T.C.
NECMETTİN ERBAKAN ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

AYRIK ORKA YIRTICI ALGORİTMASININ GELİŞTİRİLMESİ

Hamdi KILINÇ

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı

Haziran-2023
KONYA

TEZ KABUL VE ONAYI

Hamdi KILINÇ tarafından hazırlanan "Ayrık Orka Yırtıcı Algoritmasının Geliştirilmesi" adlı tez çalışması … / … / 20…. tarihinde aşağıdaki jüri tarafından oy birliği ile Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Jüri Üyeleri **İmza**

Danışman
Doç. Dr. İlhan İLHAN

Üye
Doç. Dr. Hüseyin HAKLI

Üye
Dr. Öğr. Üyesi Vahit TONGUR

Fen Bilimleri Enstitüsü Yönetim Kurulu'nun …./…/20.. gün ve …….. sayılı kararıyla onaylanmıştır.

Prof. Dr. Şerife Yurdagül KUMCU
FBE Müdürü

TEZ BİLDİRİMİ

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

DECLARATION PAGE

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

Hamdi KILINÇ

Tarih:…./…./……

ÖZET

YÜKSEK LİSANS TEZİ

AYRIK ORKA YIRTICI ALGORİTMASININ GELİŞTİRİLMESİ

Hamdi KILINÇ

Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. İlhan İLHAN

2023, 65 Sayfa

Jüri
Doç. Dr. İlhan İLHAN
Doç. Dr. Hüseyin HAKLI
Dr. Öğr. Üyesi Vahit TONGUR

Gezgin Satıcı Problemi (Traveling Salesman Problem - TSP), günümüzde en sık çalışılan kombinatoryal optimizasyon problemlerinden birisidir. Çizelgeleme, devre tasarlama, fabrikalarda tesislerin yerleşim tasarımı, rota planlama ve baskı devre tasarlama gibi birçok gerçek dünya problemlerini çözmek için kullanılır. Bu nedenle, ayrık optimizasyon yöntemleri alanında çalışan araştırmacılar, onu gerçekçi bir test ortamı olarak kabul ederler ve geliştirdikleri yeni algoritmaların performansını onun üzerinde değerlendirirler.

Bu çalışmada, orka yırtıcı algoritmasının (Orca Predation Algorithm - OPA) ayrık bir versiyonu geliştirilmiş, ayrık orka yırtıcı algoritması (Discrete Orca Predation Algorithm - DOPA) olarak adlandırılmıştır ve TSP'yi çözmek için kullanılmıştır. OPA gibi DOPA da kovalama ve saldırı olmak üzere iki fazdan oluşmaktadır. Kovalama fazında orkalar arasındaki mesafeler Hamming mesafesi ile hesaplanmış, hesaplanan bu değerler kullanılarak hız değerleri elde edilmiştir. Hız değerleri ve 2-opt algoritması kullanılarak orkaların konumları güncellenmiştir. Saldırı fazında orkaların konumları sıralı çaprazlama (Order Crossover - OX1) operatörü ile hesaplanmıştır. Pozisyon ayarlama prosedüründe ise takas (swap) lokal arama operatörü kullanılmıştır. DOPA'nın parametreleri Taguchi istatistiksel yöntemi ile ayarlanmıştır. DOPA 67 iyi bilinen TSP örneği üzerinde test edilmiştir. Ayrıca, DOPA ile diğer güncel dokuz yöntem arasında önemli farklılıklar olup olmadığını kontrol etmek için Friedman ve Wilcoxon işaretli sıra testleri uygulanmıştır. Deneysel sonuçlar DOPA'nın diğer yöntemlere alternatif ve oldukça rekabetçi bir yöntem olduğunu göstermiştir.

Anahtar Kelimeler: Ayrık Algoritma, Hamming Mesafesi, Gezgin Satıcı Problemi, Sıralı Çaprazlama, Orka Yırtıcı Algoritması, 2-Opt

ABSTRACT

MS THESIS

DEVELOPMENT OF THE DISCRETE ORCA PREDATION ALGORITHM

Hamdi KILINÇ

**THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE
OF NECMETTİN ERBAKAN UNIVERSITY
THE DEGREE OF MASTER OF SCIENCE OF PHILOSOPHY
IN COMPUTER ENGINEERING**

Advisor: Assoc. Prof. Dr. İlhan İLHAN

2023, 65 Pages

Jury
Advisor Assoc. Prof. Dr. İlhan İLHAN
Assoc. Prof. Dr. Hüseyin HAKLI
Asst. Prof. Dr. Vahit TONGUR

The Traveling Salesman Problem (TSP) is one of the most frequently studied combinatorial optimization problems today. It is used to solve many real-global problems such as scheduling, circuit design, layout design of facilities in factories, route planning and printed circuit design. Therefore, researchers in the field of discrete optimization methods consider it as a realistic testbed and evaluate the performance of new algorithms on it.

In this study, a discrete version of the Orca Predation Algorithm (OPA), called the Discrete Orca Predation Algorithm (DOPA), is developed and used to solve the TSP. Like OPA, DOPA consists of two phases: chase and attack. In the chase phase, the distances between orcas are calculated using Hamming distance and the velocity values are obtained using these calculated values. The locations of the orcas are updated using the velocity values and the 2 opt algorithm. In the attack phase, the positions of the orcas are calculated with the Order Crossover (OX1) operator. In the position adjustment procedure, the swap local search operator is used. The parameters of the DOPA are tuned by the Taguchi statistical method. DOPA is tested on 67 well-known TSP samples. Furthermore, Friedman and Wilcoxon signed rank tests are applied to check whether there are significant differences between DOPA and the other nine current methods. The experimental results have shown that DOPA is an alternative and highly competitive method to the other methods.

Keywords: Discrete Algorithm, Hamming Distance, Traveling Salesman Problem, Sequential Crossover, Orca Predator Algorithm, 2-Opt

ÖNSÖZ

Tez çalışmamın her aşamasında fikirlerini, görüşlerini ve desteklerini esirgemeyen tez süreci boyunca her problemimde tecrübeleri ve bilgileri ile beni yönlendiren değerli danışmanım Doç. Dr. İlhan İLHAN'a, çalışmam boyunca maddi manevi desteklerini esirgemeyen aileme sonsuz teşekkürlerimi sunarım.

Hamdi KILINÇ
KONYA-2023

İÇİNDEKİLER

ŞEKİLLER

ÇİZELGELER

SİMGELER VE KISALTMALAR

Kısaltmalar

TSP	: Gezgin Satıcı Problemi
STSP	: Simetrik Gezgin Satıcı Problemi
ATSP	: Asimetrik Gezgin Satıcı Problemi
DFJ	: Dantzig Fulkerson Johnson
D-GWO	: Ayrık Gri Kurt Optimizasyon Algoritması
OX1	: Sıralı Çaprazlama
PSO	: Parçacık Sürü Optimizasyon Algoritması
VTPSO	: Hız Geçici Parçacık Sürü Optimizasyon Algoritması
SS	: Takas Dizisi
DSMO	: Ayrık Örümcek Maymunu Optimizasyon Algoritması
OPA	: Orka Yırtıcı Algoritması
DOPA	: Ayrık Orka Yırtıcı Algoritması
ER	: Kenar Rekombinasyon Çaprazlama
IBA	: Geliştirilmiş Ayrık Yarasa Algoritması
LBSA-CO	: Çaprazlama Operatörlü, Liste Tabanlı Benzetilmiş Tavlama Algoritması
DSSA	: Ayrık Serçe Arama Algoritması
GA-JGHO	: Atlama Genli ve Sezgisel Operatörlü Genetik Algoritma
ABCSS	: Takas Dizisi Tabanlı Yapay Arı Kolonisi Algoritması
DBAL	: Levy Uçuşlarına Dayanan Ayrık Yarasa Algoritması

1. GİRİŞ

Gezgin Satıcı Problemi (Traveling Salesman Problem - TSP), günümüzde en sık çalışılan optimizasyon problemlerinden birisidir. İlk olarak 1759'da Leonhard Euler tarafından ortaya atılmıştır. 1932'de Karl Menger tarafından matematiksel olarak formüle edilmiştir (Pfluger, 1968). TSP; çizelgeleme, devre tasarlama, fabrikalarda tesislerin yerleşim tasarımı, rota planlama ve baskı devre tasarlama gibi birçok gerçek dünya problemlerini çözmek için kullanılır. Bu nedenle, optimizasyon yöntemleri alanında çalışan araştırmacılar, onu gerçekçi bir test ortamı olarak tercih ederler. Ayrıca, yapay zekâ teknolojisinin sürekli gelişim göstermesine ve beraberinde birtakım yeniliklerin oluşmasına olanak sağlarlar. Bu yeniliklerin gelişimi insanların yaşam kalitesini arttırmak ve oluşacak maliyeti azaltmak için kullanılmaktadır. Kullanılan yenilikler satıcının algısını geliştirerek farklı şehirlerde, müşterilerle olan ilişkisini güçlendirecek ve kullanılabilir zamanın boşa gitmesini engelleyecektir. Gelişimin sağlanmasıyla birlikte hesaplanması zor olan problemlere cevap bulmak daha da kolaylaşacaktır. Dolayısıyla malzeme veya hizmetin transferi hızlı bir şekilde yapılacaktır. Burada ihtiyaç duyulan zorunluluk, lojistik uygulamalarının önemini arttırmaktadır. Lojistik uygulamaların hem etkin hem de ekonomik olması oluşacak rekabet ortamında satıcı için önemli avantajlar sağlayacaktır. Ayrıca, hizmetin en düşük rota maliyetinde tamamlanması önem arz etmektedir.

Lojistik uygulamaların her geçen gün gelişmesi TSP'ye olan ilgiyi artırmıştır. Günümüzde ulaşım ve lojistik uygulamaları kullanarak faaliyet gösteren satıcıların optimum rotayı bulmaları oldukça kolay bir hâle gelmiştir. Özellikle ulaşım ve lojistik uygulamaları arasında artan rekabet anlayışı, gelişen teknolojiyle birlikte satıcının bu yeniliklere kolaylıkla adapte olmasına ve sürekli kendisini bu alanda geliştirmesine veya ilerletmesine bağlı olarak çözüme kavuşacaktır. Satıcının tüm şehirleri kısa zamanda, düşük maliyette ve eksiksiz dolaşması, anlaşılması kolay ve hesaplanması karmaşık bir optimizasyon problemi olan TSP'nin kullanılmasına bağlıdır (İlhan ve Gökmen, 2022).

Bu problemin üzerinde sıklıkla çalışılmasının nedeni ise kargo şirketlerinin dağıtım planlaması, uçaklar için en kısa rotaların hesaplanması ve satıcının en kısa zamanda dolaşacağı rotanın belirlenmesi gibi hayatta karşılaşılabilecek durumları ve çözülmesi zor olan problemleri içermesinden kaynaklıdır. TSP'nin çözümünde tercih edilen algoritmalar kesin çözüm üreten ve kesin çözüm üretmeyen algoritmalar olmak üzere iki grupta incelenir. Kesin çözüm üreten algoritmalar bir kuvvet uygulama mantığı

ile probleme yaklaşır ve tüm yolları deneyerek en iyi sonuca ulaşmayı hedefler. Kesin çözüm üreten algoritmalar, diğerlerine kıyasla az sayıda düğüme sahip TSP üzerinde iyi sonuçlar üretse de fazla düğüm sayısına sahip problemler için uygulamanın karmaşıklık zamanını oldukça arttırmaktadır. Bu sebeple fazla sayıda düğüme sahip problemlerde kesin çözümlerin kullanılması verimli sonuçlara ulaşmada etkili olmaz. Kesin çözüm üretmeyen algoritmalar ise problemlerde yaklaşım ve sezgisel yöntem anlayışlarını kullanarak optimum sonuca yakın çözümler üretir. Kesin çözüm üretmeyen algoritmalar büyük boyutlu problemler için zaman tasarrufu sağlarlar. Bu algoritmalar en iyi olmasa da en iyi çözüme yakın sonuç üretmeyi amaçlarlar. TSP, NP-zor problem sınıfında yer alır. Bu nedenle, en uygun çözümler farklı optimizasyon algoritmaları kullanılarak kısa sürede elde edilir (Pulat ve Karakoç, 2017).

Bu çalışmada, orka yırtıcı algoritmasının (Orca Predation Algorithm - OPA) ayrık bir versiyonu geliştirilmiş, ayrık orka yırtıcı algoritması (Discrete Orca Predation Algorithm - DOPA) olarak adlandırılmış ve TSP'yi çözmek için kullanılmıştır. OPA gibi DOPA da kovalama ve saldırı olmak üzere iki fazdan oluşmaktadır. Kovalama fazında orkalar arasındaki mesafeler Hamming mesafesi ile hesaplanmış, hesaplanan bu değerler kullanılarak hız değerleri elde edilmiştir. Hız değerleri ve 2-opt algoritması kullanılarak orkaların konumları güncellenmiştir. Saldırı fazında orkaların konumları sıralı çaprazlama (Order Crossover - OX1) operatörü ile hesaplanmıştır. Pozisyon ayarlama prosedüründe ise takas lokal arama operatörü kullanılmıştır. DOPA'nın parametreleri Taguchi istatistiksel yöntemi ile ayarlanmıştır. DOPA 67 iyi bilinen TSP örneği üzerinde test edilmiştir. Ayrıca, DOPA ile diğer güncel dokuz yöntem arasında önemli farklılıklar olup olmadığını kontrol etmek için Friedman ve Wilcoxon işaretli sıra testleri uygulanmıştır.

Bu çalışmanın literatüre katkıları aşağıdaki gibi sıralanabilir:

(1) OPA'nın, DOPA olarak adlandırılan, ayrık bir versiyonu geliştirilmiştir.

(2) DOPA'da OPA'nın yapısına bağlı kalınmış, kovalama ve saldırı fazları TSP'ye göre düzenlenmiştir.

(3) Saldırı fazında pozisyon ayarlama prosedüründe takas operatörü kullanılmış, yakınsama hızlandırılmıştır.

(4) DOPA'nın parametreleri Taguchi yöntemi ile ayarlanmıştır.

(5) DOPA'nın, TSP'yi çözmek için, alternatif ve rekabetçi bir yaklaşım olarak kullanılabileceği 67 örnek üzerinde farklı performans ölçütleri ile gösterilmiştir.

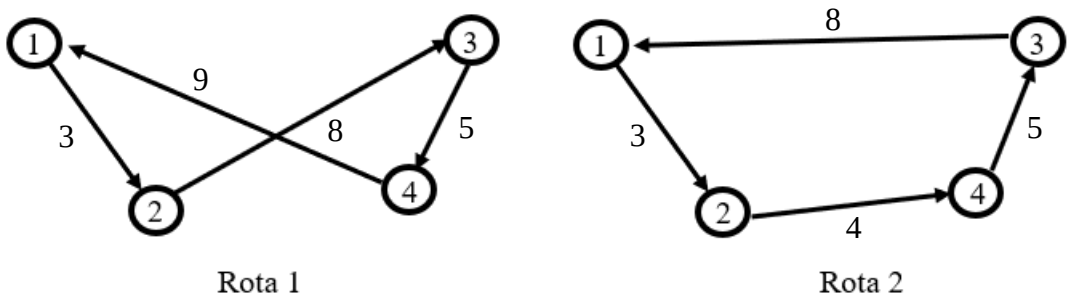
Tezin organizasyonu şu şekildedir: İkinci bölümde TSP hakkında bilgi verilmiştir. Üçüncü bölümde TSP'yi çözmek için literatürde yapılan çalışmalar ayrıntılı olarak sunulmuştur. Dördüncü bölümde orijinal OPA'nın kısa bir açıklaması yapılmıştır. Beşinci bölümde TSP'yi çözmek için önerilen DOPA anlatılmıştır. Altıncı bölümde DOPA'nın sayısal sonuçları verilmiştir. Son bölümde ise çalışmadan elde edilen sonuçlar değerlendirilmiştir.

2. GEZGİN SATICI PROBLEMİ (TSP)

2.1. TSP Nedir?

Bir seyyar satıcının, Hamilton yolu probleminde kullanılan kurallar dahilinde, belli rota üzerinde bulunan şehirler arasında yolculuk yaparken şehirlere bir defa uğramak şartıyla en kısa yoldan tüm şehirlere olan yolculuğunu tamamlamasıdır. TSP'nin çözümü lojistik, ulaştırma gibi birçok alanda kullanılmaktadır.

4 şehirden oluşan örnek TSP problemine ait iki farklı rotanın Rota 1 = 1234 ve Rota 2 = 1243 şeklinde şehirler dizisi olarak verildiği ve bu rotalara ait grafiksel gösterimin de Şekil 2.1'de olduğu gibi kabul edilsin. Bu durumda, Rota 1 için toplam mesafe (uygunluk değeri) 25, Rota 2 için ise 20 olarak hesaplanır. Rota 2, Rota 1'e göre daha iyi bir çözüm olarak tespit edilir.



Şekil 2.1. 4 şehirden oluşan TSP'ye ait örnek rotalar

2.2. TSP Türleri

Her bir düğüm çifti için, i şehrinden j şehrine olan mesafenin j şehrinden i şehrine olan mesafeye eşit olması durumuna Simetrik TSP (STSP) denir. Bu çalışmada kullanılan veri örneklerinin tamamı STSP grubuna aittir.

Her bir düğüm çifti için, i şehrinden j şehrine olan mesafenin j şehrinden i şehrine olan mesafeden farklı olması durumuna Asimetrik TSP (ATSP) denir

2.3. Matematiksel Model

TSP'nin matematiksel modeli aşağıdaki verilmiştir (Dantzig ve ark., 1954):

Amaç Fonksiyonu:

$$Min\ F = \sum_{i=1}^{v} \sum_{j=1, i \neq j}^{v} d_{ij} z_{ij} \quad (1)$$

Kısıtlar:

$$\sum_{i=1, j \neq i}^{v} z_{ij} = 1, j \in V \quad (2)$$

$$\sum_{j=1, i \neq j}^{v} z_{ij} = 1, i \in V \quad (3)$$

$$\sum_{i,j \in S, i \neq j}^{v} z_{ij} \leq |S| - 1, \forall S \subset V \quad (4)$$

$$z_{ij} \in \{0, 1\}, i \neq j; i, j \in V \quad (5)$$

Minimize edilmesi gereken amaç fonksiyonu Denklem 1'de sunulmuştur. Burada d_{ij}, i şehrinden j şehrine olan seyahat mesafesini gösterir ve d_{ij}>0'dır. z_{ij} ise karar değişkenidir ve satıcı i şehrinden j şehrine seyahat ederse 1 aksi durumda 0'dır. Denklem 2 ve 3, her şehrin sadece bir kez ziyaret edilmesini garanti eder. Herhangi olası alt turlardan kaçınılması gerekir. Bu kısıt Denklem 4 ile sağlanır. z_{ij} değişkeni 0 veya 1 olmak üzere sadece iki farklı tamsayı değer alır. Bu kısıt Denklem 5'de gösterilir.

2.4. TSP'nin Uygulama Alanları

TSP, endüstriyel sistemlerde ya da diğer alanlarda karşılaşılabilecek gerçek dünya problemlerini çözmek için kullanılabilir. En önemli uygulamalarından biri gidilecek yola göre rotanın planlanmasıdır. Rota üzerinde bulunan şehirler, problemin adından da anlaşılacağı gibi bir satıcının satış yapmaya gideceği yerler ya da tatil yapmak üzere yola çıkan bir ailenin dolaşmak istediği tatil köyleri olabilir. Sonuçta, gezgin satıcı da tatile çıkan aile de gidilecek şehirlerin dolaşımını en az maliyetle yapmak ister. Bunu sağlamak için de gezmek istedikleri şehirlerden geçen en kısa turu hesaplayarak rotalarını bu tura göre harita üzerinde çizerler. Otobüs ve uçaklarla gerçekleştirilen posta veya kargo sistemlerinin dağıtımı bu yöntemden faydalanarak uygulanır. Aslında kullanılan bu yöntem tüm toplama ve dağıtım problemlerinde uygulanabilir.

Fazla sayıda insan tarafından gerçekleştirilen işin tek bir makine kullanılarak yapıldığı bir fabrikada, farklı işlerin de yapılabilmesi için makinenin farklı durumlarda

(sıcaklık değeri ya da basınç değerinin yüksek olması gibi) çalıştırılması gerekiyorsa bu durumların hangi sıraya göre gerçekleşeceğini belirleme TSP'nin esas uygulama alanlarından olan planlama sürecine girer. İş planlarının oluşturulmasının yanı sıra ders programı hazırlama ve sınav notlarına bağlı çizelgeleme gibi çeşitli problemler de TSP'ye benzer problemlerdir.

