme sonucu.

(20×10)'luk la30 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.31'de gösterilmiştir (C_{max}=1355 sn.).



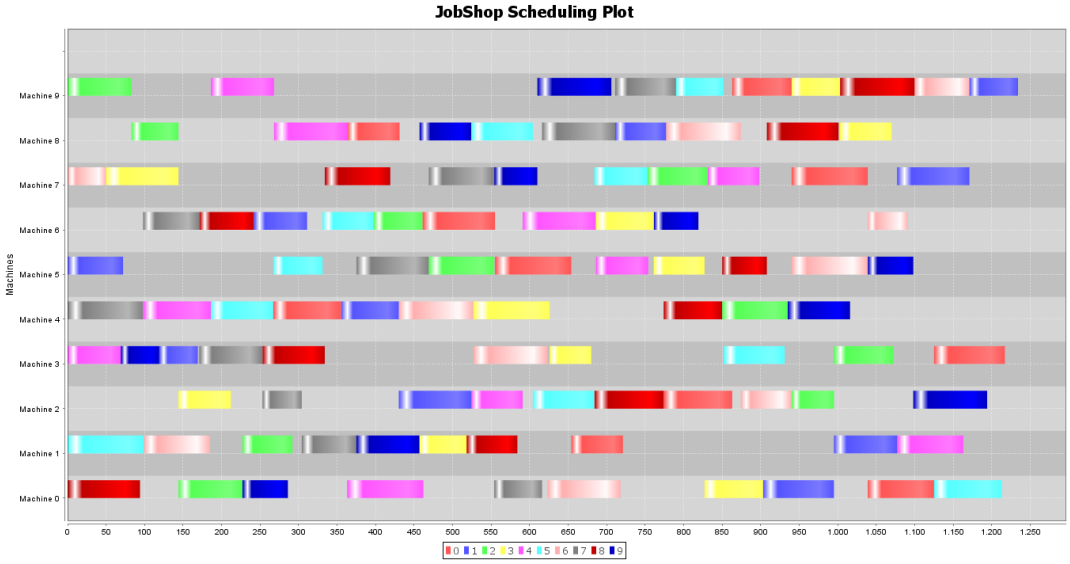
Şekil 4.31. la30 data set için en iyi çizelgeleme sonucu.

(30×10)'luk la35 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.32'de gösterilmiştir (C_{max}=1888 sn.).



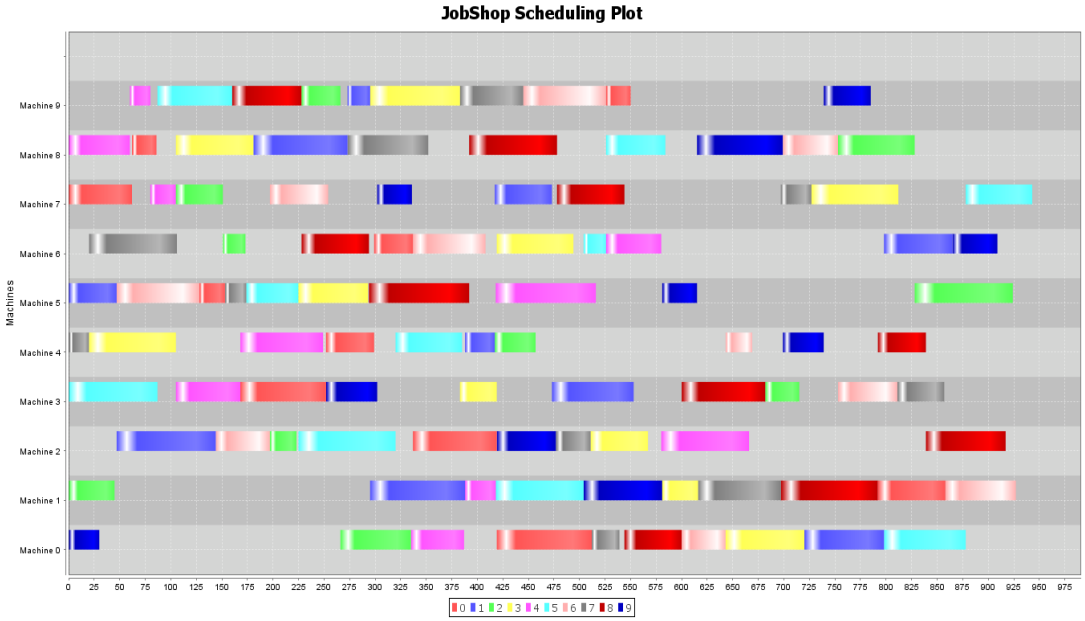
Şekil 4.32. la35 data set için en iyi çizelgeleme sonucu.

(10×10)'luk abz05 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.33'te gösterilmiştir (C_{max}=1234 sn.).



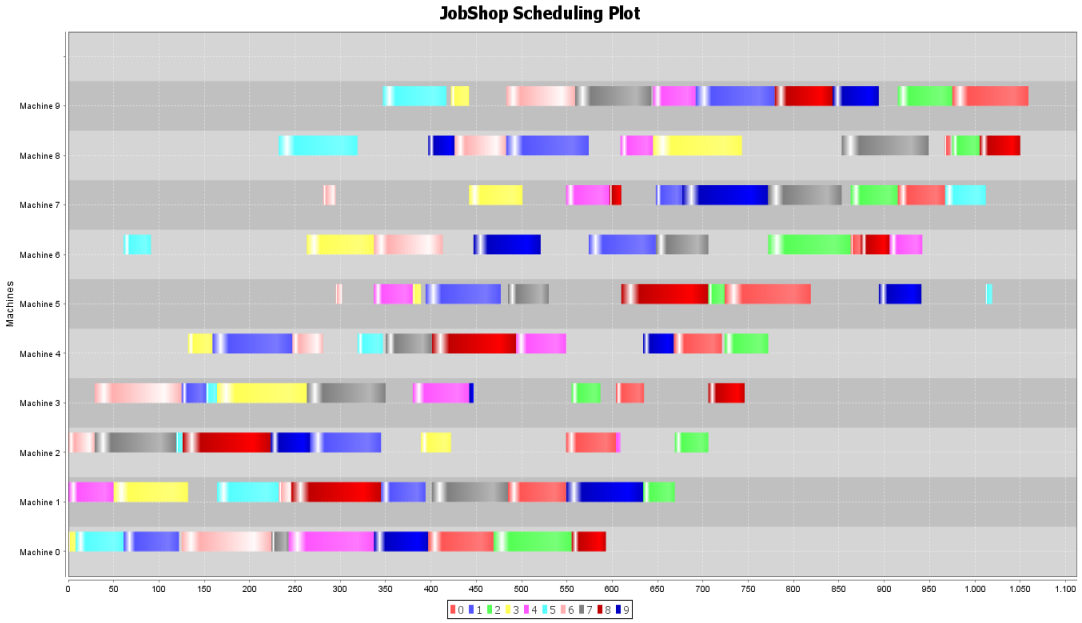
Şekil 4.33 abz05 data set için en iyi çizelgeleme sonucu.

(10×10)'luk abz06 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.34'te gösterilmiştir (C_{max}=943 sn.).



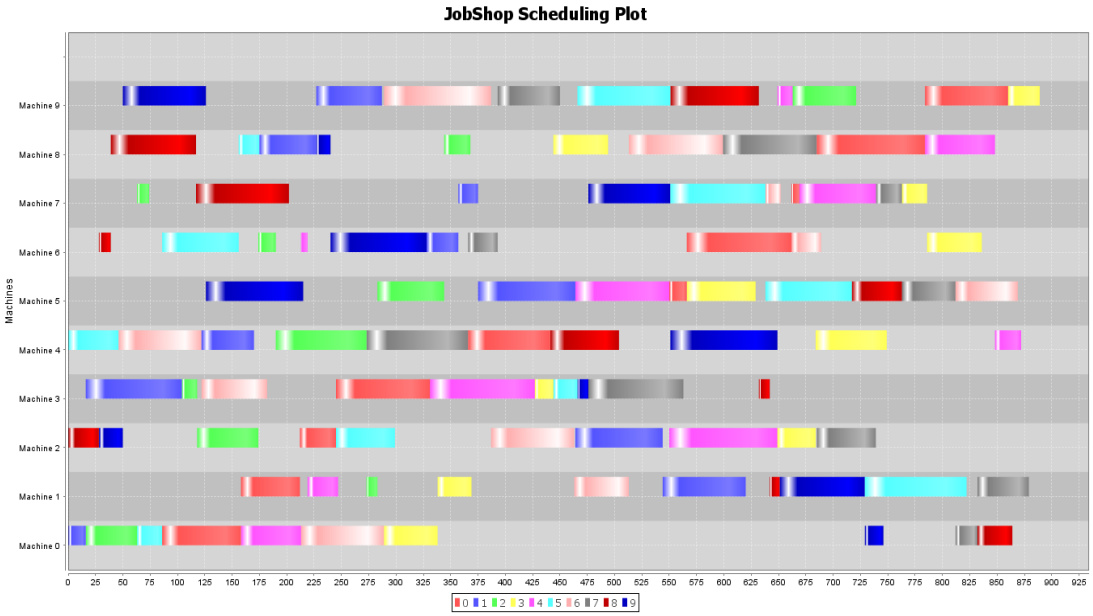
Şekil 4.34. abz06 data set için en iyi çizelgeleme sonucu.

10×10'luk (iş×makine sayısı) orb01 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.35'te gösterilmiştir (C_{max}=1059 sn.).



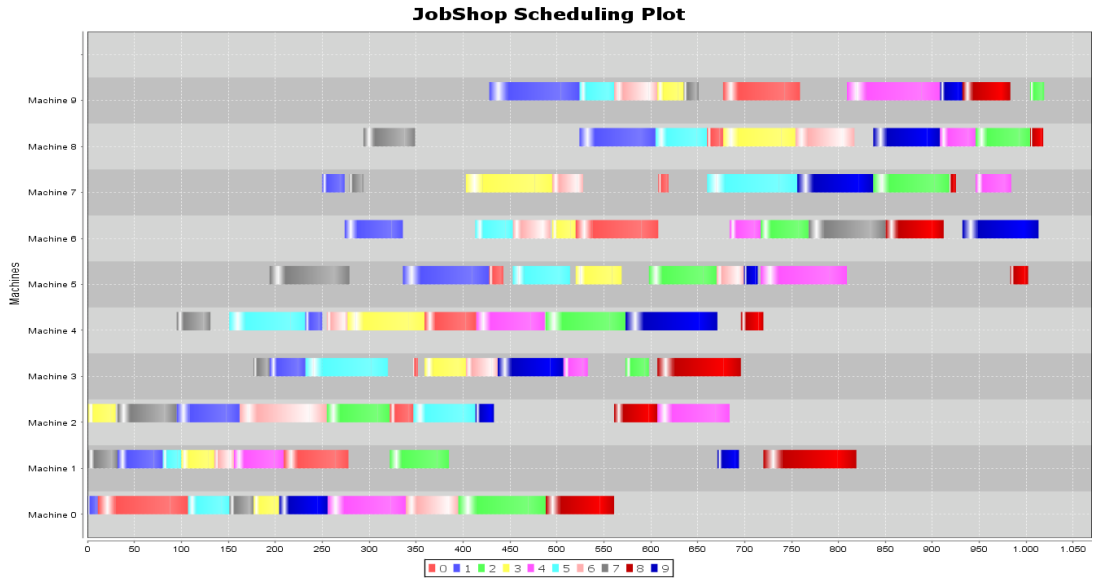
Şekil 4.35. orb01 data set için en iyi çizelgeleme sonucu.

(10×10)'luk orb02 data seti için elde edilen en iyi çözüm Şekil 4.36'da gösterilmiştir (C_{max}=888 sn.).



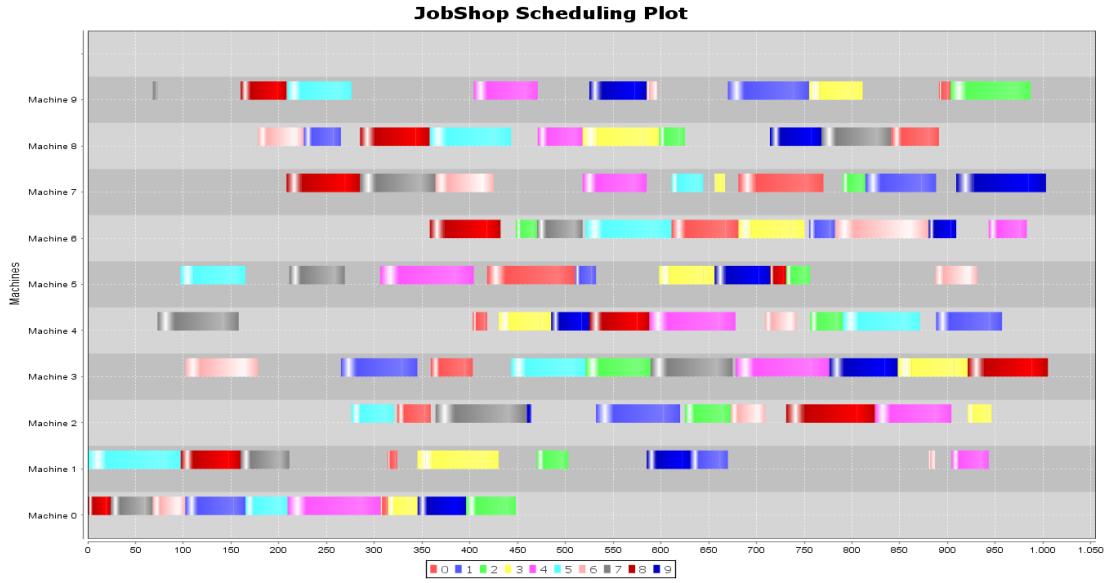
Şekil 4.36. orb02 data set için en iyi çizelgeleme sonucu.

(10×10)'luk orb03 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.37'de gösterilmiştir (C_{max}=1005 sn.).



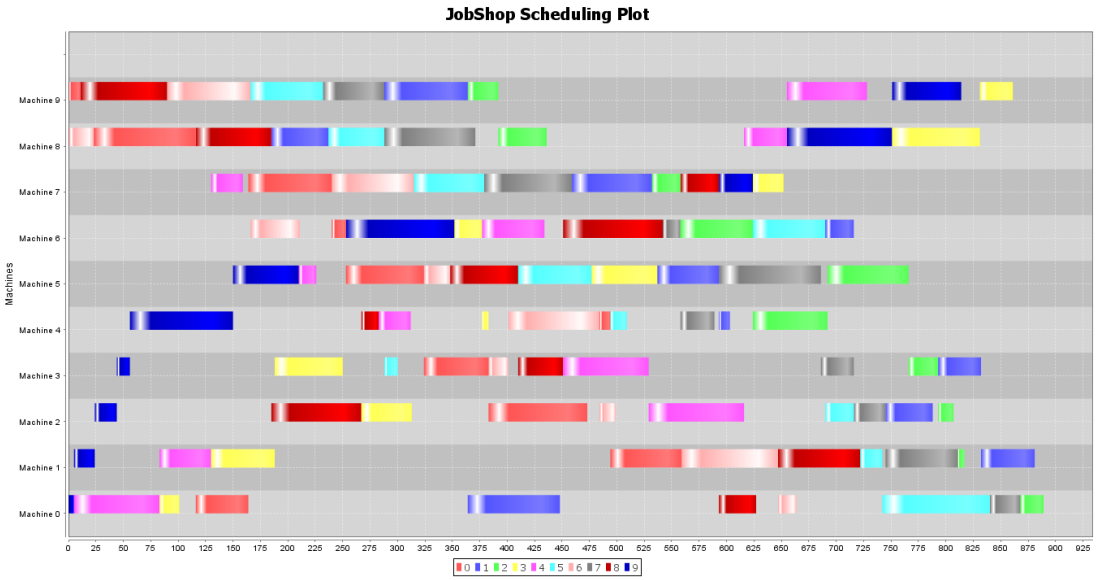
Şekil 4.37. orb03 data set için en iyi çizelgeleme sonucu.

(10×10)'luk orb04 data seti için elde edilen en iyi çözüm Şekil 4.38'de gösterilmiştir (C_{max}=1005 sn.).



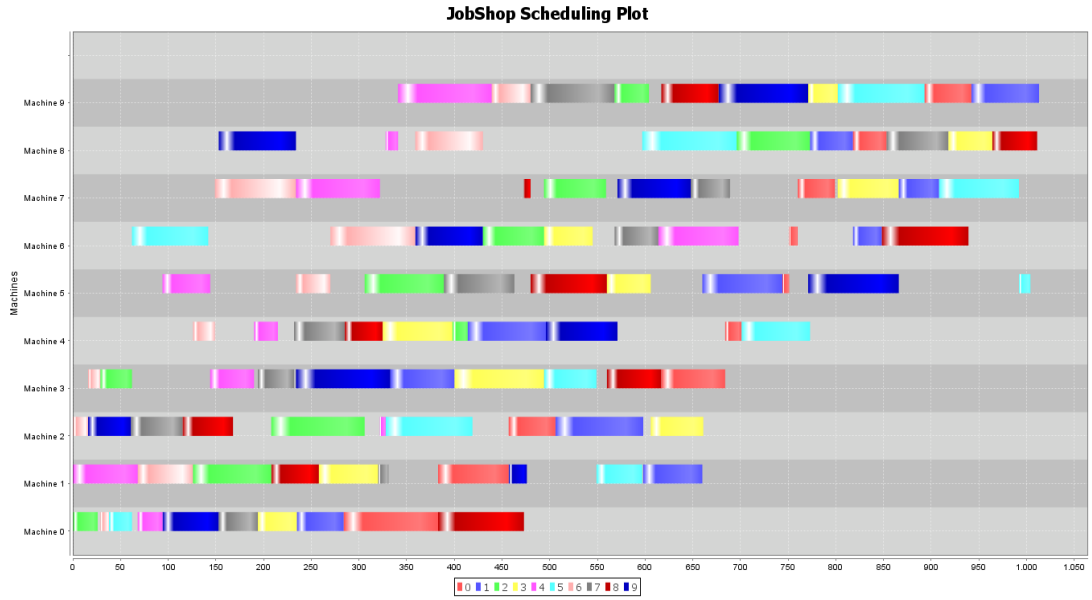
Şekil 4.38. orb04 data set için en iyi çizelgeleme sonucu.

(10×10)'luk orb05 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.39'da gösterilmiştir (C_{max}=887 sn.).



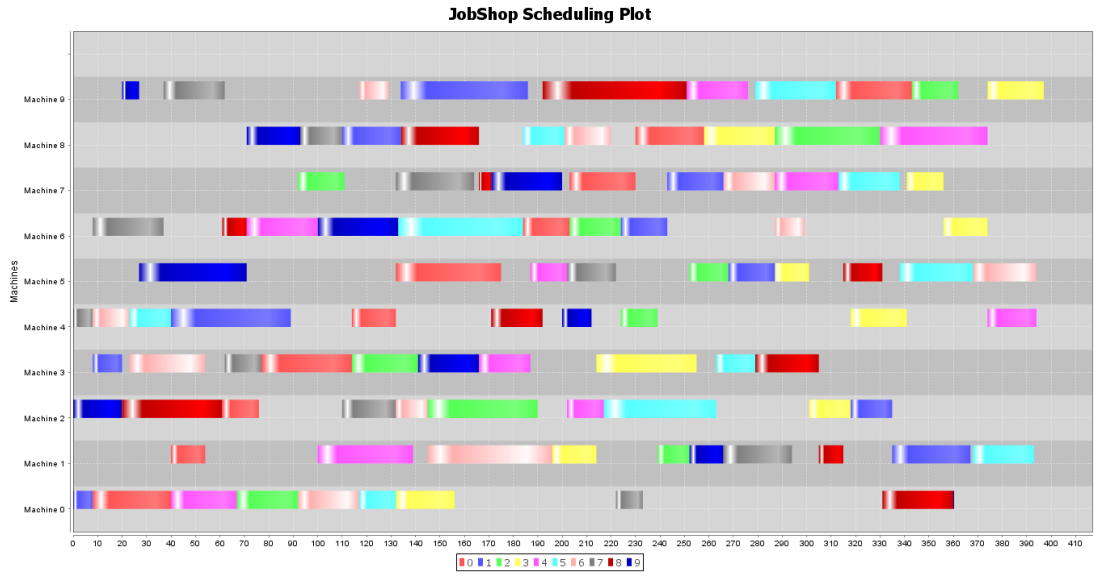
Şekil 4.39. orb05 data set için en iyi çizelgeleme sonucu.

(10×10)'luk orb06 data seti için elde edilen en iyi çözüm Şekil 4.40'ta gösterilmiştir (C_{max}=1010 sn.).



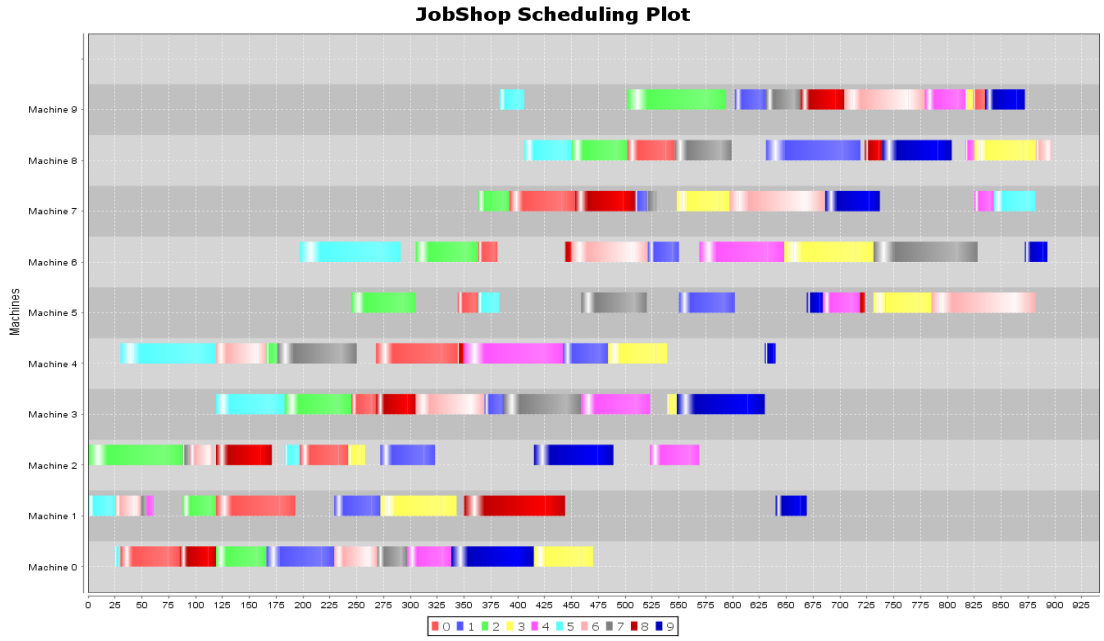
Şekil 4.40. orb06 data set için en iyi çizelgeleme sonucu.

(10×10)'luk orb07 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.41'de gösterilmiştir (C_{max}=397 sn.).



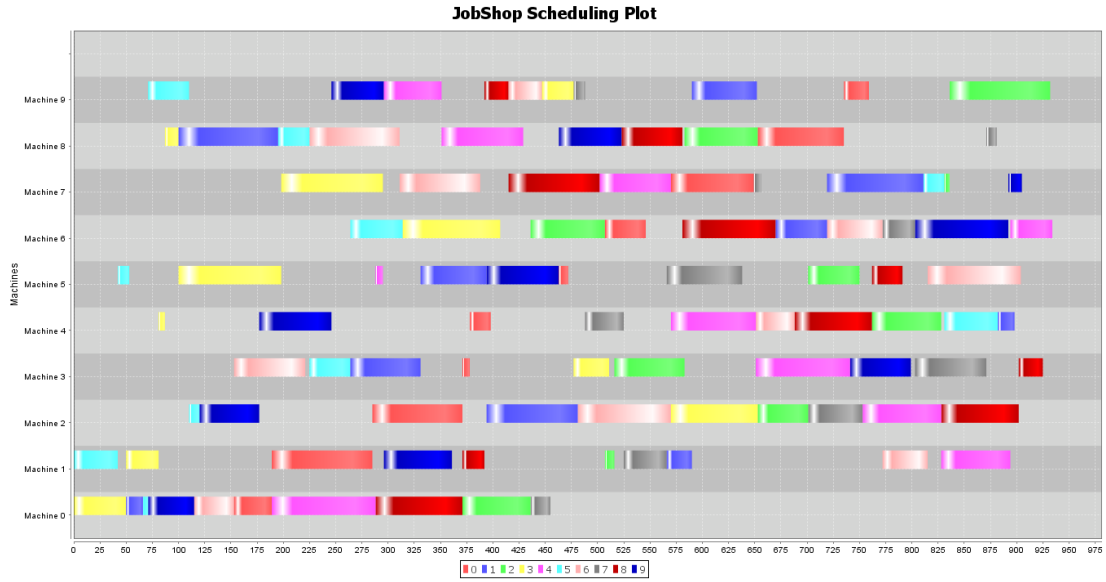
Şekil 4.41. orb07 data set için en iyi çizelgeleme sonucu.

(10×10)'luk orb08 data seti için elde edilen en iyi çözüm Şekil 4.42'de gösterilmiştir (C_{max}=899 sn.).



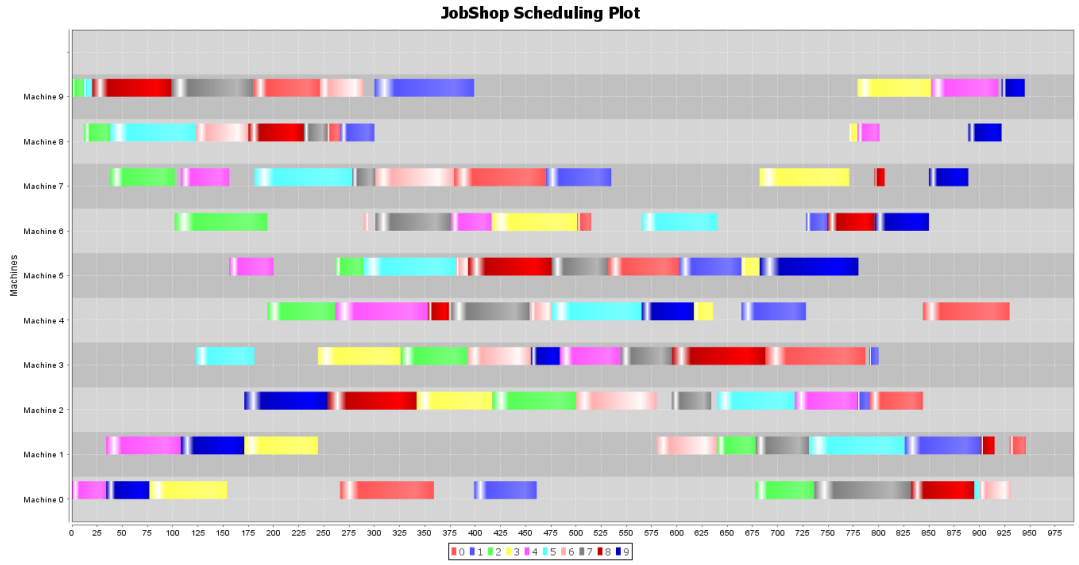
Şekil 4.42. orb08 data set için en iyi çizelgeleme sonucu.

(10×10)'luk orb09 data seti için elde edilen en iyi çizelgeleme sonucu Şekil 4.43'te gösterilmiştir (C_{max}=934 sn.).



Şekil 4.43. orb09 data set için en iyi çizelgeleme sonucu.

(10×10)'luk orb10 data seti için elde edilen en iyi çözüm Şekil 4.44'te gösterilmiştir (C_{max}=944 sn.).



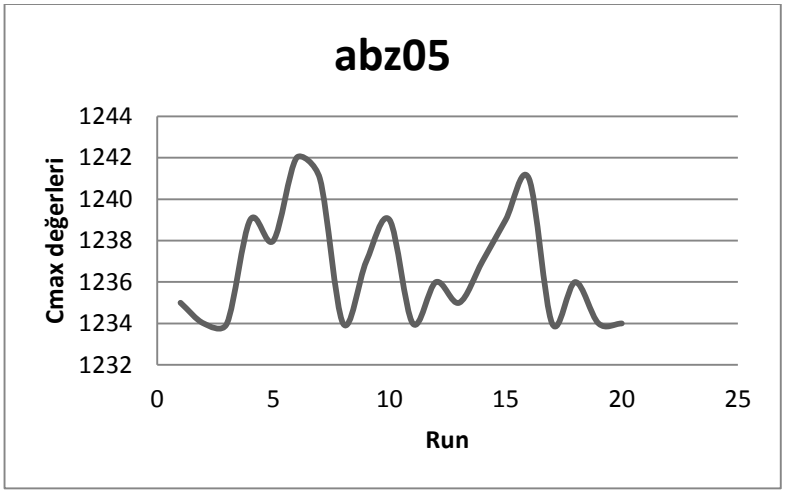
Şekil 4.44. orb10 data set için en iyi çizelgeleme sonucu.

Yukarıda en iyi çizelgelemeleri verilmiş olan 36 data setinin toplam tamamlanma zamanlarını ifade eden C_{max} değerine göre bakıldığında, ilgili data setlerinin literatürdeki [179] en iyi sonuçlarına göre bakıldığında 36 data seti için de en iyi sonuçlar ABC-EA bütünleşik yaklaşımı ile elde edilmiştir.

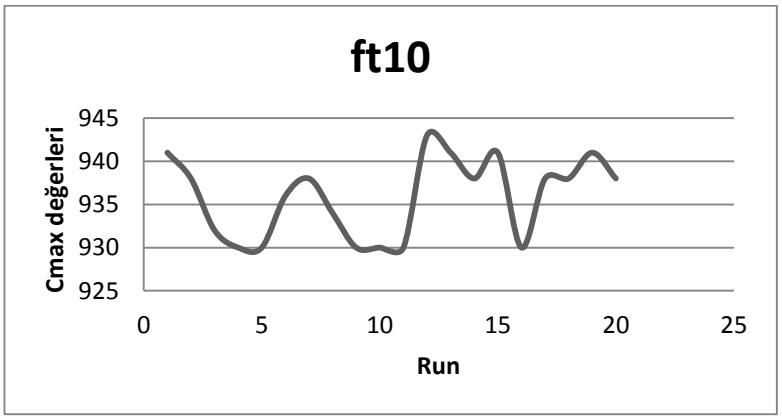
4.3.2. Farklı iterasyonlardan elde edilen sonuçlar

4.3.2.1. Elde edilen 20 iterasyonun sonuçları

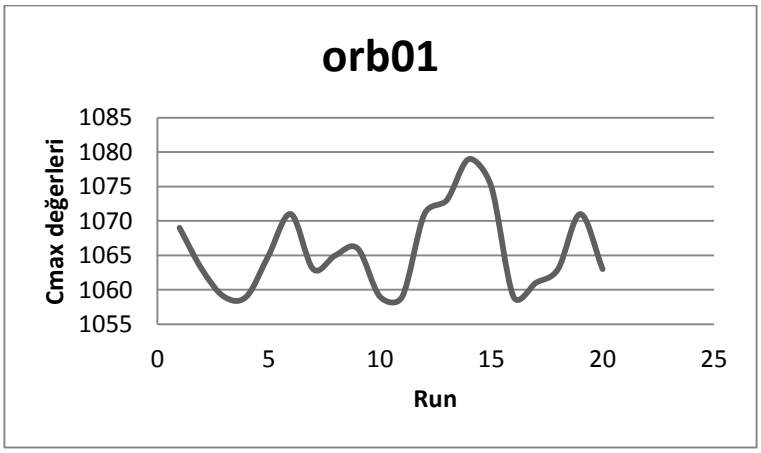
Atölye tipi çizelgeleme problemleriyle ilgili önerilen metot farklı iterasyonlarda çalıştırılarak önerilen metodolojinin farklı iterasyon sayılarına nasıl cevap verdiği gözlemlenmiştir. Yine bu çalışmanın "Sonuç ve Öneriler" bölümünde önerilen metodun farklı optimizasyon metotları ile kıyaslanmasında kullanıldığı için ilk olarak kıyaslamada kullanılmak üzere farklı data setleri için kurulan program 20 kez tekrar (iterasyon) çalıştırılmış (run) ve sonuçları alınarak, aşağıda grafiklerde verilmiştir. Şekil 4.45 ile Şekil 4.72 arasındaki grafikler bu data setleri için 20 iterasyon sonuçlarını göstermektedir.