Elektronik baskı devre üretiminde üretici firmalar, elektronik parçaların kart üzerinde lehimlenebileceği iğnelere karşılık gelen yerlerin delinmesine ihtiyaç duymaktadır. Burada amaç, kullanılacak kartların üzerinde yer alan bu noktaların delinmesi işleminin süresini en aza indirerek üretimin hızlanmasını sağlamaktır. Elektronik baskı devre üretiminde delme faaliyetinin hangi sırayla yapılacağı da TSP'nin esas uygulama alanlarından biridir.

3. LİTERATÜR TARAMASI

TSP'yi çözmek için kullanılan yaklaşımlar kesin, sezgisel ve metasezgisel yöntemler olmak üzere üç grupta toplanabilir. Bu yöntemlerle ilgili detaylı bilgiler ve literatürde yapılan çalışmalar aşağıdaki alt bölümlerde verilmiştir.

3.1. Kesin Yöntemler

Kesin yöntemler, küçük boyutlu TSP'leri makul bir zaman diliminde çözebilirler. Ancak problemin boyutu arttıkça bu yöntemlerin uygulama süreleri de artar. Sonuç olarak, bu yöntemlerin optimum sonuçlara ulaşması imkânsız hale gelir. Kesin yöntemlere dinamik programlama (Held ve Karp, 1962), dal-sınır (Lawler ve Wood, 1966), dal kesme (Padberg ve Rinaldi, 1991), kesme düzlemi (Miliotis, 1978), doğrusal programlama (LP) gevşemesi (Dantzig, 1954) ve sayma yöntemi (Berbeglia ve Hanh, 2009) gibi yöntemler örnek olarak gösterilebilir. Bu yöntemlerden bazıları aşağıda açıklanmıştır.

Dinamik Programlama

Sayma yöntemine benzerlik gösteren kesin çözüm yöntemidir. Ayrıca tekrar eden hesaplamaların önüne geçebilmek için sayma yönteminin kullanılması daha kısa sürelerde çözüme ulaşmayı sağlayabilir. TSP'yi çözmek için dinamik programlama tabanlı bir yöntem Held ve Karp (1962) tarafından önerilmiş ve kullanılmıştır. Held ve Karp bu yöntemi, iki fazlı hesaplama tekniğinin yinelemeli olarak kendini tekrarlaması sonucunda elde etmişlerdir. Ayrıca, büyük problemlerin çözümüne ulaşmak için algoritmaları ardışık yaklaşım yöntemiyle birleştirerek kullanmışlardır.

Dal – Sınır Algoritması

Rasgele olarak üretilen bir çözümden hareketle problemi çözmeye çalışır. Dallara ayırma işlemi, çözüm uzayının parçalara ayrılmasını ve böylelikle çözüm uzayının sıkıştırılarak araştırılmasını sağlar. Dal-sınır algoritması, TSP için belirlenen başlangıç alt sınırı ile kısıtların daraltılması sonucu Dantzig Fulkerson Johnson (DFJ) formülasyonunun birlikte kullanılmasından elde edilir. Yapılan birçok çalışmada dal–

sınır algoritması kullanılmıştır (Lawler ve Wood, 1966). Maliyeti en aza indirmek için farklı bir yöntem olarak küçük örten ağaç problemi de kullanılmaktadır.

Dal Kesme Algoritması

Dal-sınır algoritması ile çok yüzlü (polyhedral) tekniklerin birleştirilmesi sonucunda oluşan dal kesme algoritması tam sayılı programlama problemleri için kullanılan en etkili yöntemlerden birisidir. Dal kesme algoritması, dal-sınır algoritmasına ve çok yüzlü tekniklere benzerlik göstermektedir. Tamsayıların kullanılmasıyla birlikte problemin başlatılması, doğrusal programlama sonucunda elde edilecek çözümlere bağlıdır. Tam sayılı programlama problemine etkin bir çözüm sunabilmek için yalnız dal kesme algoritmasının kullanılması yeterli değildir. Ayrıca dallara ayırma işleminin gerçekleştirilmesiyle optimum sonuca ulaşmak gerekir. Dal-sınır algoritması, çalışma hızı açısından değerlendirildiğinde dal kesme algoritmasının kullanılmasıyla bir hayli hızlandırılabilir. Dallara ayırma işlemini yapmadan kesme eklenebilir, ayrıca çözüm sürecinde de ağacın her düğümü için kesme işlemi uygulanabilir. Dal ve sınırın aksine, düzlem kesme prosedürü optimal bir çözümle sona ermediğinde, dallanmadan sonra kesimler üretmeye devam eden bir ağaç arama stratejisi kullanılır (Padberg ve Rinaldi, 1991).

Kesme Düzlemi

Doğrusal olmayan programlama sorunlarında çözüme ulaşmak ve farklılaşmayı sağlayan dışbükey optimizasyon problemlerinde çoğunlukla tercih edilmektedir. Dışbükey optimizasyon problemleri ve çeşitleri farklı isimlerle bilinmektedir. Bir kesme düzlemi, LP gevşemesine bağlı olarak çözülebilmektedir. Ancak, LP gevşemesinin de çözüme ulaşması garanti edilemez. Başka bir kesirli sayıya ulaşmak muhtemeldir. Bu sebeple, işlemlerin tekrar edilmesi gerekmektedir (Miliotis, 1978).

Doğrusal Programlama (LP) Gevşemesi

Büyük boyutlu problemlerin çözümüne hızlı bir şekilde ulaşmak için kullanılan yöntemlerden birisidir. LP gevşemesi, problemin karmaşık olmasını sağlayarak çözülmesini zorlaştıran bazı faktörlerin gevşetilmesi sonucunda bir kat sayıyla çarpılarak

elde edilen amaç fonksiyonunu ifade etmektedir. LP gevşemesi, rasgele bir çözümle başlatılır. Çözümde kullanılan alt ve üst sınırların birbirlerine yaklaşması sağlanmaktadır. Alt ve üst arasındaki fark, bazı hallerde çok azalabilmektedir. Bu da algoritmanın çok uzun süreli çalışmasına ve sonucun geç bulunmasına neden olabilmektedir (Dantzig, 1954).

Sayma Yöntemi

En iyi sonucu bulmak için oluşabilecek tüm kombinasyonlar değerlendirilir. k şehir sayısına sahip bir problem için (k-1)! adet farklı çözüm içerisinden en kısa tur uzunluğuna sahip çözüm yolunun bulunması gerekmektedir. Büyük boyutlu problemler için hesaplanan zaman karmaşıklığının O((k-1)!) olduğu düşünüldüğünde, onaylanabilir bir zaman aralığında optimum çözümü bulmak imkansızdır (Berbeglia ve Hanh, 2009).

3.2. Sezgisel Yöntemler

Sezgisel yöntemler, iyi bilinen TSP örnekleri için makul bir zaman aralığında kaliteli çözümler bulabilir. Ancak optimal çözümlere ulaşabilme garantileri yoktur. Bu algoritmalar deneme yanılma yoluyla çözümler bulur. Her zaman değil ancak çoğu zaman çalıştıkları kabul edilir. Bu algoritmalar en iyi çözümlerden daha ziyade kolay ve ulaşılabilir iyi çözümlerin gerekli olduğu durumlarda kullanılabilir (İlhan ve Gökmen, 2022). Sezgisel yöntemlere k-opt, tur birleştirme (Cook ve Seymour, 2003), Helsgaun'un LK (Lin-Kernighan) algoritması (Helsgaun, 2007), LKH-2 (Değiştirilmiş Lin-Kernighan) algoritması (Helsgaun, 2009) ve Dong'un yaklaşımı (Dong ve ark., 2009) gibi yöntemler örnek olarak gösterilebilir. Bu yöntemlerden bazıları aşağıda açıklanmıştır.

Tur Birleştirme

Tur birleştirme, gezgin satıcı probleminde bulunan düğümlerin arasında değişmez yeni bir bağlantının oluşturulmasıdır. Ancak değişmeze dayalı parçalamalar, optimizasyon problemlerine çözüm üretmek için dinamik programlamaya doğal bir çerçeve sağlamaktadır. Dal-kesmelerini tespit etmek için sezgisel bir yöntem olarak tur bileştirme kullanılmaktadır. Ayrıca, gezgin satıcı problemlerinde üretilen tur koleksiyonların birleştirilmesiyle probleme cevap olabilecek yüksek kaliteli turların elde

edilmesi mümkün hale gelecektir. Çözülmesi zor problemler için arama uzayı sonucunun optimuma yakın bir değer olarak elde edilmesi için tur birleştirme yönteminin kullanılması gerekmektedir. Tur birleştirme yönteminden farklı olarak optimum mesafenin de elde edilmesi sağlanacaktır (Cook ve Seymour, 2003).

Helsgaun'un LK Algoritması

Yerel arama algoritmasına ait olan bir algoritmadır. Arama uzayında rasgele belirtilen bir konumdan başlayarak komşu konumlar üzerinden hareket etme eğilimi göstermektedir. Aday çözümünü farklı bir çözüme dönüştürebilen değişim şekli olarak ifade edilmektedir. Bir turdaki k kenarını daha kısa bir tur elde edecek şekilde farklı bir k kenarıyla değiştirerek yeni bir tur oluşturulmaktadır (Helsgaun, 2007).

LKH-2 Algoritması

Helsgaun'un LK algoritması ile bu yöntem arasındaki fark, düğüm sayısının fazla olduğu örneklerin çözülmesi durumunda zaman ve hesaplama açısından maliyetinin az olmasıdır. LKH-2 algoritması, Helsgaun'un LK algoritmasındaki eksiklikleri ve maliyeti yok denecek kadar azaltmaktadır. Aynı zamanda yüksek boyutlu örnekler için daha kaliteli çözümler elde etme olasılığını arttırmaktadır (Helsgaun, 2009).

Dong'un Yaklaşımı

Ön işlem adımı olarak hesaplanan kenarların daraltılmasıyla daha küçük boyutlu bir tur elde etme işlemidir. Rotanın küçük boyutlu olması, yeniden dönüştürülemeyeceği anlamına gelmemektedir. Helsgaun'un LK ve LKH-2 yöntemlerinin bu yaklaşımla karşılaştırılması önemli ölçüde güçsüz olduklarını göstermektedir. Problem boyutunun küçültülmesine istinaden daha iyi turların oluşturulabileceği ifade edilmektedir (Dong ve ark., 2009).

3.3. Metasezgisel Yöntemler

Metasezgisel yöntemler, yerel arama ve rastgeleleştirme prosedürlerini birleştiren gelişmiş sezgisel algoritmalardır. Bu yöntemler, çözüm uzayında olasılığa dayalı mantıklı

bir arama yaparak istenilen optimum sonuca ulaşmayı sağlamaktadır. Bu yöntemler genellikle basit sezgisel yöntemlerden daha iyi performans göstermekte ve kısa çalışma sürelerinde tatmin edici sonuçlar üretebilmektedir. TSP'yi çözmek için metasezgisel algoritmalar kullanılarak birçok çalışma yapılmış, bu çalışmalar aşağıda özetlenmiştir.

Akhand ve ark. (2015) tarafından TSP'yi çözmek için parçacık sürü optimizasyonu (PSO) tabanlı Hız Geçici PSO (VTPSO) algoritması önerilmiştir. Bu algoritma parçacıkların hız işlemini gerçekleştirmek için takas dizisini (SS) dikkate almaktadır. Bir hızın (SS) her turda değişecek iki konumu belirtmesi, birkaç takas operatörü (SO) topluluğuyla aynı anlamı taşımaktadır. Aktif kullanılan yöntemler, yeni bir çözümde hesaplanan SS'nin tüm SO'larını bir çözüme uygulayarak kullanmaktadır. VTPSO ise hesaplanan SS'yi geçici hız olarak kabul etmekte ve SO'ları art arda uyguladığında geçici çözümleri kontrol altına almaktadır. SS'nin bir kısmı ile en iyi çözüm turu elde edilirken VTPSO'da bir parçacık çözüm noktası olarak kabul edilmektedir. Bu şekilde ara geçici tur değerlendirme işlemi, yalnızca daha iyi bir çözüm üretmekle kalmaz, oluşacak hesaplama süresini de azaltır (Yang ve ark., 2012).

Osaba ve ark. (2016) çok kullanılan simetrik ve asimetrik TSP'ye çözüm üretmek için geliştirilmiş ayrık yarasa algoritmasını (IBA) önermişlerdir. IBA'nın umut verici bir yaklaşım olduğunu ispatlamak için beş farklı yöntemle 37 örnek üzerinde performans değerlendirmesi yapılmıştır. Adil ve eksiksiz karşılaştırma için üç farklı istatistiksel test analizi uygulanmıştır. Ayrıca önerilen algoritmanın yakınsama davranışı, evrimsel benzetilmiş tavlama ve ayrık ateşböceği algoritmalarının yakınsama davranışıyla karşılaştırılmıştır (Bora ve ark., 2012).

Khan ve Maiti (2019) yapay arı kolonisi algoritmasını (ABC) k-opt operatörü ile birleştirerek TSP'ye çözüm bulmayı amaçlamışlardır. ABCSS olarak isimlendirilen algoritmada elde edilen çözümleri güncellemek için sekiz farklı kural kullanılmıştır. Bir çözümün sonuçlarının görevli arı veya gözcü arı tarafından güncellenmesi, rulet tekerleği yöntemi kullanılarak kural setinden rastgele seçilen bir kural ile yapılmaktadır. Algoritmada k-opt işlemi, herhangi bir çözümde olası iyileştirme için gerekli olan sabit sayı kadar çalışmaktadır. Çözümün kalitesini artırmak için arama uzayının sonunda yine k-opt işlemi kullanılmaktadır (Pan ve ark., 2011; Kiran ve ark., 2015; Gündüz ve ark., 2015).

Akhand ve ark. (2019) tarafından yapılan bir çalışmada ayrık örümcek maymunu optimizasyon (DSMO) algoritması önerilmiştir. DSMO'da her örümcek maymunu, optimum çözümü elde edebilmek için maymunlar arasında etkileşimi sağlayan takas

dizisi (SS) ve takas operatörü (SO) tabanlı işlemleri birlikte kullanarak bir TSP çözümünü temsil etmektedir. SO'lar, belirli bir örümcek maymununun yaşamından elde edilen tecrübelerden farklı olarak grupta bulunan diğer üyelerin (yerel lider, küresel lider veya rastgele seçilen bir örümcek maymunu) tecrübelerinin kullanmasıyla elde edilmektedir. Önerilen yöntemin performansı ve etkililiği, çok sayıda TSP veri örneği üzerinde test edilmiş ve sonuçlar diğer iyi bilinen yöntemlerle karşılaştırılmıştır (Bansal ve ark., 2014).

Saji ve Barkatou (2021) yeni bir ayrık yarasa algoritması önermişler ve DBAL olarak isimlendirmişlerdir. Arama taktiklerini geliştirmek ve lokal optimuma takılmaktan kaçınmak için L´evy uçuşlarına bağlı olarak rastgele oluşturulan yürüyüşler yarasanın hareketleriyle birleştirilmiştir. Popülasyon çeşitliliğini ve yakınsamayı geliştirmek için önerilen algoritma nötr çaprazlama operatörüyle birlikte kullanılmıştır. DBAL'ın performansı, 38 TSP örneği üzerinde sekiz farklı algoritma ile karşılaştırılmıştır (Yang, 2010).

Panwar ve Deep (2021) yeni ayrık gri kurt optimizasyon (D-GWO) algoritmasını önermişlerdir. Önerilen algoritmada üretilen çözümler 2-opt algoritması ile geliştirilmiştir. D-GWO'nun performansını değerlendirmek için üç farklı algoritma ile karşılaştırmalar yapılmıştır. Tarafsız ve net bir karşılaştırma için ortalama ve standart sapma gibi belirleyici istatistikler kullanılmıştır. Ayrıca Friedman ve Holm testi gibi istatistiksel testler de uygulanmıştır.

İlhan ve Gökmen (2022) çaprazlama operatörlü, liste tabanlı benzetilmiş tavlama (LBSA-CO) algoritmasını önermişlerdir. Bu algoritma, popülasyon tabanlı metasezgisel bir yöntemdir ve problemin çözüm uzayına entegre olabilen liste tabanlı sıcaklık soğutma çizelgesi kullanmaktadır. Popülasyondaki çözümler ters çevirme, ekleme ve 2-opt yerel arama operatörleri ile geliştirilmiştir. Yakınsama hızını arttırmak için geliştirilmiş çözümlere sıralı çaprazlama (OX1) ve kenar rekombinasyon çaprazlama (ER) operatörleri uygulanmıştır. Algoritmanın parametrelerini ayarlamak için Taguchi yöntemi kullanılmıştır. Önerilen yöntem, iyi bilinen 65 TSP örneği üzerinde test edilmiştir.

Zhang ve Han (2022) simetrik TSP'yi çözmek için ayrık serçe arama algoritmasını (DSSA) önermişlerdir. DSSA'da, serçelerin konum güncellemeleri sıra tabanlı kodlama ve kod çözme yöntemi ile yapılmıştır. Yoğunlaşma ve çeşitlendirme arasındaki denge, Gauss mutasyonu ve takas operatörü ile birlikte küresel pertürbasyon mekanizmasının kullanılmasıyla sağlanmaktadır. Çözümlerin kalitesini daha da artırmak için 2-opt yerel

arama operatörü kullanılmıştır. DSSA, 34 iyi bilinen TSP örneği üzerinde test edilmiştir (Zhang ve Ding, 2021).

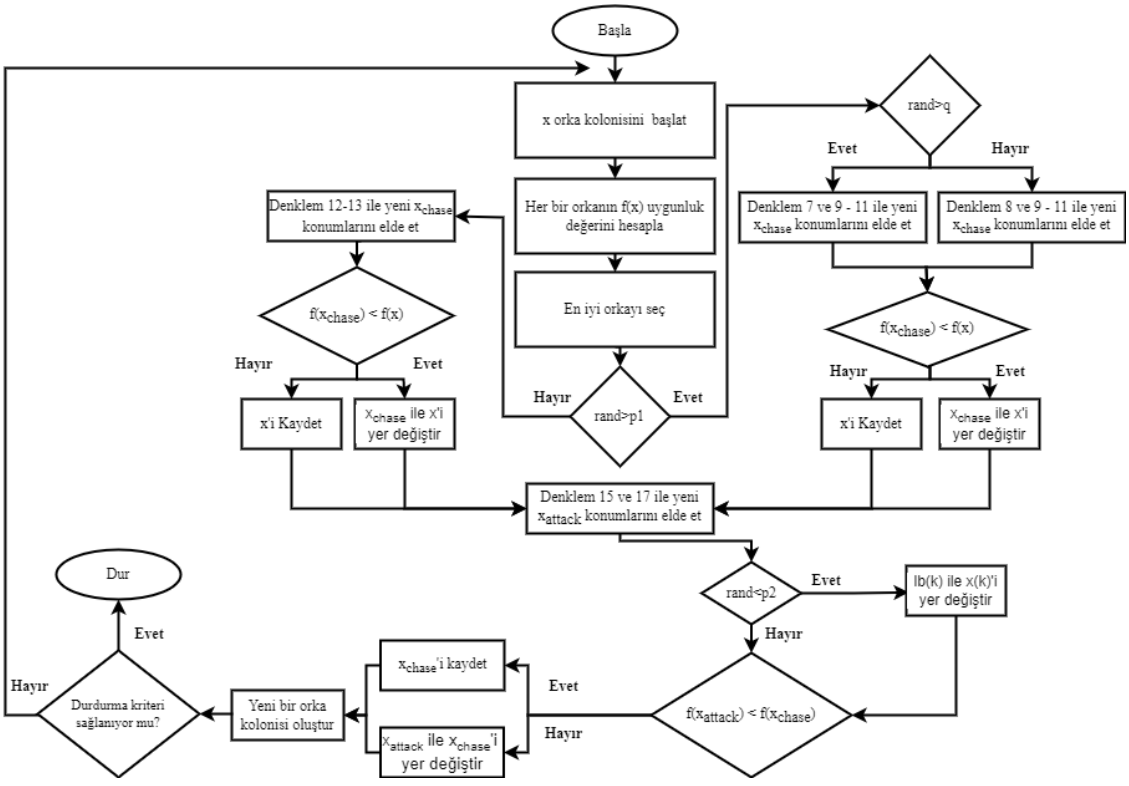
Zhang ve ark. (2022) yavaş yakınsama, düşük çözüm kalitesi ve lokal optimuma takılma problemlerini çözmeyi hedefleyen atlama genli ve sezgisel operatörlü genetik algoritmayı (GA-JGHO) önermişlerdir. GA-JGHO'da çift yönlü sezgisel çaprazlama operatörü kullanılmıştır. Yoğunlaştırma ve çeşitlendirme arasındaki denge birleştirilmiş mutasyon operatörü ile sağlanmıştır. Atlama geni operatörü arama uzayını genişletmek ve lokal optimumda takılmayı engellemek için tasarlanmıştır. GA-JGHO'nun performansını doğrulamak için TSPLIB kütüphanesinden çok sayıda veri örneği üzerinde testler yapılmıştır (Kasat ve Gupta, 2003; Ramteke ve ark, 2015).

4. ORKA YIRTICI ALGORİTMASI (OPA)

4.1. Orka Yırtıcı Algoritmasına (OPA) Genel Bakış

Orkalar, yunus ailesinden gelen yüksek sosyalleşme derecelerine sahip çok zeki etoburlardır. Birbirlerine sıkı sıkıya bağlı olan aile grupları şeklinde yaşarlar. Gruptaki üye sayısı otuza kadar çıkabilir. Tipik bir orka grubu, %20 buzağı, %20 olgun erkek ve %60 dişiden oluşur. Bir grubun bireyleri birbirlerinden en fazla 100 m uzakta yüzerler ve nadiren gruptan birkaç saatten fazla ayrılırlar (Ford, 2018; Jiang ve ark., 2022). Orkaların görüşü oldukça iyidir, ancak görüşleri karanlık su altı ortamında avlanmak veya gezinmek için işe yaramaz. Diğer dişli balinalar gibi, orkalar da su ortamını algılamak için büyük ölçüde sonarlarına bağımlıdırlar (Ford, 2018; Jiang ve ark., 2022).

OPA, Jiang ve ark. (2022) tarafından önerilmiş metasezgisel bir optimizasyon yöntemidir. OPA, orka balinalarının, kovalama ve saldırı aşamalarından oluşan avlanma davranışlarını simüle eder ve birkaç matematiksel model şeklinde onları özetler. OPA'nın akış şeması Şekil 4.1'de verilmiştir. Akış şemasındaki bazı bölümler ise aşağıdaki alt bölümlerde açıklanmıştır.



Şekil 4.1. OPA'nın akış diyagramı

4.2. Orka Popülasyonunun Kurulması

OPA'da bir orka grubu Denklem 6'daki matematiksel model ile temsil edilir. Burada X, problemin tüm aday çözümlerinin kümesine karşılık gelen orka popülasyonunu gösterir. x_N, problemin N'inci aday çözümüne karşılık gelen N'inci orkanın konumunu ve $x_{N,D}$, problemin N'inci aday çözümün D'inci boyutunun değerine karşılık gelen N'inci orkanın D'inci boyutunun konumunu temsil eder.

$$X = [x_1, x_2, .. x_N] = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,D} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,D} \\ \vdots & \vdots & & \vdots \\ x_{N,1} & x_{N,2} & \cdots & x_{N,D} \end{bmatrix} \quad (6)$$

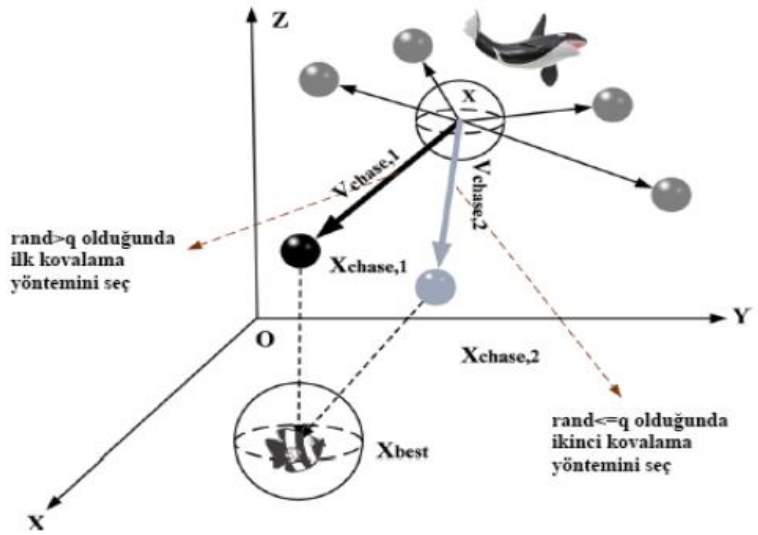
OPA, sürekli optimizasyon problemlerini çözmek için geliştirilmiş bir algoritmadır. Sürekli optimizasyon problemlerinde karar değişkenleri reel sayılar kullanılarak oluşturulur (Fığlalı, 2008). Bu nedenle, OPA'da da popülasyondaki karar değişkenleri reel sayılardan oluşur ve çözüm uzayını temsil edebilecek şekilde rastgele olarak üretilirler.

4.3. Kovalama Aşaması

Orkalar bir balık sürüsüyle karşılaştıklarında sonarları aracılığıyla birbirleriyle iletişim kurarak iş birliği yaparlar. Orka grubu dağılır, balık sürüsünü yüzeye çıkarır ve ardından onları kontrollü bir şekilde muhafaza içerisine alırlar. Bu davranışa göre, orkaların avlanma sürecindeki kovalama aşaması avın sürülmesi ve avın çevrelenmesi olmak üzere iki farklı davranışa indirgenebilir. OPA'da, bu iki davranışın gerçekleştirilme olasılığı için p1 parametresi kullanılır. p1, [0, 1] arasında sabit bir sayı olarak ayarlanır. [0, 1] arasında rastgele üretilen sayı p1'den büyük ise avın sürülmesi, aksi takdirde avın çevrelenmesi aşaması gerçekleştirilir.

Bir balık sürüsü tespit edildikten sonra, sürünün yüzeye çıkarılması gerekir. Orka sürüsü küçükse avın konumu doğru ve hızlı bir şekilde tespit edilebilir. Aksine, orka sürüsü büyükse sürünün dağılması kolay olacak ve doğru bir şekilde istenen konuma ulaşmak zor olacaktır. Orka sürüsünün büyüklüğüne bağlı olarak, iki takip yönteminin kullanılması gerekir. Orka grubu büyük (rand > q) olduğunda ilk yöntemin, orka grubu

küçük (rand ≤ q) olduğunda ise ikinci yöntemin kullanılması gerekir. Orkanın av sürüş modeli Şekil 4.2'de, hız ve konum formülasyonu ise Denklem 7-11'de verilmiştir (Jiang ve ark., 2022).



Şekil 4.2. Orkanın av sürüş modeli

$$v_{chase,1,i}^{t} = a \times (d \times x_{best}^{t} - F \times (b \times M^{t} + c \times x_{i}^{t})) \quad (7)$$

$$v_{chase,2,i}^{t} = e \times x_{best}^{t} - x_{i}^{t} \quad (8)$$

$$M = \frac{\sum_{i=1}^{N} x_{i}^{t}}{N} \quad (9)$$

$$c = 1 - b \quad (10)$$

$$\begin{cases} x_{chase,1,i}^{t} = x_{i}^{t} + v_{chase,1,i}^{t}, & Eğer\ rand > q \\ x_{chase,2,i}^{t} = x_{i}^{t} + v_{chase,2,i}^{t}, & Eğer\ rand \leq q \end{cases} \quad (11)$$

Burada, $v_{chase,1,i}^{t}$ ve $v^{t}_{chase,2,i}$ sırasıyla, birinci ve ikinci takip yöntemleri için i'inci orkanın t'inci iterasyondaki kovalama hızlarını gösterirler. M orka grubunun ortalama konumunu temsil eder. $x^{t}_{chase,1,i}$ ve $x^{t}_{chase,2,i}$ sırasıyla, birinci ve ikinci takip yöntemleri için i'inci orkanın t'inci iterasyondaki konumlarını gösterirler. a, b ve d, sırasıyla [0, 1] arasında rasgele üretilen sayıları, e, [0, 2] arasında rasgele üretilen bir sayıyı, F değeri 2 sabit sayısını ifade eder. q belirli bir takip yönteminin seçilme olasılığını temsil eder ve [0, 1] arasında sabit bir sayıdır.

Avın çevrelenmesi, orkaların balık sürüsünü yüzeye çıkardıktan sonra bu sürünün kontrollü bir top haline getirilmesi işlemidir. Orkalar sonarları aracılığıyla birbirleriyle

iletişim kurarlar ve yakınlarında bulunan orkaların konumlarına göre bir sonraki konumlarını belirlerler. OPA'da, orkalar rastgele seçilen üç orkanın konumuna göre kendilerini konumlandırırlar ve bir sonraki pozisyonları Denklem 12 ve 13'deki gibi hesaplanıp verilmiştir (Jiang ve ark., 2022).

$$x^t_{chase,3,i,k} = x^t_{j1,k} + u * (x^t_{j2,k} - x^t_{j3,k}) \quad (12)$$

$$u = 2 \times (rand - 1 / 2) \times \frac{Max\_iter - t}{Max\_iter} \quad (13)$$

Burada, Max_iter maksimum iterasyon sayısını temsil eder. j1, j2 ve j3, birbirinden farklı (j1≠ j2≠ j3), rastgele seçilen üç orkayı gösterir. $x^t_{chase,3,i}$, üçüncü kovalama yöntemi için i'inci orkanın t'inci iterasyondaki konumunu temsil eder.

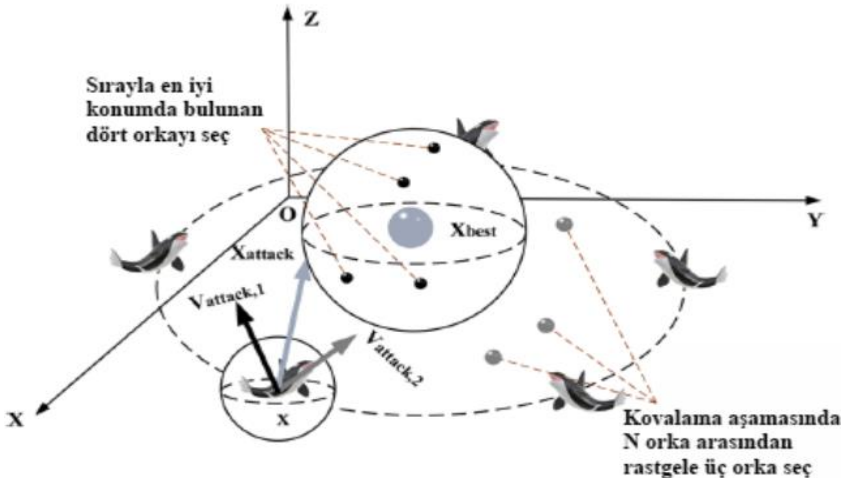
Orkalar, kovalama işlemi esnasında balıkların yaklaştığını algılarsa, kendi konumlarını güncelleyerek kovalamaya devam edeceklerdir. Aksi takdirde onlar orijinal konumlarında kalacaklardır. Konumları Denklem 14 kullanılarak ayarlanır (Jiang ve ark., 2022).

$$x^t_i = \begin{cases} x^t_{chase,i}, & Eğer\ f(x^t_{chase,i}) < f(x^t_i) \\ x^t_i, & Aksi\ durumda \end{cases} \quad (14)$$

$f(x^t_{chase,i})$ ve $f(x^t_i)$, sırasıyla, $x^t_{chase,i}$ ve x^t_i'ye karşılık gelen uygunluk fonksiyonu değerlerini gösterir.

4.4. Saldırı Aşaması

Orkalar avlarına saldırmak için sırayla muhafazaya girerler. Kuyruklarını çembere vurarak kuşattıkları balıkları sersemletirler ve onları yerler. Daha sonra başka bir orkayla yer değiştirmek için eski konumlarına geri dönerler. Çemberde dört optimal saldırı pozisyonuna karşılık gelen dört orka olduğu kabul edilir. Diğer orkalar muhafazaya girmek isterlerse dört orkanın konumları yönünde hareket ederler. Orkalar beslendikten sonra diğer orkaların yerini almak için muhafazaya geri dönmek isterlerse, yakınlarda rastgele seçilen orkaların konumlarına göre hareket yönleri belirlenir. Saldırı aşamasında orkaların hareket hızları ve konumları Denklem 15-17 ile hesaplanır ve bu işlem Şekil 4.3'deki model ile özetlenir (Jiang ve ark., 2022).



Şekil 4.3. Orkanın ava saldırı modeli

$$v^t_{attack,1,i} = (x^t_{first} + x^t_{second} + x^t_{third} + x^t_{four}) \,/\, 4 - x^t_{chase,i} \quad (15)$$

$$v^t_{attack,2,i} = (x^t_{chase,j1} + x^t_{chase,j2} + x^t_{chase,j3}) \,/\, 3 - x^t_i \quad (16)$$

$$x^t_{attack,i} = x^t_{chase,i} + g1 \times v^t_{attack,1,i} + g2 \times v^t_{attack,2,i} \quad (17)$$

Burada $v^t_{attack,1,i}$, t anında avını avlamak için i'inci orkanın hız vektörünü ve $v^t_{attack,2,i}$, t zamanında muhafazaya dönmek için i'inci orkanın hız vektörünü temsil eder. x^t_{first}, x^t_{second}, x^t_{third} ve x^t_{four} sırasıyla en iyi konumdaki dört orkayı gösterir. $x^t_{chase,j1}$, $x^t_{chase,j2}$ ve $x^t_{chase,j3}$ (j1 ≠ j2 ≠ j3) kovalama aşamasında orka popülasyonu içerisinden rastgele seçilen üç orkayı temsil eder. $x^t_{attack,i}$, saldırı aşamasından sonra t anındaki i'inci orkanın konumu temsil eder. g1 ve g2 sırasıyla [0, 2] ve [-2.5, 2.5] arasındaki rastgele sayıları gösterir (Jiang ve ark., 2022).

Orkalar sonar sistemlerini, kovalama aşamasında olduğu gibi, avlarının yerlerini belirlemek ve konumlarını güncellemek için kullanırlar. Balık sürüsü kontrol altındayken orkalardan biri balık sürüsünün kenarına ilerler ve avını almak için kuyruğuyla sürüye vurur. Orkanın konumu, problemin uygun aralığının minimum sınır değeri (lb) olarak atanır. Bu da Algoritma 1'de verilen sözde kod ile gösterilir (Jiang ve ark., 2022).

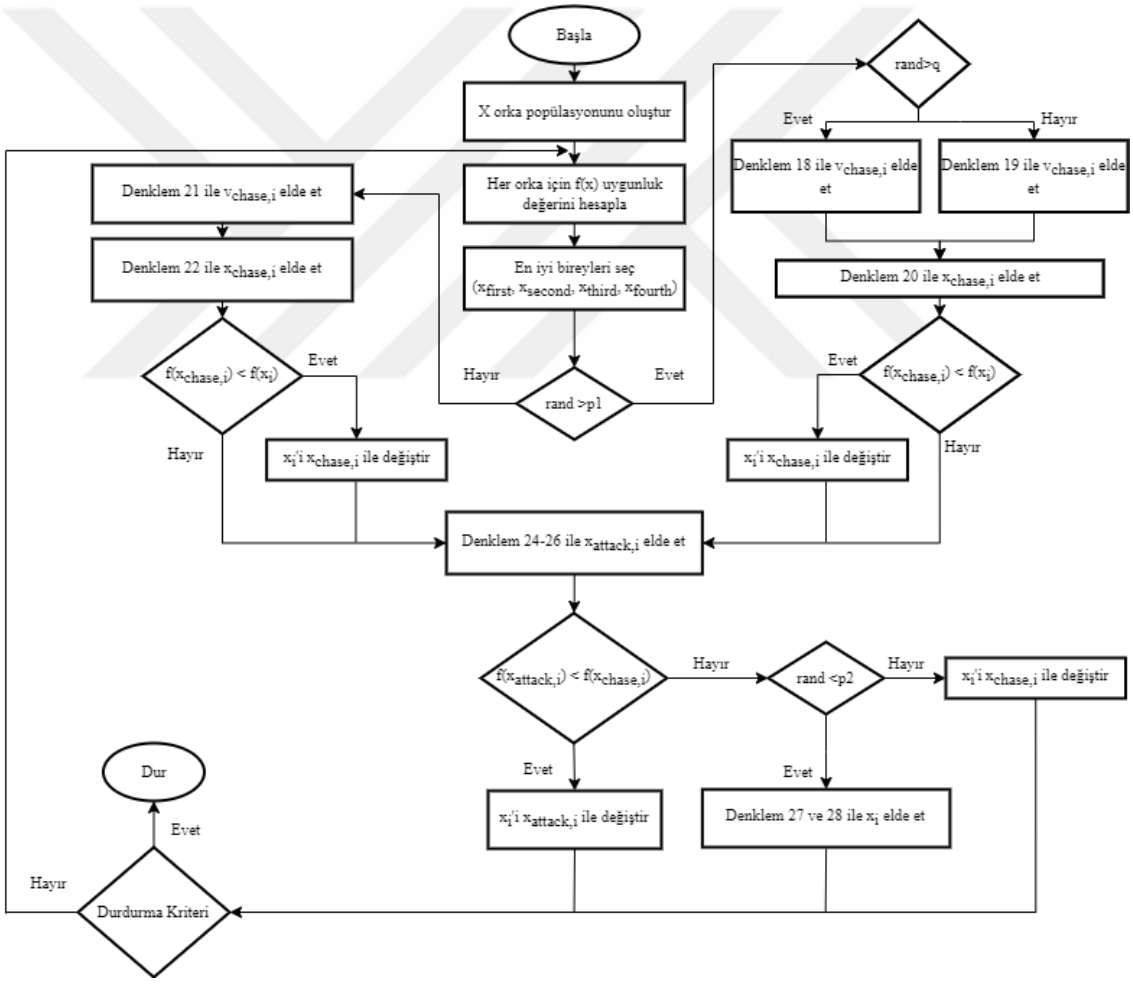
Algoritma 1. Saldırı sırasında pozisyon ayarlama aşaması

```
if f(x^t_attack,i) < f(x^t_chase,i)
    x^{t+1}_i = x^t_attack,i
else
    Q = rand;
    for k = 1 to D
        if Q < p2
            x^{t+1}_j,k = lb(k)
        else
            x^{t+1}_j,k = x^t_chase,i,k
        end if
    end for
end if
```

Burada p2, [0, 1] arasında problemlere göre farklı değerler olarak seçilen bir sabittir.

5. ÖNERİLEN AYRIK ORKA YIRTICI ALGORİTMASI (DOPA)

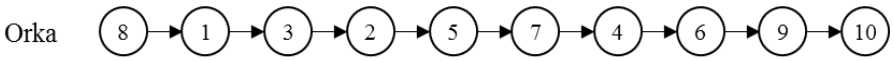
Bir önceki bölümde kısaca açıklandığı gibi OPA, orkaların kovalama ve saldırı aşamalarını taklit eder. Sürekli optimizasyon problemlerini çözmek için kullanılır. Bu nedenle, mevcut haliyle TSP gibi bir kombinatoryal optimizasyon problemlerini çözmek için kullanılamaz. Klasik OPA üzerinde birtakım değişikliklerin yapılması ve kombinatoryal optimizasyon problemlerine uygun olan ayrık versiyonunun geliştirilmesi gerekir. Bu çalışmada TSP'yi çözmek için ayrık orka yırtıcı algoritması geliştirilmiş ve DOPA olarak adlandırılmıştır. DOPA'nın akış şeması Şekil 5.1'de verilmiştir. Akış şemasını oluşturan prosedürler takip eden alt bölümlerde detaylı olarak açıklanmıştır.



Şekil 5.1. DOPA'nın akış şeması

5.1. Orka Popülasyonunun Kurulması

Şekil 5.1'den de görülebildiği gibi DOPA'da öncelikle $X = \{x_i : i = 1, 2, \ldots, N\}$ başlangıç orka popülasyonu oluşturulur. Orkaların sayısı (N), popülasyon büyüklüğüne karşılık gelir ve DOPA'ya giriş olarak verilir. Popülasyondaki her bir orka, bir çözüme karşılık gelir ve şehirler kümesinin bir permütasyonu şeklinde rastgele olarak oluşturulur. Şekil 5.2'de örnek bir orka verilmiştir. Orkanın uzunluğu problemdeki şehir sayısına yani 10'a eşittir. Orkadaki indisler şehirleri temsil eder.