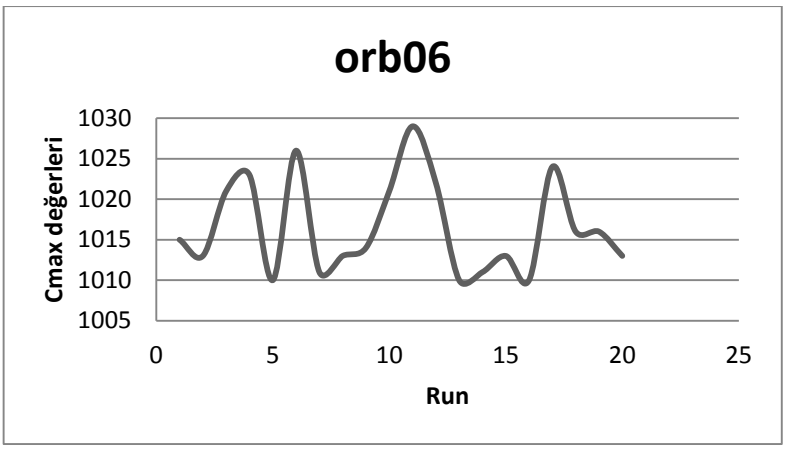
Şekil 4.45. abz05 data set için 20 run değeri.



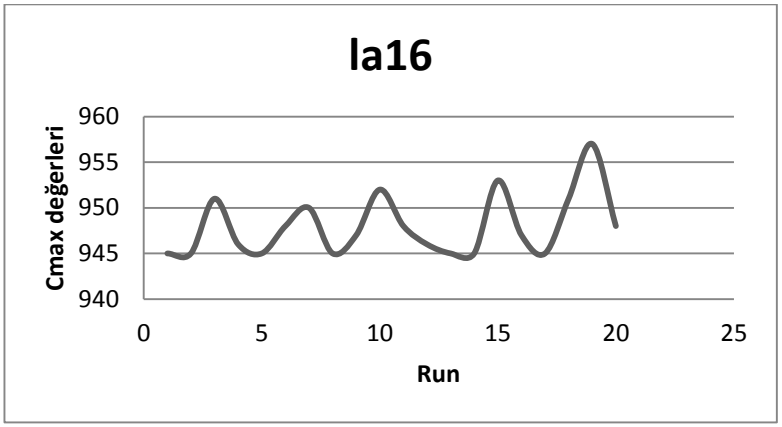
Şekil 4.46. ft10 data set için 20 run değeri.



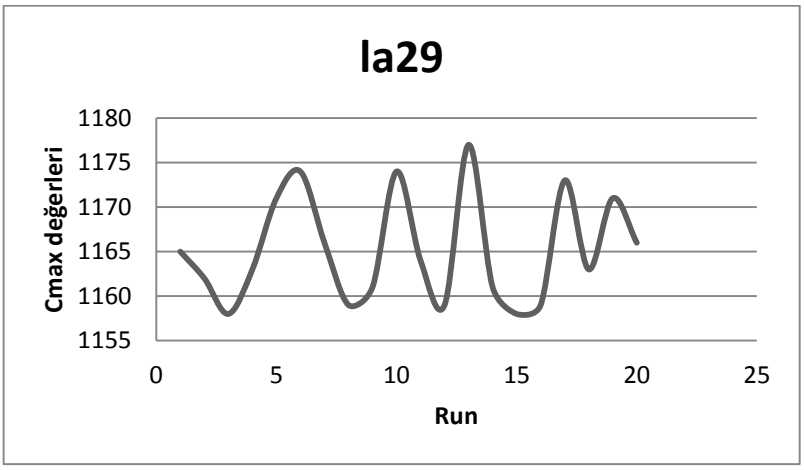
Şekil 4.47. orb01 data set için 20 run değeri.



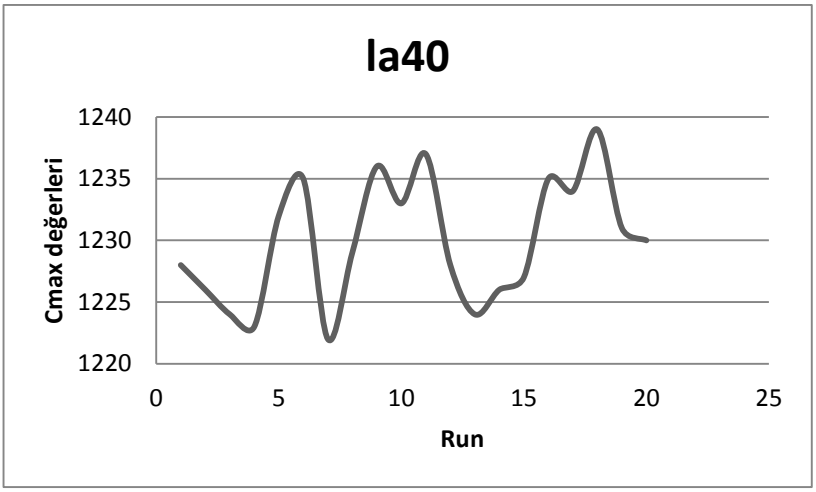
Şekil 4.48. orb06 data set için 20 run değeri.



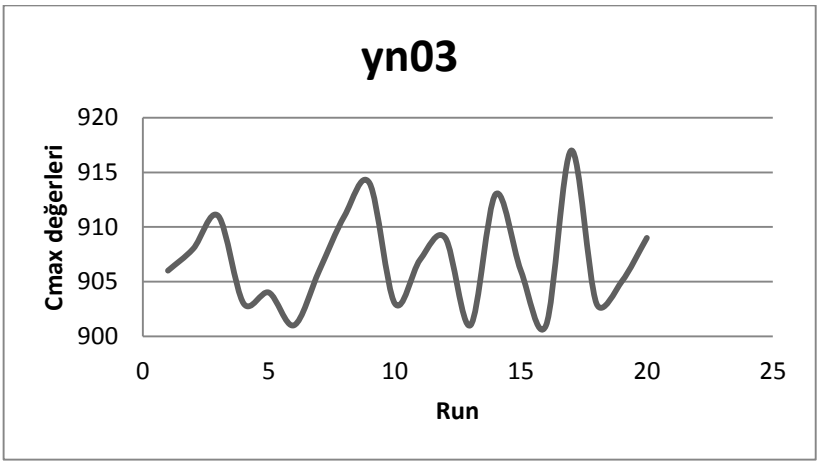
Şekil 4.49. la16 data set için 20 run değeri.



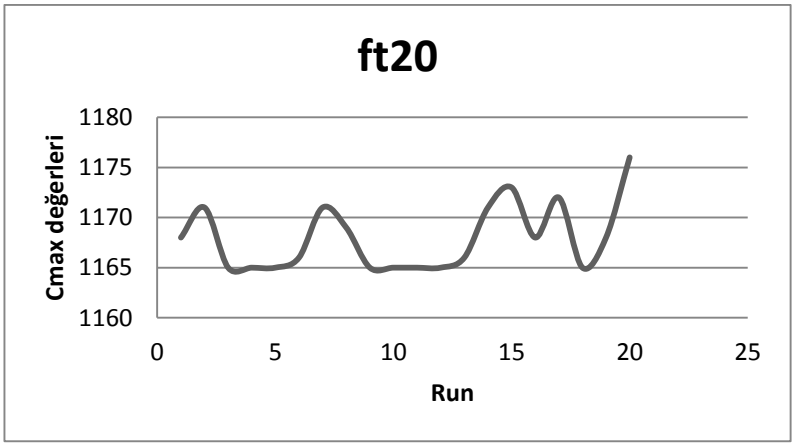
Şekil 4.50. la29 data set için 20 run değeri.



Şekil 4.51. la40 data set için 20 run değeri.

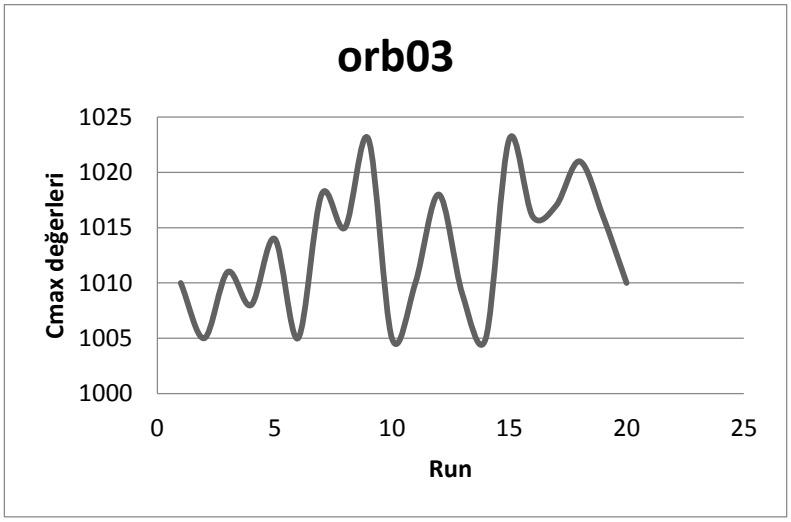


Şekil 4.52. yn03 data set için 20 run değeri.

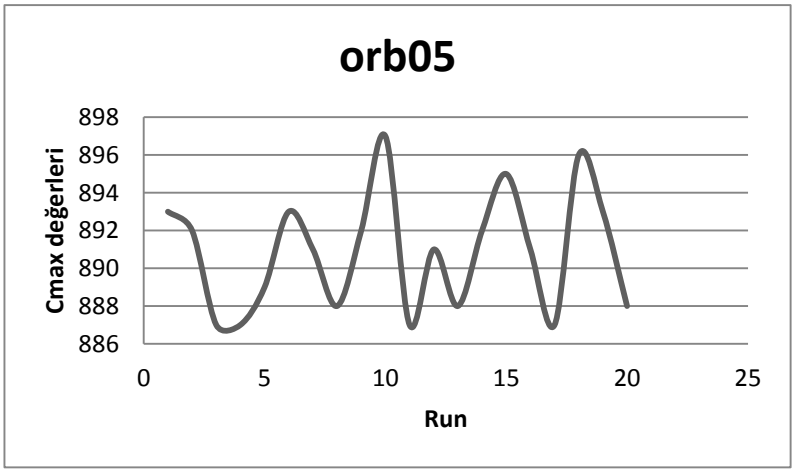


Şekil 4.53. ft20 data set için 20 run değeri.

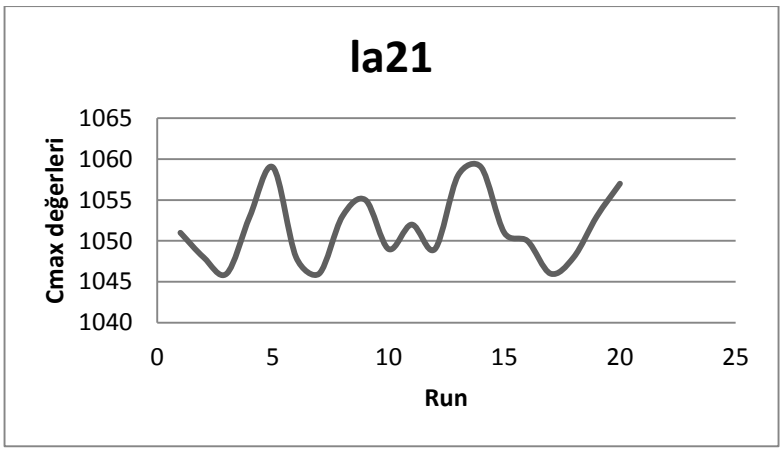
Şekil 4.54. orb02 data set için 20 run değeri.



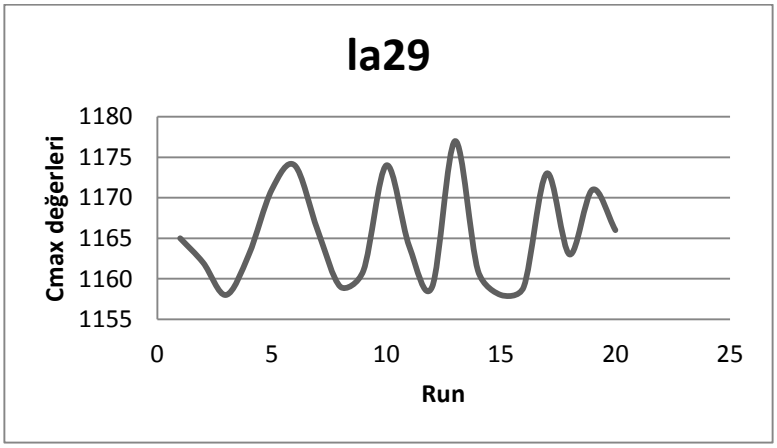
Şekil 4.55. orb03 data set için 20 run değeri.



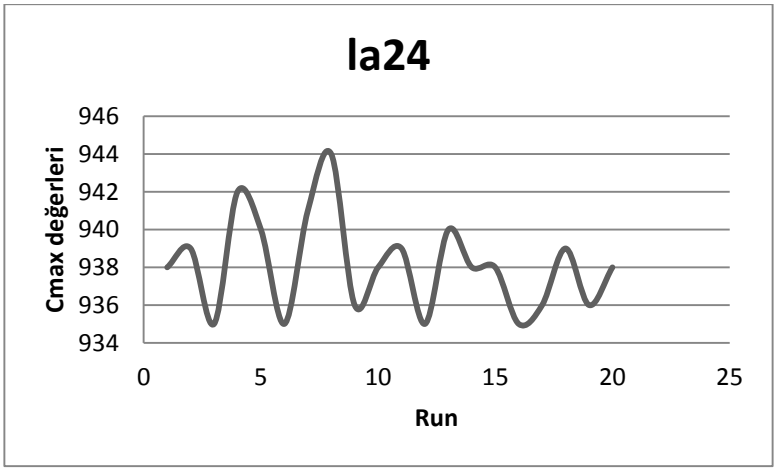
Şekil 4.56. orb05 data set için 20 run değeri.



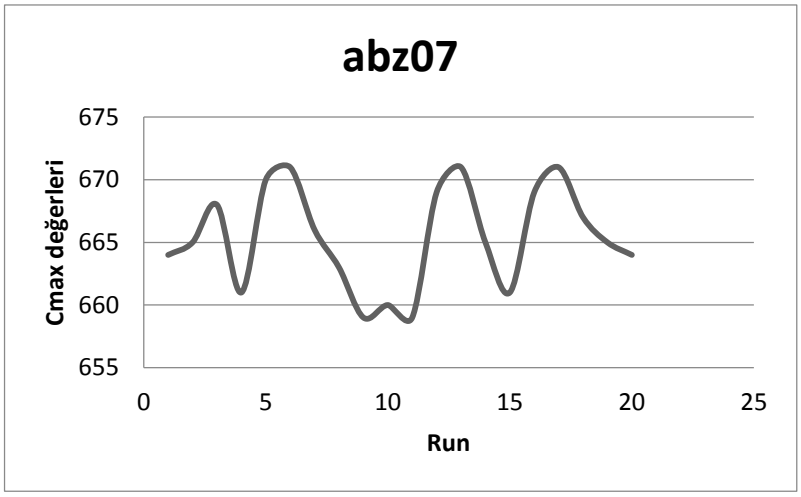
Şekil 4.57. la21 data set için 20 run değeri.



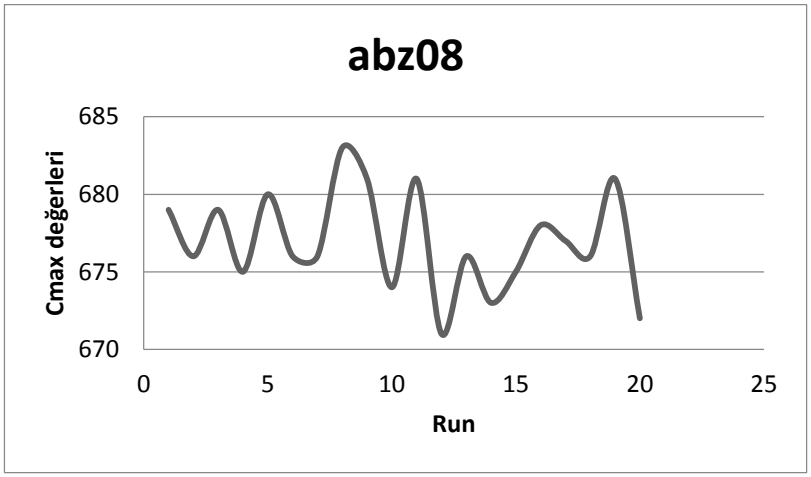
Şekil 4.58. la29 data set için 20 run değeri.



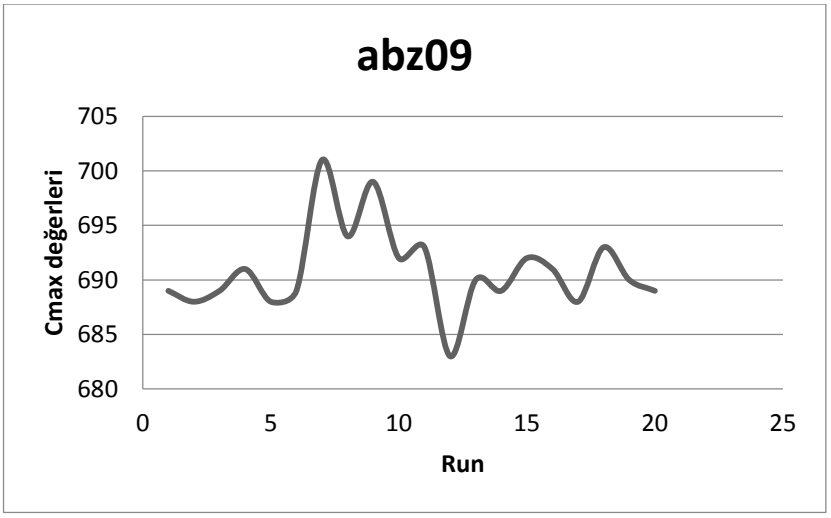
Şekil 4.59. la24 data set için 20 run değeri.



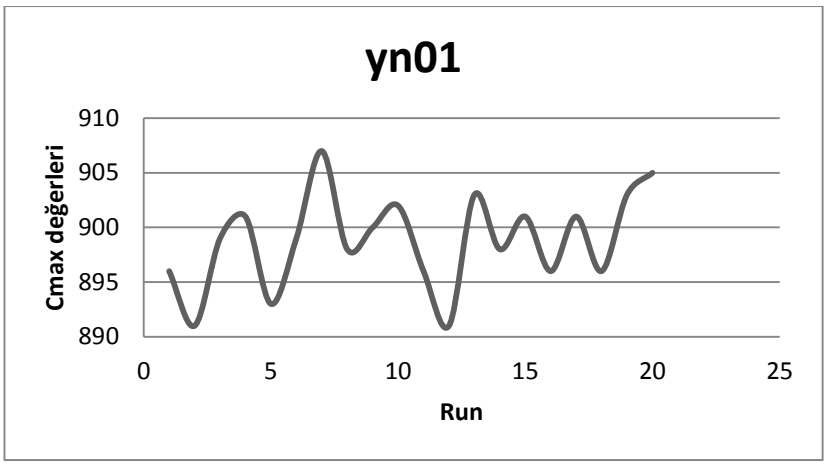
Şekil 4.60. abz07 data set için 20 run değeri.



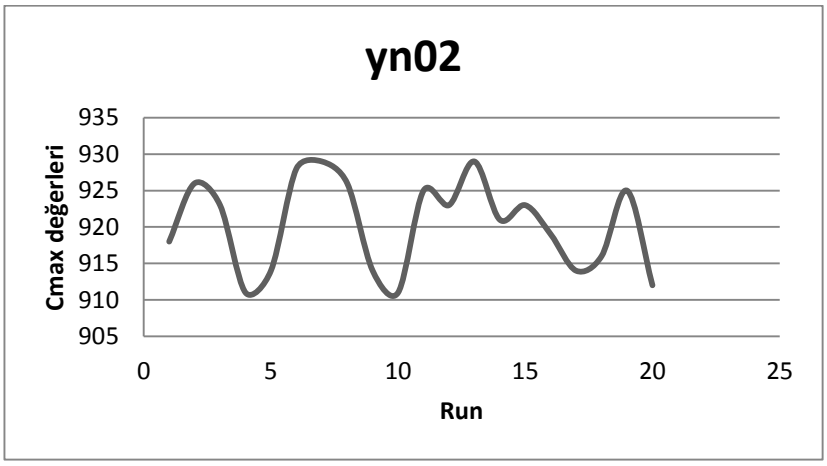
Şekil 4.61. abz08 data set için 20 run değeri.



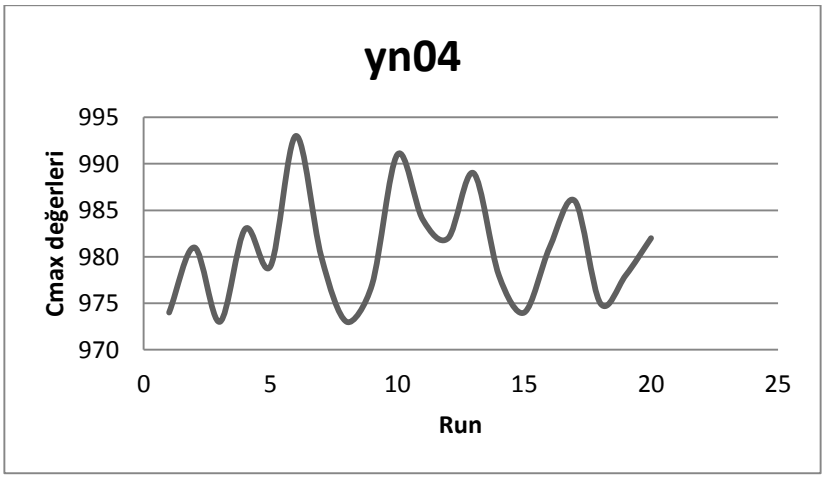
Şekil 4.62. abz09 data set için 20 run değeri.



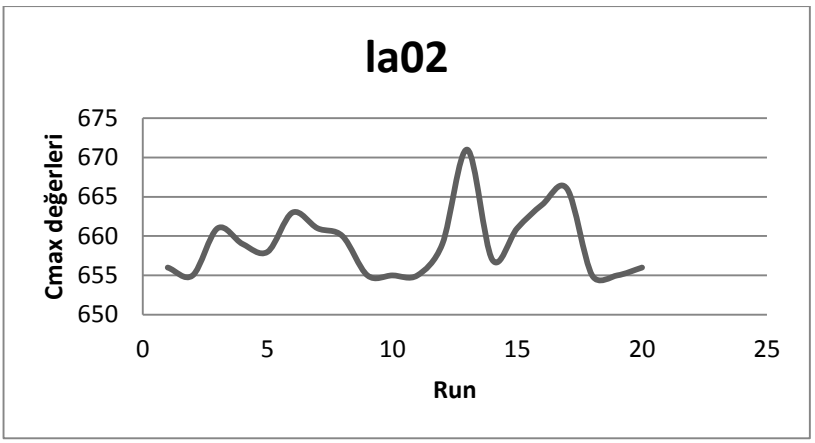
Şekil 4.63. yn01 data set için 20 run değeri.



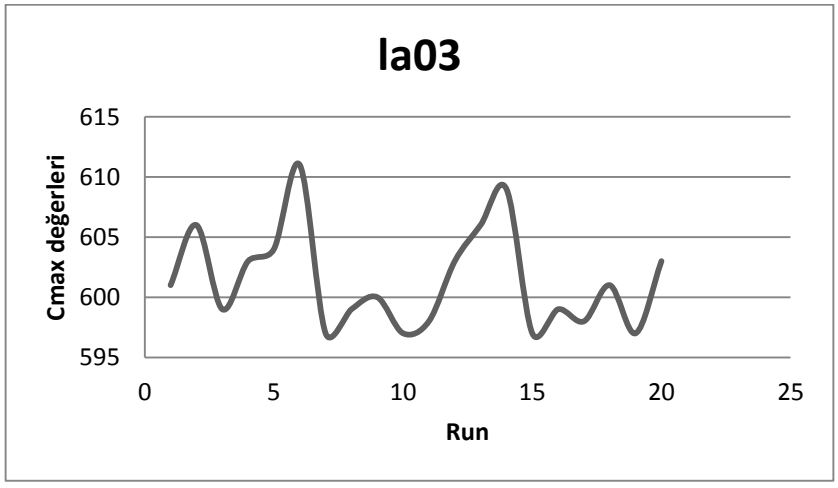
Şekil 4.64. yn02 data set için 20 run değeri.



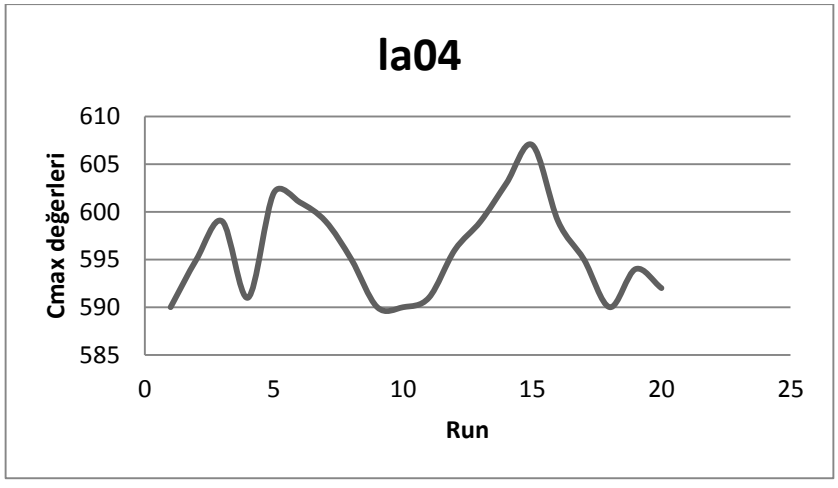
Şekil 4.65. yn04 data set için 20 run değeri.



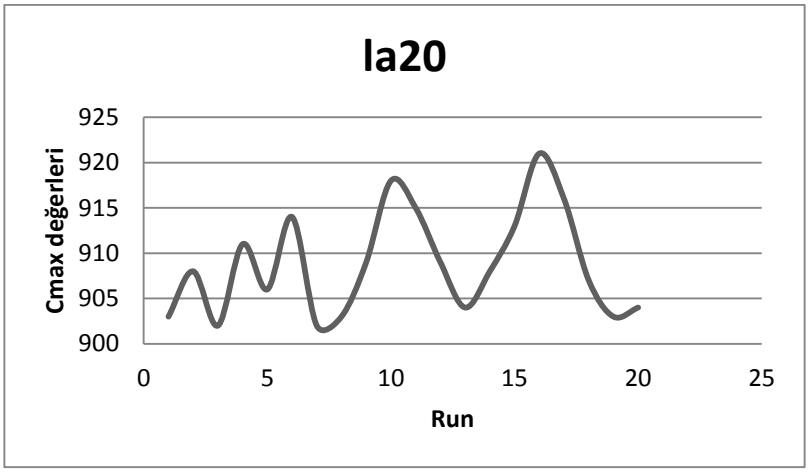
Şekil 4.66 la02 data set için 20 run değeri.



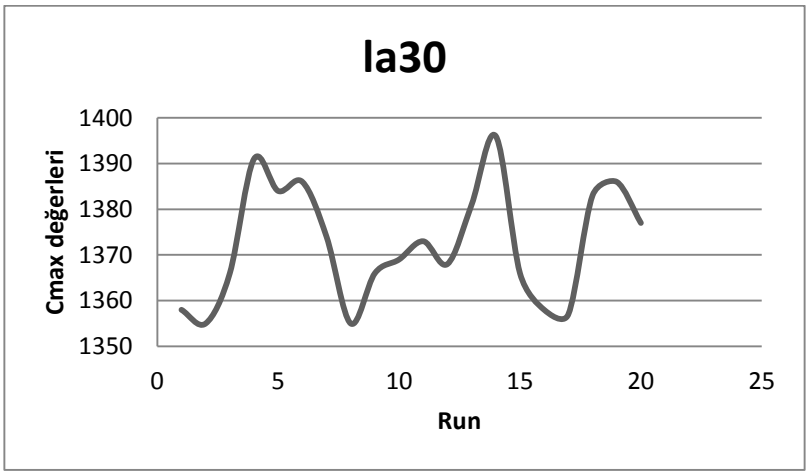
Şekil 4.67. la03 data set için 20 run değeri.



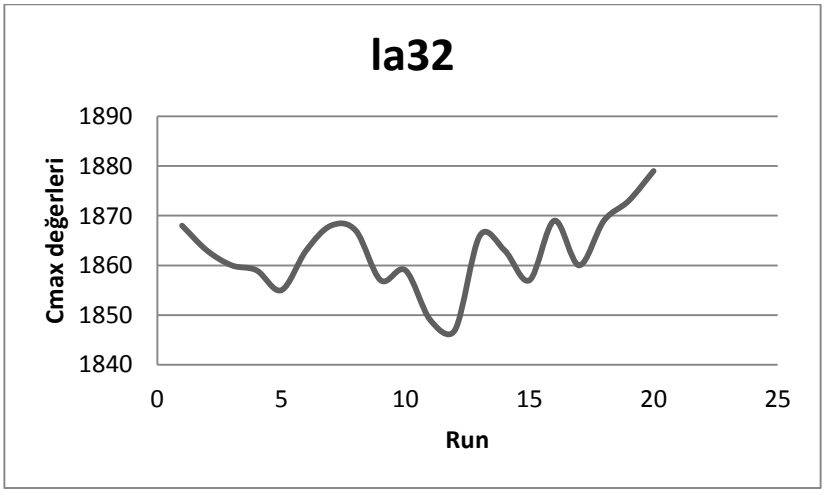
Şekil 4.68. la04 data set için 20 run değeri.