Şekil 5.2. Örnek bir orka

5.2. Orkaların Uygunluk Değerinin Hesaplanması

Herhangi bir çözümün uygunluk değeri o çözümü temsil eden rotanın toplam mesafesidir. Bu mesafe, rotayı oluşturan şehirler arasındaki mesafelerin toplanmasıyla hesaplanır. Matematiksel olarak Denklem 1 ile temsil edilir. TSP örnek dosyalarında şehirler arasındaki mesafeler Öklid veya coğrafi mesafeler şeklinde verilebilir. Hatta bu dosyalarda mesafeler özel uzaklık fonksiyonları şeklinde de ifade edilebilir. Bu çalışmadaki TSP örnekleri, büyük çoğunluğu Öklid mesafesi olmak üzere, her üç mesafe ölçütünü de kullanan örneklerden oluşur.

5.3. Kovalama Aşaması

OPA'da olduğu gibi DOPA'da da kovalama aşaması avın sürülmesi ve avın çevrelenmesi olmak üzere iki farklı davranışa indirgenir. Bu iki davranışın gerçekleştirilme olasılığı için p1 parametresi kullanılır. p1, [0, 1] aralığında sabit bir sayı olarak belirlenir. [0, 1] aralığında rastgele üretilen sayı p1'den büyük ise avın sürülmesi, aksi takdirde avın çevrelenmesi aşaması gerçekleştirilir.

Avın sürülmesi aşamasında, OPA'da olduğu gibi DOPA'da da orka sürüsünün büyüklüğüne bağlı olarak iki farklı takip yöntemi kullanılır. Orka sürüsü büyük olduğunda, yani $\text{rand} > \text{q}$ için, Denklem 18 kullanılarak kovalama hızları belirlenir. Orka sürüsü küçük olduğunda ise, yani $\text{rand} \leq \text{q}$ için, Denklem 19 kullanılarak kovalama hızları belirlenir. Belirlenen kovalama hızlarına göre orkaların kovalama aşamasındaki yeni konumları Denklem 20 ile hesaplanır.

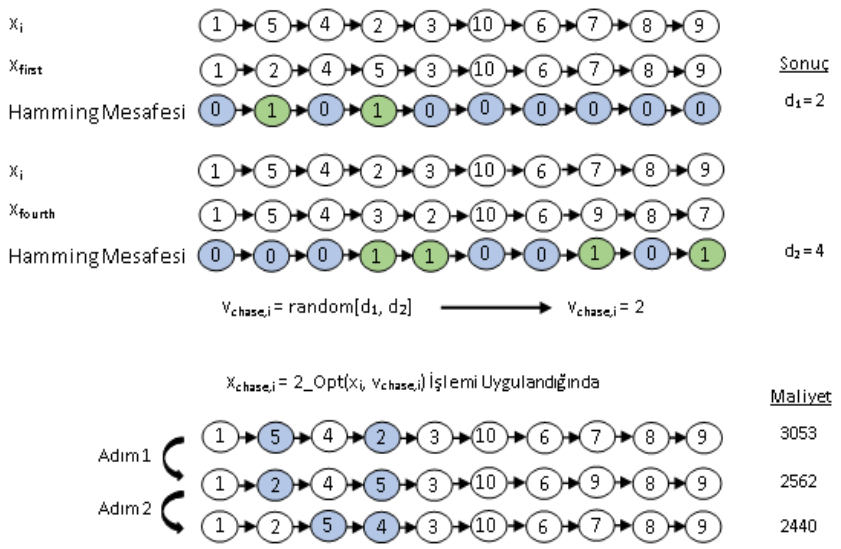
$$\left\{\begin{array}{c} d_1 = HammingDistance(x_i, x_{first}) \\ d_2 = HammingDistance(x_i, x_{fourth}) \\ v_{chase,i} = random[d_1, d_2] \end{array}\right\}, \text{ Eğer rand} > \text{q} \quad (18)$$

$$\left\{\begin{array}{c} d_1 = HammingDistance(x_i, x_{first}) \\ v_{chase,i} = random[1, d_1] \end{array}\right\}, \text{ Eğer rand} \leq \text{q} \quad (19)$$

$$x_{chase,i} = 2\_Opt(x_i, v_{chase,i}) \quad (20)$$

Burada, q takip yöntemlerinin seçilme olasılığını temsil eder ve [0, 1] aralığında sabit bir sayı olarak belirlenir. x_i i'inci orkanın konumunu, x_{first} ve x_{fourth}, sırasıyla, birinci ve dördüncü en iyi uygunluk değerlerine sahip orkaların konumlarını, $v_{chase,i}$ i'inci orkanın kovalama hızını, $x_{chase,i}$ i'inci orkanın kovalama aşamasındaki yeni konumunu gösterir. d_1, x_i ve x_{first} arasındaki, d_2, x_i ve x_{fourth} arasındaki Hamming mesafesidir. $v_{chase,i}$ orka sürüsü büyük olduğunda $[d_1, d_2]$ aralığında, orka sürüsü küçük olduğunda $[1, d_1]$ aralığında rastgele olarak üretilir. $x_{chase,i}$, i'inci orkanın x_i konumu ve $v_{chase,i}$ kovalama hızı kullanılarak 2-opt algoritması ile hesaplanır.

Şekil 5.3'de, Denklem 18 ve 20'nin örnek bir uygulaması verilmiştir. Öncelikle x_i ile x_{first} ve x_{fourth} arasındaki Hamming mesafeleri (d_1 = 2 ve d_2 = 4) ayrı ayrı hesaplanmıştır. Daha sonra $[d_1, d_2]$ aralığında rastgele bir değer üretilerek $v_{chase,i} = 2$ kovalama hızı bulunmuştur. x_i üzerinde bu hız değeri kadar 2-opt operatörü uygulanmıştır. Böylece i. orkanın kovalama aşamasındaki yeni konumu $x_{chase,i}$ bulunmuştur. x_i'den $x_{chase,i}$ elde edilirken maliyetin yani uygunluk değerinin azaldığı (3053'den 2440'a doğru) kabul edilmiştir.



Şekil 5.3. Örnek bir orkanın kovalama aşamasındaki yeni konumunun belirlenmesi

Avın çevrelenmesi aşamasında, OPA'da olduğu gibi DOPA'da da orkalar yakınlarında bulunan orkaların konumlarına göre bir sonraki konumlarını belirlerler. Bunun için rastgele olarak üç orka seçerler ve bu orkaların konumlarına göre kendilerini konumlandırırlar. Bu işlem matematiksel olarak Denklem 21 ve 22 ile temsil edilir.

$$\left\{ \begin{array}{c} d_1 = HammingDistance(x_{j2}, x_{j3}) \\ v_{chase,i} = random[1, d_1] \end{array} \right\} \tag{21}$$

$$x_{chase,i} = 2\_Opt(x_{j1}, v_{chase,i}) \tag{22}$$

Burada, x_{j1}, x_{j2} ve x_{j3} birbirlerinden farklı olarak rastgele seçilen üç orkanın konumunu, $v_{chase,i}$ i'inci orkanın kovalama hızını, $x_{chase,i}$ i'inci orkanın kovalama aşamasındaki yeni konumunu gösterir. d_1, x_{j2} ve x_{j3} arasındaki Hamming mesafesidir. $v_{chase,i}$ $[1, d_1]$ aralığında rastgele olarak üretilir. $x_{chase,i}$, rastgele seçilen j1 orkasının x_{j1} konumu ve $v_{chase,i}$ kovalama hızı kullanılarak 2-opt algoritması ile hesaplanır.

Kovalama işlemi esnasında orkalar balıkların yaklaştığını algılarlarsa kendi konumlarını güncellerler. Aksi takdirde orijinal konumlarında kalırlar. Bu durum DOPA'da Denklem 23 ile formüle edilir.

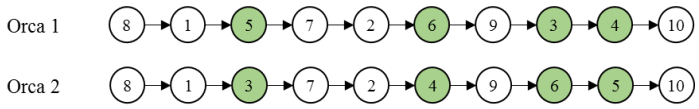
$$x_i = \begin{cases} x_{chase,i}, & Eğer\ f(x_{chase,i}) < f(x_i) \\ x_i, & Aksi\ durumda \end{cases} \tag{23}$$

Burada, $f(x_{chase,i})$ ve $f(x_i)$, sırasıyla, $x_{chase,i}$ ve x_i'ye karşılık gelen uygunluk fonksiyonu değerlerini gösterir.

OPA'nın kovalama aşamasında kullanılan a, b, d, e ve F parametreleri ayrık versiyon olan DOPA'da kullanılmamıştır. Ancak, DOPA'yı mümkün olduğu kadar orijinal OPA'ya benzetebilmek için kovalama hızları Hamming mesafe ölçütü kullanılarak hesaplanmıştır. Orkaların yeni konumları ise 2-opt algoritması kullanılarak belirlenmiştir. Hamming mesafesi ve 2-opt algoritması aşağıdaki altbölümlerde açıklanmıştır.

5.3.1. Hamming Mesafesi

Hamming mesafesi, Amerikalı matematikçi Richard Hamming tarafından bulunmuştur (Richard Hamming, 2023). Vektör bazlı kodlama teorisinde geçen bir karşılaştırma algoritması olarak bilinir. Hamming mesafesi, aynı uzunluktaki iki dizi arasında karşılıklı sembollerdeki farklılıkların sayısıdır. Bu çalışmada, popülasyondaki bir orka ile farklı bir orka arasındaki mesafeyi ölçmek için kullanılmıştır. Şekil 5.4'de bununla ilgili bir örnek verilmiştir. Şekilden de görülebildiği gibi orka 1 ve 2, 10 şehirden oluşan rotaları içermektedir. Rotadaki şehirler karşılıklı olarak karşılaştırıldığında karşılıklı dört şehirde farklılık olduğu gözükmektedir. Buradan orka 1 ve orka 2 arasındaki Hamming mesafesinin 4 olduğu sonucu çıkarılır.



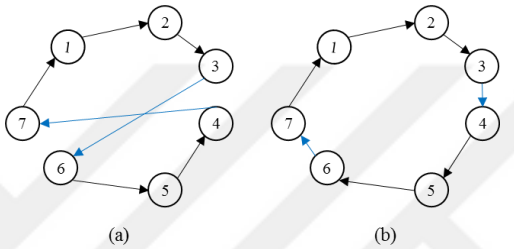
Şekil 5.4. İki orka arasındaki Hamming mesafesi

5.3.2. 2-Opt Algoritması

Bir yerel arama prosedürü, aday bir çözüm ile başlar ve onu geliştirmek için iteratif olarak kendini tekrarlar. Bu çalışmada yerel arama prosedürlerinden biri olan 2-opt algoritması kullanılmıştır. Bu algoritma ilk olarak 1958 yılında Croes (1958) tarafından önerilmiştir. Ancak genelleştirilmiş bir kavram olarak Lin (Lin, 1965) tarafından formüle edildikten sonra yaygın olarak kullanılmaya başlamıştır. 2-opt

algoritması çözümlerin kalitesini geliştirmeye ve yakınsamayı hızlandırmaya yardım eder.

2-opt algoritması öncelikle mevcut rotadan rasgele iki kenar seçer ve onları rotadan çıkarır. Daha sonra çıkarılan bu kenarları daha kısa bir seyahat mesafesi elde edecek şekilde birleştirir. Bu işlem daha kısa bir seyahat mesafesi elde edilemeyinceye kadar devam eder. Bu esnada alt turların oluşmasından kaçınılır. Şekil 5.5'de bununla ilgili bir örnek verilmiştir. Şekil 5.5a'daki orijinal rotadan 3-6 ve 4-7 kenarları çıkarılmış, bu kenarlar daha kısa bir seyahat mesafesi elde edilecek şekilde Şekil 5.5b'deki gibi birleştirilmiştir.



Şekil 5.5. 2-opt algoritması için a) Orijinal rota b) Yeni rota

5.4. Saldırı Aşaması

OPA'da çemberde dört optimal saldırı pozisyonuna karşılık gelen dört orka olduğu kabul edilir. Eğer diğer orkalar muhafazaya girmek isterlerse dört orkanın konumları yönünde hareket ederler. Eğer orkalar beslendikten sonra diğer orkaların yerini almak için muhafazaya geri dönmek isterlerse, hareket yönleri yakınlarında rastgele seçilen orkaların konumlarına göre belirlenir. DOPA'nın saldırı aşamasında da klasik OPA'daki bu yapıya dikkat edilmiş, ancak orkaların hareket hızları yerine doğrudan konumları hesaplanmıştır. Böylece, klasik OPA'da farklı aralıklarda rastgele üretilmesi gereken g1 ve g2 sayıları DOPA'da kullanılmamıştır. Saldırı aşamasında orkaların konumları Denklem 24-26 ile hesaplanmıştır.

$$x_{attack1,i} = FindMin\left[OX1\left(x_{chase,i}, x_{first}\right), OX1\left(x_{chase,i}, x_{second}\right), OX1\left(x_{chase,i}, x_{third}\right), OX1\left(x_{chase,i}, x_{fourth}\right)\right] \quad (24)$$

$$x_{attack2,i} = FindMin\left[OX1\left(x_{chase,j1}, x_i\right), OX1\left(x_{chase,j2}, x_i\right), OX1\left(x_{chase,j3}, x_i\right)\right] \quad (25)$$

$$x_{attack,i} = FindMin\left[x_{chase,i},\ x_{attack1,i}, x_{attack2,i}\right] \quad (26)$$

Burada, $x_{chase,i}$ i'inci orkanın kovalama aşamasındaki konumunu, x_{first}, x_{second}, x_{third} ve x_{fourth} en iyi uygunluk değerlerine sahip ilk dört orkanın konumunu, $x_{chase,j1}$, $x_{chase,j2}$ ve $x_{chase,j3}$ kovalama aşamasında birbirlerinden farklı olarak rastgele seçilen üç orkanın konumunu ve x_i i'inci orkanın konumunu gösterir. $x_{attack1,i}$ ve $x_{attack2,i}$, sırasıyla, avı avlamak isteyen ve muhafazaya geri dönmek isteyen orkaların hesaplanan konumlarıdır. $x_{attack,i}$ i[th] orkanın saldırı aşamasındaki konumudur. OX1 sıralı çaprazlama operatörünü temsil eder ki bu operatör altbölümlerde detaylı olarak açıklanır. İki ebeveyn orkayı çaprazlamak ve iki yavru orka elde etmek için kullanılır. FindMin ise kendisine parametre olarak gönderilen orkalar içerisinden en iyi uygunluk değerine sahip orkayı bulmak için kullanılır. Örneğin, Denklem 26'da $x_{chase,i}$, $x_{attack1,i}$ ve $x_{attack2,i}$ orkalarından en iyi uygunluk değerine sahip olan orka FindMin ile belirlenir. Bu orka saldırı aşamasındaki i'inci orkanın yeni konumu olarak tespit edilir.

OPA'da saldırı aşamasında kullanılan pozisyon ayarlama prosedürü DOPA'da da benzer şekilde kullanılır. Ancak ayrık problemlerin doğası gereği bir orkanın konumunun problemin uygun aralığının minimum sınır değeri (lb) olarak belirlenmesi mümkün değildir. Bunun yerine orkanın yeni konumu Denklem 27 ve 28 kullanılarak belirlenir.

$$x_{min} = FindMin\ [x] \tag{27}$$

$$x_i = Swap(x_{min}) \tag{28}$$

Burada, FindMin orka popülasyonundaki en iyi uygunluk değerine sahip orkayı bulmak için kullanılır. Swap ise bulunan en iyi orkaya takas lokal arama operatörünü ki bu operatör altbölümlerde detaylı olarak açıklanır, uygular. Saldırı aşamasında kullanılan konum ayarlama prosedürü Algoritma 2'de verilir.

Algoritma 2. DOPA'da bir saldırı sırasında konum ayarlama aşaması

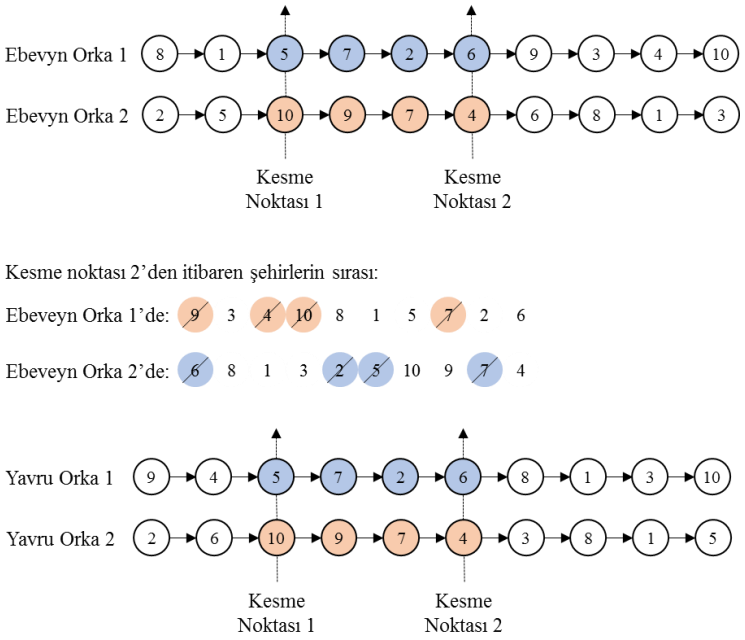
---

if f$(x_{attack,i}) <$ f$(x_{chase,i})$
  $x_i = x_{attack,i}$
else
  if rand < p2
    $x_{min} = FindMin\ [x]$
    $x_i = Swap(x_{min})$
  else
    $x_i = x_{chase,i}$
  end if
end if

Burada, p2 [0, 1] aralığında seçilen bir sabittir. Bu prosedürde kullanılan takas lokal arama operatörü birtakım deneyler sonucunda tespit edilmiştir.

5.4.1. Sıralı Çaprazlama (OX1) Operatörü

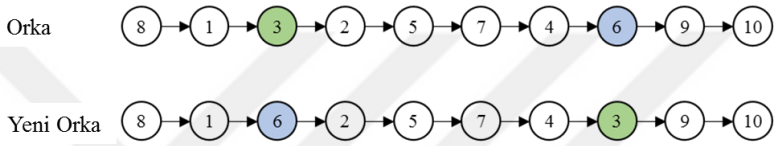
OX1 operatörü (Davis, 1985) iki ebeveyn orkadan iki yavru orka elde etmek için kullanılır. Öncelikle ebeveyn orkalar üzerinde rasgele iki kesme noktası seçilir ve bu kesme noktaları arasındaki alt turlar yavru orkalara kopyalanır. Devamında, birinci ebeveyn orkanın ikinci kesme noktasından başlanarak şehirler, ikinci yavru orkanın ikinci kesme noktasından itibaren sırasıyla aktarılır. Dizinin sonuna gelindiğinde kopyalama işlemine ikinci yavru orkanın ilk konumundan itibaren devam edilir. Bu sırada, ikinci yavru orkanın iki kesme noktası arasındaki şehirlerin dizide tekrar yer almamasına özellikle dikkat edilir. Bunun için tamir operatörü kullanılır. Bu operatör, ikinci kesim noktasından itibaren şehirleri aktarırken tekrar eden şehirleri çıkarır. İlk yavru orkanın elde edilmesi için de aynı prosedür uygulanır. OX1 operatörünün uygulanışı ve tamir işlemi Şekil 5.6'da gösterilmiştir. Şekilden de gözlemlendiği üzere ilk yavru orka, ikinci ebeveyn orkanın kesme noktaları dışında kalan şehirler ve ilk ebeveyn orkanın kesme noktaları içinde kalan şehirler dikkate alınarak oluşturulur. İkinci yavru orka ise, ilk ebeveyn orkanın kesme noktaları dışında kalan şehirler ve ikinci ebeveyn orkanın kesme noktaları içinde kalan şehirler dikkate alınarak oluşturulur.



Şekil 5.6. OX1 operatörünün uygulanışı (İlhan ve Gökmen, 2022)

5.4.2. Takas (Swap) Operatörü

Takas operatörü permütasyon kodlu optimizasyon problemlerinde yaygın olarak kullanılan lokal arama operatörüdür. Bu operatör DOPA'da saldırı aşamasında konum ayarlama prosedüründe kullanılmıştır ve Şekil 5.7'daki gibi uygulanmıştır. Şekilden de görülebildiği gibi herhangi bir orka üzerinde rastgele iki şehir seçilir. Bu şehirler takas edilir ve yeni orka elde edilir. Elde edilen yeni orka, araştırma uzayı içerisinde, başlangıçtaki orkanın komşuluğunda bir orka olur.



Şekil 5.7. Takas operatörünün uygulanışı

6. ARAŞTIRMA SONUÇLARI VE TARTIŞMA

DOPA, Matlab 2019a'da kodlanmıştır. Deneyler, Windows 10 Home Single Language 64 Bit İşletim Sistemine sahip bir masaüstü bilgisayarda uygulanmıştır. Bilgisayar Intel Core i5-10300 2.50 GHz CPU'ya ve 16 GB RAM'e sahiptir. DOPA'nın performansı TSPLIB (Symmetric TSPs, 2023) kütüphanesinden alınan, araştırmacılar tarafından yaygın olarak kullanılan ve iyi bilinen 67 TSP örneği üzerinde test edilmiştir. Bu örneklerin en küçüğü 14 şehir, en büyüğü ise 1002 şehir içermektedir. Seçilen TSP örneklerinin küçük, orta ve büyük ölçekli örneklerden oluşmasına özellikle dikkat edilmiştir.

6.1. Parametre Optimizasyonu

Taguchi yöntemi, deneysel tasarım yapmak için Genichi Taguchi (Taguchi, 1986) tarafından önerilmiştir. Bu yöntem parametreleri kontrol edilebilir ve kontrol edilemeyen parametreler olarak iki gruba ayırır ve istatistiksel bir analiz yapar. Kontrol edilebilen parametreleri kararlılık açısından değerlendirirken kontrol edilemeyen parametreleri de ortadan kaldırmaya çalışır. Bunun için parametrelerin algoritmanın performansına olan etkilerini inceler (Mozdgir ve ark., 2013).

Bu çalışmada Taguchi yöntemi DOPA'nın parametrelerini ayarlamak için kullanılmıştır. DOPA beş parametreye sahiptir. İlk parametre iterasyon sayısıdır ve tüm Taguchi deneylerinde 500 olarak kabul edilmiştir. Geriye kalan parametrelerin her biri dört seviyeye sahiptir. Çizelge 6.1'de bu parametrelere ait seviyeler görülmektedir. Bu seviyelere göre Taguchi yöntemi, 16 deneyi içeren L16 ortogonal dizisini üretmiştir. Her bir deney için farklı boyutlarda 15 TSP örneği kullanılmıştır. Güvenilirliği artırmak için her bir örnek üzerinde 10 çalıştırma uygulanmıştır. Çizelge 6.2'de bu çalıştırmalara ait ortalama değerler verilmiştir. Bu değerler Minitab 19 yazılımı ile analiz edilmiş ve sinyal-gürültü (S/N) oranlarına dönüştürülmüştür. S/N oranı, ortalama değerin (sinyal) standart sapmaya (gürültü) oranıdır. S/N oranının hesaplanmasında en yüksek değer formülü kullanılmıştır ve Denklem 29'da verilmiştir (Ghani ve ark., 2004). Burada n deney sayısını, y_i karakteristik değeri (kesme kuvvetini) gösterir.

$$\frac{S}{N}: -10\log\left(\frac{1}{n}\sum_{i:1}^{n}\frac{1}{y_i^2}\right) \tag{29}$$

S/N oranları Çizelge 6.3'te sunulmuştur. Çizelgede her bir parametreye ait maksimum değerler koyu ile gösterilmiştir. Koyu değerler her bir parametrenin sahip olduğu en iyi seviyeleri gösterir. Bu seviyelere karşılık gelen parametre değerlerine Çizelge 6.1'den bakıldığında PS = 50, p1 = 0,2, q = 0,4 ve p2 = 0,9 olması gerektiği anlaşılır. Bu değerler Taguchi yöntemi ile DOPA için bulunan en iyi parametre değerleridir. Çizelge 6.3'te "Delta" satırındaki değerler herhangi bir sütundaki en büyük değer ile en küçük değerin farkı alınarak hesaplanır. Bu değerlere göre "Rank" satırı bulunur ve parametrelerin önem sırasını gösterir. Bu satıra göre, DOPA'nın performansını en çok etkileyen parametrenin p2, en az etkileyen parametrenin q olduğu görülmektedir.

Çizelge 6.1. DOPA parametreleri ve seviyeleri

Parametre İsim	Sembol	Seviye 1	Seviye 2	Seviye 3	Seviye 4
Popülasyon büyüklüğü	PS	20	30	40	50
Avın sürülme ve kuşatılma davranışlarının seçilme olasılığı	p1	0.2	0.4	0.6	0.8
İki sürüş yönteminden birinin seçilme olasılığı	q	0.4	0.6	0.8	0.9
Bir parçacığın konum ataması yapma olasılığı	p2	0.1	0.3	0.6	0.9

Çizelge 6.2. Ortogonal dizi L16 için elde edilen ortalama değerler

Ortogonal Dizi					Örnekler													
#	PS	p1	q	p2	eil51	berlin52	pr76	ch130	rat99	kroE100	eil101	pr144	tsp225	pr226	gil262	u574	linhp318	pcb442	rat783
1	20	0.2	0.4	0.1	426.7	7542.0	108327	6162.3	1216.2	22102.6	637.8	58537	3987.8	80399.4	2405.8	37869.2	42598.4	51636.1	9101.3
2	20	0.4	0.6	0.3	427.1	7542.0	108433	6142.0	1216.5	22134.7	634.0	58537	3962.7	80369.8	2399.7	37630.1	42459.5	51452.7	9040.7
3	20	0.6	0.8	0.6	426.7	7542.0	108159	6145.6	1213.5	22119.9	634.6	58537	3946.0	80369.8	2400.3	37479.1	42433.4	51424.5	9005.4
4	20	0.8	0.9	0.9	426.3	7542.0	108340	6146.5	1212.8	22129.6	632.9	58537	3956.8	80370.2	2395.2	37527.7	42344.4	51266.5	8991.9
5	30	0.2	0.6	0.6	427.1	7542.0	108159	6124.0	1212.0	22112.3	631.0	58537	3955.9	80369.4	2389.2	37497.5	42362.9	51276.2	8945.4
6	30	0.4	0.4	0.9	426.7	7542.0	108159	6125.2	1211.1	22114.6	632.7	58537	3946.4	80369.0	2388.7	37353.9	42266.6	51268.0	8951.6
7	30	0.6	0.9	0.1	427.0	7542.0	108344	6147.1	1214.7	22117.5	634.7	58537	3970.2	80482.5	2408.0	37973.3	42595.3	51768.3	9107.7
8	30	0.8	0.8	0.3	427.0	7542.0	108336	6141.6	1213.1	22132.8	633.5	58537	3966.6	80370.6	2398.0	37601.2	42414.6	51563.0	9036.0
9	40	0.2	0.8	0.9	426.8	7542.0	108159	6126.9	1213.1	22112.7	631.6	58537	3950.1	80369.4	2397.2	37421.2	42272.4	51198.6	8935.2
10	40	0.4	0.9	0.6	426.6	7542.0	108159	6145.2	1211.9	22101.0	630.8	58537	3956.9	80369.4	2389.2	37492.2	42384.8	51319.4	8946.0
11	40	0.6	0.4	0.3	426.4	7542.0	108159	6134.6	1212.6	22109.3	632.7	58537	3959.4	80369.4	2391.0	37529.0	42373.2	51376.0	8981.2
12	40	0.8	0.6	0.1	427.4	7559.3	108252	6158.6	1215.0	22125.7	636.6	58537	3973.3	80450.6	2408.1	37714.6	42647.3	51607.9	9144.3
13	50	0.2	0.9	0.3	426.6	7542.0	108159	6127.5	1212.6	22126.2	634.3	58537	3946.8	80369.4	2397.9	37455.4	42398.3	51328.5	8962.9
14	50	0.4	0.8	0.1	426.7	7565.3	108252	6143.9	1214.1	22140.0	633.8	58537	3962.7	80370.2	2400.3	37622.4	42382.7	51393.9	9034.2
15	50	0.6	0.6	0.9	427.1	7542.0	108159	6122.9	1211.9	22119.9	632.3	58537	3947.4	80369.0	2390.3	37357.2	42259.9	51296.4	8929.8
16	50	0.8	0.4	0.6	426.7	7542.0	108159	6131.2	1211.0	22116.1	631.3	58537	3963.5	80369.8	2391.5	37438.8	42244.9	51332.7	8937.0

Çizelge 6.3. DOPA parametreleri için S/N oranları

Seviye	PS	p1	q	p2
1	-92.7375	**-92.7307**	**-92.7307**	-92.7439
2	-92.7352	-92.7314	-92.7338	-92.7341
3	-92.7306	-92.7335	-92.7313	-92.7270
4	**-92.7273**	-92.7351	-92.7348	**-92.7256**
Delta	0.0101	0.0044	0.0041	0.0183
Rank	2	3	4	1

6.2. Operatör Seçimi

OPA'da saldırı aşamasında orkaların pozisyonları p2 parametresi ve rasgele üretilen değere bağlı olarak problem uzayının minimum sınır değeri olarak belirlenir (Algoritma 1). TSP için minimum veya maksimum sınır değeri olmadığı için ayrık versiyonda bu işlem ya rastgele bir rota üretilerek yapılır ya da mevcut rotalar üzerinde lokal arama operatörleri kullanılarak değişiklik yapılır. Böylece yeni bir rota elde edilir. DOPA'da bu aşamada, Algoritma 2'den de görülebildiği gibi, takas operatörü kullanılmıştır. Ancak bu operatörün kullanımına karar vermek için birtakım deneyler yapılmıştır. Deneylerde takas (swap), ekleme (insertion) ve ters çevirme (reversion) lokal arama operatörlerinin performansları ve rastgele bir rota üretimini temsil eden rastgele (random) operatörünün performansı karşılaştırılmıştır. Deneylerde farklı boyutlarda 15 TSP örneği kullanılmış ve her bir örnek üzerinde 10 çalıştırma uygulanmıştır. Elde edilen sonuçlar Çizelge 6.4'te sunulmuştur. Bu çizelgede "Instance" TSP örneklerinin adını, "Optimum" bilinen en iyi sonucu ifade eder. "Random", "Swap", "Insertion" ve "Reversion" bu operatörler ile elde edilen sonuçları gösterir. "PDB" ve "PDW", sırasıyla, tüm çalıştırmalar içerisindeki en iyi ve en kötü çözümlerin Optimum'a göre yüzde hatalarını temsil eder (Denklem 30 ve 32). "PDA", tüm çalıştırmaların ortalama değerinin Optimum'a göre yüzde hatasını gösterir (Denklem 31). "Average" satırı ise ilgili sütunlardaki değerlerin ortalamasını temsil eder. Average satırındaki değerlerden de görülebildiği gibi takas operatörü PDB ve PDA açısından diğer operatörlere kıyasla daha düşük değerlere sahiptir. Yani daha iyi performans göstermiştir. Bu nedenle DOPA'nın saldırı aşamasında pozisyon ayarlama fazında bu operatör kullanılmıştır.

$$PDB = \frac{(\text{Best} - \text{Optimum})}{\text{Optimum}} * 100, \qquad Best\text{: Bütün çalıştırmalar içerisindeki en iyi çözüm} \tag{30}$$

$$PDA = \frac{(\text{Avg.} - \text{Optimum})}{\text{Optimum}} * 100, \qquad Avg.\text{: Bütün çalıştırmaların ortalama değeri} \tag{31}$$

$$PDW = \frac{(\text{Worst} - \text{Optimum})}{\text{Optimum}} * 100, \qquad Worst\text{: Bütün çalıştırmalar içerisindeki en kötü çözüm} \tag{32}$$

Çizelge 6.4. Operatörlerin performanslarının 15 örnek üzerinde karşılaştırılması

Instance	Optimum	Random			Swap			Insertion			Reversion		
		PDB (%)	PDA (%)	PDW (%)	PDB (%)	PDA (%)	PDW (%)	PDB (%)	PDA (%)	PDW (%)	PDB (%)	PDA (%)	PDW (%)
eil51	426	0.00	0.14	0.23	0.00	0.12	0.23	0.00	**0.09**	0.23	0.00	0.28	1.41
berlin52	7542	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
pr76	108159	0.00	**0.00**	0.00	0.00	**0.00**	0.00	0.00	0.09	0.86	0.00	**0.00**	0.00

ch130	6110	0.00	**0.40**	0.93	0.00	0.47	1.19	0.00	0.60	1.21	0.00	0.42	0.74
rat99	1211	0.00	**0.02**	0.08	0.00	0.12	0.58	0.00	0.06	0.58	0.00	0.20	0.58
kroE100	22068	0.00	0.21	0.24	0.00	**0.20**	0.33	0.00	0.26	0.78	0.00	0.22	0.33
eil101	629	0.00	**0.27**	0.79	0.00	0.64	1.91	0.00	0.54	1.27	0.00	0.73	1.59
pr144	58537	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
tsp225	3916	0.10	0.98	1.92	0.00	**0.53**	1.28	0.08	0.87	1.46	0.36	0.85	1.25
pr226	80369	0.00	**0.00**	0.00	0.00	**0.00**	0.00	0.00	**0.00**	0.00	0.00	0.03	0.29
gil262	2378	0.00	0.72	1.43	0.08	**0.52**	1.47	0.04	0.54	0.93	0.04	0.56	1.09
u574	36905	0.75	1.44	1.97	0.14	1.25	1.82	0.54	**1.07**	1.88	0.64	1.14	1.89
linhp318	41345	1.65	2.24	2.75	1.71	2.43	3.15	1.65	2.33	2.95	1.86	**2.20**	2.75
pcb442	50778	0.70	1.29	1.82	0.44	0.87	1.59	0.45	1.12	1.75	0.45	**0.79**	1.37
rat783	8806	1.67	2.26	2.79	1.00	**1.45**	1.86	1.15	1.69	2.06	1.11	1.47	1.91
Average		0.32	0.66	0.99	0.22	**0.56**	1.03	0.26	0.61	1.06	0.29	0.59	1.01

DOPA'nın saldırı aşamasında, iki ebeveyn orkayı çaprazlamak ve iki yavru orka elde etmek için kullanılacak operatöre birtakım deneyler yapılarak karar verilmiştir. Deneylerde OX1 ve ER (Edge Recombination) operatörlerinin performansları karşılaştırılmıştır. Elde edilen sonuçlar, OX1 operatörünün optimum çözüme daha hızlı yakınsadığını ve daha iyi sonuçlar elde ettiği göstermiştir. Bu nedenle, DOPA'da çaprazlama operatörü olarak OX1 kullanılmıştır.

6.3. Diğer Yöntemlerle Karşılaştırma

Çizelge 6.5, farklı boyutlardaki TSP örnekleri için DOPA tarafından yuvarlatılmış Öklid mesafeleri kullanılarak hesaplanan sonuçları göstermektedir. Güvenilir sonuçlar elde etmek için her örnek üzerinde 20 çalıştırma uygulanmıştır. Bu çizelgede "Instance" TSP örneklerinin adını, "Optimum" bilinen en iyi çözümü, "Best" ve "Worst", sırasıyla, tüm çalıştırmalar içerisindeki en iyi ve en kötü çözümü, "Avg." tüm çalıştırmaların ortalama değerini ifade eder. "PDB" ve "PDW", sırasıyla, tüm çalıştırmalar içerisindeki en iyi ve en kötü çözümlerin Optimum'a göre yüzde hatalarını temsil eder. "PDA", tüm çalıştırmaların ortalama değerinin Optimum'a göre yüzde hatasını gösterir. "SD" ve "Time (s)", sırasıyla, standart sapmayı ve saniye cinsinden tüm çalıştırmaların ortalama süresini temsil eder. Bu çizelgeden DOPA'nın 67 örneğin 50'sinde en iyi sonucu elde ettiği görülmektedir. Geriye kalan 17 örneğin 14'ünde ise, PDB %1'lik dilim içerisindedir. PDA, 67 örneğin 23'ünde %0.00'dır. Geriye kalan 44 örneğin 31'inde ise PDA %1'lik dilim içerisindedir. Time açısından bakıldığında örneklerin boyutu artarken DOPA'nın çalışma süresinin de doğal olarak arttığı görülmektedir.

Çizelge 6.5. 67 örnek için DOPA'nın sayısal sonuçları

No	Instance	Optimum	Best	Avg.	Worst	PDB (%)	PDA (%)	PDW (%)	SD	Time (s)
1	burma14	3323	3323	3323.00	3323	0.00	0.00	0.00	0.00	3.79
2	ulysses16	6859	6859	6859.00	6859	0.00	0.00	0.00	0.00	3.86
3	gr17	2085	2085	2085.00	2085	0.00	0.00	0.00	0.00	3.89
4	ulysses22	7013	7013	7013.00	7013	0.00	0.00	0.00	0.00	4.15
5	bays29	2020	2020	2020.00	2020	0.00	0.00	0.00	0.00	4.46
6	dantzig42	699	699	699.00	699	0.00	0.00	0.00	0.00	5.29
7	swiss42	1273	1273	1273.00	1273	0.00	0.00	0.00	0.00	5.24
8	att48	10628	10628	10628.00	10628	0.00	0.00	0.00	0.00	7.42
9	rand50	5553	5553	5553.00	5553	0.00	0.00	0.00	0.00	6.00
10	eil51	426	426	426.80	427	0.00	0.19	0.23	0.41	5.97
11	berlin52	7542	7542	7542.00	7542	0.00	0.00	0.00	0.00	6.13
12	st70	675	675	675.00	675	0.00	0.00	0.00	0.00	7.58
13	eil76	538	538	539.70	546	0.00	0.32	1.49	2.90	8.17
14	pr76	108159	108159	108159.00	108159	0.00	0.00	0.00	0.00	8.39
15	gr96	55209	55209	55265.30	55403	0.00	0.10	0.35	61.73	10.51
16	rat99	1211	1211	1211.10	1213	0.00	0.01	0.17	0.45	10.47
17	kroA100	21282	21282	21282.00	21282	0.00	0.00	0.00	0.00	10.84
18	kroB100	22141	22141	22160.30	22237	0.00	0.09	0.43	31.28	10.89
19	kroC100	20749	20749	20749.00	20749	0.00	0.00	0.00	0.00	10.83
20	kroD100	21294	21294	21294.00	21294	0.00	0.00	0.00	0.00	10.79
21	kroE100	22068	22068	22114.20	22121	0.00	0.21	0.24	16.45	10.81
22	rd100	7910	7910	7911.70	7944	0.00	0.02	0.43	7.60	10.89
23	eil101	629	629	631.35	636	0.00	0.37	1.11	2.35	10.61
24	lin105	14379	14379	14379.00	14379	0.00	0.00	0.00	0.00	11.76
25	pr107	44303	44303	44303.00	44303	0.00	0.00	0.00	0.00	11.33
26	pr124	59030	59030	59030.00	59030	0.00	0.00	0.00	0.00	13.95
27	bier127	118282	118282	118579.90	120073	0.00	0.25	1.51	476.73	14.94
28	ch130	6110	6110	6131.75	6171	0.00	0.36	1.00	22.81	15.43
29	pr136	96772	96772	96830.15	96920	0.00	0.06	0.15	68.06	15.59
30	gr137	69853	69853	69853.00	69853	0.00	0.00	0.00	0.00	15.73
31	pr144	58537	58537	58537.00	58537	0.00	0.00	0.00	0.00	17.54
32	ch150	6528	6528	6549.30	6566	0.00	0.33	0.58	11.83	18.57
33	kroA150	26524	26524	26529.35	26621	0.00	0.02	0.37	21.58	18.62
34	kroB150	26130	26130	26140.90	26218	0.00	0.04	0.34	26.59	18.25
35	pr152	73682	73682	73791.20	73818	0.00	0.15	0.18	55.02	19.11
36	u159	42080	42080	42111.60	42396	0.00	0.08	0.75	97.26	19.37
37	rat195	2323	2323	2335.35	2349	0.00	0.53	1.12	7.16	26.50
38	d198	15780	15780	15791.20	15841	0.00	0.07	0.39	14.88	26.83
39	kroA200	29368	29368	29443.70	29721	0.00	0.26	1.20	107.67	28.98
40	kroB200	29437	29437	29515.85	29870	0.00	0.27	1.47	126.74	29.08
41	gr202	40160	40160	40309.65	40626	0.00	0.37	1.16	141.46	36.40
42	tsp225	3916	3916	3949.55	3997	0.00	0.86	2.07	18.96	20.35
43	ts225	126643	126643	126643.00	126643	0.00	0.00	0.00	0.00	37.36
44	pr226	80369	80369	80369.00	80369	0.00	0.00	0.00	0.00	33.26
45	gr229	134602	134616	135047.85	135647	0.01	0.33	0.78	318.13	35.18