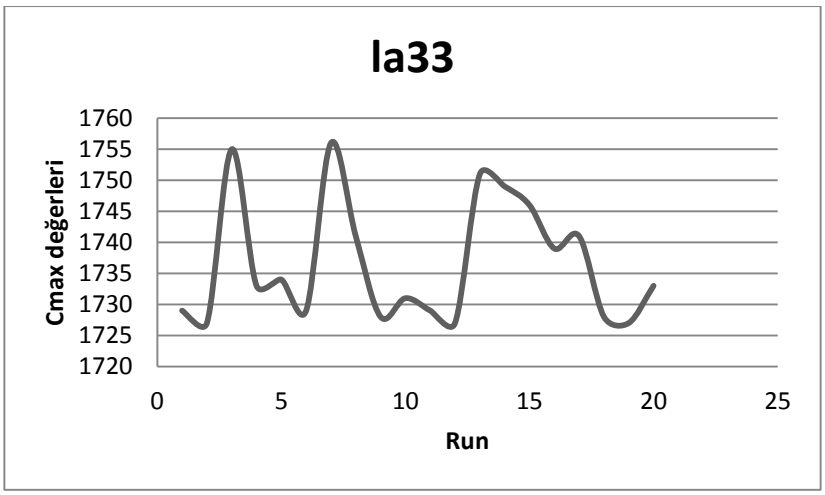
Şekil 4.69. la20 data set için 20 run değeri.



Şekil 4.70. la30 data set için 20 run değeri.



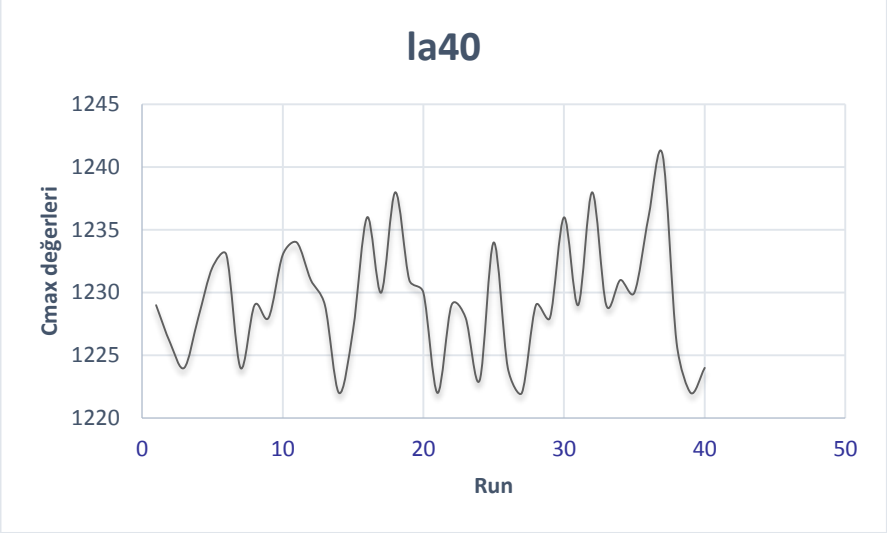
Şekil 4.71 la32 data set için 20 run değeri.



Şekil 4.72. la33 data set için 20 run değeri.

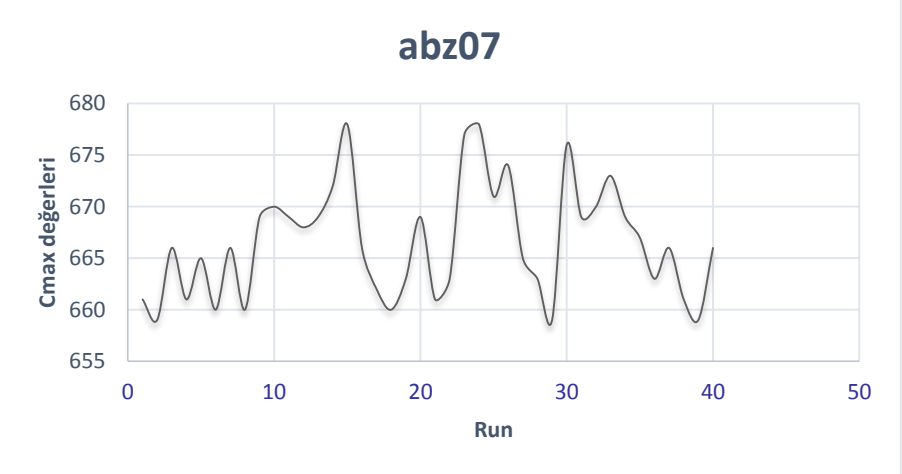
4.3.2.2. Elde edilen 40 iterasyon sonuçları

Önerilen metodolojinin farklı iterasyon sayılarına nasıl cevap verdiğini gözlemlemek için farklı data seti gruplarından 7 data seti alınarak 40 ve 60 iterasyon sonuçlar alınmış ve yine aşağıdaki şekillerde gösterilmiştir. Şekil 4.73 ile Şekil 4.79 arasındaki grafikler bu data setleri için 40 iterasyon sonuçlarını göstermektedir.

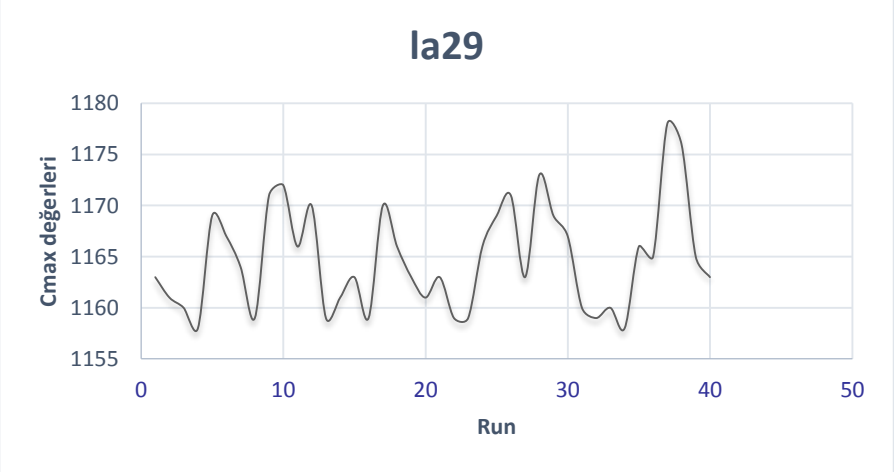


Şekil 4.73. la40 data set için 40 run değeri.

Şekil 4.74. yn02 data set için 40 run değeri.



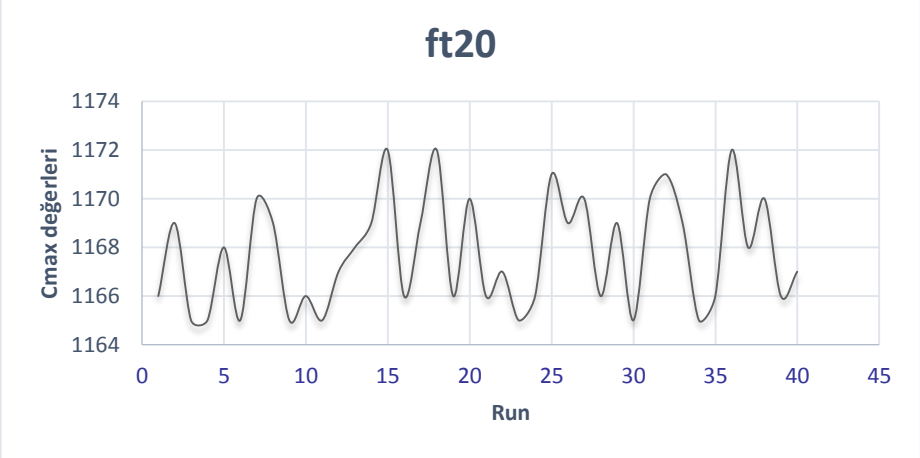
Şekil 4.75. abz07 data set için 40 run değeri.



Şekil 4.76. la29 data set için 40 run değeri.

Şekil 4.77. la21 data set için 40 run değeri.

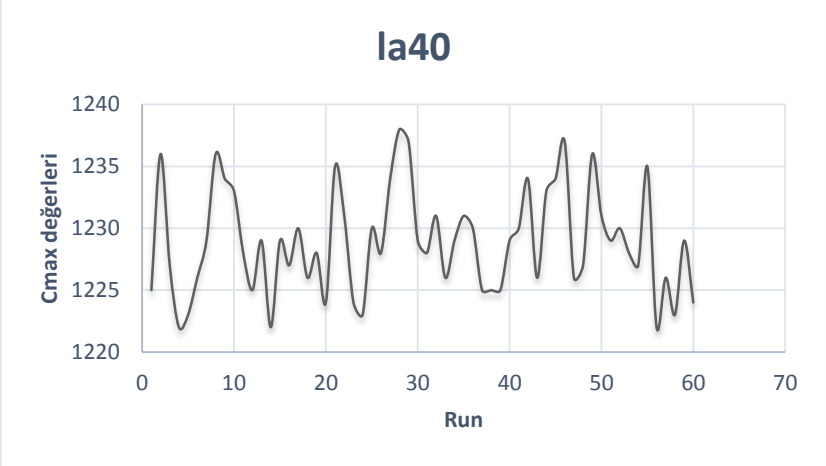
Şekil 4.78. orb02 data set için 40 run değeri.



Şekil 4.79. ft20 data set için 40 run değeri.

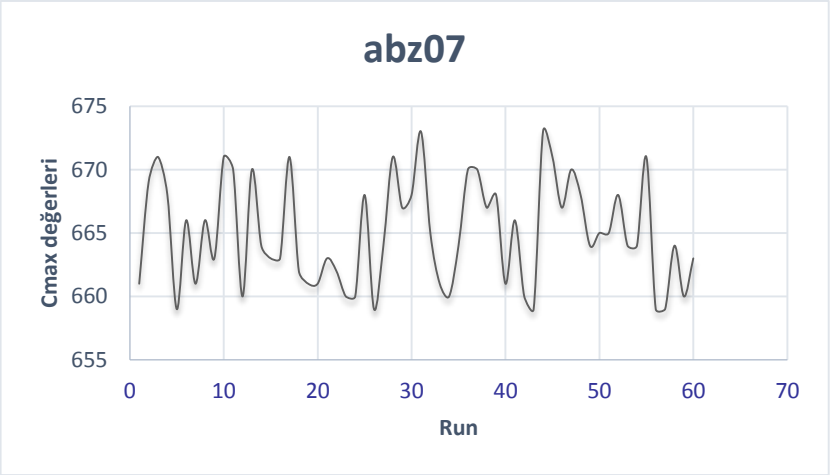
4.3.2.3. Elde edilen 60 iterasyon sonuçları

Yine aynı şekilde önerilen metodun ilgili 7 data seti alınarak 60 iterasyon sonuçlar alınmış ve yine aşağıdaki şekillerde gösterilmiştir. Şekil 4.80 ile Şekil 4.86 arasındaki grafikler bu data setleri için 60 iterasyon sonuçlarını göstermektedir.

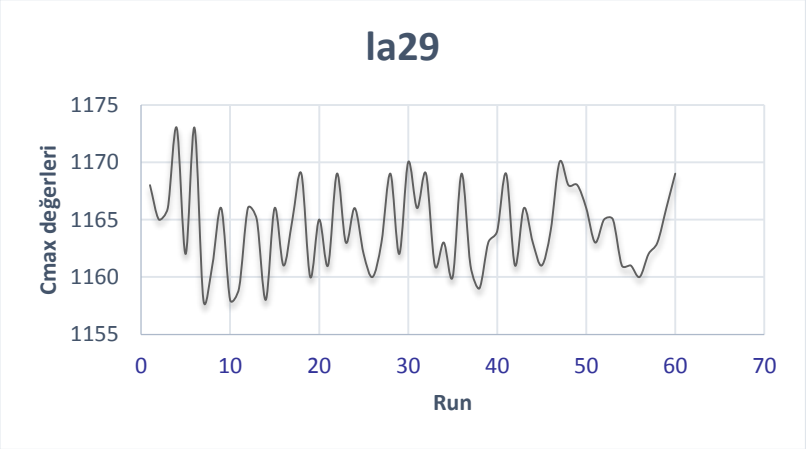


Şekil 4.80. la40 data set için 60 run değeri.

Şekil 4.81. yn02 data set için 60 run değeri.

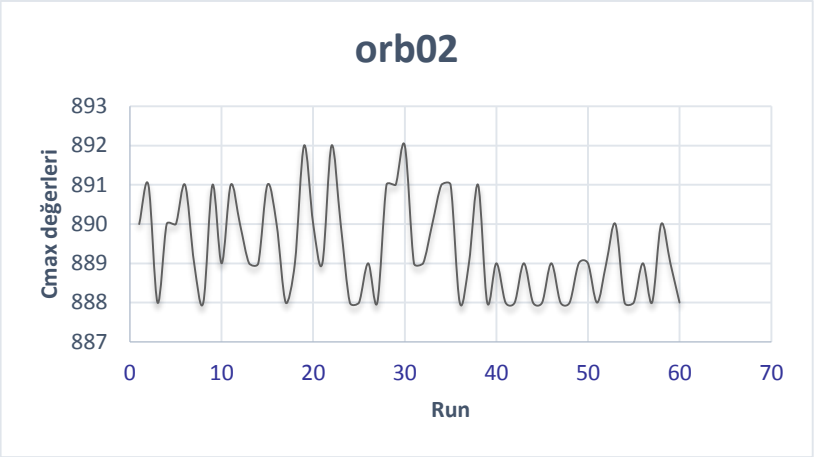


Şekil 4.82. abz07 data set için 60 run değeri.

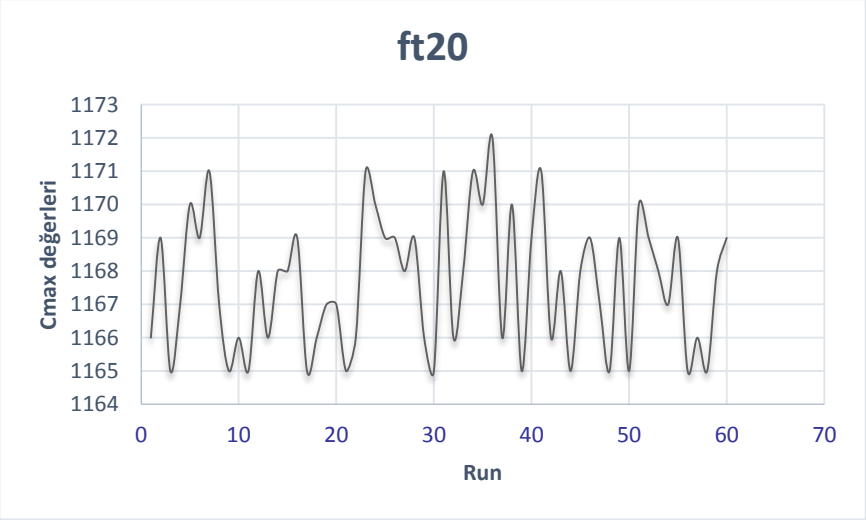


Şekil 4.83. la29 data set için 60 run değeri.

Şekil 4.84. la21 data set için 60 run değeri



Şekil 4.85. orb02 data set için 60 run değeri.



Şekil 4.86. ft20 data set için 60 run değeri.

4.3.3. Farklı iterasyonlardan elde edilen sonuçların kıyaslanması

4.3.3.1. Farklı iterasyonlardan elde edilen verilere varyans analizi uygulaması

Elde edilen verilerin kıyaslanmasında parametrik testleri (varyans analizi gibi) kullanabilmemiz için öncelikle verilerin % 95 güven aralığında normal dağılım göstermesi gerekmektedir [180, 181]. Çalışmanın bu kısmında SPSS 17.0 paket program kullanılarak kıyaslanan veri setleri için normal dağılım testi yapılmıştır. Bu doğrulama yapıldıktan sonra veri setlerinin varyanslarının % 95 güven aralığında

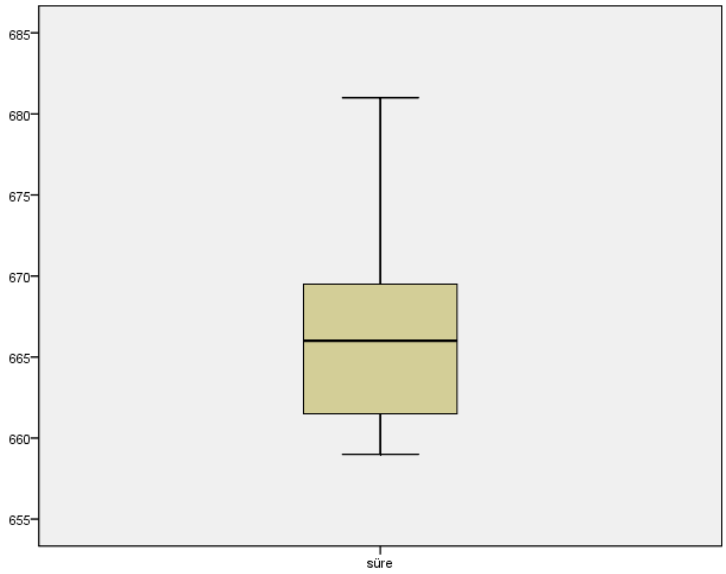
homojen dağılıp dağılmadığını kontrol etmemiz gerekmektedir. Bunun için de yine SPSS paket programı kullanılarak verilerin homojenliği gösterilmiştir. Veri setlerinin normal dağılımı ve varyanslarının homojen dağılıp dağılmadığı ile ilgili olarak sırasıyla aşağıdaki hipotezler kurulmuş ve bu hipotezlere göre sonuçlar alınmıştır.

- H_0: %95 güven aralığında veriler normal dağılımlıdır.
- H_1: %95 güven aralığında veriler normal dağılımlı değildir.

- H_0: %95 güven aralığında grup varyansları homojendir.
- H_1: %95 güven aralığında grup varyansları homojen değildir.

Hipotezler oluşturulduktan sonra 7 farklı data seti için 20,40 ve 60 iterasyon sonuçlarının normal dağılıma uygunluğu ve varyanslarının homojen olup olmadığı SPSS paket programında test edilmiş ve sonuçlar aşağıda sunulmuştur. Normal dağılıma uygunluğu için aşağıdaki şekilde "Descriptives" tablosundaki "Skewness" ve "Kurtosis" satırlarındaki değerlere bakılır.

Veri setinin normal dağılıma uygun olması için belirtilen değerlerin (-1,5) ile (+1,5) arasında olması gerekmektedir. Varyansların homojenliği için ise "Test of Homogeneity of Variances" tablosundaki "Sig." sütunundaki değere bakılır. Veri setinin homojen olması için bu değerin 0,05 değerinden büyük olması gerekmektedir [180, 181]. İlk olarak abz07 data setini ele alırsak bu data seti için C_{max} değerlerinin dağılımını gösteren kutu diyagramı Şekil 4.87'de gösterilmiştir.



Şekil 4.87. abz07 data seti için kutu diyagramı.

Kutu diyagramı ile ilgili data setindeki C_{max} değerlerinin dağılımı gösterildikten sonra sırasıyla normal dağılım ve varyansların homojenliği ile ilgili kurulmuş olan hipotezler test edilir.

Descriptives

			Statistic	Std. Error
C_{max}	Mean		666,07	,457
	95% Confidence Interval for Mean	Lower Bound	665,16	
		Upper Bound	666,97	
	5% Trimmed Mean		665,81	
	Median		666,00	
	Variance		25,054	
	Std. Deviation		5,005	
	Minimum		659	
	Maximum		681	
	Range		22	
	Interquartile Range		9	
	Skewness		,506	,221
	Kurtosis		-,220	,438

Şekil 4.88. abz07 data seti için normallik testi sonuçları.

Şekil 4.88 incelendiğinde "Skewness" ve "Kurtosis" satırlarındaki değerler (-1,5) ile (+1,5) arasında olduğu için H_0 hipotezi reddedilemez ve abz07 data seti için "%95 güven aralığında veriler normal dağılımlıdır" sonucuna ulaşılır.

Test of Homogeneity of Variances

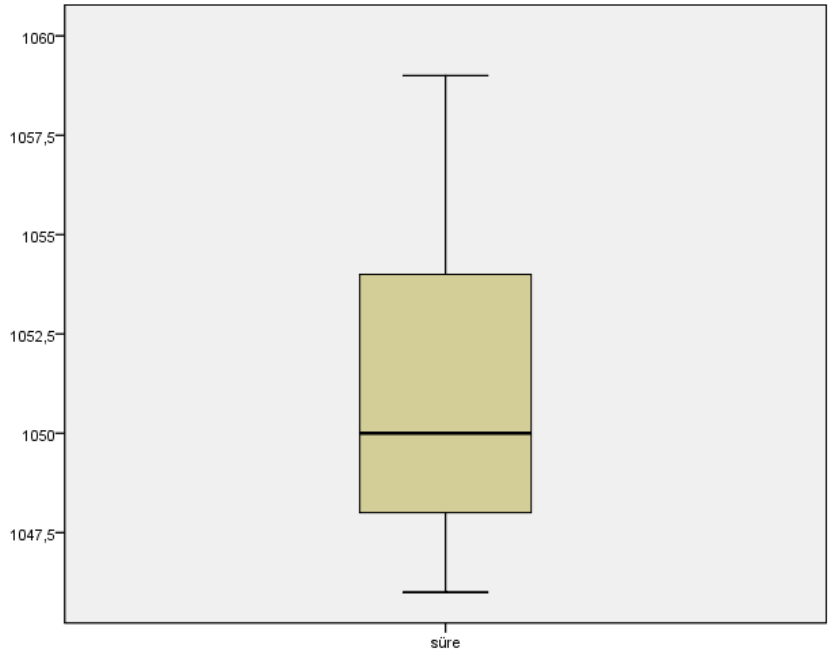
Cmax

Levene Statistic	df1	df2	Sig.
1,564	2	117	,214

Şekil 4.89. abz07 data seti için varyansların homojenlik testi sonuçları.

Şekil 4.89 incelendiğinde "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi reddedilemez ve abz07 data seti için "%95 güven aralığında grup varyansları homojendir" sonucuna ulaşılır.

Bir sonraki data seti la21 için C_{max} değerlerinin dağılımını gösteren kutu diyagramı Şekil 4.90'da gösterilmiştir.



Şekil 4.90. la21 data seti için kutu diyagramı.

Descriptives

			Statistic	Std. Error
Cmax	Mean		1051,00	,350
	95% Confidence Interval for Mean	Lower Bound	1050,31	
		Upper Bound	1051,69	
	5% Trimmed Mean		1050,83	
	Median		1050,00	
	Variance		14,689	
	Std. Deviation		3,833	
	Minimum		1046	
	Maximum		1059	
	Range		13	
	Interquartile Range		6	
	Skewness		,478	,221
	Kurtosis		-,708	,438

Şekil 4.91. la21 data seti için normallik testi sonuçları.

Şekil 4.91 incelendiğinde "Skewness" ve "Kurtosis" satırlarındaki değerler (-1,5) ile (+1,5) arasında olduğu için H_0 hipotezi reddedilemez ve la21 data seti için "%95 güven aralığında veriler normal dağılımlıdır" sonucuna ulaşılır.

Test of Homogeneity of Variances

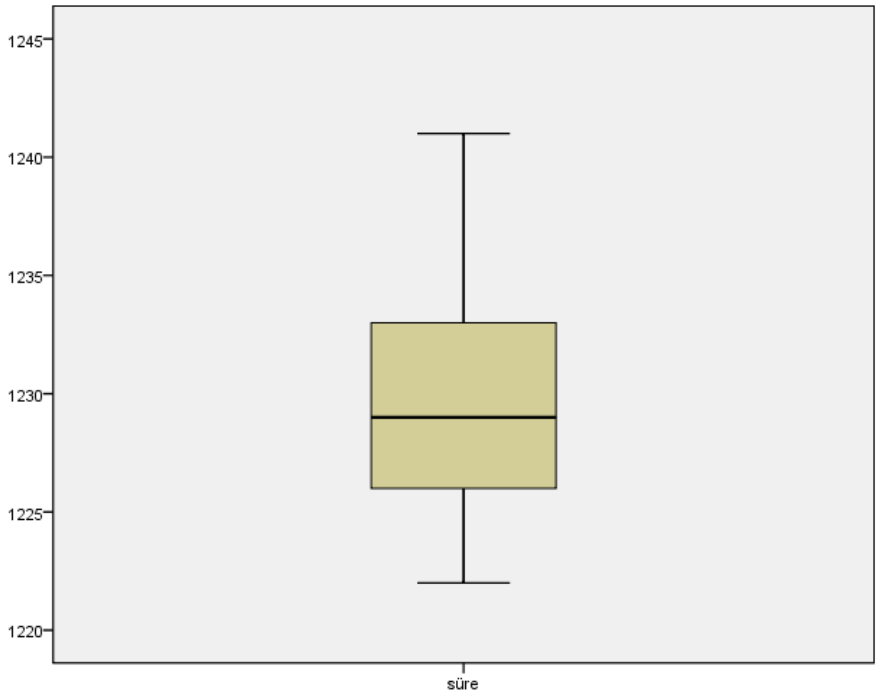
C_{max}

Levene Statistic	df1	df2	Sig.
,301	2	117	,741

Şekil 4.92. la21 data seti için varyansların homojenlik testi sonuçları.

Şekil 4.92 incelendiğinde "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi reddedilemez ve la21 data seti için "%95 güven aralığında grup varyansları homojendir" sonucuna ulaşılır.

la40 data seti için C_{max} değerlerinin dağılımını gösteren kutu diyagramı Şekil 4.93'te gösterilmiştir.



Şekil 4.93. la40 data seti için kutu diyagramı.

Descriptives

			Statistic	Std. Error
Cmax	Mean		1229,23	,416
	95% Confidence Interval for Mean	Lower Bound	1228,41	
		Upper Bound	1230,06	
	5% Trimmed Mean		1229,12	
	Median		1229,00	
	Variance		20,802	
	Std. Deviation		4,561	
	Minimum		1222	
	Maximum		1241	
	Range		19	
	Interquartile Range		7	
	Skewness		,334	,221
	Kurtosis		-,628	,438

Şekil 4.94. la40 data seti için normallik testi sonuçları.

Şekil 4.94 incelendiğinde "Skewness" ve "Kurtosis" satırlarındaki değerler (-1,5) ile (+1,5) arasında olduğu için H_0 hipotezi reddedilemez ve la40 data seti için "%95 güven aralığında veriler normal dağılımlıdır" sonucuna ulaşılır.

Test of Homogeneity of Variances

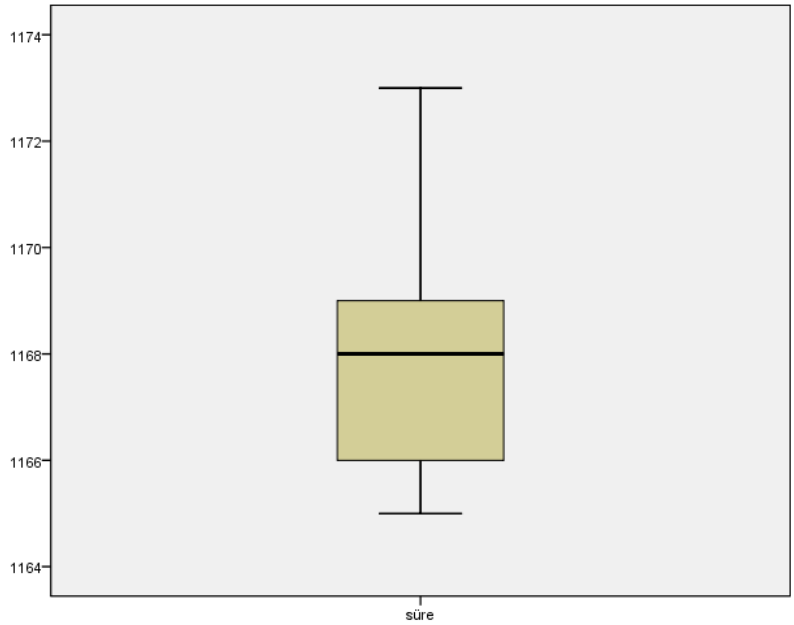
Cmax

Levene Statistic	df1	df2	Sig.
,770	2	117	,465

Şekil 4.95. la40 data seti için varyansların homojenlik testi sonuçları.

Şekil 4.95 incelendiğinde "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi reddedilemez ve la40 data seti için "%95 güven aralığında grup varyansları homojendir" sonucuna ulaşılır.

ft20 data seti için C_{max} değerlerinin dağılımını gösteren kutu diyagramı Şekil 4.96'da gösterilmiştir.



Şekil 4.96. ft20 data seti için kutu diyagramı.

Descriptives

			Statistic	Std. Error
Cmax	Mean		1167,65	,203
	95% Confidence Interval for Mean	Lower Bound	1167,25	
		Upper Bound	1168,05	
	5% Trimmed Mean		1167,55	
	Median		1168,00	
	Variance		4,952	
	Std. Deviation		2,225	
	Minimum		1165	
	Maximum		1173	
	Range		8	
	Interquartile Range		3	
	Skewness		,386	,221
	Kurtosis		-,992	,438

Şekil 4.97. ft20 data seti için normallik testi sonuçları.

Şekil 4.97 incelendiğinde "Skewness" ve "Kurtosis" satırlarındaki değerler (-1,5) ile (+1,5) arasında olduğu için H_0 hipotezi reddedilemez ve ft20 data seti için "%95 güven aralığında veriler normal dağılımlıdır" sonucuna ulaşılır.

Test of Homogeneity of Variances

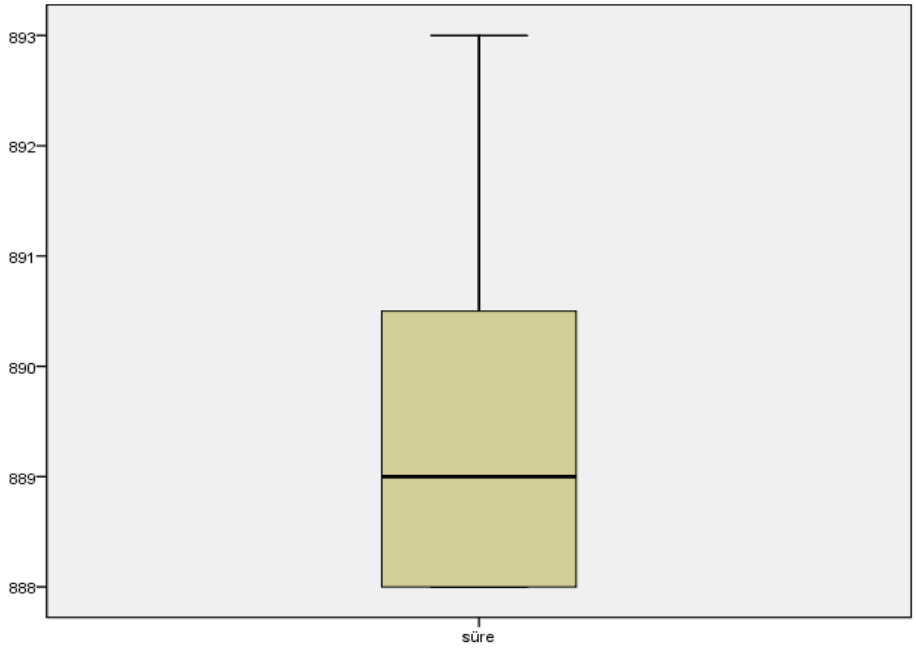
Cmax

Levene Statistic	df1	df2	Sig.
2,968	2	117	,055

Şekil 4.98. ft20 data seti için varyansların homojenlik testi sonuçları.

Şekil 4.98 incelendiğinde "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi reddedilemez ve ft20 data seti için "%95 güven aralığında grup varyansları homojendir" sonucuna ulaşılır.

Bir sonraki data seti orb02 için C_{max} değerlerinin dağılımını gösteren kutu diyagramı Şekil 4.99'da gösterilmiştir.