46	gil262	2378	2378	2388.40	2404	0.00	0.44	1.09	8.23	46.21
47	pr264	49135	49135	49135.00	49135	0.00	0.00	0.00	0.00	42.89
48	a280	2579	2579	2585.80	2604	0.00	0.26	0.97	10.68	48.24
49	pr299	48191	48191	48292.60	48627	0.00	0.21	0.90	114.20	69.27
50	lin318	42029	42029	42283.55	42465	0.00	0.61	1.04	116.39	81.72
51	linhp318	41345	42143	42411.25	42760	1.93	2.58	3.42	181.14	81.88
52	rd400	15281	15352	15415.20	15494	0.46	0.88	1.39	39.69	105.76
53	fl417	11861	11862	11873.35	11878	0.01	0.10	0.14	4.88	96.34
54	gr431	171414	172836	173594.80	174908	0.83	1.27	2.04	502.14	115.33
55	pr439	107217	107277	107624.85	108612	0.06	0.38	1.30	444.62	112.66
56	pcb442	50778	51138	51324.35	51607	0.71	1.08	1.63	120.27	120.88
57	d493	35002	35148	35381.65	35617	0.42	1.08	1.76	120.31	145.70
58	att532	27686	27822	27996.20	28175	0.49	1.12	1.77	97.10	177.06
59	ali535	202310	202442	204457.65	207616	0.07	1.06	2.62	1445.53	186.14
60	u574	36905	37068	37347.80	37582	0.44	1.20	1.83	135.35	192.84
61	rat575	6773	6843	6877.45	6938	1.03	1.54	2.44	26.41	195.72
62	p654	34643	34643	34663.75	34689	0.00	0.06	0.13	9.56	227.09
63	d657	48912	49101	49333.50	49596	0.39	0.86	1.40	135.05	259.68
64	gr666	294358	296239	299088.05	302022	0.64	1.61	2.60	1631.01	273.10
65	u724	41910	42171	42387.65	42633	0.62	1.14	1.73	103.64	316.95
66	rat783	8806	8946	8978.95	9020	1.59	1.96	2.43	21.23	378.06
67	pr1002	259045	261488	263527.25	265602	0.94	1.73	2.53	1079.57	622.01

Çizelge 6.6, 32 örnek üzerinde DOPA, VTPSO (Akhand ve ark., 2015), ABCSS (Khan ve Maiti, 2019), DSMO (Akhand ve ark., 2020) ve LBSA-CO (İlhan ve Gökmen, 2022)'nun sonuçlarını göstermektedir. DOPA'nın sonuçları her bir örnek üzerinde 20 kez çalıştırılarak elde edilmiştir. LBSA-CO'nun sonuçları orijinal sayfasından, diğer yöntemlerin sonuçları ise Akhand ve ark. (2020) tarafından uygulanan çalışmadan alınmıştır. İlgili sayfalarda VTPSO, ABCSS, DSMO ve LBSA-CO'nun sonuçlarının her bir örnek üzerinde 20 kez çalıştırılarak elde edildiği raporlanmaktadır. Çizelge kayan noktalı Öklid mesafeleri kullanılarak hesaplanan sonuçları içermektedir. Çizelgeden de görülebildiği gibi Best açısından DOPA, 32 örneğin tamamında VTPSO'dan, 31'inde hem ABCSS hem de DSMO'dan, 12'sinde ise LBSA-CO'dan daha iyi performans göstermiştir. Benzer durum Average satırındaki değerlerden de görülebilmektedir. Bu satırdaki değerlere göre DOPA'nın Best, Avg. ve SD'deki ortalama değerleri diğer yöntemlerin tamamından daha iyidir. Çizelgedeki "W/D/L" satırı, sırasıyla galibiyet (win), beraberlik (draw) ve mağlubiyet (loss) anlamına gelir ve ortalama değerler açısından DOPA ile diğer yöntemler arasında bir karşılaştırma sunar.

Çizelge 6.6. 32 örnekte DOPA ve diğer yöntemlerin performans karşılaştırması

Instance	VTPSO			ABCSS			DSMO			LBSA-CO			DOPA		
	Best	Avg.	SD	Best	Avg.	SD	Best	Avg.	SD	Best	Avg.	SD	Best	Avg.	SD
gr17	2332.58	2332.58	0.00	2332.58	2332.58	0.00	2332.58	2332.58	0.00	**2085.00**	**2085.00**	0.00	**2085.00**	**2085.00**	0.00
eil51	429.51	439.87	5.65	428.98	437.01	4.98	428.86	436.96	4.73	428.86	428.97	0.14	**428.86**	**428.94**	0.06
berlin52	7544.37	7728.00	161.80	7544.37	7807.86	177.55	7544.37	7633.60	85.40	**7544.32**	**7544.32**	0.00	**7544.32**	**7544.32**	0.00
st70	682.57	711.50	14.57	682.57	690.50	5.35	677.11	702.64	15.04	**677.06**	**677.06**	0.00	**677.06**	**677.06**	0.00
eil76	559.25	569.10	9.12	550.24	561.48	7.21	558.68	572.70	7.56	544.33	545.52	2.29	**544.33**	**545.23**	1.97
pr76	109586.10	112549.20	1420.00	108879.70	109758.57	850.64	108159.40	111299.30	2050.48	**108159.40**	**108159.40**	0.00	**108159.40**	**108159.40**	0.00
rat99	1256.25	1325.80	34.83	1242.32	1265.93	13.54	1225.56	1291.93	21.07	1219.25	1221.40	2.11	**1219.25**	**1219.94**	1.77
kroa100	21307.44	22400.48	529.13	21299.00	21878.83	455.63	21298.21	22024.27	508.89	21285.47	21296.26	25.80	**21285.47**	**21285.47**	0.00
krob100	22475.67	23236.43	522.63	22229.71	22707.96	259.83	22308.00	23022.37	277.32	22139.03	22170.84	38.04	**22139.03**	22175.85	31.90
rd100	8094.75	8502.73	167.70	7944.32	8207.80	172.52	8041.30	8377.76	209.40	7910.41	7926.51	33.73	**7910.41**	**7913.80**	10.44
eil101	653.16	679.14	13.29	646.05	662.63	7.13	648.66	674.40	10.97	640.13	644.76	3.24	**640.13**	**641.73**	2.79
lin105	14581.58	15684.64	610.38	14406.12	14766.55	263.01	14383.00	15114.00	500.76	14382.94	14386.41	8.47	**14382.94**	**14382.94**	0.00
pr107	44436.25	45287.90	1207.52	44525.68	44927.27	319.03	44385.86	45666.99	1300.43	44301.64	44331.93	60.20	**44301.64**	**44301.64**	0.00
pr124	61076.73	63939.97	1739.40	**59030.74**	**59772.68**	516.56	60285.21	62443.49	1644.93	59030.75	59041.77	19.57	59030.75	59030.75	0.00
pr136	99247.01	102945.70	1688.20	97853.91	101795.57	1916.45	97538.68	102872.00	2855.28	96795.51	97221.60	230.97	**96771.04**	**96787.04**	33.79
kroa150	27232.10	28262.98	455.95	26981.98	27971.36	554.50	27591.44	28354.09	524.91	26524.85	26586.03	74.29	**26524.85**	**26534.60**	43.45
krob150	26579.73	27987.85	620.72	26760.79	27653.49	509.24	26601.94	27576.16	625.26	26127.37	26218.44	107.37	**26127.37**	**26137.41**	22.46
pr152	74414.17	76272.37	1199.90	74337.62	76097.48	904.45	74243.91	76526.77	1663.08	73683.52	73814.66	99.80	**73683.52**	**73773.14**	67.10
u159	43579.82	45441.55	1178.06	42862.51	45234.92	1212.54	42598.30	42598.30	0.00	**42075.73**	**42163.52**	147.06	42075.73	42164.62	141.77
rat195	2452.92	2554.10	47.56	2469.31	2579.27	44.14	2372.89	2488.55	50.48	2342.82	2357.47	10.31	**2333.85**	**2345.16**	6.62
d198	16066.44	16489.28	206.79	16270.22	16483.73	162.08	15978.13	16270.47	171.20	15818.02	15860.97	32.64	**15808.65**	**15823.07**	10.89
kroa200	30602.81	31502.33	531.41	30701.86	31938.81	656.84	30481.35	31828.64	652.32	29369.37	29472.87	90.86	**29369.37**	**29438.46**	115.92
krob200	30767.52	31923.15	553.02	31508.85	32208.73	526.77	30716.50	31781.62	487.39	29450.48	29595.98	145.68	**29440.40**	**29507.66**	94.26
tsp225	4095.01	4210.65	66.44	4140.24	4276.92	72.80	4013.68	4162.79	66.08	**3867.81**	**3922.14**	28.45	3916	3942,60	18,57
pr226	81050.23	88031.31	3461.96	82266.00	87400.60	3482.64	83587.98	85935.69	2105.13	80370.27	80430.30	144.03	**80370.27**	**80370.27**	0.00
gil262	2547.16	2631.69	39.44	2713.75	2839.11	73.01	2543.15	2627.87	42.39	2397.34	2422.19	15.85	**2385.76**	**2396.28**	8.43
pr299	50571.83	53154.34	1389.36	64464.76	67620.95	2092.63	50579.82	51747.99	863.32	48248.90	48525.24	200.33	**48194.82**	**48292.37**	119.01
lin318	44724.38	46209.53	773.01	55744.52	61902.84	3824.65	44118.66	45460.25	660.47	42094.04	42710.49	219.32	**42042.45**	**42370.90**	197.11
linhp318	44337.02	46329.87	948.05	56834.19	60853.66	2306.85	43831.44	45730.57	929.73	42062.38	42606.31	274.77	**42042.45**	**42328.45**	177.02
fl417	12376.53	12980.24	311.88	22253.99	27237.13	2479.90	12218.98	12950.77	360.99	11934.12	11956.66	28.96	**11914.24**	**11926.32**	4.94
pr439	112088.00	119442.20	2770.58	206233.14	244192.44	21576.50	112105.20	116379.20	2462.82	107573.76	108506.44	646.96	**107248.99**	**107526.57**	339.88
d493	37132.09	38159.18	603.84	68556.68	78968.31	5776.85	36844.63	37861.14	426.97	35376.00	35781.19	198.47	**35229.69**	**35409.56**	126.85
Average	32340.03	33747.36	727.57	37646.77	40407.28	1600.81	32195.11	33273.31	674.53	31451.90	31581.65	90.30	**31430.35**	**31482.02**	49.47
W/D/L		0/0/32			1/0/31			0/0/32			2/4/26				

Çizelge 6.7'de DOPA ve LBSA-CO'nun 65 örnek üzerindeki sonuçları sunulmuştur. LBSA-CO'nun sonuçları, Time sütunundakiler hariç, orijinal sayfasından alınmıştır. Time sütunundaki sonuçlar ise adil bir karşılaştırma yapabilmek için DOPA'nın sonuçlarını elde etmek için kullanılan bilgisayar ile elde edilmiştir. Çizelgeden de görülebildiği gibi DOPA 65 örneğin 49'unda optimum sonucu elde etmiştir. PDB açısından DOPA'nın başarı oranı %75'tir. LBSA-CO ise 65 örneğin 37'sinde optimum sonucu elde etmiştir. Bu yöntemin başarı oranı ise %57'dir. Çizelgedeki Time sütunundan da görülebildiği gibi LBSA-CO küçük ve orta ölçekli örnekler üzerinde DOPA'dan daha hızlı çalışmaktadır. Ancak örneğin boyutu büyüdükçe LBSA-CO ile DOPA'nın çalışma süreleri eşitlenmekte, hatta büyük ölçekli örneklerde DOPA daha hızlı sonuç üretmektedir.

Çizelge 6.7. DOPA ve LBSA-CO yöntemlerinin 65 örnek üzerinde performans karşılaştırması

Instance	Optimum	LBSA-CO					DOPA				
		PDB (%)	PDA (%)	PDW (%)	SD	Time (s)	PDB (%)	PDA (%)	PDW (%)	SD	Time (s)
burma14	3323	**0.00**	**0.00**	0.00	0.00	3.61	**0.00**	**0.00**	0.00	0.00	3.79
ulysses16	6859	**0.00**	**0.00**	0.00	0.00	3.53	**0.00**	**0.00**	0.00	0.00	3.86
gr17	2085	**0.00**	**0.00**	0.00	0.00	3.78	**0.00**	**0.00**	0.00	0.00	3.89
ulysses22	7013	**0.00**	**0.00**	0.00	0.00	4.11	**0.00**	**0.00**	0.00	0.00	4.15
bays29	2020	**0.00**	**0.00**	0.00	0.00	4.24	**0.00**	**0.00**	0.00	0.00	4.46
dantzig42	699	**0.00**	**0.00**	0.00	0.00	5.18	**0.00**	**0.00**	0.00	0.00	5.29
swiss42	1273	**0.00**	**0.00**	0.00	0.00	5.00	**0.00**	**0.00**	0.00	0.00	5.24
rand50	5553	**0.00**	**0.00**	0.00	0.00	5.73	**0.00**	**0.00**	0.00	0.00	6.00
eil51	426	**0.00**	**0.19**	0.47	0.52	5.45	**0.00**	**0.19**	0.23	0.41	5.97
berlin52	7542	**0.00**	**0.00**	0.00	0.00	5.35	**0.00**	**0.00**	0.00	0.00	6.13
st70	675	0.00	0.39	1.33	2.85	6.63	**0.00**	**0.00**	0.00	0.00	7.58
eil76	538	0.00	0.44	1.86	3.53	6.88	**0.00**	**0.32**	1.49	2.90	8.17
pr76	108159	0.00	0.17	0.86	380.02	6.41	**0.00**	**0.00**	0.00	0.00	8.39
gr96	55209	0.00	0.27	0.68	122.07	7.47	**0.00**	**0.10**	0.35	61.73	10.51
rat99	1211	0.00	0.52	1.57	5.26	7.81	**0.00**	**0.01**	0.17	0.45	10.47
kroa100	21282	0.00	0.02	0.11	7.57	7.68	**0.00**	**0.00**	0.00	0.00	10.84
krob100	22141	0.00	0.18	0.98	53.70	7.82	**0.00**	**0.09**	0.43	31.28	11.89
kroc100	20749	0.00	0.16	1.13	65.31	7.61	**0.00**	**0.00**	0.00	0.00	10.83
krod100	21294	0.00	0.35	1.40	101.15	7.59	**0.00**	**0.00**	0.00	0.00	10.79
kroe100	22068	0.15	0.42	1.05	55.37	7.68	**0.00**	**0.21**	0.24	16.45	10.81
rd100	7910	0.00	0.39	1.67	33.79	7.65	**0.00**	**0.02**	0.43	7.60	10.89
eil101	629	0.00	0.85	2.23	4.80	8.37	**0.00**	**0.37**	1.11	2.35	10.61
lin105	14379	**0.00**	**0.00**	0.00	0.00	7.73	**0.00**	**0.00**	0.00	0.00	11.76
pr107	44303	0.00	0.21	0.61	85.67	8.07	**0.00**	**0.00**	0.00	0.00	11.33
pr124	59030	0.00	0.06	0.88	116.09	8.61	**0.00**	**0.00**	0.00	0.00	13.95
bier127	118282	0.00	0.48	1.36	546.97	9.11	**0.00**	**0.25**	1.51	476.73	14.94
ch130	6110	0.00	0.77	1.67	31.47	9.65	**0.00**	**0.36**	1.00	22.81	15.43
pr136	96772	0.01	0.60	1.83	488.74	10.00	**0.00**	**0.06**	0.15	68.06	15.59
gr137	69853	0.00	0.31	1.23	300.26	9.49	**0.00**	**0.00**	0.00	0.00	15.73

pr144	58537	0.00	0.05	0.09	27.19	9.56	**0.00**	**0.00**	0.00	0.00	17.54
ch150	6528	0.00	0.66	1.75	31.93	10.88	**0.00**	**0.33**	0.58	11.83	18.57
kroa150	26524	0.00	0.60	1.56	114.29	10.75	**0.00**	**0.02**	0.37	21.58	18.62
krob150	26130	0.00	0.48	2.04	118.53	10.71	**0.00**	**0.04**	0.34	26.59	18.25
pr152	73682	0.00	0.25	1.02	178.45	11.59	**0.00**	**0.15**	0.18	55.02	19.11
u159	42080	0.00	0.60	1.46	164.05	11.28	**0.00**	**0.08**	0.75	97.26	19.37
rat195	2323	0.52	1.33	2.88	15.25	14.49	**0.00**	**0.53**	1.12	7.16	26.50
d198	15780	0.05	0.29	0.61	24.30	17.91	**0.00**	**0.07**	0.39	14.88	26.83
kroa200	29368	0.16	0.72	1.56	133.09	15.32	**0.00**	**0.26**	1.20	107.67	28.98
krob200	29437	0.04	0.61	1.51	132.24	15.23	**0.00**	**0.27**	1.47	126.74	29.08
gr202	40160	0.34	1.20	2.19	196.39	19.69	**0.00**	**0.37**	1.16	141.46	36.40
tsp225	3916	0.89	1.44	2.32	16.28	27.94	**0.00**	**0.86**	2.07	18.96	20.35
ts225	126643	**0.00**	**0.00**	0.00	0.00	16.71	**0.00**	**0.00**	0.00	0.00	37.36
pr226	80369	0.00	0.13	0.62	183.23	28.74	**0.00**	**0.00**	0.00	0.00	33.26
gr229	134602	0.38	1.21	2.02	583.61	22.50	**0.01**	**0.33**	0.78	318.13	35.18
gil262	2378	0.21	1.59	3.20	16.22	36.29	**0.00**	**0.44**	1.09	8.23	46.21
pr264	49135	0.00	0.19	0.63	108.38	21.45	**0.00**	**0.00**	0.00	0.00	42.89
a280	2579	0.00	0.76	2.02	16.19	25.38	**0.00**	**0.26**	0.97	10.68	48.24
pr299	48191	0.10	0.53	1.28	164.67	37.87	**0.00**	**0.21**	0.90	114.20	69.27
lin318	42029	0.17	1.27	2.12	221.16	32.90	**0.00**	**0.61**	1.04	116.39	81.72
linhp318	41345	2.41	3.30	4.48	199.12	43.19	**1.93**	**2.58**	3.42	181.14	81.88
rd400	15281	1.38	2.23	3.13	58.81	77.56	**0.46**	**0.88**	1.39	39.69	105.76
fl417	11861	0.08	0.29	1.02	22.73	82.42	**0.01**	**0.10**	0.14	4.88	96.34
gr431	171414	0.91	1.82	2.89	746.11	91.48	**0.83**	**1.27**	2.04	502.14	115.33
pr439	107217	0.32	1.07	2.50	705.03	94.58	**0.06**	**0.38**	1.30	444.62	112.66
pcb442	50778	1.10	1.95	2.53	243.67	109.94	**0.71**	**1.08**	1.63	120.27	120.88
d493	35002	1.46	2.12	2.94	155.97	132.51	**0.42**	**1.08**	1.76	120.31	145.70
ali535	202310	0.80	1.98	2.70	1060.81	142.37	**0.07**	**1.06**	2.62	1445.53	186.14
u574	36905	1.49	2.90	3.82	211.55	206.30	**0.44**	**1.20**	1.83	135.35	192.84
rat575	6773	2.14	3.28	4.03	33.50	215.64	**1.03**	**1.54**	2.44	26.41	195.72
p654	34643	0.18	0.38	0.85	64.40	218.09	**0.00**	**0.06**	0.13	9.56	227.09
d657	48912	1.49	2.42	3.68	289.54	403.95	**0.39**	**0.86**	1.40	135.05	259.68
gr666	294358	2.06	2.94	3.89	1422.28	510.39	**0.64**	**1.61**	2.60	1631.01	273.10
u724	41910	2.60	3.02	3.82	160.26	538.97	**0.62**	**1.14**	1.73	103.64	316.95
rat783	8806	2.71	3.74	4.63	37.89	611.20	**1.59**	**1.96**	2.43	21.23	378.06
pr1002	259045	3.39	4.10	5.14	1238.28	1026.83	**0.94**	**1.73**	2.53	1079.57	622.01
Average		0.42	0.90	1.63	176.93	78.47	**0.16**	**0.39**	0.77	121.37	67.50
W/D/L			0/12/53								

DOPA, 36 örnek üzerinde IBA (Osaba ve ark., 2016) ile karşılaştırılmış, sonuçlar Çizelge 6.8'de sunulmuştur. IBA'nın sonuçları, Time sütunundakiler hariç, İlhan ve Gökmen (2022) tarafından uygulanan çalışmadan alınmıştır. Time sütunundaki sonuçlar ise adil bir karşılaştırma için DOPA'nın çalıştırıldığı bilgisayar ile elde edilmiştir. Çizelgeden de görülebildiği gibi DOPA'nın PDB'si örneklerin %100'ünde %0'dır. Başka bir ifadeyle DOPA, 36 örneğin tamamında optimum sonucu elde etmiştir. IBA'nın PDB'si ise örneklerin ancak %31'inde %0'dır. Yani IBA, 36 örneğin sadece 11'inde

optimum sonucu elde etmiştir. DOPA, sadece PDB değil, PDA, PDW, SD ve Time açısından da IBA'ya göre üstün performans göstermiştir. En büyük performans farkının, özellikle çalışma süreleri açısından olduğu görülmektedir. DOPA, IBA'ya göre, küçük boyutlu birkaç örnek haricinde, çok daha hızlı çalışmaktadır.

Çizelge 6.8. DOPA ve IBA yöntemlerinin 36 örnek üzerinde performans karşılaştırması

Instance	Optimum	IBA					DOPA				
		PDB (%)	PDA (%)	PDW (%)	SD	Time (s)	PDB (%)	PDA (%)	PDW (%)	SD	Time (s)
burma14	3323	0.00	0.02	0.39	2.91	0.73	**0.00**	**0.00**	0.00	0.00	3.79
ulysses16	6859	0.00	0.03	0.16	3.77	1.13	**0.00**	**0.00**	0.00	0.00	3.86
gr17	2085	0.00	0.15	0.24	2.01	2.22	**0.00**	**0.00**	0.00	0.00	3.89
ulysses22	7013	0.00	0.24	1.06	30.50	3.09	**0.00**	**0.00**	0.00	0.00	4.15
bays29	2020	0.00	0.96	2.38	14.68	8.37	**0.00**	**0.00**	0.00	0.00	4.46
dantzig42	699	0.00	0.36	1.57	3.47	29.07	**0.00**	**0.00**	0.00	0.00	5.29
swiss42	1273	0.00	1.03	3.38	13.50	28.79	**0.00**	**0.00**	0.00	0.00	5.24
eil51	426	0.23	1.91	3.05	3.23	56.68	**0.00**	**0.19**	0.23	0.41	5.97
berlin52	7542	0.00	2.69	6.39	142.33	98.73	**0.00**	**0.00**	0.00	0.00	6.13
st70	675	0.15	1.40	2.52	4.55	151.54	**0.00**	**0.00**	0.00	0.00	7.58
eil76	538	1.86	3.03	4.28	3.76	205.77	**0.00**	**0.32**	1.49	2.90	8.17
pr76	108159	0.15	1.08	1.97	515.38	201.25	**0.00**	**0.00**	0.00	0.00	8.39
gr96	55209	0.81	1.89	3.36	424.22	387.37	**0.00**	**0.10**	0.35	61.73	10.51
rat99	1211	1.82	4.22	7.35	15.83	474.92	**0.00**	**0.01**	0.17	0.45	10.47
kroa100	21282	0.16	0.87	2.30	121.11	490.84	**0.00**	**0.00**	0.00	0.00	10.84
krob100	22141	0.41	1.81	3.20	186.56	480.80	**0.00**	**0.09**	0.43	31.28	10.89
kroc100	20749	0.17	1.73	3.81	225.12	470.35	**0.00**	**0.00**	0.00	0.00	10.83
krod100	21294	0.45	1.53	4.60	188.78	501.24	**0.00**	**0.00**	0.00	0.00	10.79
kroe100	22068	1.06	1.79	2.58	105.57	490.69	**0.00**	**0.21**	0.24	16.45	10.81
rd100	7910	0.04	2.36	4.78	109.32	467.74	**0.00**	**0.02**	0.43	7.60	10.89
eil101	629	2.86	4.00	6.36	5.93	488.77	**0.00**	**0.37**	1.11	2.35	10.61
lin105	14379	0.00	1.25	2.65	116.57	547.70	**0.00**	**0.00**	0.00	0.00	11.76
pr107	44303	0.10	1.03	1.69	168.27	584.30	**0.00**	**0.00**	0.00	0.00	11.33
pr124	59030	0.00	0.49	1.34	245.42	898.64	**0.00**	**0.00**	0.00	0.00	13.95
bier127	118282	0.50	2.04	4.00	1012.46	1021.27	**0.00**	**0.25**	1.51	476.73	14.94
ch130	6110	1.49	2.73	4.44	42.21	1372.94	**0.00**	**0.36**	1.00	22.81	15.43
pr136	96772	1.56	3.14	4.85	906.11	1576.61	**0.00**	**0.06**	0.15	68.06	15.59
gr137	69853	1.45	2.69	4.12	552.08	1308.32	**0.00**	**0.00**	0.00	0.00	15.73
pr144	58537	0.00	0.12	0.39	86.08	1531.39	**0.00**	**0.00**	0.00	0.00	17.54
ch150	6528	1.53	3.21	5.41	57.47	1787.10	**0.00**	**0.33**	0.58	11.83	18.57
kroa150	26524	1.52	2.54	3.88	146.33	1674.13	**0.00**	**0.02**	0.37	21.58	18.62
krob150	26130	1.04	2.3	4.23	216.11	1736.65	**0.00**	**0.04**	0.34	26.59	18.25
pr152	73682	0.27	1.18	2.18	381.15	1802.40	**0.00**	**0.15**	0.18	55.02	19.11
u159	42080	1.11	2.9	5.12	476.98	2078.13	**0.00**	**0.08**	0.75	97.26	19.37
rat195	2323	3.31	4.78	6.97	19.88	3988.08	**0.00**	**0.53**	1.12	7.16	26.50
d198	15780	1.29	1.69	2.28	35.64	4227.08	**0.00**	**0.07**	0.39	14.88	26.83
Average		0.70	1.81	3.31	182.92	865.97	**0.00**	**0.09**	0.30	25.70	11.70
W/D/L			0/0/36								

Çizelge 6.9, DOPA ve DSSA'nın (Zhang ve Han, 2022) 32 örnek üzerindeki sonuçlarını içermektedir. DSSA'nın sonuçları orijinal sayfasından alınmıştır. Çizelgeden de görülebildiği gibi DOPA 32 örneğin 30'unda optimum sonucu elde etmiştir. Dolayısıyla, DOPA'nın başarı yüzdesi %94'tür. DSSA ise 32 örneğin 24'ünde optimum sonucu, birinde ise optimumdan daha iyi bir sonuç elde etmiştir. Dolayısıyla, DSSA'nın başarı yüzdesi %78'dir. PDB değerlerine bakıldığında DOPA, 7 örnek üzerinde DSSA'dan daha iyi sonuç elde etmiştir. DSSA ise sadece bir örnek üzerinde DOPA'ya göre daha iyi sonuç üretmiştir. DOPA'nın diğer yönteme göre üstünlüğü Average satırındaki tüm değerlerden de görülebilmektedir. Bu satırdaki DOPA'ya ait tüm değerler diğer yönteme ait değerlerden daha düşüktür.

Çizelge 6.9. DOPA ve DSSA yöntemlerinin 32 örnek üzerinde performans karşılaştırması

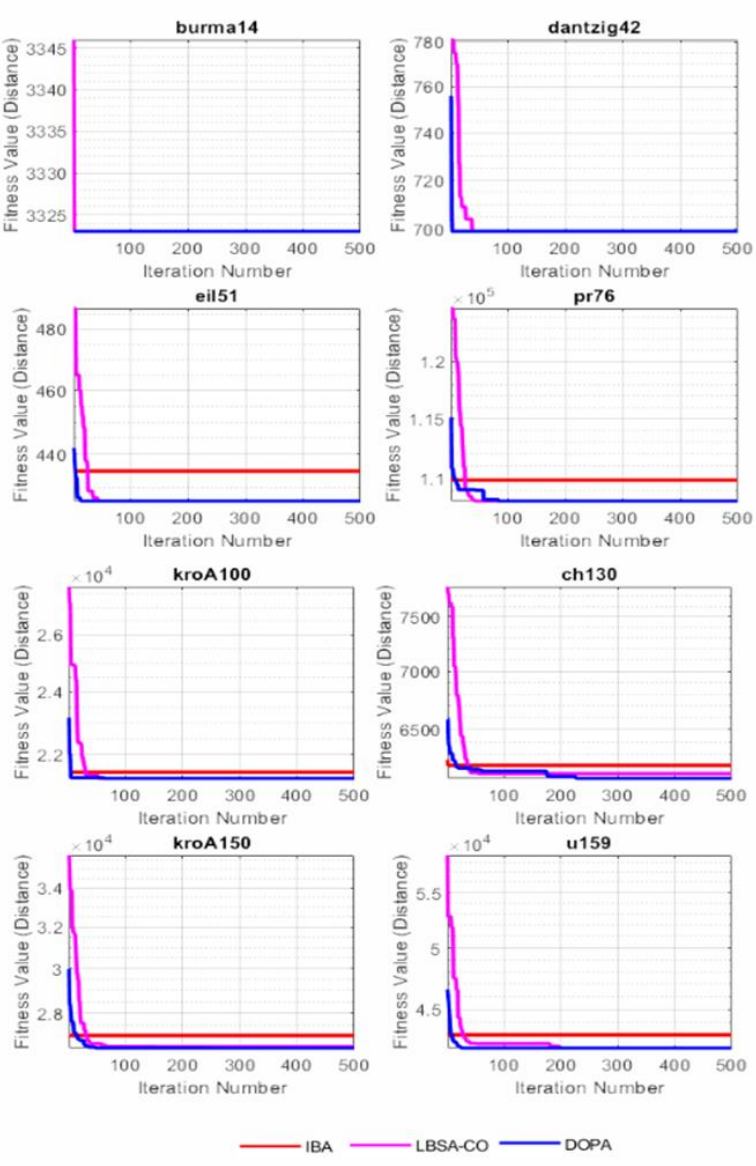
Instance	Optimum	DSSA							DOPA						
		Best	Avg.	Worst	PDB (%)	PDA (%)	PDW (%)	SD	Best	Avg.	Worst	PDB (%)	PDA (%)	PDW (%)	SD
dantzig42	699	**675**	**675.00**	675	-3.43	-3.43	-3.43	0.00	699	699.00	699	0.00	0.00	0.00	0.00
rand50	5553	**5553**	**5553.00**	5553	0.00	0.00	0.00	0.00	**5553**	**5553.00**	5553	0.00	0.00	0.00	0.00
eil51	426	426	427.00	428	0.00	0.14	0.47	0.68	**426**	**426.80**	427	0.00	0.19	0.23	0.41
berlin52	7542	**7542**	**7542.00**	7542	0.00	0.00	0.00	0.00	**7542**	**7542.00**	7542	0.00	0.00	0.00	0.00
st70	675	675	675.15	678	0.00	0.02	0.44	0.67	**675**	**675.00**	675	0.00	0.00	0.00	0.00
eil76	538	538	543.10	548	0.00	0.95	1.86	2.83	**538**	**539.70**	546	0.00	0.32	1.49	2.90
pr76	108159	**108159**	**108159.00**	108159	0.00	0.00	0.00	0.00	**108159**	**108159.00**	108159	0.00	0.00	0.00	0.00
kroa100	21282	21282	21290.20	21353	0.00	0.04	0.33	17.14	**21282**	**21282.00**	21282	0.00	0.00	0.00	0.00
krob100	22141	22141	22173.10	22258	0.00	0.14	0.53	36.30	**22141**	**22160.30**	22237	0.00	0.09	0.43	31.28
kroc100	20749	20749	20770.50	20880	0.00	0.10	0.63	42.54	**20749**	**20749.00**	20749	0.00	0.00	0.00	0.00
krod100	21294	21294	21319.05	21410	0.00	0.12	0.54	44.15	**21294**	**21294.00**	21294	0.00	0.00	0.00	0.00
kroe100	22068	**22068**	**22091.90**	22174	0.00	0.11	0.48	32.85	22068	22114.20	22121	0.00	0.21	0.24	16.45
eil101	629	634	641.50	646	0.79	1.99	2.70	3.62	**629**	**631.35**	636	0.00	0.37	1.11	2.35
lin105	14379	**14379**	**14379.00**	14379	0.00	0.00	0.00	0.00	**14379**	**14379.00**	14379	0.00	0.00	0.00	0.00
pr107	44303	44303	44322.00	44387	0.00	0.04	0.19	34.75	**44303**	**44303.00**	44303	0.00	0.00	0.00	0.00
pr124	59030	**59030**	**59030.00**	59030	0.00	0.00	0.00	0.00	**59030**	**59030.00**	59030	0.00	0.00	0.00	0.00
ch130	6110	6110	6153.65	6205	0.00	0.71	1.55	26.38	**6110**	**6131.75**	6171	0.00	0.36	1.00	22.81
pr136	96772	96920	97302.35	97956	0.15	0.55	1.22	265.94	**96772**	**96830.15**	96920	0.00	0.06	0.15	68.06
pr144	58537	**58537**	**58537.00**	58537	0.00	0.00	0.00	0.00	**58537**	**58537.00**	58537	0.00	0.00	0.00	0.00
ch150	6528	6528	6590.15	6658	0.00	0.95	1.99	34.53	**6528**	**6549.30**	6566	0.00	0.33	0.58	11.83
kroa150	26524	26525	26699.85	27031	0.00	0.66	1.91	144.19	**26524**	**26529.35**	26621	0.00	0.02	0.37	21.58
krob150	26130	26130	26220.40	26325	0.00	0.35	0.75	69.01	**26130**	**26140.90**	26218	0.00	0.04	0.34	26.59
pr152	73682	**73682**	**73731.35**	73841	0.00	0.07	0.22	67.91	73682	73791.20	73818	0.00	0.15	0.18	55.02
u159	42080	42080	42262.75	42438	0.00	0.43	0.85	157.55	**42080**	**42111.60**	42396	0.00	0.08	0.75	97.26
kroa200	29368	29459	29682.15	29882	0.31	1.07	1.75	130.55	**29368**	**29443.70**	29721	0.00	0.26	1.20	107.67
krob200	29437	29564	29850.55	30126	0.43	1.40	2.34	178.27	**29437**	**29515.85**	29870	0.00	0.27	1.47	126.74
tsp225	3916	**3916**	**3926.05**	3952	0.00	0.26	0.92	12.53	3916	3949.55	3995	0.00	0.86	2.07	18.96
pr226	80369	80369	80369.20	80373	0.00	0.00	0.00	0.89	**80369**	**80369.00**	80369	0.00	0.00	0.00	0.00
pr264	49135	49135	49271.85	49564	0.00	0.28	0.87	155.73	**49135**	**49135.00**	49135	0.00	0.00	0.00	0.00
lin318	42029	42495	42742.70	43298	1.11	1.70	3.02	184.42	**42029**	**42283.55**	42465	0.00	0.61	1.04	116.39
pr439	107217	107494	107844.90	108376	0.26	0.59	1.08	263.11	**107277**	**107624.85**	108612	0.06	0.38	1.30	444.62
pr1002	259045	264212	266352.35	268693	1.99	2.82	3.72	1146.43	**261488**	**263527.25**	265602	0.94	1.73	2.53	1079.57
Average		40393.88	40535.26	40729.84	0.05	0.38	0.84	95.41	**40276.53**	**40375.42**	40519.63	0.03	0.20	0.50	70.36
W/D/L			4/6/22												

Çizelge 6.10, 27 örnek üzerinde DOPA, GA- JGHO (Zhang ve ark., 2022), D-GWO (Panwar ve Deep, 2021) ve DBAL (Saji ve Barkatou, 2021)'ın sonuçlarını göstermektedir. GA-JGHO, D-GWO ve DBAL'ın sonuçları Zhang ve ark. (2022) tarafından uygulanan çalışmadan alınmıştır. İlgili sayfada bu yöntemlere ait sonuçların her bir örnek üzerinde 20 kez çalıştırılarak elde edildiği raporlanmaktadır. Çizelge 6.10 yuvarlatılmış Öklid mesafeleri kullanılarak hesaplanan sonuçları içermektedir. Çizelgeden de görülebildiği gibi PDB açısından DOPA, 27 örneğin 22'sinde optimum sonucu elde ederek %81'lik bir başarı yüzdesi elde etmiştir. Aynı açıdan GA-JGHO, D-GWO ve DBAL'ın başarı yüzdeleri ise, sırasıyla, %78, %33 ve %67'dir. PDA açısından ise DOPA, 27 örneğin 8'inde %0.00 sonucunu elde ederek %30'luk bir başarı yüzdesi elde etmiştir. Aynı açıdan GA-JGHO, D-GWO ve DBAL'ın başarı yüzdeleri ise, sırasıyla, %26, %4 ve %22'dir. Average satırındaki değerlere bakıldığında DOPA, D-GWO ve DBAL yöntemlerine göre daha iyi sonuç elde ederken GA-JGHO yöntemine göre daha kötü sonuç elde etmiştir. Bu noktada, bu çalışmanın klasik OPA'nın yapısına bağlı kalarak onun ayrık bir versiyonunu geliştirmek için yapıldığını hatırlatmakta fayda var. DOPA üzerinde geliştirme, iyileştirme ve GA-JGHO'da olduğu gibi diğer algoritmalarla hibritleme yapılarak daha rekabetçi ve üstün yöntemler geliştirilebilir.