Şekil 4.99. orb02 data seti için kutu diyagramı.

Descriptives

			Statistic	Std. Error
Cmax	Mean		889,50	,122
	95% Confidence Interval for Mean	Lower Bound	889,26	
		Upper Bound	889,74	
	5% Trimmed Mean		889,42	
	Median		889,00	
	Variance		1,782	
	Std. Deviation		1,335	
	Minimum		888	
	Maximum		893	
	Range		5	
	Interquartile Range		3	
	Skewness		,701	,221
	Kurtosis		-,319	,438

Şekil 4.100. orb02 data seti için normallik testi sonuçları.

Şekil 4.100 incelendiğinde “Skewness” ve “Kurtosis” satırlarındaki değerler (-1,5) ile (+1,5) arasında olduğu için H_0 hipotezi reddedilemez ve orb02 data seti için “%95 güven aralığında veriler normal dağılımlıdır” sonucuna ulaşılır.

Test of Homogeneity of Variances

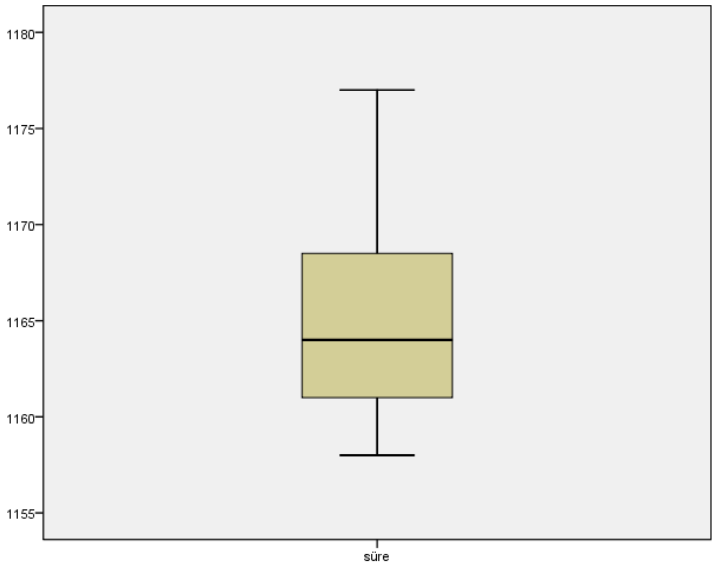
Cmax

Levene Statistic	df1	df2	Sig.
,895	2	117	,411

Şekil 4.101. orb02 data seti için varyansların homojenlik testi sonuçları.

Şekil 4.101 incelendiğinde “Sig.” değeri 0,05’ten büyük olduğu için H_0 hipotezi reddedilemez ve orb02 data seti için “%95 güven aralığında grup varyansları homojendir” sonucuna ulaşılır.

la29 data seti için C_{max} değerlerinin dağılımını gösteren kutu diyagramı Şekil 4.102’de gösterilmiştir.



Şekil 4.102. la29 data seti için kutu diyagramı.

Descriptives

			Statistic	Std. Error
Cmax	Mean		1164,63	,404
	95% Confidence Interval for Mean	Lower Bound	1163,83	
		Upper Bound	1165,42	
	5% Trimmed Mean		1164,44	
	Median		1164,00	
	Variance		19,547	
	Std. Deviation		4,421	
	Minimum		1158	
	Maximum		1177	
	Range		19	
	Interquartile Range		8	
	Skewness		,503	,221
	Kurtosis		-,380	,438

Şekil 4.103. la29 data seti için normallik testi sonuçları.

Şekil 4.103 incelendiğinde "Skewness" ve "Kurtosis" satırlarındaki değerler (-1,5) ile (+1,5) arasında olduğu için H_0 hipotezi reddedilemez ve la29 data seti için "%95 güven aralığında veriler normal dağılımlıdır" sonucuna ulaşılır.

Test of Homogeneity of Variances

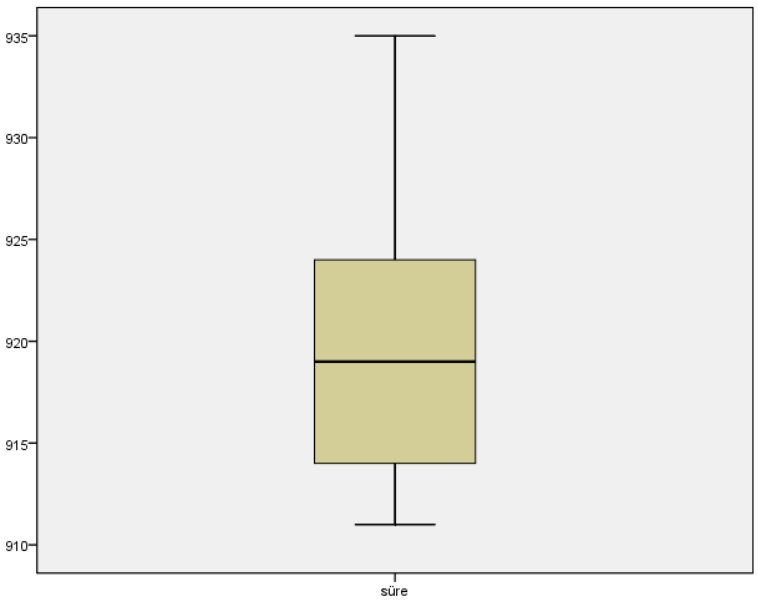
Cmax

Levene Statistic	df1	df2	Sig.
2,203	2	117	,115

Şekil 4.104. la29 data seti için varyansların homojenlik testi sonuçları.

Şekil 4.104 incelendiğinde "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi reddedilemez ve la29 data seti için "%95 güven aralığında grup varyansları homojendir" sonucuna ulaşılır.

Bir sonraki data seti yn02 için C_{max} değerlerinin dağılımını gösteren kutu diyagramı Şekil 4.105'te gösterilmiştir.



Şekil 4.105. yn02 data seti için kutu diyagramı.

Descriptives

			Statistic	Std. Error
Cmax	Mean		919,37	,528
	95% Confidence Interval for Mean	Lower Bound	918,32	
		Upper Bound	920,41	
	5% Trimmed Mean		919,21	
	Median		919,00	
	Variance		33,511	
	Std. Deviation		5,789	
	Minimum		911	
	Maximum		935	
	Range		24	
	Interquartile Range		10	
	Skewness		,291	,221
	Kurtosis		-,923	,438

Şekil 4.106. yn02 data seti için normallik testi sonuçları.

Şekil 4.106 incelendiğinde "Skewness" ve "Kurtosis" satırlarındaki değerler (-1,5) ile (+1,5) arasında olduğu için H_0 hipotezi reddedilemez ve yn02 data seti için "%95 güven aralığında veriler normal dağılımlıdır" sonucuna ulaşılır.

Test of Homogeneity of Variances

Cmax

Levene Statistic	df1	df2	Sig.
,275	2	117	,760

Şekil 4.107. yn02 data seti için varyansların homojenlik testi sonuçları.

Şekil 4.107 incelendiğinde "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi reddedilemez ve yn02 data seti için "%95 güven aralığında grup varyansları homojendir" sonucuna ulaşılır.

Yapılan hipotez testleri sonucunda ilgili data setleri için veri setlerinin normal dağıldığı ve varyanslarının homojen olduğu gözlemlenmiştir. Bu aşamadan sonra 7 farklı data setinin 20,40 ve 60 iterasyon sonuçları için varyans analizi yapılacaktır. Bunun için "önerilen metodun veri setlerine uygulanması sonucu C_{max} değerlerinin ortalaması farklı iterasyonlarda istatistiksel olarak farklılık gösterir mi?" hipotezi test edilecektir. Farklı data setlerinin herbiri için konu ile ilgili aşağıdaki hipotezler kurulmuş ve bu hipotezlere göre sonuçlar alınmıştır.

- H_0: %95 güven aralığında farklı iterasyonlarda C_{max} değerlerinin ortalaması için gruplar arasında istatistiksel olarak anlamlı bir fark yoktur.
- H_1: %95 güven aralığında farklı iterasyonlarda C_{max} değerlerinin ortalaması için en az bir grubun ortalaması diğerlerinden farklıdır.

Hipotezler oluşturulduktan sonra 7 farklı data seti için farklı iterasyonlarda C_{max} değerlerinin ortalaması için gruplar arasında anlamlı bir fark olup olmadığının analiz edilmesinde yine SPSS paket programı kullanılmış ve tek yönlü varyans analizi yapılmıştır. Varyans analizi yapılırken "ANOVA" tablosundaki "Sig." sütunundaki değere bakılır. Gruplar arasında anlamlı bir fark olması için bu değerin 0,05'ten küçük

olması gerekmektedir [180, 181]. İlk olarak abz07 data setini ele alırsak bu data seti için tek tönlü varyans analizi Şekil 4.108'de gösterilmiştir.

ANOVA

Cmax

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	167,508	2	83,754	3,482	,034
Within Groups	2813,958	117	24,051		
Total	2981,467	119			

Şekil 4.108. abz07 data seti için tek yönlü varyans analizi sonuçları.

Şekil 4.108 incelendiğinde "Sig." değeri 0,05'ten küçük olduğu için H_0 hipotezi reddedilir ve abz07 data seti için "%95 güven aralığında farklı iterasyonlarda C_{max} değerlerinin ortalaması için en az bir grubun ortalaması diğerlerinden farklıdır" sonucuna ulaşılır. Diğer data setleri de sırasıyla ele alınarak tek yönlü varyans analizi uygulanmış ve kurulan hipoteze göre sonuçlar alınmıştır.

ANOVA

Cmax

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	7,742	2	3,871	,260	,771
Within Groups	1740,258	117	14,874		
Total	1748,000	119			

Şekil 4.109. la21 data seti için tek yönlü varyans analizi sonuçları.

Şekil 4.109 incelendiğinde "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi kabul edilir ve la21 data seti için "%95 güven aralığında farklı iterasyonlarda C_{max} değerlerinin ortalaması için gruplar arasında istatistiksel olarak anlamlı bir fark yoktur" sonucuna ulaşılır.

ANOVA

Cmax

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	17,742	2	8,871	,422	,657
Within Groups	2457,725	117	21,006		
Total	2475,467	119			

Şekil 4.110. la40 data seti için tek yönlü varyans analizi sonuçları.

Şekil 4.110 incelendiğinde "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi kabul edilir ve la40 data seti için "%95 güven aralığında farklı iterasyonlarda C_{max} değerlerinin ortalaması için gruplar arasında istatistiksel olarak anlamlı bir fark yoktur" sonucuna ulaşılır.

ANOVA

Cmax

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1,200	2	,600	,119	,888
Within Groups	588,100	117	5,026		
Total	589,300	119			

Şekil 4.111. ft20 data seti için tek yönlü varyans analizi sonuçları.

Şekil 4.111 incelendiğinde "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi kabul edilir ve ft20 data seti için "%95 güven aralığında farklı iterasyonlarda C_{max} değerlerinin ortalaması için gruplar arasında istatistiksel olarak anlamlı bir fark yoktur" sonucuna ulaşılır.

ANOVA

Cmax

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	3,467	2	1,733	,973	,381
Within Groups	208,533	117	1,782		
Total	212,000	119			

Şekil 4.112. orb02 data seti için tek yönlü varyans analizi sonuçları.

Şekil 4.112 incelendiğinde "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi kabul edilir ve orb02 data seti için "%95 güven aralığında farklı iterasyonlarda C_{max}

değerlerinin ortalaması için gruplar arasında istatistiksel olarak anlamlı bir fark yoktur" sonucuna ulaşılır.

ANOVA

Cmax

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	,017	2	,008	,000	1,000
Within Groups	2326,108	117	19,881		
Total	2326,125	119			

Şekil 4.113. la29 data seti için tek yönlü varyans analizi sonuçları.

Şekil 4.113 incelendiğinde "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi kabul edilir ve la29 data seti için "%95 güven aralığında farklı iterasyonlarda C_{max} değerlerinin ortalaması için gruplar arasında istatistiksel olarak anlamlı bir fark yoktur" sonucuna ulaşılır.

ANOVA

Cmax

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	26,733	2	13,367	,395	,675
Within Groups	3961,133	117	33,856		
Total	3987,867	119			

Şekil 4.114. yn02 data seti için tek yönlü varyans analizi sonuçları.

Şekil 4.114 incelendiğinde "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi kabul edilir ve yn02 data seti için "%95 güven aralığında farklı iterasyonlarda C_{max} değerlerinin ortalaması için gruplar arasında istatistiksel olarak anlamlı bir fark yoktur" sonucuna ulaşılır.

Kurulan hipoteze göre 7 farklı data seti için tek yönlü varyans analizi uygulaması sonucu genel olarak %95 güven aralığında farklı iterasyonlarda C_{max} değerlerinin ortalaması için gruplar arasında istatistiksel olarak anlamlı bir fark olmadığı sonucuna varılmıştır. Sadece abz07 data seti için istatistiksel olarak anlamlı bir fark olduğu tespit edilmiştir. Gruplararası farkın olduğu durumda, farklılığın hangi gruptan kaynaklı olduğunu tespit eden istatistik post-hoc olarak bilinmektedir [180, 181]. abz07 data

seti için hangi iterasyonlarda anlamlı bir fark olduğunu anlayabilmek için SPSS'te "Tukey" testini uygularsak Şekil 4.115 ve 4.116'daki sonuçları elde ederiz.

Multiple Comparisons

Tukey HSD

(I) abz07	(J) abz07	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
					Lower Bound	Upper Bound
20 iterasyon	40 iterasyon	1,625	1,343	,450	-1,56	4,81
	60 iterasyon	3,183*	1,266	,035	,18	6,19
40 iterasyon	20 iterasyon	-1,625	1,343	,450	-4,81	1,56
	60 iterasyon	1,558	1,001	,269	-,82	3,93
60 iterasyon	20 iterasyon	-3,183*	1,266	,035	-6,19	-,18
	40 iterasyon	-1,558	1,001	,269	-3,93	,82

Şekil 4.115. abz07 data seti için tukey testi sonuçları.

Şekil 4.109 abz07 data seti için farklı iterasyonların 2'li olarak karşılaştırmasını göstermektedir. "*" olan satırlardaki iterasyonlar birbirinden anlamlı olarak farklıdır. Bu bağlamda abz07 data setindeki gruplararası farkın 20 ve 60 iterasyon sonuçları arasında olduğu görülmektedir. Bu farkı belirledikten sonra Şekil 4.110'daki "Descriptives" tablosunu incelersek; "Mean" sütunundan 60 iterasyon için C_{max} ortalamasının daha düşük sonuç verdiğini, yine "Std. Deviation" sütunundan standart sapmasının da daha az olduğunu görebiliriz.

Descriptives

Cmax

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean		Minimum	Maximum
					Lower Bound	Upper Bound		
20 iterasyon	20	668,20	5,709	1,277	665,53	670,87	659	681
40 iterasyon	40	666,58	5,477	,866	664,82	668,33	659	678
60 iterasyon	60	665,02	4,168	,538	663,94	666,09	659	673
Total	120	666,07	5,005	,457	665,16	666,97	659	681

Şekil 4.116. abz07 data seti için tukey testi sonuçları.

Sonuç olarak değişik data setlerine önerilen metot uygulandığında elde edilen sonuçlar farklı iterasyonlar için tek yönlü varyans analizi uygulanarak kıyaslandığında genel olarak C_{max} değerlerinin ortalaması için gruplar arasında anlamlı bir fark olmadığı sonucuna varılmış, olabilecek anlamlı farklılıkların ise 60 ile 20 iterasyon sonuçları arasında olduğu görülmüştür.

4.3.3.2. C_{max} ve standart sapma kriterlerine göre kıyaslanması

Farklı iterasyonlardan elde edilen verilere tek yönlü varyans analizi uygulanmasından sonra yine aynı data setleri C_{max} ve standart sapmalarına göre kıyaslanmıştır. Bu kısımda amaç, data setlerinin sonuçları arasındaki değişkenliği gözlemlemektir. Önerilen metot 7 farklı data setine farklı iterasyonlarda uygulanarak elde edilen sonuçlar Tablo 4.1'de kıyaslanmıştır. Kıyaslama yapılırken belirtildiği üzere işlerin toplam bitirme zamanları olan C_{max} değerlerinin ve standart sapmalarının ortalamalarına bakılmıştır. Bu şekilde farklı iterasyonlar için ne derece sapmaların olup olmadığı gözlemlenmiştir.

Tablo 4.1'de "Instance" sütunu data setinin ismini, "Best Known" şimdiye kadar literatürde bulunan en iyi sonucu, "Best" ABC-EA bütünleşik yaklaşımı ile elde edilen en iyi değeri, "Average" ilgili data seti için farklı iterasyonlar için ortalama C_{max}'ı, "Std.Sapma" ilgili data setinin farklı iterasyonlar için standart sapmasını, "Time limit" ise ne kadar süre ile o data setinin çalıştırıldığı (sn. olarak) göstermektedir. Herbir data seti aynı koşullar altında farklı iterasyonlarda çalıştırılmıştır.

Sonuçlara bakıldığında 20 iterasyon sonucunda ortalama işlerin tamamlanma süresi C_{max}'ın ortalama olarak 7 data seti için 1013,51 sn., 40 iterasyon için bu sürenin 1012,59 sn., 60 iterasyon için ise 1012,14 sn. olduğu görülmüştür. Yine ortalama işlerin tamamlanma süresi C_{max}'ın farklı iterasyonlar için standart sapmalarına bakarsak, 20 iterasyon sonucundaki ortalama olarak standart sapma 4,648 sn., 40 iterasyon sonucundaki ortalama olarak standart sapma 4,164 sn., 60 iterasyon sonucundaki ortalama olarak standart sapma 3,522 sn., olarak bulunmuştur. Bu sonuçlara göre iterasyon sayısı artıkça standart sapmanın daha da düştüğü, işlerin

ortalama tamamlanma süresinin ise çok az da olsa azaldığı gözlemlenmiştir. Sonuç olarak, iterasyon sayısı arttıkça C_{max} değerlerinin ortalaması yaklaşık olarak binde 1 azalmaktadır. Bu sonuç, C_{max} değerlerinin ortalaması için "gruplar arasında istatistiksel olarak anlamlı bir fark yoktur" hipotezini desteklemektedir.

4.4. Sonuçları Farklı Optimizasyon Teknikleri ile Kıyaslama Kriterleri

4.4.1. Ortalama bağıl hata yüzdesi

Literatürde çeşitli sezgisel optimizasyon teknikleri ile çözülen atölye tipi çizelgeleme problemleri incelendiğinde yapılan çalışmaların kıyaslanmasında genellikle Ortalama Bağıl Hata Yüzdesi (Average Relative Percent Error-ARPE) [182, 183, 184] kullanılmaktadır. Ortalama bağıl hata yüzdesinin hesaplanması Denklem 4.1'de gösterilmiştir.

$$\text{ARPE} = \frac{\sum_{i=1}^{R} \frac{(Hi - Ui) \times 100}{Ui}}{R} \tag{4.1}$$

Bu eşitlikte Hi sembolü, önerilen metodoloji ile ilgili data seti için bulunan en iyi C_{max} değerini, Ui ise literatürde ilgili datanın çeşitli optimizasyon teknikleri ile çözülmesi sonucu bulunmuş en iyi veya optimum değeri simgelemektedir. R ise çözümü yapılmış toplam data seti sayısıdır. Öncelikle her bir ilgili data seti için RPE değeri hesaplanır. Bulunan değerler toplamının ortalaması ise ARPE değerini bulmamızı sağlar. Bulunan ARPE değeri 0'a ne kadar yakınsa veya eşitse ilgili data seti için elde edilen netice de o kadar etkilidir. Literatürde atölye tipi çizelgeleme problemleriyle ilgili çeşitli data setleri ARPE değerleri hesaplanarak ortalaması alınmakta ve diğer bulunan değerlerle kıyaslanma yapılmaktadır.

4.4.2. Ortalama bağıl sapma yüzdesi

Literatürdeki diğer bir kıyaslama kriteri de Ortalama Bağıl Sapma Yüzdesidir (Average Relative Percent Deviation-ARPD) [185, 186, 187]. Ortalama bağıl sapma yüzdesinin hesaplanması Denklem 4.2'de gösterilmiştir.

$$\text{ARPD} = \frac{\sum_{i=1}^{R} \frac{(Oi-Ui)\times 100}{Ui}}{R} \tag{4.2}$$

Bu eşitlikte Oi sembolü, önerilen metodoloji ile ilgili data seti için bulunan C_{max} değerlerinin ortalamasını, Ui ise literatürde ilgili datanın çeşitli optimizasyon teknikleri ile çözülmesi sonucu bulunmuş en iyi veya optimum değeri simgelemektedir. R ise çözümü yapılmış toplam data seti sayısıdır. Öncelikle ilgili data setlerinin herbiri için RPD değeri hesaplanır. Bulunan değerler toplamının ortalaması ise ARPD değerini bulmamızı sağlar. Bulunan ARPD değeri ne kadar küçükse, elde edilen sonuçlar da o kadar kararlı ve etkilidir.

Bu çalışmada literatürde karınca kolonisi optimizasyonu [188], kuş sürüsü optimizasyon tekniği ve differential evolution algoritması [189] kullanılarak çözülmüş çeşitli atölye tipi çizelgeleme problem data setlerinin sonuçları ile ABC-EA bütünleşik yaklaşımı ile elde edilen sonuçlar ARPE ve ARPD faktörleri kullanılarak kıyaslanmıştır. Literatürdeki belirtilen çalışmalarda 2.6 GHz işlemcisi 256 MB belleği olan bilgisayar kullanılmıştır. Yine aynı şekilde özelliklere sahip bir PC kullanılarak ABC-EA bütünleşik yaklaşımı, ilgili data setlerine uygulanmıştır.

Bulunan bu sonuçlar ise tezin son kısmında literatürdeki farklı optimizasyon algoritmalarından olan karınca kolonisi optimizasyonu(ACO), kuş sürüsü optimizasyon tekniği (PSO) ve differential evolution (DE) algoritması kullanılarak elde edilen çözümler ile kıyaslanmış ve sonuçlar ortaya konulmuştur. Her bir data seti için (iş*makine) sayısının büyüklüğüne göre kıyaslaması yapılan literatürdeki optimizasyon tekniklerinde araştırmacıların kullandığı bir time limitlere göre yazılım çalıştırılmıştır. Bu belirlenen süre kadar sistem çalıştırılarak sonuçlar üretilmiş ve

tablolarda kıyaslanması sunulmuştur. Literatürdeki çalışmalar incelendiğinde her bir data seti için sistem 20 kez çalıştırılmış olduğu görülmektedir. Bundan dolayı ABC-EA bütünleşik yaklaşımı ile de her bir data seti için kurulan program 20 kez çalıştırılmış ve sonuçların ortalamaları alınmıştır.

Tablo 4.1. ABC-EA bütünleşik yaklaşımının farklı iterasyonlardaki elde edilen değerlerinin kıyaslanması.

20 run için ABC-EA bütünleşik yaklaşımının sonuçları

Instance	Best Known	Best	Average	Std.Sapma	Time limit
ft20	1165	1165	1167,95	3,347	76,15
orb02	888	888	889,45	1,877	76,15
la21	1046	1046	1051,55	4,273	380,77
la29	1152	1158	1165,2	6,066	761,54
la40	1222	1222	1229,95	5,031	761,54
abz07	656	659	670,1	5,75	761,54
yn02	909	911	920,35	6,192	1523,08
ORTALAMA			**1013,51**	**4,648**	

40 run için ABC-EA bütünleşik yaklaşımının sonuçları

Instance	Best Known	Best	Average	Std.Sapma	Time limit
ft20	1165	1165	1167,75	2,273	76,15
orb02	888	888	889,35	1,511	76,15
la21	1046	1046	1050,97	3,833	380,77
la29	1152	1158	1164,77	5,176	761,54
la40	1222	1222	1229,37	4,834	761,54
abz07	656	659	666,575	5,476	761,54
yn02	909	911	919,4	6,046	1523,08
ORTALAMA			**1012,59**	**4,164**	

60 run için ABC-EA bütünleşik yaklaşımının sonuçları

Instance	Best Known	Best	Average	Std.Sapma	Time limit
ft20	1165	1165	1167,65	2,015	76,15
orb02	888	888	889,33	1,23	76,15
la21	1046	1046	1050,83	3,729	380,77
la29	1152	1158	1164,25	3,731	761,54
la40	1222	1222	1228,9	4,25	761,54
abz07	656	659	665,01	4,168	761,54
yn02	909	911	919,016	5,53	1523,08
ORTALAMA			**1012,14**	**3,522**	

BÖLÜM 5. SONUÇLAR VE ÖNERİLER

Bu tez çalışmasında, öncelikle günümüzde doğru ve etkin bir çizelgelemenin hem insanlar hem de işletmeler açısından önemi vurgulanmış, bu bağlamda çizelgeleme problemlerinin çözümünde özellikle son yıllarda araştırmacılar tarafından yoğun bir biçimde kullanılan sezgisel algoritmalar hakkında detaylı bilgiler verilmiştir. Yine son zamanlarda optimizasyon problemlerinde önemli bir yer tutan yapay arı kolonisi algoritması, evrimsel algoritmalar ile bütünleşik olarak kullanılarak bir eniyileme yöntemi önerilmiştir. Önerilen yaklaşım, atölye tipi çizelgeleme problemlerinin çözümünün eniyilenmesi için kullanılmıştır.

Yapılan bu çalışmada uygulama kısmına geçilmeden önce önerilen metodun yapısındaki evrimsel algoritmaların hibrit olarak kullanıldığında eniyileme konusunda etkili olduğunu göstermek amacıyla evrimsel yapay sinir ağları metodu ile bir eniyileme uygulaması yapılmıştır. Bu bağlamda literatürde daha önce farklı bir optimizasyon metodu ile çözülmüş olan bir çalışmaya evrimsel yapay sinir ağları metodu uygulanmış ve sonuçlar kıyaslanarak evrimsel algoritmaların hibrit kullanımındaki etkinliği gösterilmeye çalışılmıştır. Uygulama bölümünde ise öncelikle üzerinde çalışma yapılan atölye tipi çizelgeleme problemlerinden bahsedilmiş, bu konu hakkında literatürde yapılmış olan çalışmalar hakkında bilgi verildikten sonra önerilmiş olan metodun yapısı detaylı bir şekilde anlatılmıştır. Önerilen metot, atölye tipi çizelgeleme problemleri ile ilgili data setlerine uygulanmış ve sonuçlar tablo ile şekillerde gösterilmiştir. Daha sonra önerilen yaklaşım farklı iterasyonlarda çalıştırılmış ve elde edilen sonuçlara tek yönlü varyans analizi uygulanarak, önerilen metodun farklı iterasyonlardaki sonuçları için gruplar arasında anlamlı bir farkın olup olmadığı araştırılmıştır.

Tezin bu son kısmında ise önerilen yaklaşım kullanılarak elde edilen çözümlere, parametrik ve parametrik olmayan testler uygulanarak, literatürdeki farklı optimizasyon teknikleri ile kıyaslanmıştır. Kıyaslama yapılan eniyileme teknikleri karınca kolonisi optimizasyonu, kuş sürüsü optimizasyon tekniği ve diferansiyel gelişim algoritmalarıdır. Önerilen metot ile elde edilen RPE, RPD değerleri ile literatürdeki ACO [188], PSO ve DE [189] algoritmaları kullanılarak elde edilen RPE, RPD değerleri sırasıyla Tablo (5.1-5.2-5.3)'te verilmiştir. Kıyaslamalarda kullanılan ilgili data setleri seçilirken, özellikle literatürde RPE değerleri 0'dan fazla çıkan (RPE'nin sıfır olması mevcut çözümün literatürdeki en iyi çözümü yakaladığını gösterir) data setleri alınarak eniyileme yapılmıştır. ACO ile yapılan kıyaslama çalışmasında la02, la03, la04, la17, la19, la20, la21, la24, la26, la29, la30, la31, la32, la33, la35, la39, la40 data setleri (2 grup olduğu için bağımsız-t testi uygulanmıştır); PSO ve DE ile yapılan kıyaslama çalışmasında ise ft20, orb01, orb02, orb03, orb05, orb06, la21, la24, la29, la40, abz07, abz08, abz09, yn01, yn02, yn03 ve yn04 data setleri (2'den fazla grup olduğu için varyans analizi uygulanmıştır) kullanılmıştır.

İlk olarak önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPE değerleri ile literatürdeki ilgili eniyileme tekniklerinin sonuçları arasında istatistiksel olarak anlamlı bir fark olup olmadığını araştırmak amacıyla parametrik testler uygulanmıştır. Bunun için de elde edilen RPE değerlerinin kıyaslanmasında parametrik testleri kullanabilmemiz için öncelikle verilerin % 95 güven aralığında normal dağılım göstermesi ve varyanslarının homojen dağılması gerekmektedir [180,181]. Çalışmanın bu kısmında SPSS 17.0 paket program kullanılarak kıyaslanan RPE değerleri için normal dağılım ve varyansların homojenlik testleri yapılmıştır. Veri setlerinin normal dağılımı ve varyanslarının homojen dağılıp dağılmadığı ile ilgili olarak sırasıyla aşağıdaki hipotezler kurulmuş ve bu hipotezlere göre sonuçlar alınmıştır.

- H_0: %95 güven aralığında veriler normal dağılımlıdır.
- H_1: %95 güven aralığında veriler normal dağılımlı değildir.

- H_0: %95 güven aralığında varyanslar homojen dağılmıştır.
- H_1: %95 güven aralığında varyanslar homojen dağılmamıştır.

Hipotezler oluşturulduktan sonra kıyaslanacak RPE değerlerinin normal dağılıma uygunluğu ve varyanslarının homojen olup olmadığı SPSS paket programında test edilmiş ve sonuçlar aşağıda sunulmuştur. Daha önceki bölümde de ifade edildiği üzere normal dağılıma uygunluğu için aşağıdaki şekilde "Descriptives" tablosundaki "Skewness" ve "Kurtosis" satırlarındaki değerlere bakılır. Veri setinin normal dağılıma uygun olması için belirtilen değerlerin (-1,5) ile (+1,5) arasında olması gerekmektedir. Varyansların homojenliği için ise "Test of Homogeneity of Variances" tablosundaki "Sig." sütunundaki değere bakılır. Veri setinin homojen olması için bu değerin 0,05 değerinden büyük olması gerekmektedir [180,181]. İlk olarak literatürdeki karınca kolonisi optimizasyon tekniği alınmış ve önerilen metot ile ile elde edilen RPE değerleri için bu testler yapılmıştır.

Descriptives

			Statistic	Std. Error
RPE	Mean		2,3815	,59873
	95% Confidence Interval for Mean	Lower Bound	1,1633	
		Upper Bound	3,5996	
	5% Trimmed Mean		1,9003	
	Median		,6500	
	Variance		12,188	
	Std. Deviation		3,49116	
	Minimum		,00	
	Maximum		15,73	
	Range		15,73	
	Interquartile Range		3,96	
	Skewness		2,180	,403
	Kurtosis		5,704	,788

Şekil 5.1. Karınca kolonisi algoritması ve önerilen metot ile elde edilen RPE değerleri için normallik testi sonuçları.

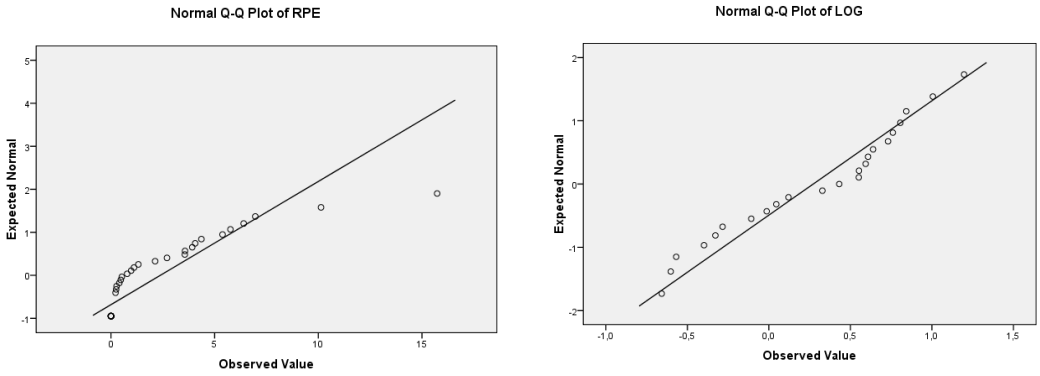
Şekil 5.1 incelendiğinde "Skewness" ve "Kurtosis" satırlarındaki değerler (-1,5) ile (+1,5) arasında olmadığı için H_0 hipotezi reddedilir "%95 güven aralığında veriler normal dağılımlı değildir" sonucuna ulaşılır. Daha önce de vurgulandığı üzere parametrik testleri kullanabilmemiz için verilerin normal dağılması gerekmektedir. Bunun için veri setine SPSS'te logaritmik dönüşümler uygulanarak veri seti normal dağılıma uygun hale getirilmeye çalışılır. Bu dönüşüm yapıldıktan sonra elde edilen değerler Şekil 5.2'de gösterilmiştir. Şekil 5.2 incelendiğinde "Skewness" ve "Kurtosis" satırlarındaki değerler (-1,5) ile (+1,5) arasında olduğu için H_0 hipotezi kabul edilir "%95 güven aralığında veriler normal dağılımlıdır" sonucuna ulaşılır.

Descriptives

			Statistic	Std. Error
LOG	Mean		,27191927	,115437338
	95% Confidence Interval for Mean	Lower Bound	,03251688	
		Upper Bound	,51132166	
	5% Trimmed Mean		,27316229	
	Median		,43136376	
	Variance		,306	
	Std. Deviation		,553618025	
	Minimum		-,657577	
	Maximum		1,196729	
	Range		1,854306	
	Interquartile Range		1,014779	
	Skewness		-,276	,481
	Kurtosis		-1,135	,935

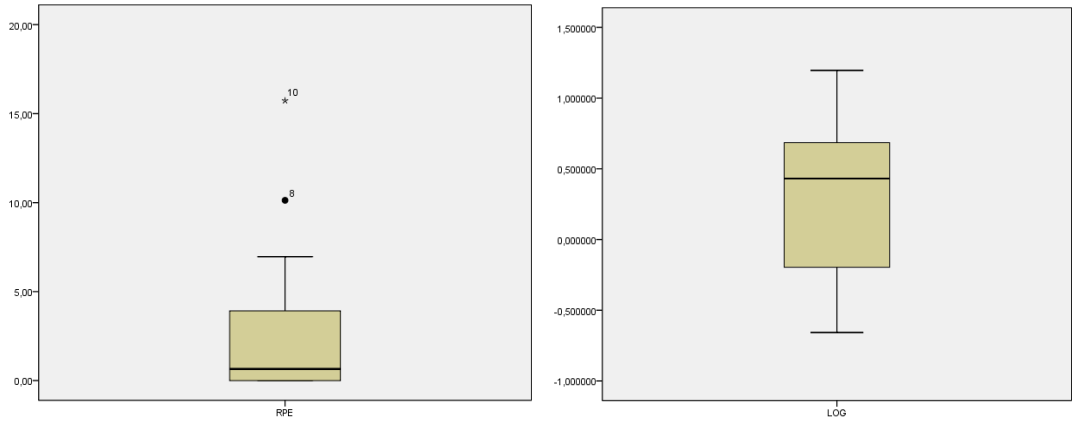
Şekil 5.2. Karınca kolonisi algoritması ve önerilen metot ile elde edilen RPE değerleri için logaritmik dönüşüm sonrası normallik testi sonuçları.

Veri setlerinin normal dağılımını grafiksel olarak görebilmek amacıyla logaritmik dönüşüm yapılmadan önce ve yapıldıktan sonraki Q-Q grafikleri (Quantile-Quantile plot) ve kutu diyagramları (Box-and-Whisker plot) sırasıyla Şekil 5.3(a)-(b) ve Şekil 5.4(a)-(b)'de gösterilmiştir.



(a) (b)

Şekil 5.3. Karınca kolonisi algoritması ve önerilen metot ile elde edilen RPE değerleri için logaritmik dönüşüm öncesi (a) ve sonrası (b) Q-Q grafikleri.



(a) (b)

Şekil 5.4. Karınca kolonisi algoritması ve önerilen metot ile elde edilen RPE değerleri için logaritmik dönüşüm öncesi (a) ve sonrası (b) kutu diyagramları.

Bu doğrulama yapıldıktan sonra logaritmik dönüşüm sonrası veri setlerinin varyanslarının % 95 güven aralığında homojen dağılıp dağılmadığını kontrol etmemiz gerekmektedir. Bunun için de yine SPSS paket programı kullanılarak varyansların homojenliği incelenmiştir.

Test of Homogeneity of Variances

LOG

Levene Statistic	df1	df2	Sig.
2,608	1	21	,121

Şekil 5.5. Karınca kolonisi algoritması ve önerilen metot ile elde edilen RPE değerleri için logaritmik dönüşüm sonrası varyansların homojenlik testi sonuçları.

Şekil 5.5 incelendiğinde "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi reddedilemez ve "%95 güven aralığında varyanslar homojendir" sonucuna ulaşılır.

Diğer kıyaslanacak metot olan kuş sürüsü algoritması ve önerilen metot ile ile elde edilen RPE değerleri için yine aynı testler yapılmıştır. Şekil 5.6 incelendiğinde "Skewness" ve "Kurtosis" satırlarındaki değerler (-1,5) ile (+1,5) arasında olmadığı için H_0 hipotezi reddedilir "%95 güven aralığında veriler normal dağılımlı değildir" sonucuna ulaşılır.

Daha önce de vurgulandığı üzere parametrik testleri kullanabilmemiz için verilerin normal dağılması gerekmektedir. Bunun için veri setine logaritmik dönüşümler uygulanarak veri seti yine normal dağılıma uygun hale getirilmeye çalışılır. Bu dönüşüm yapıldıktan sonra elde edilen değerler Şekil 5.7'de gösterilmiştir.

Descriptives

			Statistic	Std. Error
RPE	Mean		,3729	,08016
	95% Confidence Interval for Mean	Lower Bound	,2099	
		Upper Bound	,5360	
	5% Trimmed Mean		,3394	
	Median		,1350	
	Variance		,218	
	Std. Deviation		,46741	
	Minimum		,00	
	Maximum		1,35	
	Range		1,35	
	Interquartile Range		,53	
	Skewness		2,114	,403
	Kurtosis		-,075	,788

Şekil 5.6. Kuş sürüsü algoritması ve önerilen metot ile elde edilen RPE değerleri için normallik testi sonuçları.

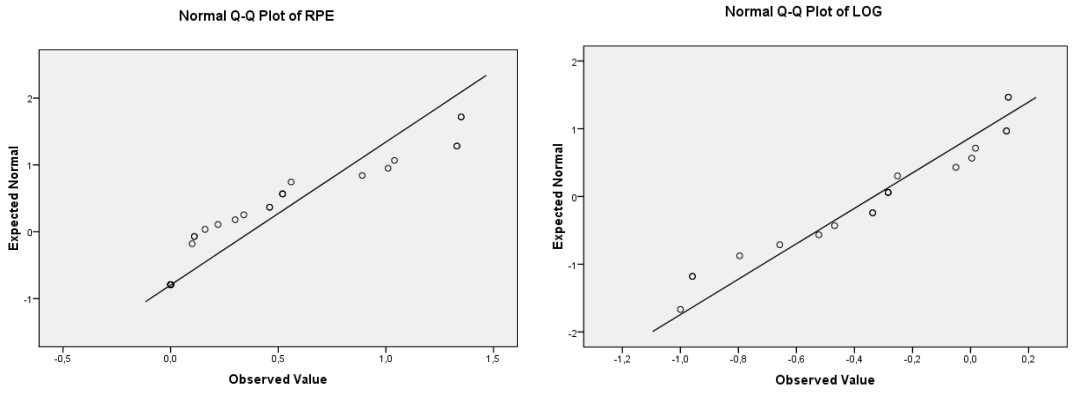
Şekil 5.7 incelendiğinde "Skewness" ve "Kurtosis" satırlarındaki değerler (-1,5) ile (+1,5) arasında olduğu için H_0 hipotezi kabul edilir "%95 güven aralığında veriler normal dağılımlıdır" sonucuna ulaşılır.

Veri setlerinin normal dağılımını grafiksel olarak görebilmek amacıyla logaritmik dönüşüm yapılmadan önce ve yapıldıktan sonraki Q-Q grafikleri (Quantile-Quantile plot) ve kutu diyagramları (Box-and-Whisker plot) sırasıyla Şekil 5.8 (a)-(b) ve Şekil 5.9 (a)-(b)'de gösterilmiştir.

Descriptives

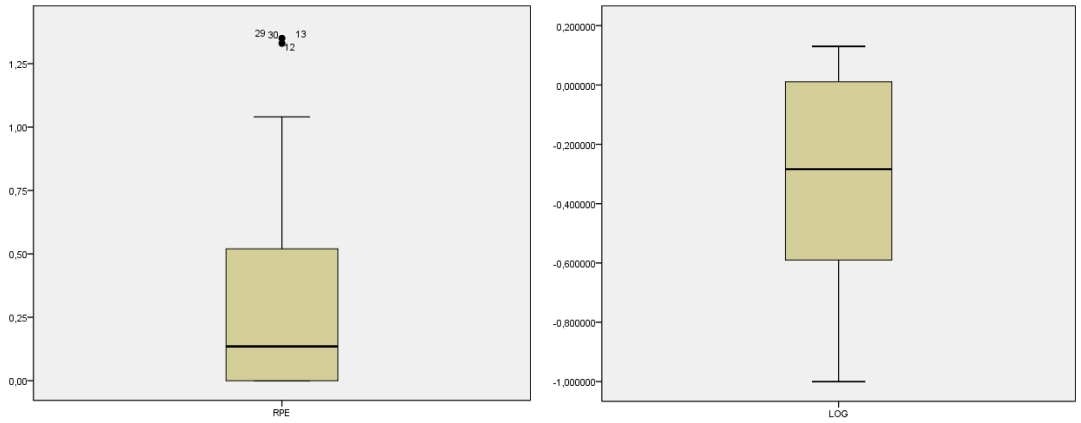
			Statistic	Std. Error
LOG	Mean		-,33306213	,085507064
	95% Confidence Interval for Mean	Lower Bound	-,51203047	
		Upper Bound	-,15409378	
	5% Trimmed Mean		-,32175424	
	Median		-,28399666	
	Variance		,146	
	Std. Deviation		,382399215	
	Minimum		-1,000000	
	Maximum		,130334	
	Range		1,130334	
	Interquartile Range		,637758	
	Skewness		-,454	,512
	Kurtosis		-,916	,992

Şekil 5.7. Kuş sürüsü algoritması ve önerilen metot ile elde edilen RPE değerleri için logaritmik dönüşüm sonrası normallik testi sonuçları.



(a) (b)

Şekil 5.8. Kuş sürüsü algoritması ve önerilen metot ile elde edilen RPE değerleri için logaritmik dönüşüm öncesi (a) ve sonrası (b) Q-Q grafikleri.



(a) (b)

Şekil 5.9. Kuş sürüsü algoritması ve önerilen metot ile elde edilen RPE değerleri için logaritmik dönüşüm öncesi (a) ve sonrası (b) kutu diyagramları.

Bu doğrulama yapıldıktan sonra logaritmik dönüşüm sonrası veri setlerinin varyanslarının % 95 güven aralığında homojen dağılıp dağılmadığını kontrol etmemiz gerekmektedir. Bunun için de yine SPSS paket programı kullanılarak varyansların homojenliği incelenmiştir.

Test of Homogeneity of Variances

LOG

Levene Statistic	df1	df2	Sig.
3,203	1	18	,090

Şekil 5.10. Kuş sürüsü algoritması ve önerilen metot ile elde edilen RPE değerleri için logaritmik dönüşüm sonrası varyansların homojenlik testi sonuçları.

Şekil 5.10 incelendiğinde "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi reddedilemez ve "%95 güven aralığında varyanslar homojendir" sonucuna ulaşılır. Son olarak diferansiyel gelişim algoritması ve önerilen metot ile ile elde edilen RPE değerlerine aynı testler uygulanmıştır.

Descriptives

			Statistic	Std. Error
RPE	Mean		,4206	,08317
	95% Confidence Interval for Mean	Lower Bound	,2514	
		Upper Bound	,5898	
	5% Trimmed Mean		,3868	
	Median		,2250	
	Variance		,235	
	Std. Deviation		,48496	
	Minimum		,00	
	Maximum		1,52	
	Range		1,52	
	Interquartile Range		,73	
	Skewness		1,618	,403
	Kurtosis		-,225	,788

Şekil 5.11. Diferansiyel gelişim algoritması ve önerilen metot ile elde edilen RPE değerleri için normallik testi sonuçları.

Şekil 5.11 incelendiğinde "Skewness" ve "Kurtosis" satırlarındaki değerler (-1,5) ile (+1,5) arasında olmadığı için H_0 hipotezi reddedilir "%95 güven aralığında veriler normal dağılımlı değildir" sonucuna ulaşılır. Bundan dolayı veri setine logaritmik dönüşümler uygulanarak veri seti normal dağılıma uygun hale getirilmeye çalışılır. Bu dönüşüm yapıldıktan sonra elde edilen değerler Şekil 5.12'de gösterilmiştir.

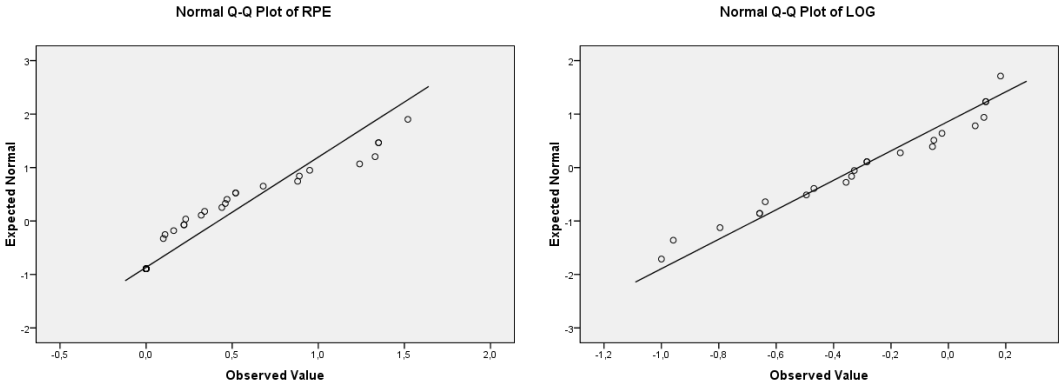
Şekil 5.12 incelendiğinde "Skewness" ve "Kurtosis" satırlarındaki değerler (-1,5) ile (+1,5) arasında olduğu için H_0 hipotezi kabul edilir "%95 güven aralığında veriler normal dağılımlıdır" sonucuna ulaşılır.

Descriptives

			Statistic	Std. Error
LOG	Mean		-,31350366	,077315285
	95% Confidence Interval for Mean	Lower Bound	-,47428960	
		Upper Bound	-,15271772	
	5% Trimmed Mean		-,30283317	
	Median		-,30594940	
	Variance		,132	
	Std. Deviation		,362640830	
	Minimum		-1,000000	
	Maximum		,181844	
	Range		1,181844	
	Interquartile Range		,649747	
	Skewness		-,350	,491
	Kurtosis		-,882	,953

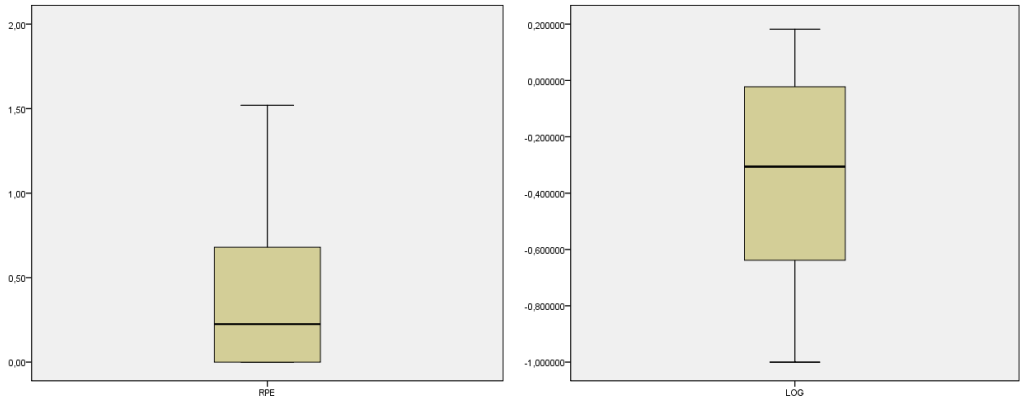
Şekil 5.12. Diferansiyel gelişim algoritması ve önerilen metot ile elde edilen RPE değerleri için logaritmik dönüşüm sonrası normallik testi sonuçları.

Veri setlerinin normal dağılımını grafiksel olarak görebilmek amacıyla logaritmik dönüşüm yapılmadan önce ve yapıldıktan sonraki Q-Q grafikleri ve kutu diyagramları sırasıyla Şekil 5.13 (a)-(b) ve Şekil 5.14 (a)-(b)'de gösterilmiştir.



(a) (b)

Şekil 5.13. Diferansiyel gelişim algoritması ve önerilen metot ile elde edilen RPE değerleri için logaritmik dönüşüm öncesi (a) ve sonrası (b) Q-Q grafikleri.



(a) (b)

Şekil 5.14. Diferansiyel gelişim algoritması ve önerilen metot ile elde edilen RPE değerleri için logaritmik dönüşüm öncesi (a) ve sonrası (b) kutu diyagramları.

Bu doğrulama da yapıldıktan sonra logaritmik dönüşüm sonrası veri setlerinin varyanslarının % 95 güven aralığında homojen dağılıp dağılmadığını kontrol etmemiz gerekmektedir. Bunun için de yine SPSS paket programı kullanılarak varyansların homojenliği incelenmiştir.

Test of Homogeneity of Variances

LOG

Levene Statistic	df1	df2	Sig.
3,203	1	18	,090

Şekil 5.15. Diferansiyel gelişim algoritması ve önerilen metot ile elde edilen RPE değerleri için logaritmik dönüşüm sonrası varyansların homojenlik testi sonuçları.

Şekil 5.15 incelendiğinde "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi reddedilemez ve "%95 güven aralığında varyanslar homojendir" sonucuna ulaşılır.

Yapılan hipotez testleri sonucunda RPE verilerinin normal dağılıma uygunluğu ve varyanslarının homojenliği sağlandıktan sonra PSO, DE ve önerilen metot ile elde edilen değerlere SPSS paket programı kullanılarak tek yönlü varyans analizi uygulaması yapılmıştır. Bunun için "farklı optimizasyon tekniklerinin ilgili veri setlerine uygulanması sonucu elde edilen RPE değerleri için uygulanan metotlar arasında istatistiksel olarak anlamlı bir fark var mıdır?" hipotezi, varyans analizi yapılarak test edilmiştir. Konu ile ilgili olarak aşağıdaki hipotez kurulmuş ve bu hipoteze göre sonuçlar alınmıştır.

- H_0: %95 güven aralığında farklı optimizasyon tekniklerinin ilgili veri setlerine uygulanması sonucu elde edilen RPE değerleri için uygulanan metotlar arasında istatistiksel olarak anlamlı bir fark yoktur.
- H_1: %95 güven aralığında farklı optimizasyon tekniklerinin ilgili veri setlerine uygulanması sonucu elde edilen RPE değerleri için uygulanan metotlar arasında istatistiksel olarak anlamlı bir fark vardır.

Hipotezler oluşturulduktan sonra ilgili RPE değerleri için gruplar arasında anlamlı bir fark olup olmadığının analiz edilmesinde yine SPSS paket programı kullanılmış ve tek yönlü varyans analizi yapılmıştır. Varyans analizi yapılırken "ANOVA" tablosundaki "Sig." sütunundaki değere bakılır. Gruplar arasında anlamlı bir fark olması için bu değerin 0,05'ten küçük olması gerekmektedir [180, 181]. Yapılan varyans analizi sonucu Şekil 5.16'da gösterilmiştir.

ANOVA

LOG

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	,052	2	,026	,174	,841
Within Groups	4,337	29	,150		
Total	4,389	31			

Şekil 5.16. İlgili optimizasyon metotlarının RPE değerleri için varyans analizi sonuçları.

Şekil 5.16 incelendiğinde "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi kabul edilir ve "%95 güven aralığında farklı optimizasyon tekniklerinin ilgili veri setlerine uygulanması sonucu elde edilen RPE değerleri için uygulanan metotlar arasında istatistiksel olarak anlamlı bir fark yoktur" sonucuna ulaşılır. Diğer optimizasyon algoritması ACO ve önerilen metot ile elde edilen değerlere ise bağımsız-t testi uygulanmıştır. Daha sonra varyans analizi yapılarak istatistiksel olarak anlamlı farkın bulunmadığı 3 metot için 2'li olarak bağımsız-t testi uygulanmış ve daha detaylı olarak hipotez sonuçlarına bakılmıştır. Bunun için "önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPE değerleri ile literatürdeki diğer optimizasyon algoritmaları ile elde edilmiş RPE değerleri arasında istatistiksel olarak anlamlı bir

fark var mıdır?" hipotezi herbir metot için sırasıyla test edilmiştir. Farklı optimizasyon tekniklerinin herbiri için konu ile ilgili aşağıdaki hipotezler kurulmuş ve bu hipotezlere göre sonuçlar alınmıştır. İlk olarak önerilen metot ile karınca kolonisi optimizasyon tekniği sonuçları değerlendirilmiştir. Bu bağlamda aşağıdaki hipotez kurulmuştur.

- H_0: %95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPE değerleri ile literatürdeki karınca kolonisi optimizasyon tekniği sonuçları arasında istatistiksel olarak anlamlı bir fark yoktur.
- H_1: %95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPE değerleri ile literatürdeki karınca kolonisi optimizasyon tekniği sonuçları arasında istatistiksel olarak anlamlı bir fark vardır.

Hipotezler oluşturulduktan sonra iki farklı metot ile elde edilen RPE değerleri SPSS paket programındaki bağımsız t-testi uygulamasına tabi tutulmuştur. Bağımsız t-testi yapılırken "Independent Samples Test" tablosundaki "Sig. (2-tailed)" sütunundaki değere bakılır. İki metodun RPE değerleri arasında anlamlı bir fark olması için bu değerin 0,05'ten küçük olması gerekmektedir [180, 181].

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
LOG	Eq.var. assumed	2,608	,121	6,501	21	,000	1,007	,155	,6854	1,330
	Eq. Var. not assumed			9,257	19,781	,000	1,007	,108	,7805	1,235

Şekil 5.17. Karınca kolonisi algoritması ve önerilen metot ile elde edilen RPE değerleri için bağımsız t-testi sonuçları.

Şekil 5.17 incelendiğinde "Sig. (2-tailed)" değeri 0,05'ten küçük olduğu için H_0 hipotezi reddedilir ve "%95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPE değerleri ile literatürdeki karınca kolonisi optimizasyon tekniği sonuçları arasında istatistiksel olarak anlamlı bir fark vardır" sonucuna ulaşılır.

Daha sonra diğer kuş sürüsü algoritması ile önerilen metodun sonuçları incelenmiştir. Bunun için aşağıdaki hipotez kurulmuştur.

- H_0: %95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPE değerleri ile literatürdeki kuş sürüsü optimizasyon tekniği sonuçları arasında istatistiksel olarak anlamlı bir fark yoktur.
- H_1: %95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPE değerleri ile literatürdeki kuş sürüsü optimizasyon tekniği sonuçları arasında istatistiksel olarak anlamlı bir fark vardır.

Hipotezler oluşturulduktan sonra iki farklı metot ile elde edilen RPE değerleri SPSS paket programındaki bağımsız t-testi uygulamasına tabi tutulmuştur.

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
LOG	Eq. var. assumed	3,203	,090	-,999	18	,331	-,1743	,174	-,541	,1924
	Eq. var. not assumed			-1,091	17,995	,290	-,1743	,159	-,510	,1614

Şekil 5.18. Kuş sürüsü algoritması ve önerilen metot ile elde edilen RPE değerleri için bağımsız t-testi sonuçları.

Şekil 5.18 incelendiğinde "Sig. (2-tailed)" değeri 0,05'ten büyük olduğu için H_0 hipotezi kabul edilir ve "%95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPE değerleri ile literatürdeki kuş sürüsü optimizasyon tekniği sonuçları arasında istatistiksel olarak anlamlı bir fark yoktur" sonucuna ulaşılır.

Son olarak diferansiyel gelişim algoritması ile önerilen metot sonuçları incelenmiştir. Bunun için aşağıdaki hipotez kurulmuştur.

- H_0: %95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPE değerleri ile literatürdeki diferansiyel gelişim algoritması sonuçları arasında istatistiksel olarak anlamlı bir fark yoktur.
- H_1: %95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPE değerleri ile literatürdeki diferansiyel gelişim algoritması sonuçları arasında istatistiksel olarak anlamlı bir fark vardır.

Hipotezler oluşturulduktan sonra iki farklı metot ile elde edilen RPE değerleri SPSS paket programındaki bağımsız t-testi uygulamasına tabi tutulmuştur.

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
LOG	Equal variances assumed	2,078	,165	-,825	20	,419	-,133	,161	-,471	,204
	Equal variances not assumed			-,913	19,091	,372	-,133	,146	-,439	,172

Şekil 5.19. Diferansiyel gelişim algoritması ve önerilen metot ile elde edilen RPE değerleri için bağımsız t-testi sonuçları.

Şekil 5.19 incelendiğinde "Sig. (2-tailed)" değeri 0,05'ten büyük olduğu için H_0 hipotezi kabul edilir ve "%95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPE değerleri ile literatürdeki diferansiyel gelişim algoritması sonuçları arasında istatistiksel olarak anlamlı bir fark yoktur" sonucuna ulaşılır.

Sonuç olarak, önerilen metot ile elde edilen RPE değerleri ile literatürdeki 3 farklı eniyileme teknikleri ile elde edilen RPE değerlerinin bağımsız-t testi uygulanarak kıyaslanması neticesinde, önerilen metot ve karınca kolonisi algoritması ile elde edilen sonuçlar arasında istatistiksel olarak anlamlı farklılıklar olduğu görülmüştür. Önerilen metot ile diğer eniyileme teknikleri olan kuş sürüsü algoritması ve diferansiyel gelişim algoritması sonuçları arasında ise istatistiksel olarak anlamlı farklılıklar olmadığı görülmüştür.

Bağımsız-t testi kullanılarak yukarıda yapılan kıyaslama çalışması, önerilen metot ve diğer eniyileme algoritmaları ile elde edilen sonuçlar arasında istatistiksel olarak anlamlı farklar olup olmadığını farklı bir uygulama ile test etmek amacıyla SPSS paket programda parametrik olmayan testlerden olan Mann-Whitney U testi kullanılarak incelenmiştir. Bilindiği gibi uygulamanın başında ilgili RPE değerlerinin normal olarak dağılmadığı tespit edildiğinden, parametrik olmayan testler burada kullanılabilmektedir. İlk olarak önerilen metot ile karınca kolonisi optimizasyon tekniği sonuçları değerlendirilmiştir.

Test Statistics

	RPE
Mann-Whitney U	,000
Wilcoxon W	153,000
Z	-5,063
Asymp. Sig. (2-tailed)	,000
Exact Sig. [2*(1-tailed Sig.)]	,000

Şekil 5.20. Karınca kolonisi algoritması ve önerilen metot ile elde edilen RPE değerleri için Mann-Whitney U testi sonuçları.

Mann-Whitney U testi yapılırken "Test Statistics" tablosundaki "Asymp. Sig. (2-tailed)" satırındaki değere bakılır. İki metodun RPE değerleri arasında anlamlı bir fark olması için bu değerin 0,05'ten küçük olması gerekmektedir [180, 181]. Şekil 5.20 incelendiğinde "Asymp. Sig. (2-tailed)" değeri 0,05'ten küçük olduğu için "önerilen metot sonucu elde edilen RPE değerleri ile literatürdeki karınca kolonisi optimizasyon tekniği sonuçları arasında istatistiksel olarak anlamlı bir fark vardır" sonucuna ulaşılır. Diğer eniyileme tekniği olan kuş sürüsü algoritması ile önerilen metot sonuçları Şekil 5.21'de gösterilmiştir.

Test Statistics

	RPE
Mann-Whitney U	120,000
Wilcoxon W	273,000
Z	-,875
Asymp. Sig. (2-tailed)	,381
Exact Sig. [2*(1-tailed Sig.)]	,413

Şekil 5.21. Kuş sürüsü algoritması ve önerilen metot ile elde edilen RPE değerleri için Mann-Whitney U testi sonuçları.

Şekil 5.21 incelendiğinde "Asymp. Sig. (2-tailed)" değeri 0,05'ten büyük olduğu için "önerilen metot sonucu elde edilen RPE değerleri ile literatürdeki kuş sürüsü algoritması sonuçları arasında istatistiksel olarak anlamlı bir fark yoktur" sonucuna ulaşılır. Son olarak diferansiyel gelişim algoritması ile önerilen metot sonuçları incelenmiş ve Şekil 5.22'de gösterilmiştir.

Test Statistics

	RPE
Mann-Whitney U	104,500
Wilcoxon W	257,500
Z	-1,409
Asymp. Sig. (2-tailed)	,159
Exact Sig. [2*(1-tailed Sig.)]	,170

Şekil 5.22. Diferansiyel gelişim algoritması ve önerilen metot ile elde edilen RPE değerleri için Mann-Whitney U testi sonuçları.

Şekil 5.22 incelendiğinde “Asymp. Sig. (2-tailed)” değeri 0,05’ten büyük olduğu için “önerilen metot sonucu elde edilen RPE değerleri ile literatürdeki diferansiyel gelişim algoritması sonuçları arasında istatistiksel olarak anlamlı bir fark yoktur” sonucuna ulaşılır. Parametrik olmayan bir test olan Mann-Whitney U testi sonuçları, parametrik testler olan bağımsız-t testi ve varyans analizi ile kıyaslandığında aynı sonuçlar verdiği gözlemlenmiştir.

İkinci kıyaslama kriteri olan RPD değerleri için de, farklı optimizasyon metodolojilerinin sonuçları arasında istatistiksel olarak anlamlı bir fark olup olmadığını araştırmak amacıyla da parametrik testler uygulanmıştır. Bunun için de elde edilen RPD değerlerinin kıyaslanmasında parametrik testleri kullanabilmemiz için öncelikle verilerin % 95 güven aralığında normal dağılım göstermesi ve varyanslarının homojen dağılması gerekmektedir [180,181]. RPD değerleri için normal dağılım ve varyansların homojenlik testleri yapılmıştır. Veri setlerinin normal dağılımı ve varyanslarının homojen dağılıp dağılmadığı ile ilgili olarak sırasıyla aşağıdaki hipotezler kurulmuş ve bu hipotezlere göre sonuçlar alınmıştır.

- H_0: %95 güven aralığında veriler normal dağılımlıdır.
- H_1: %95 güven aralığında veriler normal dağılımlı değildir.

- H_0: %95 güven aralığında varyanslar homojen dağılmıştır.
- H_1: %95 güven aralığında varyanslar homojen dağılmamıştır.

Hipotezler oluşturulduktan sonra kıyaslanacak RPD değerlerinin normal dağılıma uygunluğu ve varyanslarının homojen olup olmadığı SPSS paket programında test edilmiş ve sonuçlar aşağıda sunulmuştur.

Descriptives

			Statistic	Std. Error
RPD	Mean		4,0112757	,74955742
	95% Confidence Interval for Mean	Lower Bound	2,4862896	
		Upper Bound	5,5362617	
	5% Trimmed Mean		3,5504715	
	Median		1,9042833	
	Variance		19,102	
	Std. Deviation		4,37063328	
	Minimum		,33155	
	Maximum		19,30556	
	Range		18,97400	
	Interquartile Range		5,79361	
	Skewness		1,652	,403
	Kurtosis		3,123	,788

Şekil 5.23. Karınca kolonisi algoritması ve önerilen metot ile elde edilen RPD değerleri için normallik testi sonuçları.

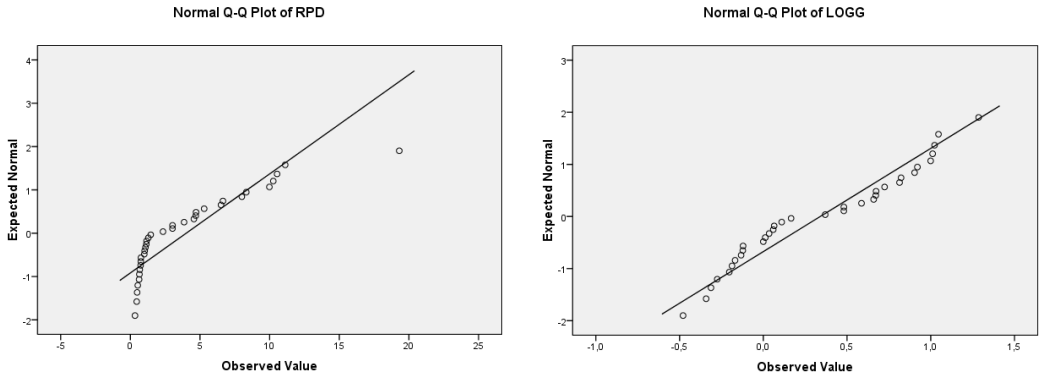
Şekil 5.23 incelendiğinde "Skewness" ve "Kurtosis" satırlarındaki değerler (-1,5) ile (+1,5) arasında olmadığı için H_0 hipotezi reddedilir "%95 güven aralığında veriler normal dağılımlı değildir" sonucuna ulaşılır. Daha önce de vurgulandığı üzere parametrik testleri kullanabilmemiz için verilerin normal dağılması gerekmektedir. Bunun için veri setine SPSS'te logaritmik dönüşümler uygulanarak veri seti normal dağılıma uygun hale getirilmeye çalışılır. Bu dönüşüm yapıldıktan sonra elde edilen değerler Şekil 5.24'te gösterilmiştir. Şekil 5.24 incelendiğinde "Skewness" ve "Kurtosis" satırlarındaki değerler (-1,5) ile (+1,5) arasında olduğu için H_0 hipotezi kabul edilir "%95 güven aralığında veriler normal dağılımlıdır" sonucuna ulaşılır.

Descriptives

			Statistic	Std. Error
LOG	Mean		,34037079	,086541617
	95% Confidence Interval for Mean	Lower Bound	,16430055	
		Upper Bound	,51644104	
	5% Trimmed Mean		,33572528	
	Median		,26788618	
	Variance		,255	
	Std. Deviation		,504620005	
	Minimum		-,479450	
	Maximum		1,285682	
	Range		1,765132	
	Interquartile Range		,940931	
	Skewness		,148	,403
	Kurtosis		-1,360	,788

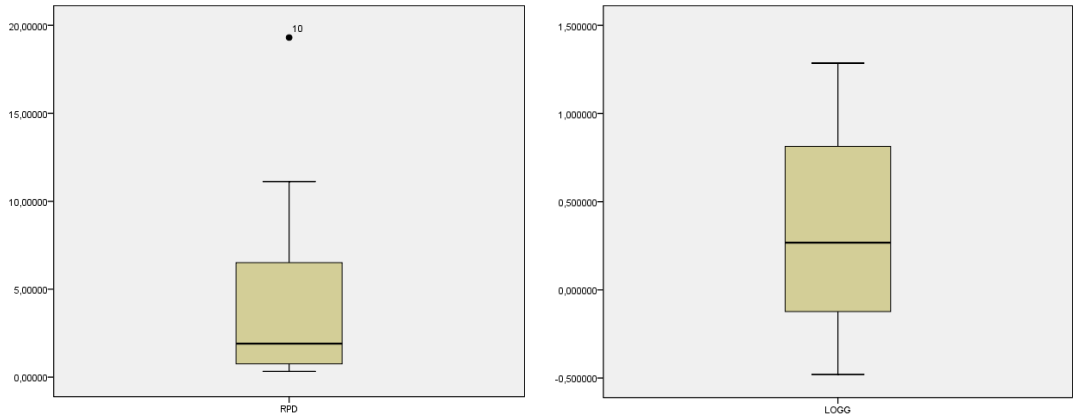
Şekil 5.24. Karınca kolonisi algoritması ve önerilen metot ile elde edilen RPD değerleri için logaritmik dönüşüm sonrası normallik testi sonuçları.

Veri setlerinin normal dağılımını grafiksel olarak görebilmek amacıyla logaritmik dönüşüm yapılmadan önce ve yapıldıktan sonraki Q-Q grafikleri (Quantile-Quantile plot) ve kutu diyagramları (Box-and-Whisker plot) sırasıyla Şekil 5.25(a)-(b) ve Şekil 5.26 (a)-(b)'de gösterilmiştir.



(a) (b)

Şekil 5.25. Karınca kolonisi algoritması ve önerilen metot ile elde edilen RPD değerleri için logaritmik dönüşüm öncesi (a) ve sonrası (b) Q-Q grafikleri.



(a) (b)

Şekil 5.26. Karınca kolonisi algoritması ve önerilen metodu ile elde edilen RPD değerleri için logaritmik dönüşüm öncesi (a) ve sonrası (b) kutu diyagramları.

Bu doğrulama yapıldıktan sonra logaritmik dönüşüm sonrası veri setlerinin varyanslarının % 95 güven aralığında homojen dağılıp dağılmadığını kontrol etmemiz gerekmektedir. Bunun için de yine SPSS paket programı kullanılarak varyansların homojenliği incelenmiştir.

Test of Homogeneity of Variances

LOG

Levene Statistic	df1	df2	Sig.
1,911	1	32	,176

Şekil 5.27. Karınca kolonisi algoritması ve önerilen metot ile elde edilen RPD değerleri için logaritmik dönüşüm sonrası varyansların homojenlik testi sonuçları.

Şekil 5.27 incelendiğinde "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi reddedilemez ve "%95 güven aralığında varyanslar homojendir" sonucuna ulaşılır.

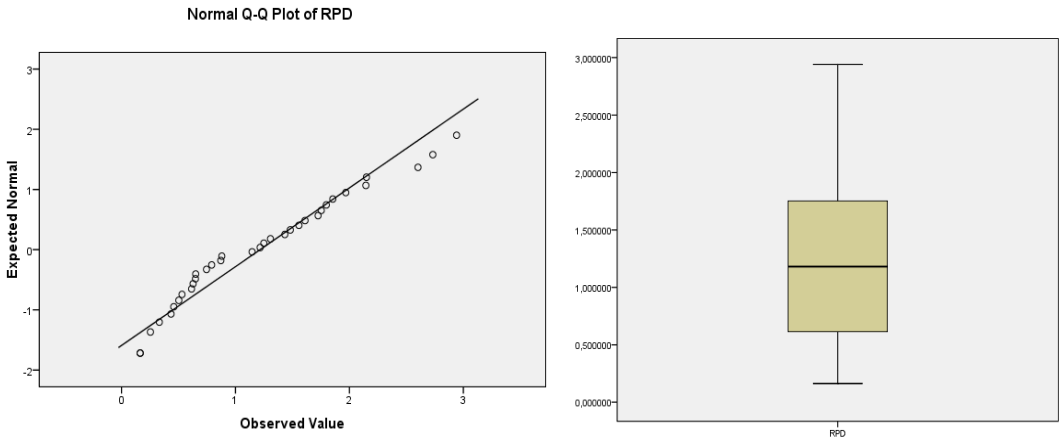
Diğer kıyaslanacak metot olan kuş sürüsü algoritması ve önerilen metot ile ile elde edilen RPD değerleri için yine aynı testler yapılmıştır. Şekil 5.28 incelendiğinde "Skewness" ve "Kurtosis" satırlarındaki değerler (-1,5) ile (+1,5) arasında olduğu için H_0 hipotezi kabul edilir "%95 güven aralığında veriler normal dağılımlıdır" sonucuna ulaşılır.

Descriptives

			Statistic	Std. Error
RPD	Mean		1,21550268	,131002032
	95% Confidence Interval for Mean	Lower Bound	,94897705	
		Upper Bound	1,48202832	
	5% Trimmed Mean		1,18290299	
	Median		1,18102477	
	Variance		,583	
	Std. Deviation		,763866544	
	Minimum		,163288	
	Maximum		2,940299	
	Range		2,777010	
	Interquartile Range		1,170210	
	Skewness		,552	,403
	Kurtosis		-,540	,788

Şekil 5.28. Kuş sürüsü algoritması ve önerilen metot ile elde edilen RPD değerleri için normallik testi sonuçları.

Veri setlerinin normal dağılımını grafiksel olarak görebilmek amacıyla Q-Q grafiği (Quantile-Quantile plot) ve kutu diyagramı (Box-and-Whisker plot) sırasıyla Şekil 5.29 (a)-(b)'de gösterilmiştir.



(a) (b)

Şekil 5.29. Kuş sürüsü algoritması ve önerilen metot ile elde edilen RPD değerleri için (a) Q-Q grafiği ve (b) kutu diyagramı.

Son olarak diferansiyel gelişim algoritması ve önerilen metot ile ile elde edilen RPD değerlerine aynı testler uygulanmıştır.

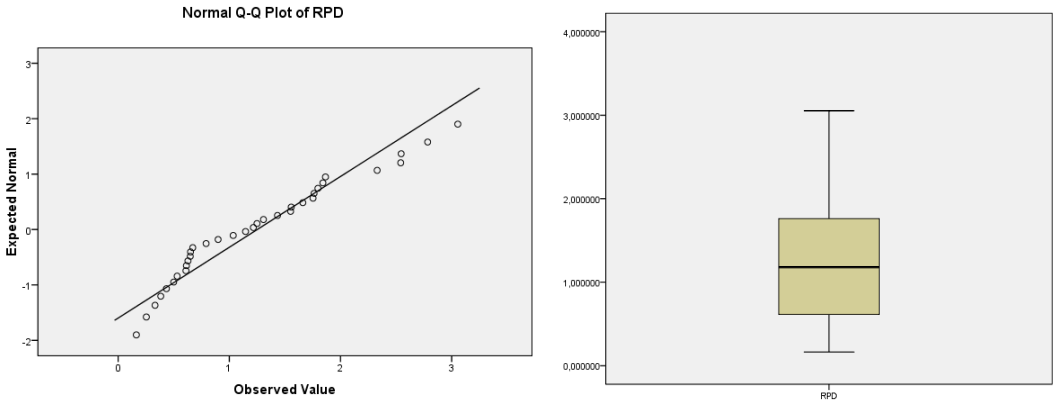
Descriptives

			Statistic	Std. Error
RPD	Mean		1,25093853	,134244703
	95% Confidence Interval for Mean	Lower Bound	,97781563	
		Upper Bound	1,52406144	
	5% Trimmed Mean		1,21530028	
	Median		1,18102477	
	Variance		,613	
	Std. Deviation		,782774404	
	Minimum		,163288	
	Maximum		3,054726	
	Range		2,891438	
	Interquartile Range		1,158156	
	Skewness		,633	,403
	Kurtosis		-,466	,788

Şekil 5.30. Diferansiyel gelişim algoritması ve önerilen metot ile elde edilen RPD değerleri için normallik testi sonuçları.

Şekil 5.30 incelendiğinde "Skewness" ve "Kurtosis" satırlarındaki değerler (-1,5) ile (+1,5) arasında olduğu için H_0 hipotezi kabul edilir "%95 güven aralığında veriler normal dağılımlıdır" sonucuna ulaşılır.

Veri setlerinin normal dağılımını grafiksel olarak görebilmek amacıyla Q-Q grafiği (Quantile-Quantile plot) ve kutu diyagramı (Box-and-Whisker plot) sırasıyla Şekil 5.31 (a)-(b)'de gösterilmiştir.



(a) (b)

Şekil 5.31. Diferansiyel gelişim algoritması ve önerilen metot ile elde edilen RPD değerleri için (a) Q-Q grafiği ve (b) kutu diyagramı.

Yapılan hipotez testleri sonucunda RPD verilerinin normal dağılıma uygunluğu ve varyanslarının homojenliği (PSO ve DE için varyansların homojenliği testi, varyans analizi ile birlikte verilecektir) sağlandıktan sonra PSO, DE ve önerilen metot ile elde edilen değerlere SPSS paket programı kullanılarak tek yönlü varyans analizi uygulaması yapılmıştır. Bunun için "farklı optimizasyon tekniklerinin ilgili veri setlerine uygulanması sonucu elde edilen RPD değerleri için uygulanan metotlar arasında istatistiksel olarak anlamlı bir fark var mıdır?" hipotezi, varyans analizi yapılarak test edilmiştir. Konu ile ilgili olarak aşağıdaki hipotez kurulmuş ve bu hipoteze göre sonuçlar alınmıştır.

- H_0: %95 güven aralığında farklı optimizasyon tekniklerinin ilgili veri setlerine uygulanması sonucu elde edilen RPD değerleri için uygulanan metotlar arasında istatistiksel olarak anlamlı bir fark yoktur.
- H_1: %95 güven aralığında farklı optimizasyon tekniklerinin ilgili veri setlerine uygulanması sonucu elde edilen RPD değerleri için uygulanan metotlar arasında istatistiksel olarak anlamlı bir fark vardır.

Hipotezler oluşturulduktan sonra ilgili RPD değerleri için gruplar arasında anlamlı bir fark olup olmadığının analiz edilmesinde yine SPSS paket programı kullanılmış ve tek yönlü varyans analizi yapılmıştır. Yapılan varyans analizi sonucu Şekil 5.32'de gösterilmiştir.

ANOVA

RPD

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	4,087	2	2,043	3,398	,042
Within Groups	28,865	48	,601		
Total	32,952	50			

Şekil 5.32. İlgili optimizasyon metotlarının RPD değerleri için varyans analizi sonuçları.

Şekil 5.32 incelendiğinde "Sig." değeri 0,05'ten küçük olduğu için H_0 hipotezi reddedilir ve "%95 güven aralığında farklı optimizasyon tekniklerinin ilgili veri setlerine uygulanması sonucu elde edilen RPD değerleri için uygulanan metotlar arasında istatistiksel olarak anlamlı bir fark vardır" sonucuna ulaşılır. Şekil 5.33 te de ilgili veriler için varyansların homojenliği testi sonuçları görülebilir. "Sig." değeri 0,05'ten büyük olduğu için H_0 hipotezi reddedilemez ve "%95 güven aralığında varyanslar homojendir" sonucuna ulaşılır.

Test of Homogeneity of Variances

RPD

Levene Statistic	df1	df2	Sig.
2,957	2	48	,062

Şekil 5.33. PSO, DE algoritmaları ve önerilen metot ile elde edilen RPD değerleri için varyansların homojenlik testi sonuçları.

Diğer optimizasyon algoritması ACO ve önerilen metot ile elde edilen değerlere ise bağımsız-t testi uygulanmıştır. Daha sonra varyans analizi yapılarak istatistiksel olarak anlamlı farkların bulunduğu 3 metot için de 2'li olarak bağımsız-t testi uygulanmış ve daha detaylı olarak hipotez sonuçlarına bakılmıştır. Bunun için de "önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPD değerleri ile literatürdeki diğer optimizasyon algoritmaları ile elde edilmiş RPD değerleri arasında istatistiksel olarak anlamlı bir fark var mıdır?" hipotezi herbir metot için sırasıyla test edilmiştir. Farklı optimizasyon tekniklerinin herbiri için konu ile ilgili aşağıdaki hipotezler kurulmuş ve bu hipotezlere göre sonuçlar alınmıştır. İlk olarak önerilen metot ile karınca kolonisi optimizasyon tekniği sonuçları değerlendirilmiştir. Bu bağlamda aşağıdaki hipotez kurulmuştur.

- H_0: %95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPD değerleri ile literatürdeki karınca kolonisi optimizasyon tekniği sonuçları arasında istatistiksel olarak anlamlı bir fark yoktur.
- H_1: %95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPD değerleri ile literatürdeki karınca kolonisi optimizasyon tekniği sonuçları arasında istatistiksel olarak anlamlı bir fark vardır.

Hipotezler oluşturulduktan sonra iki farklı metot ile elde edilen RPD değerleri SPSS paket programındaki bağımsız t-testi uygulamasına tabi tutulmuştur.

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
LOG	Equal variances assumed	1,911	,176	12,355	32	,000	,90403	,0731704	,754993	1,05308
	Equal variances not assumed			12,355	29,333	,000	,9041	,0731704	,754460	1,05361

Şekil 5.34. Karınca kolonisi algoritması ve önerilen metot ile elde edilen RPD değerleri için bağımsız t-testi sonuçları.

Şekil 5.34 incelendiğinde "Sig. (2-tailed)" değeri 0,05'ten küçük olduğu için H_0 hipotezi reddedilir ve "%95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPD değerleri ile literatürdeki karınca kolonisi optimizasyon tekniği sonuçları arasında istatistiksel olarak anlamlı bir fark vardır" sonucuna ulaşılır.

Daha sonra diğer kuş sürüsü algoritması ile önerilen metodun sonuçları incelenmiştir. Bunun için aşağıdaki hipotez kurulmuştur.

- H_0: %95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPD değerleri ile literatürdeki kuş sürüsü optimizasyon tekniği sonuçları arasında istatistiksel olarak anlamlı bir fark yoktur.
- H_1: %95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPD değerleri ile literatürdeki kuş sürüsü optimizasyon tekniği sonuçları arasında istatistiksel olarak anlamlı bir fark vardır.

Hipotezler oluşturulduktan sonra iki farklı metot ile elde edilen RPD değerleri SPSS paket programındaki bağımsız t-testi uygulamasına tabi tutulmuştur.

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
RPD	Equal variances assumed	5,358	,057	-2,277	32	,030	-,5619	,2468	-1,064	-,0591
	Equal variances not assumed			-2,277	26,529	,031	-,5619	,2468	-1,0687	-,055

Şekil 5.35. Kuş sürüsü algoritması ve önerilen metot ile elde edilen RPD değerleri için bağımsız t-testi sonuçları.

Şekil 5.35 incelendiğinde "Sig. (2-tailed)" değeri 0,05'ten küçük olduğu için H_0 hipotezi reddedilir ve "%95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPD değerleri ile literatürdeki kuş sürüsü optimizasyon tekniği sonuçları arasında istatistiksel olarak anlamlı bir fark vardır" sonucuna ulaşılır.

Son olarak diferansiyel gelişim algoritması ile önerilen metot sonuçları incelenmiştir. Bunun için aşağıdaki hipotez kurulmuştur.

- H_0: %95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPD değerleri ile literatürdeki diferansiyel gelişim algoritması sonuçları arasında istatistiksel olarak anlamlı bir fark yoktur.
- H_1: %95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPD değerleri ile literatürdeki diferansiyel gelişim algoritması sonuçları arasında istatistiksel olarak anlamlı bir fark vardır.

Hipotezler oluşturulduktan sonra iki farklı metot ile elde edilen RPD değerleri SPSS paket programındaki bağımsız t-testi uygulamasına tabi tutulmuştur.

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
RPD	Equal variances assumed	5,211	,029	-2,545	32	,016	-,6327	,2486	-1,1392	-,1262
	Equal variances not assumed			-2,545	26,369	,017	-,6327	,2486	-1,1435	-,122

Şekil 5.36. Diferansiyel gelişim algoritması ve önerilen metot ile elde edilen RPD değerleri için bağımsız t-testi sonuçları.

Şekil 5.36 incelendiğinde "Sig. (2-tailed)" değeri 0,05'ten küçük olduğu için H_0 hipotezi reddedilir ve "%95 güven aralığında önerilen metodun ilgili veri setlerine uygulanması sonucu elde edilen RPD değerleri ile literatürdeki diferansiyel gelişim algoritması sonuçları arasında istatistiksel olarak anlamlı bir fark vardır" sonucuna ulaşılır.

Sonuç olarak, önerilen metot ile elde edilen RPD değerleri ile literatürdeki 3 farklı eniyileme teknikleri ile elde edilen RPD değerlerinin bağımsız-t testi uygulanarak kıyaslanması neticesinde, önerilen metot ve diğer 3 farklı optimizasyon algoritması ile elde edilen sonuçlar arasında istatistiksel olarak anlamlı farklılıklar olduğu görülmüştür.

Son olarak tezin bir önceki bölümünde bahsedilen ortalama bağıl hata yüzdesi kriteri kullanılarak önerilen metot ile elde edilen RPE değerleri, literatürdeki ACO, PSO ve DE algoritmaları kullanılarak elde edilen RPE değerleri ile sırasıyla Tablo (5.1-5.2-5.3)'te kıyaslanmıştır.

Tablolardaki "instance" sütunu data setlerinin isimlerini, "Best Known" sütunu literatürdeki en iyi bulunan çözümün değerini, "Best" ABC-EA bütünleşik yaklaşımı ile bulunan en iyi çözüm değerini, "Average" sütunu her bir data seti için kurulan sistemin 20 kez çalıştırılması sonucunda elde edilen değerlerin ortalamasını, "Std.Sapma" sütunu da yine 20 kez çalıştırılması sonucu elde edilen değerlerin standart sapmasını, "RPE" sütunu ilgili data seti için hesaplanan bağıl hata yüzdesini, "time limit" sütunu ise saniye cinsinden ilgili data seti için kullanılan süre kısıtını, "RPD" sütunu ise ilgili data seti için hesaplanan bağıl sapma yüzdesini göstermektedir.

Tablo 5.1 incelendiğinde ilgili data setleri için literatürde karınca kolonisi optimizasyon tekniği kullanılarak elde edilmiş ortalama bağıl hata yüzdesi (ARPE) değeri 4,402 ve ortalama bağıl sapma yüzdesi (ARPD) ise 7,188'dir. ABC-EA bütünleşik yaklaşımı ile ilgili data setlerinin çözümü sonucu ortalama bağıl hata yüzdesi 0,125 ve ortalama bağıl sapma yüzdesi ise 0,833 bulunmuştur.

Tablo 5.2 incelendiğinde ilgili data setleri için literatürde kuş sürüsü optimizasyon tekniği kullanılarak elde edilmiş ortalama bağıl hata yüzdesi değeri 0,415 ve ortalama bağıl sapma yüzdesi 1,496'dır. ABC-EA bütünleşik yaklaşımı ile ilgili data setlerinin çözümü sonucu ortalama bağıl hata yüzdesi 0,331 ve ortalama bağıl sapma yüzdesi de 0,934 bulunmuştur. Son olarak Tablo 5.3 incelendiğinde, ilgili data setleri için literatürde DE algoritması kullanılarak elde edilmiş ortalama bağıl hata yüzdesi değeri

0,51 ve ortalama bağıl sapma yüzdesi 1,567'dir. ABC-EA bütünleşik yaklaşımı ile ilgili data setlerinin çözümü sonucu ortalama bağıl hata yüzdesi 0,331 ve ortalama bağıl sapma yüzdesi de 0,934 olarak bulunmuştur.
Yapılan bütün testler sonucunda, ABC-EA bütünleşik yaklaşımı ile diğer üç optimizasyon tekniği sonuçlarını kıyaslayacak olursak:

- 1.sonuç: Önerilen metot ile elde edilen ortalama bağıl hata yüzdesi, ACO metodu ile elde edilen ARPE değerine göre 4,3 puan (yüzdesel değişim olarak) daha düşük olduğundan daha etkin bir netice vermiştir. Ulaşılan bu sonuç, yapılan parametrik ve parametrik olmayan testlerle bu iki metodun sonuçları arasındaki istatistiksel olarak anlamlı farklılıkların olduğu hipotezini desteklemektedir.

- 2.sonuç: Önerilen ABC-EA bütünleşik yaklaşımı ile elde edilen ortalama bağıl hata yüzdesi, PSO metodu ile elde edilen ARPE değerine göre 0,1 puan (yüzdesel değişim olarak) daha düşük ve etkin bir netice vermiştir. Ulaşılan bu sonuç, yapılan parametrik ve parametrik olmayan testlerle bu iki metodun sonuçları arasındaki istatistiksel olarak anlamlı bir farkın olmadığı hipotezini desteklemektedir.

- 3.sonuç: Önerilen yaklaşım ile elde edilen ortalama bağıl hata yüzdesi, DE metodu ile elde edilen ARPE değerine göre 0,2 puan (yüzdesel değişim olarak) daha düşük ve etkin bir netice vermiştir. Ulaşılan bu sonuç, yapılan parametrik ve parametrik olmayan testlerle bu iki metodun sonuçları arasındaki istatistiksel olarak anlamlı bir farkın olmadığı hipotezini desteklemektedir.

- 4.sonuç: Önerilen metot ile elde edilen ortalama bağıl sapma yüzdesi, ACO metodu ile elde edilen ARPD değerine göre 6,3 puan (yüzdesel değişim olarak) daha düşük olduğundan daha kararlı ve etkin bir netice vermiştir. Ulaşılan bu sonuç, yapılan parametrik testlerle bu iki metodun sonuçları arasındaki istatistiksel olarak anlamlı farklılıkların olduğu hipotezini desteklemektedir.

- 5.sonuç: Önerilen ABC-EA bütünleşik yaklaşımı ile elde edilen ortalama bağıl sapma yüzdesi, PSO metodu ile elde edilen ARPD değerine göre 0,6 puan (yüzdesel değişim olarak) daha düşük olduğundan daha kararlı ve etkin bir netice vermiştir. Ulaşılan bu sonuç, yapılan parametrik testlerle bu iki metodun sonuçları arasındaki istatistiksel olarak anlamlı farklılıkların olduğu hipotezini desteklemektedir.

- 6.sonuç: Önerilen yaklaşım ile elde edilen ortalama bağıl sapma yüzdesi, DE metodu ile elde edilen ARPD değerine göre 0,7 puan (yüzdesel değişim olarak) daha düşük olduğundan daha kararlı ve etkin bir netice vermiştir. Ulaşılan bu sonuç, yapılan parametrik testlerle bu iki metodun sonuçları arasındaki istatistiksel olarak anlamlı farklılıkların olduğu hipotezini desteklemektedir.

Yapılan bu çalışmalardan sonra ileriye yönelik olarak bu konu ile ilgili yapılacak araştırma ve çalışmalar için şu önerileri sunabiliriz:

- Öncelikle önerilen ABC-EA bütünleşik yaklaşımı ile atölye tipi çizelgeleme problemleri için etkin neticelere ulaşılmıştır. Ancak özelikle (iş*makine) sayısının çok daha büyük olduğu data setlerinde önerilen metot, (iş*makine) sayısının daha küçük olduğu data setlerine göre daha etkin derecede çizelgeleme sonuçları alması çok daha uzun sürmektedir. Bu sürelerin daha da kısaltılabilmesi, ilgili konu hakkındaki araştırmaların daha iyi seviyelere gelmesini sağlayabilir.

- Bu tez çalışmasında da değinildiği gibi literatürde çok sayıda sezgisel algoritmalar mevcuttur ve bu sezgisellerin bütünleşik olarak kullanımları sonucu daha etkin neticeler veren yaklaşımlar ortaya çıkabilmektedir. Bunun için ABC metodunun literatürde daha önce birlikte kullanılmamış farklı optimizasyon teknikleri ile birlikte kullanılması araştırılabilir.

- Önerilen ABC metodu ile evrimsel algoritmaların bütünleşik yaklaşımı ile elde edilen sonuçlar, bu çalışmadaki kıyaslanan metotların dışında diğer farklı

eniyileme teknikleri ile kıyaslanarak, metodolojinin etkinliği konusunda katkıda bulunulabilir.

Tablo 5.1. Karınca kolonisi algoritması ve ABC-EA bütünleşik yaklaşımı ile elde edilen değerlerin kıyaslanması.

KARINCA KOLONİSİ ALGORİTMASI İLE ELDE EDİLEN DEĞERLER								ABC-EA BÜTÜNLEŞİK YAKLAŞIMI İLE ELDE EDİLEN DEĞERLER							
Instance	**Best Known**	**Best**	**Average**	**Std.Sapma**	**RPD**	**RPE**	**Time limit**	**Instance**	**Best Known**	**Best**	**Average**	**Std.Sapma**	**RPD**	**RPE**	**Time limit**
la02	655	669	689,7	6,2	5,2977099	2,13	76,15	la02	655	655	659,1	4,39	0,6259542	0	76,15
la03	597	623	644,8	8	8,0067002	4,36	76,15	la03	597	597	601,4	4,15	0,73701843	0	76,15
la04	590	611	617,7	5	4,6949153	3,56	76,15	la04	590	590	595,9	5,05	1	0	76,15
la17	784	812	836,1	10,9	6,6454082	3,57	76,15	la17	784	784	793,1	6,798	1,16071429	0	76,15
la19	842	875	881,7	4,7	4,7149644	3,92	76,15	la19	842	842	846,1	4,24	0,48693587	0	76,15
la20	902	912	936,8	9,6	3,8580931	1,11	76,15	la20	902	902	908,8	5,74	0,75388027	0	76,15
la21	1046	1107	1162,3	16,4	11,118547	5,38	380,77	la21	1046	1046	1051,55	4,273	0,53059273	0	380,77
la24	935	1011	1033,5	8,3	10,534759	8,13	380,77	la24	935	935	938,1	2,511	0,3315508	0	380,77
la26	1218	1296	1339,6	16,2	9,9835796	6,4	761,54	la26	1218	1221	1235,85	9,85	1,46551724	0,25	761,54
la29	1152	1339	1374,4	11,9	19,305556	15,73	761,54	la29	1152	1158	1165,2	6,066	1,14583333	0,52	761,54
la30	1355	1410	1443,2	15	6,5092251	4,06	761,54	la30	1355	1355	1372,45	12,63	1,28782288	0	761,54
la31	1784	1798	1825,8	12,5	2,3430493	0,78	761,54	la31	1784	1788	1797,5	6,79	0,75672646	0,22	761,54
la32	1850	1868	1906	20,7	3,027027	0,97	761,54	la32	1850	1855	1862,55	7,74	0,67837838	0,27	761,54
la33	1719	1731	1771	15,1	3,0250145	0,7	761,54	la33	1719	1727	1736,65	9,92	1,02675974	0,47	761,54
la35	1888	1913	1974,1	22,8	4,5603814	1,32	761,54	la35	1888	1888	1896,6	5,65	0,45550847	0	761,54
la39	1233	1304	1359,5	16,2	10,25953	5,76	761,54	la39	1233	1238	1246,35	8,88	1,08272506	0,4	761,54
la40	1222	1307	1323,7	9,2	8,3224223	6,96	761,54	la40	1222	1222	1229,95	5,031	0,65057283	0	761,54
ORTALAMA				**12,27647**	**7,188**	**4,402**		**ORTALAMA**				**6,45347**	**0,833**	**0,125**	

Tablo 5.2. PSO algoritması ve ABC-EA bütünleşik yaklaşımı ile elde edilen değerlerin kıyaslanması.

PSO İLE ELDE EDİLEN DEĞERLER								ABC-EA BÜTÜNLEŞİK YAKLAŞIMI İLE ELDE EDİLEN DEĞERLER							
Instance	**Best Known**	**Best**	**Average**	**Std.Sapma**	**RPD**	**RPE**	**Time limit**	**Instance**	**Best Known**	**Best**	**Average**	**Std.Sapma**	**RPD**	**RPE**	**Time limit**
ft20	1165	1165	1175,25	5,3	0,8798283	0	76,15	ft20	1165	1165	1167,95	3,347	0,25321888	0	76,15
orb01	1059	1059	1076,05	10,58	1,6100094	0	76,15	orb01	1059	1059	1065,65	6,019	0,6279509	0	76,15
orb02	888	889	889,45	1,79	0,1632883	0,11	76,15	orb02	888	888	889,45	1,877	0,16328829	0	76,15
orb03	1005	1005	1034,55	22,95	2,9402985	0	76,15	orb03	1005	1005	1012,95	5,986	0,79104478	0	76,15
orb05	887	887	892,45	4,81	0,6144307	0	76,15	orb05	887	887	890,85	3,133	0,43404735	0	76,15
orb06	1010	1013	1018,8	5,73	0,8712871	0,3	76,15	orb06	1010	1010	1016,55	5,898	0,64851485	0	76,15
la21	1046	1047	1053,8	6,01	0,7456979	0,1	380,77	la21	1046	1046	1051,55	4,273	0,53059273	0	380,77
la24	935	935	939,7	3,63	0,5026738	0	380,77	la24	935	935	938,1	2,511	0,3315508	0	380,77
la29	1152	1164	1176,7	10,55	2,1440972	1,04	761,54	la29	1152	1158	1165,2	6,066	1,14583333	0,52	761,54
la40	1222	1224	1227,6	3,8	0,4582651	0,16	761,54	la40	1222	1222	1229,95	5,031	0,65057283	0	761,54
abz07	656	659	670,1	5,75	2,1493902	0,46	761,54	abz07	656	659	665,4	4,044	1,43292683	0,46	761,54
abz08	665	674	682,3	5,52	2,6015038	1,35	761,54	abz08	665	674	676,95	3,252	1,79699248	1,35	761,54
abz09	679	688	697,55	5,79	2,7319588	1,33	761,54	abz09	679	688	690,9	3,945	1,75257732	1,33	761,54
yn01	888	893	901,15	6,56	1,4808559	0,56	1523,08	yn01	888	891	898,8	4,324	1,21621622	0,34	1523,08
yn02	909	910	925,85	6,43	1,8536854	0,11	1523,08	yn02	909	911	920,35	6,192	1,24862486	0,22	1523,08
yn03	893	902	908,4	5,23	1,7245241	1,01	1523,08	yn03	893	901	906,9	4,575	1,55655095	0,89	1523,08
yn04	968	973	987,05	8,87	1,9679752	0,52	1523,08	yn04	968	973	980,65	5,833	1,30681818	0,52	1523,08
ORTALAMA				**7,017647**	**1,496**	**0,415**		**ORTALAMA**				**4,48859**	**0,9345483**	**0,331**	

Tablo 5.3. DE algoritması ve ABC-EA bütünleşik yaklaşımı ile elde edilen değerlerin kıyaslanması.

DE ALGORİTMASI İLE ELDE EDİLEN DEĞERLER								ABC-EA BÜTÜNLEŞİK YAKLAŞIMI İLE ELDE EDİLEN DEĞERLER							
Instance	**Best Known**	**Best**	**Average**	**Std.Sapma**	**RPD**	**RPE**	**Time limit**	**Instance**	**Best Known**	**Best**	**Average**	**Std.Sapma**	**RPD**	**RPE**	**Time limit**
ft20	1165	1165	1172,15	6,64	0,6137339	0	76,15	ft20	1165	1165	1167,95	3,347	0,25321888	0	76,15
orb01	1059	1064	1078,5	11,02	1,8413598	0,47	76,15	orb01	1059	1059	1065,65	6,019	0,6279509	0	76,15
orb02	888	889	891,4	3,76	0,3828829	0,11	76,15	orb02	888	888	889,45	1,877	0,16328829	0	76,15
orb03	1005	1005	1035,7	21,99	3,0547264	0	76,15	orb03	1005	1005	1012,95	5,986	0,79104478	0	76,15
orb05	887	889	892,95	7,76	0,6708005	0,23	76,15	orb05	887	887	890,85	3,133	0,43404735	0	76,15
orb06	1010	1010	1020,45	6,27	1,0346535	0	76,15	orb06	1010	1010	1016,55	5,898	0,64851485	0	76,15
la21	1046	1047	1055,4	7,39	0,8986616	0,1	380,77	la21	1046	1046	1051,55	4,273	0,53059273	0	380,77
la24	935	938	940,7	5,96	0,6096257	0,32	380,77	la24	935	935	938,1	2,511	0,3315508	0	380,77
la29	1152	1163	1172,3	6,35	1,7621528	0,95	761,54	la29	1152	1158	1165,2	6,066	1,14583333	0,52	761,54
la40	1222	1224	1228,1	5,57	0,4991817	0,16	761,54	la40	1222	1222	1229,95	5,031	0,65057283	0	761,54
abz07	656	666	672,7	3,84	2,5457317	1,52	761,54	abz07	656	659	665,4	4,044	1,43292683	0,46	761,54
abz08	665	674	681,9	6,13	2,5413534	1,35	761,54	abz08	665	674	676,95	3,252	1,79699248	1,35	761,54
abz09	679	682	697,9	8,8	2,7835052	0,44	761,54	abz09	679	688	690,9	3,945	1,75257732	1,33	761,54
yn01	888	894	902,75	5,58	1,661036	0,68	1523,08	yn01	888	891	898,8	4,324	1,21621622	0,34	1523,08
yn02	909	917	925,95	3,69	1,8646865	0,88	1523,08	yn02	909	911	920,35	6,192	1,24862486	0,22	1523,08
yn03	893	895	906,85	6,93	1,5509518	0,22	1523,08	yn03	893	901	906,9	4,575	1,55655095	0,89	1523,08
yn04	968	980	990,55	7,98	2,3295455	1,24	1523,08	yn04	968	973	980,65	5,833	1,30681818	0,52	1523,08
ORTALAMA				**7,391765**	**1,567**	**0,510**		**ORTALAMA**				**4,48859**	**0,9345483**	**0,331**	

KAYNAKLAR

[1] Eren, T., Güner, E. A Literature Survey for Multicriteria Flowshop Scheduling Problems, Journal of Engineering Sciences, Vol.10, pp.19-30, 2004.

[2] Murty, K.G. Optimization Models for Decision Making. Vol. 1, Internet Edition, 2003.

[3] Akay, B. Performance Analysis of Artificial Bee Colony Algorithm on Numerical Optimization Problems. Ph.D. Thesis, Erciyes University, Turkey, 2009.

[4] Makas, H. Parallel and Collaborative Applications of the Recent Optimization Algorithms and their Performance Analyses, Ph.D. Thesis, Sakarya University, Turkey, 2015.

[5] Bellman, R. The Theory of Dynamic Programming, Princeton University Press, New Jersey, 1954.

[6] Yücel, M., Ulutaş, A. The Application of Dynamic Programming for Minimization of the Cost of Labours, The Journal of Faculty of Economics and Administrative Sciences,Vol.15, No.3 pp.271-290, 2010.

[7] Held, M., Karp, R. M. A Dynamic Programming Approach to Sequencing Problems, SIAM Journal on Applied Mathematics, Volume 10, No: 1, pp. 196-210, 1962.

[8] Lawler, E. L. On Scheduling Problems with Deferral Costs, Managemet Science, Volume 11, pp. 280-288, 1964.

[9] Lawler, L. E., Moore, J. M. Functional Equation and Its Application to Resource Allocation and Sequencing Problems, Management Science, Volume 16, pp. 77-84, 1969.

[10] Lenstra, J. K. Sequencing by Enumerative Method, Second Printing, Mathemtisch Centrum, 1985.

[11] Morton, T. E., Pentico, D. W. Heuristic Scheduling Systems, Wiley, New York, 1993.

[12] Pinedo, M. L. Scheduling: Theory, Algorithms, and Systems, Prentice-Hall, Englewood, 1995.

[13] Eastman, W. L. A Solution to the Travelling Salesman Problem, Econometrica, Volume 27, pp. 282, 1959.

[14] Dantzig, G. A Comment On Edie's Traffic Delays at Tool Booths, Operations Research, 2(3), ss.339-341, 1954.

[15] Chandler, P.R., Patcher, M. Research Issues in Autonomous Control of Tactical UAVs. In Proc. American Control Conference, Philadelphia, Pennsylvania, pp. 394-398, 1998.

[16] Duman, E., Uysal, M., Alkaya, A.F. Migrating Birds Optimization: A new metaheuristic approach and its performance on quadratic assignment problem. Information Sciences, 217:65–77, 2012.

[17] Karaboğa, D. Yapay Zekâ Optimizasyon Algoritmaları, Nobel Yayın Dağıtım, 2011.

[18] Alataş, B. Kaotik Haritalı Parçacık Sürü Optimizasyon Algoritmaları Geliştirme, Doktora Tezi, Fırat Üniversitesi, Fen Bilimleri Enstitüsü, 2007.

[19] Dorigo, M. Optimization, Learning and Natural Algorithms, PhD thesis, Dipartimento di Elettoronica, Politecnico di Milano, IT, 1992.

[20] Dorigo, M., Maniezzo, V., Colorni, A. Positive Feedback as a Search Strategy, Technical Report 91-016, Dipartimento di Elettronica, Politecnico di Milano, IT, 1991.

[21] Deneubourg, J.L., Aron S., Goss S., Pasteels, J.M. The Self-organizing Exploratory Pattern of the Argentine Ant, Journal of Insect Behavior, 3:159-168, 1990.

[22] Kılıç, S., Kaylan, A.R. Uçak Çizelgeleme Probleminin Karınca Kolonileri Optimizasyonu ile Çözümü, Havacılık ve Uzay Teknolojileri Dergisi, Cilt:2, Sayı:1 (87-95), 2005.

[23] Kennedy, J., Eberhart, R. C. Particle Swarm Optimization, Proc. Ieee int'l conf. On neural Networks, Vol. IV, pp. 1942-1948. IEEE service center, Piscataway, NJ, 1995.

[24] Guerra, F.A., Coelho, L.S. Multi-Step Ahead Nonlinear İdentification of Lorenz's Chaotic System Using Radial Basis Neural Network With Learning by Clustering and Particle Swarm Optimization, Chaos, Solitons & Fractals, In Press, Corrected Proof, Available online, 2006.

[25] Zhou, J., Duan, Z., Li, Y., Deng, J.,Yu, D. PSO-based Neural Network Optimization and its Utilization in a Boring Machine, Journal of Materials Processing Technology, In Press, Corrected Proof, Available online, 2006.

[26] Bocaniala, C.D., Costa, J.S. Application of a Novel Fuzzy Classifier to Fault Detection and Isolation of the Damadics Benchmark Problem, Control Engineering Practice, Volume 14, Issue 6, Pages 653-669, 2006.

[27] Ghoshal, S. P. Optimizations of PID Gains by Particle Swarm Optimizations in Fuzzy Based Automatic Generation Control, Electric Power Systems Research, Volume 72, Issue 3, 15, Pages 203-212, 2004.

[28] Kökçam, A.H., Engin, O. Bulanık Proje Çizelgeleme Problemlerinin Meta Sezgisel Yöntemlerle Çözümü, Mühendislik ve Fen Bilimleri Dergisi (Sigma), 28, 86-101, 2010.

[29] Serbencu, A., Minzu, V., Serbencu, A. An Ant Colony System Based Metaheurıstic for Solving Single Machine Scheduling Problem, The Annals of Dunarea De Jos Unıversity Of Galati Fascicle III, Electrotechnıcs, Electronıcs, Automatıc Control, Informatıcs, 19- 24, 2007.

[30] Durmaz, T. Açık Atölye Çizelgeleme Problemlerinin Paralel Kanguru Algoritması ile Çözümü, Yüksek Lisans Tezi, Selçuk Üniversitesi, Fen Bilimleri Enstitüsü, 2011.

[31] Gandomi, A. H., Alavi, A. H. Krill Herd: A New Bio-Inspired Optimization Algorithm, Communications in Nonlinear Science and Numerical Simulation, Cilt 17, No. 12, s.4831-4845, 2012.

[32] Hofmann, E. E., Haskell, A. G. E., Klinck, J. M., Lascara, C. M. Lagrangian Modelling Studies of Antarctic Krill (Euphausia Superba) Swarm Formation, ICES Journal of Marine Science, Cilt 61, No. 4, s.617-631, 2004.

[33] Kirkpatrick, S., Gelatt, C. D., Vecchi, M. P. Optimization by Simulated Annealing, Science, Volume 220, pp. 671–680, 1983.

[34] Cherny, V., Thermodynamical Approach to Traveling Salesman Problem: an Efficient Simulation Algorithm, Journal of Optimization Theory and Applications, Volume 45 No: 1, pp. 41-51, 1985.

[35] Aarts, E. H. L., Korst, J. H. Simulated Annealing and Bolzmann Machines: A Stochastic Approach to Combinatorial optimization and Neural Computing, Wiley, New York, 1989.

[36] Aarts, E. H. L., Korst, J. H. Boltzmann Machiness for Traveling Salesman Problems, European Journal of Operational Research, Volume 39, No. 1, pp. 79-95, 1989.

[37] Yang, X. S. Firefly Algorithms Formultimodal Optimization, Proceedings of the Stochastic Algorithms: Foundations and Applications, Lecture Notes in Computing Sciences, vol. 5792, 178-178, Springer, Sapporo, Japan, 2009.

[38] Yang, X. S. Firefly Algorithm, Levy Flights and Global Optimization, Research and Development in Intelligent Systems, XXVI, 209-218, Springer, London, UK, 2010.

[39] Apostolopoulos, T., Vlachos, A. Application of the Firefly Algorithm for Solving the Economic Emissions Load Dispatch Problem, Hindawi Publishing Corporation International Conference Journal of Combinatorics, vol. 2011, doi:10.1155/2011/523806, 2011.

[40] Akyol, S., Alatasban, B. Güncel Sürü Zekâsı Optimizasyon Algoritmaları, Nevşehir Üniversitesi Fen Bilimleri Enstitü Dergisi 1, ss.36-50, 2012.

[41] Glover, F. Tabu Search - Part I, ORSA Journal on Computing, Volume 1, No: 3, pp. 190-206, 1989.

[42] Glover, F. Tabu Search - Part II, ORSA Journal on Computing, Volume 2, No: 1, pp. 4-32, 1990.

[43] Glover, F., Laguna, M. Tabu Search, Kluwer Academic Publishers, United Stated of America, 1997.

[44] Holland, J. H. Adaptation in Natural and Artificial systems, University of Michigan Press, 1975.

[45] De Jong, K. A. An Analysis of the Bahavior of a Class of Genetic Adaptive Systems, Ph.D. Thesis, University of Michigan, 1975.

[46] Goldberg, D. E. Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, M. A, 1989.

[47] Jiang, M., Yuan, D., Cheng, Y. Improved Artificial Fish Swarm Algorithm, Fifth International Conference on Natural Computation, 281-285, 2009.

[48] Başbuğ, S. Bakteriyel Besin Arama Algoritması ile Lineer Anten Dizilerinin Diyagram Sıfırlaması, Yüksek Lisans Tezi, Erciyes Üniversitesi, Fen Bilimleri Enstitüsü, 2008.

[49] Liu, C., Yan, X., Liu, C., Wu, H., The Wolf Colony Algorithm and Its Application, Chinese Journal of Electronics, vol. 20, no. 2, 2011.

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

[51] Santosa, B., Ningrum, M.K., Cat Swarm Optimization for Clustering, International Conference of Soft Computing and Pattern Recognition, 54-59, 2009.

[52] Wang, Z. H., Chang, C. C., Li, M. C. Optimizing Least-Significant-Bit Substitution Using Cat Swarm Optimization Strategy, Information Sciences, 10.1016/j.ins.2010.07.011, 2010.

[53] Hwang, J. C., Chen, J. C., Pan, J. S. Huang, Y.C., CSO and PSO to Solve Optimal Contract Capacity for High Tension Customers, International Conference on Power Electronics and Drive Systems, 246-251, 2009.

[54] Lin, K. C., Chien, H. Y. CSO-Based Feature Selection and Parameter Optimization for Support Vector Machine, Joint Conferences on Pervasive Computing, 783-788, 3-5, 2009.

[55] Kalaiselvan, G., Lavanya, A., Natrajan, V. Enhancing the Performance of Watermarking Based on Cat Swarm Optimization Method, International Conference on Recent Trends in Information Technology, 1081-1086, 2011.

[56] Wang, W., Wu, J. Emotion Recognition Based on CSO&SVM in E-Learning, Seventh International Conference on Natural Computation, vol. 1, 566-570, 2011.

[57] Tsai, P. W., Pan, J. S., Chen, S.M., Liao, B.Y., Hao, S.P. Parallel Cat Swarm Optimization, International Conference on Machine Learning and Cybernetics, vol. 6, 3328-3333, 2008.

[58] Atashpaz-Gargari, E., Lucas, C. Imperialist Competitive Algorithm: An Algorithm for Optimization Inspired by Imperialistic Competition", IEEE Congress on Evolutionary Computation, pp. 4661-4667, 2007.

[59] Borji, A. A New Global Optimization Algorithm Inspired by Parliamentart Political Competitions, Lecture Notes in Computer Science, 4827/2007, pp. 61-71, 2007.

[60] Borji, A., Hamidi, M. A New Approach to Global Optimization Motivated by Parliamentary Political Competitions, Int. Journal of Innovative Computing Information and Control, vol. 5, no. 6, pp. 1643- 1653, 2009.

[61] Karaboğa, D., Öztürk, C. A Novel Clustering Approach: Artificial Bee Colony (ABC) Algorithm, Applied Soft Computing 11, pp:652–657, 2011.

[62] Karaboğa, D., Baştürk, B., C. A Powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) algorithm, J. Glob. Optim., 39:459–471,DOI 10.1007/s10898-007-9149-x, 2007.

[63] Karaboğa, D., Baştürk, B. On the Performance of Artificial Bee Colony (ABC) Algorithm, Applied Soft Computing 8, pp:687–697, 2008.

[64] Abraham, A., Evolutionary Computation, In: Handbook for Measurement, Systems Design, Peter Sydenham and Richard Thorn (Eds.), John Wiley and Sons Ltd., London, ISBN: 0-470-02143-8, pp. 920–931, 2005.

[65] Denysiuk, R., Costa, L., Santo, I.E. Generalized Multiobjective Evolutionary Algorithm Guided by Descent Directions, J Math Model Algor,13:387–403, Springer. 2014.

[66] Liu,R., Fan, J., Jiao, L. Integration of improved predictive model and adaptive differential evolution based dynamic multi-objective evolutionary optimization algorithm, Appl Intell. DOI 10.1007/ s10489-014-0625-y, Springer, 2015.

[67] Bureerat, S., Srisomporn, S. Optimum Plate-Fin Heat Sinks By Using A Multi-Objective Evolutionary Algorithm, Engineering Optimization, Taylor & Francis Group, LLC, Vol. 42, No. 4,pp. 305–323, 2010.

[68] Bhuvana, J., Aravindan, C. Memetic algorithm with Preferential Local Search using adaptive weights for multi-objective optimization problems, Soft Comput.DOI 10.1007/s00500-015-1593-9, Springer, 2015.

[69] Bolourchi, A., Masri, S., Aldraihem, F. O. Development and application of computational intelligence approaches for the identification of complex nonlinear systems, Nonlinear Dyn, 79:765–786, Springer, 2015.

[70] Schonfeld, J., Louis, S.J. The effect of selection on the development of mutational robustness, Evolutionary Computation, CEC, IEEE Congress on, pp.4714-4721, 2007.

[71] Fogel, D.B., Evolutionary Computation: Toward a New Philosophy of Machine Intelligence. IEEE Press, Piscataway, NJ, Second edition, 1999.

[72] Sedki, A, Ouazar D., El Mazoud, E. Evolving neural network using real coded genetic algorithm for daily rainfall–runoff forecasting", Expert Systems with Applications 36, sayfalar:4523–4527, 2009.

[73] Kiranyaz, S., Türker, I., Yıldırım,A., Gabbouja, M. Evolutionary artificial neural networks by multi-dimensional particle swarm optimization, Neural Networks, 22,1448-1462, 2009.

[74] Hacıoğlu, A. Hızlı Evrimsel Eniyileme İçin Yapay Sinir Ağı Kullanılması", Havacılık ve Uzay Teknolojileri Dergisi, cilt 2, sayı 3 (1-8), 2006.

[75] Zhang,H., Ishikawa,M. The performance verification of an evolutionary canonical particle swarm optimizer, Neural Networks, 23, 510-511, 2010.

[76] Ashena, R., Moghadasi, J. Bottom hole pressure estimation using evolved neural networks by real coded ant colony optimization and genetic algorithm, Journal of Petroleum Science and Engineering,77, ss: 375–385, 2011.

[77] Castellani, M., Rowlands,H. Evolutionary Artificial Neural Network Design and Training for wood veneer classification", Engineering Applications of Artificial Intelligence volume 22, pages 732–741, 2009.

[78] Rocha, M., Cortez P., Neves, J. Evolution of neural networks for classification and regression, Neurocomputing journal,70,ss. 2809–2816, 2007.

[79] Tıan, L., Noore, A. Evolutionary neural network modeling for software cumulative failure time prediction, Reliability Engineering and System Safety, 87, ss:45–51, 2005.

[80] Bayhan, M. Alkaya, A. The Classification of a Simulation Data of a Servo System via Evolutionary Artificial Neural Networks, Lecture Notes in Computer Science, volume 5755, Emerging Intelligent Computing Technology and Applications with aspects of Artificial Intelligence, Pages 48-54, 2009.

[81] Abbass, H. A. An Evolutionary Artificial Neural Networks Approach for Breast Cancer Diagnosis. Artificial Intelligence in Medicine, 25(3):265–281, 2002.

[82] Yao, X. Evolving artificial neural networks. Proceedings of the IEEE, IEEE Press, USA, 87(9):1423–1447, 1999.

[83] Li, B., Nye, T.J., Metzger, D.R. Multi Objective Optimization of Forming Parameters for Tube Hydroforming Process based on the Taguchi Method, International Journal of Advanced Manufacturing Technologies, 28: 23–30, 2006.

[84] Ahlburg, D. Simple versus Complex Models: Evaluation, Accuracy, and Combining. Mathematical Population Studies 5: 281-90, 1995.

[85] Campbell, P. Evaluating Forecast Error in State Population Projections using Census 2000 Counts. Population Division Working Paper Series No. 57. Washington, D.C.: U. S. Census Bureau, 2002.

[86] Hyndman, R. Koehler, A., Another Look at Measures of Forecast Accuracy. International Journal of Forecasting 22: 679-688, 2006.

[87] Rayer, S. Population Forecast Accuracy: Does the Choice of Summary Measure of Error Matter? Population Research and Policy Review 26: 163-184, 2007.

[88] Karaboğa, D. An Idea Based On Honey Bee Swarm For Numerical Optimization, Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[89] Küçüksille, E.U., Tokmak, M. Yapay Arı Kolonisi Algoritması Kullanarak Otomatik Ders Çizelgeleme, Süleyman Demirel Üniversitesi, Fen Bilimleri Enstitüsü Dergisi, 15-3,ss.203-210, 2011.

[90] Banharnsakun, A., Achalakul, T., Sirinaovakul, B. Artificial Bee Colony Algorithm on Distributed Environments, Second World Congress on Nature and Biologically Inspired Computing, 13-18, 2010.

[91] Alam, M. S., Ul Kabir, M. W., Islam, M. M. Self-Adaptation of Mutation Step Size in Artificial Bee Colony Algorithm for Continuous Function Optimization, 13th International Conference on Computer and Information Technology, 69-74, 2010.

[92] Karaboğa, D., Görkemli, B. A Combinatorial Artificial Bee Colony Algorithm for Traveling Salesman Problem, International Symposium on Innovations in Intelligent Systems and Applications, 50-53, 2011.

[93] Pandey, S., Kumar, S., Enhanced Artificial Bee Colony Algorithm and It's Application to Travelling Salesman Problem, HCTL Open Int. J. of Technology Innovations and Research, Volume 2, 2013.

[94] Guo, P., Cheng, W., Liang, J. Global Artificial Bee Colony Search Algorithm for Numerical Function Optimization, Seventh International Conference on Natural Computation, vol. 3, 1280-1283, 2011.

[95] Ning, A., Zhang, X. A Speech Recognition System Based on Fuzzy Neural Network Trained by Artificial Bee Colony Algorithm, International Conference on Electronics, Communications and Control, 2488-2491, 9-11, 2011.

[96] Jadhav, H. T., Patel, J., Sharma, U., Roy, R. An Elitist Artificial Bee Colony Algorithm for Combined Economic Emission Dispatch Incorporating Wind Power, 2nd International Conference on Computer and Communication Technology, 640-645, 2011.

[97] Çelik, M., Karaboğa, D., Köylü, F. Artificial Bee Colony Data Miner (ABC-Miner), Innovations in Intelligent Systems and Applications (INISTA), 96-100, 2011.

[98] Zakaria, N. M. A., Rosni, A. Artificial Bee Colony Optimization Algorithm with Crossover Operator for Protein Structure Prediction, Soft Computing Applications and Intelligent Systems Communications in Computer and Information Science Volume 378, pp 147-157, 2013.

[99] Khaze, S.R., Maleki, I., Hojjatkhah, S., Bagherinia, A. Evaluation the Efficiency of Artificial Bee Colony and the Firefly Algorithm in Solving the Continuous Optimization Problem, International Journal on Computational Sciences & Applications (IJCSA) Vol.3, No.4, 2013.

[100] Brajevic, I. Crossover-Based Artificial Bee Colony Algorithm for Constrained Optimization Problems, Neural Comput & Applic. DOI 10.1007/s00521-015-1826-y, 2015.

[101] Maheshwari, V., Dutta, U. Comparative Study of Different Modified Artificial Bee Colony Algorithm with Proposed ABC Algorithm, International Journal of Soft Computing and Engineering (IJSCE),ISSN: 2231-2307, Volume-3, Issue-6, 2014.

[102] Hedayatzadeh, R., Hasanizadeh, B., Akbari, R., Ziarati, K. A Multi-Objective Artificial Bee Colony for Optimizing Multi-Objective Problems, 3rd International Conference on Advanced Computer Theory and Engineering, Vol. 5, V5-277-V5-281, 2010.

[103] Sulaiman, N., Mohamad-Saleh, J., Abro, A.G. New Enhanced Artificial Bee Colony (JA-ABC5) Algorithm with Application for Reactive Power Optimization, The Scientific World Journal, Vol. 2015, Article ID 396189, 2015.

[104] Ayat, Z., Sangar, A.B. A Hybrid Approach Artificial Bee Colony Optimization and K-Means Clustering for Software Cost Estimation, Journal of Scientific Research and Development 2 (4): 250-255, 2015.

[105] Eke, I., Taplamacıoğlu, M.C., Kocaarslan, I. Yapay Arı Kolonisi Algoritması Tabanlı Kararlı Güç Sistemi Dengeleyicisi Tasarımı, J. Fac. Eng. Arch.Vol 26, No 3, 683-690, 2011.

[106] Yıldız, A.R. Yapay Arı Koloni Algoritması ile Taşıt Salıncak Kolunun Optimum Boyutlarının Bulunması, 3. Ulusal Tasarım İmalat ve Analiz Kongresi 29-30 Kasım, 2012.

[107] Özyön, S., Yaşar, C., Ozcan, G., Temurtaş, H. Çevresel Ekonomik Güç Dağıtım Problemlerine Yapay Arı Koloni Algoritması (ABC) Yaklaşımı, http://web.firat.edu.tr/feeb/kitap/C12/119.pdf, Fırat Üniversitesi-Elazığ, 2015.

[108] Sevim, O., Sönmez, M., Sezer, R. Yapay Arı Koloni Algoritması İle Uzay Çelik Yapıların Optimum Tasarımı, 2. Uluslararası Mühendislik ve Bilim Alanında Yenilikçi Teknolojiler Sempozyumu, Karabük Üniversitesi, 18-20 Haziran, 2014.

[109] Karaboğa, D., Akay B. Artificial Bee Colony (ABC) Algorithm on Training Artificial Neural Networks, Signal Processing and Communications Applications, 1-4, Eskişehir, Türkiye, 11-13 Haziran 2007.

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

[111] Tokmak, M. Yapay Arı Kolonisi Algoritması ile Ders Çizelgeleme Probleminin Çözümü, Yüksek Lisans Tezi, Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü, Isparta, 2011.

[112] Dornhaus, A., Klugl, F., Puppe, F., Tautz, J. Task Selection in Honeybees-Experiments Using Multi-Agent Simulation. in Proc of Gwal'98, 1998.

[113] Gruter C., Farina M. The Honeybee Waggle Dance: Can We Follow The Steps?, Trends in Ecology & Evolution, 24 (5), 242-247, 2009.

[114] Gölcük, I, Baykasoğlu, A., Madenoğlu, F.S. Kril Sürüsü Algoritması ile Atölye Çizelgeleme, DEU Mühendislik Fakültesi, Mühendislik bilimleri dergisi, Cilt: 16 Sayı: 48 sh. 61-75 Eylül, 2014.

[115] Anandaraman, C. An Improved Sheep Flock Heredity Algorithm for Job Shop Scheduling and Flow Shop Scheduling Problems, International Journal of Industrial Engineering Computations, Cilt 2, No. 4, s:749-764, 2011.

[116] Luh, G. C., Chueh C. H. A Multi-Modal Immune Algorithm for the Job-Shop Scheduling Problem, Information Sciences, Cilt 179, No. 10, s.1516-1532, 2009.

[117] Brucker, P., Jurisch B., Sievers B. A Branch and Bound Algorithm for the Job-Shop Scheduling Problem, Discrete Applied Mathematics, Cilt 49, No. 3, s.107-127, 1994.

[118] Wang, G. G., Guo L., Gandomi, A. H., Alavi A. H., Duan H. Simulated Annealing-Based Krill Herd Algorithm for Global Optimization, Abstract and Applied Analysis, Cilt 11, 2013.

[119] Zhang, R., Cheng, W. A Hybrid Immune Simulated Annealing Algorithm For The Job Shop Scheduling Problem, Applied Soft Computing 10,79–89, 2010.

[120] Watson, J.P., Beck, J.C., Howe, A. E., Whitley, L. D. Problem Difficulty for Tabu Search in Job-Shop Scheduling, Artificial Intelligence, 143, ss.189–217, 2003.

[121] Nowicki, E., Smutnicki, C. A Fast Taboo Search Algorithm for the Job Shop Problem, Management Science, Cilt 42, No. 6, 797-813, 1996.

[122] Zhang, J., Hu, X., Tan, X., Zhong, J.H. , Huang., Q. Implementation of an Ant Colony Optimization technique for job shop scheduling problem, Transactions of the Institute of Measurement and Control 28, 1,ss. 93/108, 2006.

[123] Abidia, M. H., Al-Harkan, I., El-Tamimi, A. M., Al-Ahmari, A.M., Nasr, E. S. A. Ant Colony Optimization for Job Shop Scheduling to Minimize the Total Weighted Tardiness, Proceedings of the 2014 Industrial and Systems Engineering Research Conference, 2014.

[124] Surekha, P., Sümathi, S. Solving Fuzzy Based Job Shop Scheduling Problems using Ga and Aco, Journal of Emerging Trends in Computing and Information Sciences, Vol 1, no. 2, Ekim, 2010.

[125] Hasan, S. M. K., Sarker R., Essam,D., Cornforth D. Memetic Algorithms for Solving Job-Shop Scheduling Problems, Memetic Computing, Cilt 1, No. 1, 69-83, 2009.

[126] Zelenka, J. Application of Particle Swarm Optimization in Job-Shop Scheduling Problem in the Recycling Process, CINTI 2010 • 11th IEEE International Symposium on Computational Intelligence and Informatics, 18–20 November, Budapest, Hungary, 2010.

[127] Tang, J., Zhang, G., Lin, B., Zhang, B. A Hybrid PSO/GA Algorithm for Job Shop Scheduling Problem, Advances in Swarm Intelligence Lecture Notes in Computer Science, Vol. 6145, ss: 566-573, 2010.

[128] Zufeng, Z. Research On Job-Shop Scheduling Problem Based On Improved Particle Swarm Optimization, Journal of Theoretical and Applied Information Technology, Vol. 47, No.2, 2013.

[129] Zhang, R., Cheng, W. An Artificial Bee Colony Algorithm for the Job Shop Scheduling Problem with Random Processing Times, Entropy, 13, 1708-1729; doi:10.3390/e13091708, 2011.

[130] Zhang, R., Song, S., Cheng, W. A Hybrid Artificial Bee Colony Algorithm for the Job Shop Scheduling Problem, Int. J. Production Economics 141,ss: 167–178, 2013.

[131] Gupta, M., Sharma, G. An Efficient Modified Artificial Bee Colony Algorithm for Job Scheduling Problem, International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, vol.1, issue-6, 2012.

[132] Baysal, M.E., Durmaz, T., Sarucan, A., Engin, O. Açık Atölye Tipi Çizelgeleme Problemlerinin Paralel Kanguru Algoritması ile Çözümü, Journal of the Faculty of Engineering and Architecture of Gazi University, Cilt 27, No 4, 855-864, 2012.

[133] Ouaarab, A., Ahiod, B., Yang, X.S., Abbad, M. Discrete Cuckoo Search Algorithm for Job Shop Scheduling Problem, 2014 IEEE International Symposium on Intelligent Control (ISIC) Part of 2014 IEEE Multi-conference on Systems and Control, 2014.

[134] Singh, S., Singh, K.P., Cuckoo Search Optimization for Job Shop Scheduling Problem, Proceedings of Fourth International Conference on Soft Computing for Problem Solving Advances in Intelligent Systems and Computing Volume 335, ss. 99-111, 2015.

[135] Ala'a, A.S., Muhannad, A.S. Hybrid Algorithm using Genetic Algorithm and Cuckoo Search Algorithm for Job Shop Scheduling Problem, IJCSI International Journal of Computer Science Issues, Vol. 12, Issue 2, Mart, 2015.

[136] Karthıkeyan, S., Asokan, P.Nickolas, S. A Hybrid Discrete Firefly Algorithm for Multi-Objective Flexible Job Shop Scheduling Problem with Limited Resource Constraints, Int J Adv Manuf Technol. 72:1567–1579, 2014.

[137] Udaiyakumar, K.C., Chandrasekaran, M. Application of Firefly Algorithm in Job Shop Scheduling Problem for Minimization of Makespan, 12th Global Congress On Manufacturıng and Management, Gcmm 2014, Procedia Engineering 97, 1798 – 1807, 2014.

[138] Zhu, K., Jiang, M. The Optimization of Job Shop Scheduling Problem Based on Artificial Fish Swarm Algorithm with Tabu Search Strategy, Third International Workshop on Advanced Computational Intelligence, 2010.

[139] Bouzidi, A., Riffi, M.E. Cat Swarm Optimization to Solve Flow Shop Scheduling Problem, Journal of Theoretical and Applied Information Technology, Vol.72 No.2, 2015.

[140] Li, Y., Chen, Y. A Genetic Algorithm for Job-Shop Scheduling, Journal of Software, vol. 5, no. 3, Mart, 2010.

[141] Goncalves, J. F., Resende, M. G. C., An Extended Akers Graphical Method with a Biased Random-Key Genetic Algorithm for Job-Shop Scheduling, International Transactions in Operational Research, Cilt 21, No. 2, s.215-246, 2014.

[142] Demir, Y., Işleyen, S.K. An Effective Genetic Algorithm for Flexible Job-Shop Scheduling with Overlapping in Operations, International Journal of Production Research, Vol. 52, No. 13, 3905–3921, 2014.

[143] Wisittipanich, W., Kachitvichyanukul, V. A Pareto-Based Differential Evolution Algorithm for Multi-Objective Job Shop Scheduling Problems, Proceedings of the Institute of Industrial Engineers Asian Conference 2013, pp 1117-1125, 2013.

[144] Zhanga, R., Song, S., Wub, C. A Hybrid Differential Evolution Algorithm For Job Shop Scheduling Problems With Expected Total Tardiness Criterion, Applied Soft Computing vol.13, pp.1448–1458, 2013.

[145] Hosseinabadi, A.A.R., Siar, H., Shamshirband, S., Shojafar, M., Nasir, M.H.N.M. Using The Gravitational Emulation Local Search Algorithm to Solve the Multi-Objective Flexible Dynamic Job Shop Scheduling Problem in Small And Medium Enterprises, Ann. Oper. Res. vol. 229; pp.451–474, 2015.

[146] Barzegar, B., Motameni, H., Bozorgi, H. Solving Flexible Job-Shop Scheduling Problem Using Gravitational Search Algorithm and Colored Petri Net, Journal of Applied Mathematics, 2012.

[147] Gao, K. Z. , Suganthan, P. N., Pan, Q. K. , Chua, T. J., Cai, T. X., Chong, C. S. Discrete Harmony Search Algorithm For Flexible Job Shop Scheduling Problem With Multiple Objectives, Journal of Intell. Manuf., DOI, 10.1007/s10845-014-0869-8, Springer, 2014.

[148] Hymavathi, M., Rao, C.S.P. Improved Music Based Harmony Search Algorithm (Imbhsa) for Solving Job Shop Scheduling Problems (Jssps), 5 th International & 26th All India Manufacturing Technology, Design and Research Conference (AIMTDR 2014) December 12th–14th, 2014.

[149] Yuan, Y., Xu, H.,Yang, J. A Hybrid Harmony Search Algorithm for the Flexible Job Shop Scheduling Problem, Applied Soft Computing v.13, pp.3259–3272, 2013.

[150] Rossi, A., Boschi, E. A Hybrid Heuristic to Solve the Parallel Machines Job-Shop Scheduling Problem, Advances in Engineering Software,v.40, ss.118–127, 2009.

[151] Huang, M., Wu, T., Liang, X. GA-ACO in Job-Shop Schedule Problem Research, Computational Intelligence and Intelligent Systems Communications in Computer and Information Science Volume 107, pp. 226-233, 2010.

[152] Silva, C.A., Faria, J.M. , Abrantes, P., Sousa, J.M.C., Surico, M., Naso, D. Concrete Delivery Using a Combination of GA and ACO, Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference, 2005.

[153] Liua, L.L., Hua, R.S., Hub, X.P., Zhaoa, G.P., Wang, S. A Hybrid PSO-GA Algorithm for Job Shop Scheduling in Machine Tool Production, International Journal of Production Research, 2014.

[154] Yu, M., Zhang, Y., Chen, K., Zhang, D. Integration of Process Planning and Scheduling Using a Hybrid GA/PSO Algorithm, Int. J. Adv. Manuf. Technol.,78:583–592, 2015.

[155] Zhang, C., Li, P., Rao, Y., Li, S. A New Hybrid GA/SA Algorithm for the Job Shop Scheduling Problem, Evolutionary Computation in Combinatorial Optimization Lecture Notes in Computer Science Volume 3448, pp 246-259, 2005.

[156] Jiang, J., Wen, M., Ma, K., Long, X., Li, J. Hybrid Genetic Algorithm for Flexible Job-shop Scheduling with Multi-objective, Journal of Information & Computational Science 8: 11, 2197–2205, 2011.

[157] Tamilarasi, A., Kumar, T. An Enhanced Genetic Algorithm with Simulated Annealing for Job-Shop Scheduling, International Journal of Engineering, Science and Technology Vol. 2, No. 1, pp. 144-151, 2010.

[158] Surekha, P., Sumathi, S. PSO and ACO Based Approach for Solving Combinatorial Fuzzy Job Shop Scheduling Problem, Int. J. Comp. Tech. Appl., Vol 2 (1), 112-120, 2011.

[159] Kuo, R. J., Cheng, W.C. Hybrid Meta-Heuristic Algorithm for Job Shop Scheduling with Due Date Time Window and Release Time, Int J Adv Manuf Technol (2013) 67:59–71, 2013.

[160] Li, J.Q., Pan, Q.K., Xie, S.X., Wang,Y.T. An Effective Hybrid Particle Swarm Optimization Algorithm for Flexible Jobshop Scheduling Problem, IJAEA, Vol.1, Issue.3, pp.9-13, 2008.

[161] Li, J.Q., Pan,Y.X. A Hybrid Discrete Particle Swarm Optimization Algorithm for Solving Fuzzy Job Shop Scheduling Problem, Int J Adv Manuf Technol, 66:583–596, 2013.

[162] Raju, R., Babukarthik, R.G., Dhavachelvan, P. Hybrid Ant Colony Optimization and Cuckoo Search Algorithm for Job Scheduling, Advances in Computing and Information Technology Advances in Intelligent Systems and Computing Volume 177, pp 491-501, 2013.

[163] Babukartik, R.G., Dhavachelvan, P. Hybrid Algorithm using the Advantage of ACO and Cuckoo Search for Job Scheduling, International Journal of Information Technology Convergence and Services (IJITCS) Vol.2, No.4, 2012.

[164] Divya, P., Narendhar, S., Ravibabu, V. An Enhanced Bio-Stimulated Methodology to Resolve Shop Scheduling Problems, International Journal of Ambient Systems and Applications (IJASA) Vol.1, No.2, 2013.

[165] Narendhar, S., Amudha, T. A Hybrid Bacterial Foraging Algorithm for Solving Job Shop Scheduling Problems, International Journal of Programming Languages and Applications (IJPLA) Vol.2, No.4, 2012.

[166] Shivakumar, B.L., Amudha, T. A Hybrid Bacterial Swarming Methodology for Job Shop Scheduling Environment, Global Journal of Computer Science and Technology Hardware & Computation, Volume 12, Issue 10, Version 1.0, 2012.

[167] Hymavathi,M., Rao, C.S.P. Development of New Paradigms for Job Shop Scheduling Problems, 5 th International & 26th All India Manufacturing Technology, Design and Research Conference, 2014.

[168] Serbencu, A., Minzu, V., Cernega, D., Serbencu, A. A Hybrid Metaheuristic for Solving Single Machine Scheduling Problem, ICINCO 2009, The 6th International Conference on Informatics in Control, Automation and Robotics, Intelligent Control Systems and Optimization, 2009.

[169] Rohaninejad, M., Kheirkhah, A.S., Nouri,B.V., Fattahi, P. Two Hybrid Tabu Search–Firefly Algorithms for the Capacitated Job Shop Scheduling Problem with Sequence-Dependent Setup Cost, International Journal of Computer Integrated Manufacturing, Vol. 28, No. 5, 470–487, 2015.

[170] Barzegar, B., Motameni, H. Solving Flexible Job-Shop Scheduling Problem using Hybrid Algorithm Based on Gravitational Search Algorithm and Particle Swarm Optimization, Journal of Advances in Computer Research, Vol. 4, No. 3, pp: 69-81, 2013.

[171] Sun, L. An Effective PSO and AIS-Based Hybrid Intelligent Algorithm for Job-Shop Scheduling, IEEE Transactions on Systems Man and Cybernetics - Part A Systems and Humans, 2008.

[172] Qıu, X., Henry Y. K. An AIS-Based Hybrid Algorithm for Static Job Shop Scheduling Problem, J Intell Manuf.,25:489–503, 2014.

[173] Guo, P., Cheng, W., Wang, Y. Parallel Machine Scheduling with Step Deteriorating Jobs and Setup Times by a Hybrid Discrete Cuckoo Search Algorithm, Journal of Engineering Optimization, 2013.

[174] Bilge, U., Kurtulan, M., Kıraç, F. A Tabu Search Algorithm for the Single Machine Total Weighted Tardiness Problem, European Journal of Operational Research 176,1423–1435, 2007.

[175] Ergün, O., Orlin, J.B. A Fast Algorithm for Searching Insertion, Swap, and Twist Neighborhoods for the Single Machine Total Weighted Tardiness Problem, Massachusetts Institute of Technology, Cambridge, 2004.

[176] Moghaddam, S.K., Khodadadi, F., Maleki, R.E., Movaghar, A. A Hybrid Genetic Algorithm and Variable Neighborhood Search for Task Scheduling Problem in Grid Environment, Procedia Engineering 29, 3808 – 3814, 2012.

[177] Wen, Y., Xu, H., Yang, J. A Heuristic-Based Hybrid Genetic-Variable Neighborhood Search Algorithm for Task Scheduling in Heterogeneous Multiprocessor System, Information Sciences 181, 567–581, 2011.

[178] URL : http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/jobshop1.txt. Erişim tarihi: Ocak 15, 2015.

[179] Fernandes, S. Lourenco, H.R. A Simple Optimised Search Heuristic for the Job Shop Scheduling Problem, Recent Advances in Evolutionary Computation for Combinatorial Optimization Studies in Computational Intelligence, Springer, Vol.153, 203-218, 2008.

[180] Tabachnick, B.G., Fidell, L.S., Using Multivariate Statistics (sixth edition), Pearson, Boston, 2013.

[181] George, D., Mallery, M., SPSS for Windows Step by Step: A Simple Guide and Reference, 17.0 Update (10a edition), Pearson, Boston, 2010.

[182] Yazıd M., Xıaolan, X. Multiresource Shop Scheduling With Resource Flexibility and Blocking, IEEE Transactions on Automation Science and Engineerıng, Vol. 8, No. 1, January, 2011.

[183] Marco, P., Darıo, P. An Iterated Greedy Metaheuristic for the Blocking Job Shop Scheduling Problem, Rome Tre University, Dipartimento di Ingegneria, 208, 2013.

[184] Wojciech, B., Zdziław, H., Mariusz, U., Mieczysław, W., Solving the Flexible Job Shop Problem on Multi-GPU, International Conference on Computational Science, ICCS 2012, Procedia Computer Science,9, 2020-2023, 2012.

[185] Framinan, J.,M., Leisten, R., García, R.,R., Manufacturing Scheduling Systems : An Integrated View on Models, Methods and Tools, Springer, 2014.

[186] Chiong, R., Dhakal, S., Natural Intelligence for Scheduling, Planning and Packing Problems, Book, Studies in Computational Intelligence, V. 250, 2009.

[187] Cirne, W., Desai, N., Job Scheduling Strategies for Parallel Processing: 18th International Workshop, JSSPP 2014, Phoenix, AZ, USA, May 23, 2014, Revised Selected Papers (Lecture Notes in Computer Science) 2015th Edition, 2015.

[188] Blum, C., Sampels, M. An Ant Colony Optimization Algorithm for Shop Scheduling Problems, Journal of Mathematical Modelling and Algorithms, 3:285-308, 2004.

[189] Taşgetiren,M.F., Şevkli, M., Liang,Y.C., Yenisey, M. A Particle Swarm Optimization and Differential Evolution Algorithms for Job Shop Scheduling Problem, International Journal of Operations Research Vol. 3, No. 2, 120-135, 2006.

EKLER

EK A: Önerilen ABC-EA bütünleşik metodunun C_{max}'ı minimize etmek için atölye tipi çizelgeleme problemleriyle ilgili olarak çözümü için Eclipse geliştirme ortamında Java yazılım dilindeki kodları

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.Random;
import java.util.Scanner;
import java.io.FileInputStream;
import java.io.FileNotFoundException;

import org.jfree.ui.RefineryUtilities;

public class textReader {

	static Random generator = new Random();

	static int GlobalMin ;
	static int VNScounter = 0 ;
	static int ScheduleCounter;
	static int Cmin;
	static int machines;
	static int jobs;

	static List<Jobs> jobList = new ArrayList<Jobs>();
	static List<machineClass> machineList = new ArrayList<machineClass>();
	static List<machineClass> cloneMAchineListe = new ArrayList<
machineClass>();
	static List<Jobs> cLonejobList ;
	static List<Jobs> cLonejobList_1 ;
	static List<machineClass> cloneMAchineListe_1 = new ArrayList<
machineClass>();
	static List<Jobs> GlobalBestcLonejobList ;
	static List<machineClass> GlobalcloneMAchineListe = new ArrayList<
machineClass>();
	static List<Jobs> GlobalcLonejobList ;
	static List<Jobs> bestSchedule = null ;
	static List<Jobs> crossSchedule = null ;
	static List<Integer> GlobalBest = new ArrayList<Integer>();
	static List<Integer> GlobalBestCount = new ArrayList<Integer>();
	static List<String> GlobalBest_Output = new ArrayList<String>();
```

```
public static void main(String[] args) {

        ArrayList<List<Jobs>> jobLists = new ArrayList<List<Jobs>>();
        ArrayList<List<machineClass>> MachineLists = new ArrayList<List<
machineClass>>();

        List<Integer> cMin = new ArrayList<Integer>();

        System.out.println("baslangic");
        Scanner inputStream = null;
        try {
                inputStream = new Scanner(new FileInputStream
("instance.txt"));
        } catch (FileNotFoundException e) {

                System.out.println("file could not be found");
        }

        // textFile okunup input Stream'a atiliyor

        InputScanner.readFromFile(inputStream);

        System.out.println("bitis");

        for (Jobs J : jobList) {
                for (Operation O : J.OperationList) {

         // joblar hangi makinede islenecekse oraya gönderiliyor

                machineClass M = getMachine(O.machine, machineList);

                        if (M.machineName == O.machine) {
                                M.NewSchedule.addToMachineSchedule(O,

M.NewSchedule.MachineSchedule);
                        }
                }
        }

        int t = 0;
        int order[] = new int[jobs];

        for (int g = 0 ; g < order.length ;g++){

                order[g] = g ;
        }

        System.out.println("job" +jobs);
```

```
System.out.println("machines" +machines);
System.out.println("order.length" +order.length);


ScheduleCounter = machines*jobs;

int roll = 0;

while(roll < ScheduleCounter){
        int y = generator.nextInt(order.length);
        int yx = generator.nextInt(order.length);

        if(y == yx ){
while(y == yx ){
        y = generator.nextInt(order.length);
        yx = generator.nextInt(order.length);
}

}

        int temp = order[y];
        order[y] = order[yx];
        order[yx] = temp;
        roll++;
        }

 //order[] = { 8 , 5 , 3 , 1 , 6,4,0,9,2,7 };
//int order[] = { 7, 2, 9, 0, 4, 6, 1, 3, 5, 8 };
int Ccurrent;
Print print = new Print();

for (int g = 0 ; g < order.length ;g++){

        System.out.print( order[g]);
        System.out.print( " ");
}

Cmin = 99999;

ScheduleCounter = machines*jobs;
SortingClass sortingclass = new SortingClass();

// buradaki t initial population.
while (t < 500){

        ActiveScheduleClass.counter = 0;
        resetSchedule(jobList);
        ResetScheduleList(machineList);
```

```
                ResetActiveSchedule(machineList);
                int e = 0;

                while (ActiveScheduleClass.counter != ScheduleCounter) {

                        int y = order[e];

                        Jobs JB = jobList.get(y);

        ActiveScheduleClass.ActiveSchedule(JB, jobList, machineList);
                        e++;

                }
                roll =0;

                while(roll < 6){
                int y = generator.nextInt(order.length);
                int yx = generator.nextInt(order.length);

                if(y == yx ){
          while(y == yx ){
                y = generator.nextInt(order.length);
                yx = generator.nextInt(order.length);
          }
       }
                int temp = order[y];
                order[y] = order[yx];
                order[yx] = temp;
                roll++;
                }
Ccurrent = Fitness.CalculateCmaX(machineList);

                //if (Ccurrent < Cmin) {
                        bestSchedule = cloneJobList(jobList);
                        Cmin = Ccurrent;
                        jobLists.add(bestSchedule);
                        cMin.add(Cmin);
                        System.out.println("t = " + t + " Cmin ;" + Cmin);
                //}

                t++;
          }
          Cmin = 99999;

          System.out.println("**********jobLists"+jobLists.size() );

          System.out.println("**********cMin"+cMin.size() );
```

```
CrossGen cross = new CrossGen();

int counter = 0;

int size = jobLists.size()/ 25;
while(counter < size){

        int index_1 = generator.nextInt(jobLists.size());
        int index_2 = generator.nextInt(jobLists.size());

        if(index_1 == index_2 ){
        while(index_1 == index_2 ){
                index_1 = generator.nextInt(jobLists.size()-1);
                        index_2 = generator.nextInt(jobLists.size()-1);
        }
      }
crossSchedule = cross.CrossOver(jobLists.get(index_1), jobLists.get(index_2));
        jobLists.add(crossSchedule);
         counter++;
        }
counter = 0;

while(counter < size){

        int index_1 = generator.nextInt(jobLists.size());
        int index_2 = generator.nextInt(jobLists.size());

        if(index_1 == index_2 ){
        while(index_1 == index_2 ){
                index_1 = generator.nextInt(jobLists.size()-1);
                        index_2 = generator.nextInt(jobLists.size()-1);
        }
      }
crossSchedule = cross.CrossOverWorstgen(jobLists.get(index_1),
jobLists.get(index_2));
        jobLists.add(crossSchedule);
          counter++;
       }
counter = 0;
while(counter < size){

        int index_1 = generator.nextInt(jobLists.size());
        int index_2 = generator.nextInt(jobLists.size());

        if(index_1 == index_2 ){
        while(index_1 == index_2 ){
                index_1 = generator.nextInt(jobLists.size()-1);
                         index_2 = generator.nextInt(jobLists.size()-1);
```

```
                    }
                }
            crossSchedule = cross.CrossOverBestgen(jobLists.get(index_1),
jobLists.get(index_2));
                    jobLists.add(crossSchedule);
                     counter++;
            }

            for (List<Jobs> JL : jobLists) {

                    cloneMAchineListe = cloneMachineListFromJoblist(JL);

            System.out.println("CMax (CalculateCmaXFromJobList) JL = " +
Fitness.CalculateCmaXFromJobList(JL));
            }
            System.out.println("***************jobLists"+jobLists.size() );

            GlobalMin = Cmin;
            Print.makeGap();

            GlobalBestcLonejobList  = cloneJobList(bestSchedule);

            for (List<Jobs> JL : jobLists) {

                    cloneMAchineListe = cloneMachineListFromJoblist(JL);
                    MachineLists.add(cloneMAchineListe);
            }
            for (List<machineClass> ML : MachineLists) {

                    sortingclass.ClassSorting(ML);

            /*for (machineClass M : ML) {
                    System.out.println("Before Maximum C for ML  machine "
                                    + M.machineName
                                    + " = "
                                    +
M.NewSchedule.scheduleList.get(M.NewSchedule.scheduleList.size()                -
1).finish_time);
            }System.out.println("");*/

            }

            int ABCcounter = 0;

            ABC artificialBee = new ABC();

            while(ABCcounter < 1){
```

```
            System.out.println(" *artificialBee.ArtificialBe_EmployedBeesPhase*
;" + ABCcounter );

            artificialBee.ArtificialBe_EmployedBeesPhase(jobLists);

            System.out.println("   *artificialBee.ArtificialBeeOnlookerBeesPhase*
;" + ABCcounter  );
            artificialBee.ArtificialBeeOnlookerBeesPhase(jobLists);
            System.out.println(" *artificialBee.ArtificialBeeScoutBeesPhase* ;" +
ABCcounter  );
            artificialBee.ArtificialBeeScoutBeesPhase(jobLists);
            ABCcounter++;
            }

            MachineLists.clear();

            for (List<Jobs> JL : jobLists) {

                  cloneMAchineListe = cloneMachineListFromJoblist(JL);
                  MachineLists.add(cloneMAchineListe);
                  //System.out.println("CMax (CalculateCmaXFromJobList) = "
+ Fitness.CalculateCmaXFromJobList(JL));
            }
            int k = 0;

            for (List<machineClass> ML : MachineLists) {
                  sortingclass.ClassSorting(ML);
                  for (machineClass M : ML) {
                        System.out
            .println("Schedule #:" + k + "Before Maximum C for ML  machine "
                                                + M.machineName
                                                + " = "
                                                +
M.NewSchedule.scheduleList

      .get(M.NewSchedule.scheduleList.size() - 1).finish_time);

                  }k++;
                  System.out.println("");
            }

            for(Integer I : GlobalBest ){
                  System.out.println("Global  Best = " + I);
            }
            for(String I : GlobalBest_Output ){
                  System.out.println("GlobalBest_Output= " + I);
            }
```

```
            System.out.println("GlobalMin = " + GlobalMin);

            System.out.println(" *Size of JÄ±b list* ;" + jobLists.size() );

cloneMAchineListe = cloneMachineListFromJoblist(GlobalBestcLonejobList);

            for (machineClass M : cloneMAchineListe) {

                  Collections.sort(M.NewSchedule.scheduleList);
            }
               for (machineClass M : cloneMAchineListe) {
                  System.out.println("Maximum C for  GlobalBestcLonejobList "
                              + M.machineName
                              + " = "
                              +
M.NewSchedule.scheduleList.get(M.NewSchedule.scheduleList
                                          .size() - 1).finish_time);
            }
            Print.makeGap();

            System.out.println("CMax  (CalculateCmaXFromJobList)  =  "  +
Fitness.CalculateCmaXFromJobList(GlobalBestcLonejobList));
            System.out.println("counter = " + ActiveScheduleClass.counter);
            System.out.println("GlobalMin = " + GlobalMin);
            System.out.println("VNScounter = " + VNScounter);
            Print.makeGap();
            XYBarChartDemo7 xybarchartdemo7 = new XYBarChartDemo7(
                  "Global Best  job List Schedule", GlobalBestcLonejobList);
            //xybarchartdemo7.pack();
            RefineryUtilities.centerFrameOnScreen(xybarchartdemo7);
            xybarchartdemo7.setVisible(true);
            inputStream.close();
      }

      public static machineClass getMachine(int machine,
                  List<machineClass> machineListe) {

            for (machineClass MC : machineListe) {
                  if (MC.machineName == machine) {
                        //    System.out.println("machine    name    =    "    +
MC.machineName);

                        return MC;
                  }
            }
            return null;
      }
```

```
	public static void resetSchedule(List<Jobs> jobListII) {
		for (Jobs J : jobListII) {
			for (Operation O : J.OperationList) {
				O.scheduled = false;
			}
		}
	}
	public static void ResetActiveSchedule(List<machineClass> machineList) {
		for (machineClass M : machineList) {
			M.NewSchedule.ActiveSchedule.clear();
		}
	}
	public static void ResetScheduleList(List<machineClass> machineList) {
		for (machineClass M : machineList) {
			M.NewSchedule.scheduleList.clear();
		}
	}
	public static List<machineClass> cloneList(List<machineClass> machineList)
{
		List<machineClass> tempList = new ArrayList<machineClass>();
		for (machineClass M : machineList) {
	machineClass newTempMachine = new machineClass(M.machineName,
					M.NewSchedule.scheduleName);
			for (Operation O : M.NewSchedule.MachineSchedule) {
			Operation newOperation = new Operation(O.machine, O.time,
				O.order, O.JobName, O.finish_time, O.start_time,
					O.scheduled);

	newTempMachine.NewSchedule.addToMachineSchedule(newOperation,
	newTempMachine.NewSchedule.MachineSchedule);
	newTempMachine.NewSchedule.addToScheduleList(newOperation,
newTempMachine.NewSchedule.scheduleList);
			}
			tempList.add(newTempMachine);
		}
		return tempList;
	}
	public static List<Jobs> cloneJobList(List<Jobs> jobListe) {
		List<Jobs> tempJobList = new ArrayList<Jobs>();
		for (Jobs J : jobListe) {
			Jobs newTempJob = new Jobs(J.jobName);
			for (Operation O : J.OperationList) {

			newTempJob.addOperation(O.machine, O.time, O.order,
				newTempJob.jobName, O.finish_time, O.start_time,
					O.scheduled);
			}
			tempJobList.add(newTempJob);
```

```
		}
		return tempJobList;
	}
	public static List<machineClass> cloneMachineListFromJoblist( List<Jobs>
jobListe) {
		List<machineClass> tempList = new ArrayList<machineClass>();
		for (int i = 0; machines > i; i++) {
			String s = "schedule_Machine_";
			String scheduleName = s.concat(Integer.toString(i));
			// System.out.println(scheduleName);
			machineClass newMachine = new machineClass(i,
scheduleName);
			tempList.add(newMachine);
		}
		for (Jobs J : jobListe) {
			for (Operation O : J.OperationList) {
				machineClass M = getMachine(O.machine, tempList);
				if (M.machineName == O.machine) {
					M.NewSchedule.addToMachineSchedule(O,
M.NewSchedule.MachineSchedule);
					M.NewSchedule.addToScheduleList(O,
M.NewSchedule.scheduleList);
				}
			}
		}
		return tempList;
	}
}
```

ÖZGEÇMİŞ

Mümin Özcan, 09.03.1981' de Bandırma' da doğdu. 1992 yılında Bandırma Ali Fahri İşeri İlkokulu'ndan, 1995 yılında Bandırma İ.H.Ortaokulu'ndan ve 1998 yılında Bursa Nilüfer Lisesi'nden mezun oldu. 1999 yılında başladığı Sakarya Üniversitesi Endüstri Mühendisliği Bölümünü 2003 yılında bitirdi. Aynı yıl Dokuz Eylül Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Anabilim Dalı Yüksek Lisans programından 2007 yılında mezun oldu. 2009 yılında Sakarya Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Anabilim Dalında doktora eğitimine başladı. Aynı yıl İzmir Gediz Üniversitesi'nde araştırma görevlisi olarak göreve başladı. 2011 yılında aynı üniversitede öğretim görevlisi kadrosuna atandı. 4 yıldan beri aynı üniversitede yöneylem araştırması, mühendislik istatistiği, araştırma yöntemleri lisans derslerini vermektedir. 2009 yılından beri üzerinde çalıştığı tez konusu ve diğer uzmanlık alanları ile ilgili bilimsel makaleler ve sempozyum bildirileri hazırlayıp sundu.