Çizelge 6.10. DOPA, GA-JGHO, D-GWO ve DBAL yöntemlerinin 27 örnek üzerinde performans karşılaştırması

Instance	Optimum	GA-JGHO					D-GWO					DBAL					DOPA				
		Best	Avg.	PDB (%)	PDA (%)	SD	Best	Avg.	PDB (%)	PDA (%)	SD	Best	Avg.	PDB (%)	PDA (%)	SD	Best	Avg.	PDB (%)	PDA (%)	SD
eil51	426	**426**	**426.60**	0.00	0.14	0.50	**426**	**426.20**	0.00	0.05	1.42	**426**	**426.05**	0.00	0.01	0.00	426	426.80	0.00	0.19	0.41
berlin52	7542	**7542**	**7542**	0.00	0.00	0.00	**7542**	**7542**	0.00	0.00	0.00	**7542**	**7542**	0.00	0.00	0.00	**7542**	**7542**	0.00	0.00	0.00
st70	675	675	675.25	0.00	0.04	1.09	675	675.40	0.00	0.06	0.92	675	675.05	0.00	0.01	0.00	**675**	**675**	0.00	0.00	0.00
eil76	538	**538**	**538.25**	0.00	0.05	0.89	540	543.10	0.37	0.95	4.97	**538**	**538**	0.00	0.00	0.00	538	539.70	0.00	0.32	2.90
pr76	108159	**108159**	**108159**	0.00	0.00	0.00	108159	108177.80	0.00	0.02	36.79	**108159**	**108159**	0.00	0.00	0.00	**108159**	**108159**	0.00	0.00	0.00
rat99	1211	**1211**	**1211.05**	0.00	0.00	0.22	1213	1223.10	0.17	1.00	7.33	1211	1211.15	0.00	0.01	0.48	1211	1211.10	0.00	0.01	0.45
kroa100	21282	**21282**	**21282**	0.00	0.00	0.00	21282	21316	0.00	0.16	34.13	**21282**	**21282**	0.00	0.00	0.00	**21282**	**21282**	0.00	0.00	0.00
krob100	22141	22141	22185.20	0.00	0.20	46.94	22141	22189	0.00	0.21	42.94	22141	22167.05	0.00	0.12	3.49	**22141**	**22160.30**	0.00	0.09	31.28
eil101	629	**629**	**630.45**	0.00	0.23	2.69	631	637.00	0.32	1.21	3.26	629	629	0.00	0.00	0.00	629	631.35	0.00	0.37	2.35
lin105	14379	**14379**	**14379**	0.00	0.00	0.00	14379	14380.10	0.00	0.01	69.41	**14379**	**14379**	0.00	0.00	0.00	**14379**	**14379**	0.00	0.00	0.00
pr107	44303	44303	44340.05	0.00	0.08	63.05	44303	44406.90	0.00	0.23	75.40	44303	44341.70	0.00	0.09	50.44	**44303**	**44303**	0.00	0.00	0.00
pr124	59030	**59030**	**59030**	0.00	0.00	0.00	59030	59081.60	0.00	0.09	50.02	59030	59047.40	0.00	0.03	45.81	**59030**	**59030**	0.00	0.00	0.00
bier127	118282	**118282**	**118437.30**	0.00	0.13	181.20	118780	118997.30	0.42	0.60	230.14	**118282**	**118449**	0.00	0.14	193.23	118282	118579.90	0.00	0.25	476.73
ch130	6110	**6110**	**6124.95**	0.00	0.24	16.18	6133	6170.20	0.38	0.99	30.05	**6110**	**6123.15**	0.00	0.22	11.89	6110	6131.75	0.00	0.36	22.81
ch150	6528	**6528**	**6543.80**	0.00	0.24	13.92	6548	6595.70	0.31	1.04	27.76	**6528**	**6531.70**	0.00	0.06	7.37	6528	6549.30	0.00	0.33	11.83
kroa150	26524	**26524**	**26527.05**	0.00	0.01	7.69	26644	26828.30	0.45	1.15	121.77	26524	26570.65	0.00	0.18	45.86	26524	26529.35	0.00	0.02	21.58
krob150	26130	**26130**	**26139.60**	0.00	0.04	18.82	26251	26305.30	0.46	0.67	39.17	26130	26152.30	0.00	0.09	26.84	26130	26140.90	0.00	0.04	26.59
kroa200	29368	**29368**	**29413.30**	0.00	0.15	69.29	29681	29810.50	1.07	1.51	80.91	29397	29447.30	0.10	0.27	50.37	29368	29443.70	0.00	0.26	107.67
tsp225	3916	**3916**	**3925.05**	0.00	0.23	14.54	3997	4006.20	2.07	2.30	9.06	3938	3962.55	0.56	1.19	14.21	3916	3949.55	0.00	0.86	18.96
pr226	80369	**80369**	**80369**	0.00	0.00	0.00	80463	80829	0.12	0.57	206.04	80369	80397.35	0.00	0.04	25.86	**80369**	**80369**	0.00	0.00	0.00
pr299	48191	48197	48282.35	0.01	0.19	87.16	48946	49197.10	1.57	2.09	198.12	48648	48830.85	0.95	1.33	25.86	**48191**	**48292.60**	0.00	0.21	114.20
lin318	42029	**42029**	**42227.95**	0.00	0.47	146.94	42764	43040.40	1.75	2.41	119.97	42230	42469.70	0.48	1.05	111.82	42029	42283.55	0.00	0.61	116.39
pr439	107217	**107250**	**107531.40**	0.03	0.29	352.58	108633	109344.70	1.32	1.98	534.77	107786	108153.85	0.53	0.87	197.52	107277	107624.85	0.06	0.38	444.62
pcb442	50778	**50927**	**51158.40**	0.29	0.75	155.91	52173	52488.80	2.75	3.37	167.55	51712	51905.60	1.84	2.22	150.10	51138	51324.35	0.71	1.08	120.27
d493	35002	**35109**	**35371.50**	0.31	1.06	131.00	35920	36081.10	2.62	3.08	86.03	35769	35901.75	2.19	2.57	98.10	35148	35381.65	0.42	1.08	120.31
d657	48912	49156	49321.60	0.50	0.84	122.95	50366	50770	2.97	3.80	221.13	50314	50545.15	2.87	3.34	126.78	**49101**	**49333.50**	0.39	0.86	135.05
pr1002	259045	**261191**	**262966.80**	0.83	1.51	748.40	269675	270748.50	4.10	4.52	691.61	269836	271668.30	4.17	4.87	956.46	261488	263527.25	0.94	1.73	1079.57
Average		**43385.22**	**43508.85**	0.07	0.26	80.81	43973.89	44141.12	0.86	1.26	114.47	43847.70	43981.73	0.51	0.69	79.35	43404.22	43548.39	0.09	0.34	105.74
W/D/L			16/6/5					1/1/25					5/4/18								

DOPA, IBA ve LBSA-CO, 14'den 159'a kadar şehir sayılarına sahip farklı boyutlarda sekiz örnek üzerinde çalıştırılmıştır. Bu örneklere ait yakınsama eğrileri Şekil 6.1'de verilmiştir. Şekilden de görülebildiği gibi IBA, DOPA'ya kıyasla daha hızlı yakınsamaktadır. Ancak burma14 ve dantzig42 örnekleri haricindeki diğer örneklerde lokal minimumlara takılmakta ve optimum sonuca ulaşamamaktadır. LBSA-CO ise DOPA'ya göre daha yavaş yakınsamakta hatta ch130 ve kroa150 örneklerinde optimum sonuca ulaşamamaktadır. DOPA, IBA'ya ait premature yakınsama ve LBSA-CO'ya ait geç yakınsama dezavantajlarına sahip değildir. DOPA erken yakınsamakta ve optimum sonuca ulaşabilmektedir. Bu DOPA'nın çeşitlendirme ve yoğunlaştırma arasındaki dengeyi sağlama potansiyelinin bir göstergesidir.



Şekil 6.1. DOPA ve diğer yöntemlerin yakınsama eğrilerinin sekiz örnek üzerinde karşılaştırılması

6.4. İstatistiksel Analizler

DOPA'nın performansı bir takım istatistiksel testler kullanılarak diğer yöntemlerle karşılaştırılmıştır. Bu testler için Minitab 19 istatistiksel test yazılımı kullanılmıştır. Çizelge 6.11, DOPA, ABCSS (Khan ve Maiti, 2019), DSMO (Akhand ve ark., 2020), LBSA-CO (İlhan ve Gökmen, 2022) ve VTPSO (Akhand ve ark., 2015) için uygulanan Kolmogorov-Smirnov ve Shapiro-Wilk normallik testlerinin sonuçlarını göstermektedir. Bu sonuçlara göre veriler normal dağılım göstermemektedir. Bu durum Çizelge 6.11'deki p-value'lardan (Sig) anlaşılmaktadır. Bu nedenle, metotlar arasında istatistiksel olarak anlamlı bir fark olup olmadığını belirlemek için Friedman testi, ki bu test parametrik olmayan verilere uygulanır, kullanılmıştır. Bu testin sonuçlarını gösteren Çizelge 6.12'deki p-value (Asymp. Sig.) yöntemler arasında önemli farklılıklar olduğunu gösterir. Ancak, Friedman testi ikili karşılaştırma yapamaz ve hangi metotların birbirlerinden farklı olduğunu gösteremez. Bunun için parametrik olmayan bir test olan Wilcoxon işaretli sıra testi uygulanmış ve sonuçlar Çizelge 6.13'de sunulmuştur. Bu çizelgedeki p-value'lardan da görülebildiği gibi DOPA ve diğer metotlar arasında istatistiksel anlamlı farklılıklar bulunmaktadır. Çizelge 6.11 - 6.13'deki sonuçlar Çizelge 6.6'daki "Best" sütunundaki değerler baz alınarak hesaplanmıştır.

Çizelge 6.11. DOPA, ABCSS, DSMO, LBSA-CO ve VTPSO yöntemlerinin normallik testlerinin sonuçları

Method	Kolmogorov-Smirnov			Shapiro-Wilk		
	Statistic	df	Sig	Statistic	df	Sig
ABCSS	0.197	32	<0.010	0.882	32	<0.010
DSMO	0.166	32	<0.032	0.930	32	<0.010
LBSA-CO	0.167	32	<0.031	0.928	32	<0.010
VTPSO	0.166	32	<0.033	0.930	32	<0.010
DOPA	0.167	32	<0.031	0.928	32	<0.010

Çizelge 6.12. Friedman testinin sonuçları

N	32
Chi-Square	102.60
df	4
Asymp.Sig.	<0.000

Çizelge 6.13. ABCSS, DSMO, LBSA-CO ve VTPSO ile ilgili olarak DOPA'nın Wilcoxon işaretli sıra testinin sonuçları

Comparison		N	Mean Rank	Sum Of Rank	Z	P-value
DOPA --- ABCSS	Negative Ranks	31	17	527	-4.937	<0.001
	Positive Ranks	1	1	1		
	Ties					
DOPA --- DSMO	Negative Ranks	30	15.5	465	-4.78	<0.001
	Positive Ranks	0	0	0		
	Ties					
DOPA --- LBSA-CO	Negative Ranks	12	6.5	78	-3.06	<0.016
	Positive Ranks	0	0	0		
	Ties					
DOPA --- VTPSO	Negative Ranks	32	16.5	528	-4.94	<0.001
	Positive Ranks	0	0	0		
	Ties					

Çizelge 6.14, DOPA ve LBSA-CO (İlhan ve Gökmen, 2022) için uygulanan Kolmogorov-Smirnov ve Shapiro-Wilk normallik testlerinin sonuçlarını göstermektedir. Bu sonuçlara göre veriler normal dağılım göstermemektedir. Bu durum Çizelge 6.14,'deki p-value'lardan (Sig) anlaşılmaktadır. Sadece iki yöntemin olduğu durumlarda Friedman testi uygulanmaz. Doğrudan ikili karşılaştırma yapılır. Çizelge 6.15, DOPA ve LBSA-CO'ya ait Wilcoxon işaretli sıra testinin sonuçlarını göstermektedir. Bu çizelgedeki p-value'dan da görülebildiği gibi bu iki yöntem arasında istatistiksel olarak anlamlı fark bulunmaktadır. Çizelge 6.14 ve Çizelge 6.15'deki sonuçlar Çizelge 6.7'deki "PDB" sütunundaki değerler baz alınarak hesaplanmıştır.

Çizelge 6.14. DOPA ve LBSA-CO yöntemlerinin normallik testlerinin sonuçları

Method	Kolmogorov-Smirnov			Shapiro-Wilk		
	Statistic	df	Sig	Statistic	df	Sig
LBSA-CO	0.328	65	<0.010	0.894	65	<0.010
DOPA	0.433	65	<0.010	0.898	65	<0.010

Çizelge 6.15. LBSA-CO'ya göre DOPA için Wilcoxon işaretli sıra testinin sonuçları

Comparison		N	Mean Rank	Sum Of Rank	Z	P-value
DOPA --- LBSA-CO	Negative Ranks	51	27.058824	1380	-6.393	<0.000
	Positive Ranks	3	35	105		
	Ties					

Çizelge 6.16, DOPA ve IBA (Osaba ve ark., 2016) için uygulanan Kolmogorov-Smirnov ve Shapiro-Wilk normallik testlerinin sonuçlarını göstermektedir. DOPA'ya ait satırdaki * işaretleri tüm örnekler için optimum sonuca ulaşıldığını göstermektedir. Çizelge 6.16'daki p-value'lardan (Sig) verilerin normal dağılım göstermediği anlaşılmaktadır. Sadece iki yöntem olduğu için burada da Friedman testi uygulanmaz. Doğrudan Wilcoxon işaretli sıra testi uygulanır. Çizelge 6.17, bu testin sonuçlarını göstermektedir. Bu çizelgedeki p-value'dan bu iki yöntem arasında istatistiksel olarak anlamlı bir fark bulunduğu anlaşılmaktadır. Çizelge 6.16 ve Çizelge 6.17'deki sonuçlar Çizelge 6.8'deki "PDB" sütunundaki değerler baz alınarak hesaplanmıştır.

Çizelge 6.16. DOPA ve IBA yöntemlerinin normallik testlerinin sonuçları DOPA ve LBSA-CO yöntemlerinin normallik testlerinin sonuçları

Method	Kolmogorov-Smirnov			Shapiro-Wilk		
	Statistic	df	Sig	Statistic	df	Sig
IBA	0.227	37	<0.010	0.935	37	<0.010
DOPA	*	37		*	37	

Çizelge 6.17. IBA'ya göre DOPA için Wilcoxon işaretli sıra testinin sonuçları

Comparison		N	Mean Rank	Sum Of Rank	Z	P-value
DOPA --- IBA	Negative Ranks	37	19	703	-5.303	<0.001
	Positive Ranks	0	0	0		
	Ties					

Çizelge 6.18, DOPA ve DSSA (Zhang ve Han, 2022) için uygulanan Kolmogorov-Smirnov ve Shapiro-Wilk normallik testlerinin sonuçlarını göstermektedir. Bu çizelgedeki p-value'lardan (Sig) verilerin normal dağılım göstermediği anlaşılmaktadır. Sadece iki yöntem olduğu için Friedman testi uygulanmadan doğrudan Wilcoxon işaretli sıra testi uygulanır. Çizelge 6.19, bu testin sonuçlarını göstermektedir. Bu çizelgedeki p-value bu iki yöntem arasında istatistiksel olarak anlamlı bir fark olduğunu göstermektedir. Çizelge 6.18 ve Çizelge 6.19'daki sonuçlar Çizelge 6.9'daki "Best" sütunundaki değerler baz alınarak hesaplanmıştır.

Çizelge 6.18. DOPA ve DSSA yöntemlerinin normallik testlerinin sonuçları

Method	Kolmogorov-Smirnov			Shapiro-Wilk		
	Statistic	df	Sig	Statistic	df	Sig
DSSA	0.220	32	<0.010	0.827	32	<0.010
DOPA	0.219	32	<0.010	0.829	32	<0.010

Çizelge 6.19. DSSA'ya göre DOPA için Wilcoxon işaretli sıra testinin sonuçları

Comparison		N	Mean Rank	Sum Of Rank	Z	P-value
DOPA --- DSSA	Negative Ranks	8	5.25	42	-2.666	<0.029
	Positive Ranks	1	3	3		
	Ties					

Çizelge 6.20, DOPA, GA-JGHO (Zhang ve ark., 2022), D-GWO (Panwar ve Deep, 2021) ve DBAL (Saji ve Barkatou, 2021) için uygulanan Kolmogorov-Smirnov ve Shapiro-Wilk normallik testlerinin sonuçlarını göstermektedir. Bu sonuçlara göre veriler normal dağılım göstermemektedir. Bu durum Çizelge 6.20'deki p-value'lardan (Sig) anlaşılmaktadır. Bu nedenle, metotlar arasında istatistiksel olarak anlamlı bir fark olup olmadığını belirlemek için Friedman testi kullanılmıştır. Bu testin sonuçlarını gösteren Çizelge 6.21'deki p-value (Asymp. Sig.) yöntemler arasında önemli farklılıklar olduğunu göstermiştir. Yöntemleri birbirleriyle kıyaslamak için Wilcoxon işaretli sıra testi uygulanmış ve sonuçlar Çizelge 6.22'de sunulmuştur. Bu çizelgedeki p-value'lardan da görülebildiği gibi DOPA ile hem D-GWO hem de DBAL arasında istatistiksel anlamlı farklılıklar bulunmaktadır. Ancak DOPA ile GA-JGHO arasında diğer yöntemler kadar anlamlı farklılık bulunmamaktadır. Çünkü, bu iki yöntemin karşılaştırılması sonucunda elde edilen 0.059 p-value değeri 0.05 değerinden daha büyüktür.

Çizelge 6.20. DOPA, GA-JGHO, D-GWO ve DBAL yöntemlerinin normallik testlerinin sonuçları

Method	Kolmogorov-Smirnov			Shapiro-Wilk		
	Statistic	df	Sig	Statistic	df	Sig
D-GWO	0.222	27	<0.010	0.835	27	<0.010
DBAL	0.223	27	<0.010	0.834	27	<0.010
GA-JHGO	0.224	27	<0.010	0.840	27	<0.010
DOPA	0.223	27	<0.010	0.840	27	<0.010

Çizelge 6.21. Friedman testinin sonuçları

N	27
Chi-Square	44.40
df	3
Asymp.Sig.	<0.000

Çizelge 6.22. GA-JGHO, D-GWO ve DBAL ile ilgili olarak DOPA'nın Wilcoxon işaretli sıra testinin sonuçları

Comparison		N	Mean Rank	Sum Of Rank	Z	P-value
DOPA --- D-GWO	Negative Ranks	18	9.5	171	-3.724	<0.006
	Positive Ranks	0	0	0		
	Ties					
DOPA --- DBAL	Negative Ranks	9	5	45	-2,67	<0.029
	Positive Ranks	0	0	0		
	Ties					
DOPA --- GA-JGHO	Negative Ranks	2	2.5	5	-2,2	<0.059
	Positive Ranks	4	4	16		
	Ties					

7. SONUÇLAR VE ÖNERİLER

Bu çalışmada, OPA'nın yapısına bağlı kalınarak onun ayrık bir versiyonu geliştirilmiş ve DOPA olarak adlandırılmıştır. DOPA, TSP'yi çözmek için kullanılmıştır. OPA gibi DOPA da kovalama ve saldırı olmak üzere iki fazdan oluşmaktadır. Kovalama fazında orkalar arasındaki mesafeler Hamming mesafesi ile hesaplanmış, hesaplanan bu değerler kullanılarak hız değerleri elde edilmiştir. Hız değerleri ve 2-opt algoritması kullanılarak orkaların konumları güncellenmiştir. Saldırı fazında orkaların konumları OX1 operatörü ile hesaplanmıştır. Pozisyon ayarlama prosedüründe ise takas (swap) lokal arama operatörü kullanılmıştır. DOPA'nın parametreleri Taguchi istatistiksel yöntemi ile ayarlanmıştır. DOPA 67 iyi bilinen TSP örneği üzerinde test edilmiş ve 50 örnek üzerinde bilinen en iyi sonuç elde edilmiştir. Ayrıca, DOPA ile diğer 9 yöntem arasında önemli farklılıklar olup olmadığı Friedman ve Wilcoxon işaretli sıra testleri ile kontrol edilmiştir. Bu testlerin sonuçları DOPA ile diğer 8 yöntem arasında önemli farklılıklar olduğunu göstermiştir.

OPA algoritması gibi DOPA algoritması da beş adet parametreye sahiptir. Parametre optimizasyonu kısmında da bahsedildiği gibi bu parametrelerden en önemli olanı p2 parametresidir. Bu parametredeki küçük bir değişim DOPA'nın performansını diğer parametrelere göre daha çok etkilemektedir. Gelecekte bu parametrenin DOPA'nın performansı üzerindeki etkisini azaltmaya yönelik birtakım iyileştirmeler yapılabilir. Hatta DOPA'nın parametre sayısını azaltmak için yapısal değişiklikler üzerine çalışılabilir.

DOPA ile diğer algoritmalar hibritlenerek daha alternatif ve rekabetçi yöntemler geliştirilebilir. Özellikle, p2 parametresinin de kullanıldığı saldırı fazını güçlendirmek için hibrit algoritmalar önerilebilir. DOPA, araç rotalama problemi (VRP) gibi diğer yaygın kombinatoryal optimizasyon problemlerine uyarlanabilir.

KAYNAKLAR

Akhand, M. A. H., Akter, S., Rashid, M. A., and Yaakob, S. B., 2015, Velocity Tentative PSO: An Optimal Velocity Implementation based Particle Swarm Optimization to Solve Traveling Salesman Problem, *IAENG International Journal of Computer Science*, 42(3).

Akhand, M. A. H., Ayon, S. I., Shahriyar, S. A., Siddique, N., and Adeli, H., 2020, Discrete spider monkey optimization for travelling salesman problem, Applied Soft Computing, 86, 105887.

Bansal, J. C., Sharma, H., Jadon, S. S., and Clerc, M., 2014, Spider monkey optimization algorithm for numerical optimization, *Memetic computing*, 6, 31-47.

Berbeglia, G., and Hahn, G., 2009, Counting feasible solutions of the traveling salesman problem with pickups and deliveries is# P-complete, *Discrete Applied Mathematics*, 157(11), 2541-2547.

Bora, T. C., Coelho, L. D. S., and Lebensztajn, L., 2012, Bat-inspired optimization approach for the brushless DC wheel motor problem, *IEEE Transactions on magnetics*, 48(2), 947-950.

Cook, W., and Seymour, P., 2003, Tour merging via branch-decomposition, *INFORMS Journal on Computing*, 15(3), 233-248.

Dantzig, G., Fulkerson, R., and Johnson, S., 1954, Solution of a large-scale traveling-salesman problem, *Journal of the operations research society of America*, 2(4), 393-410.

Davis, L., 1985, Applying adaptive algorithms to epistatic domains, *In IJCAI*, 162-164.

Dong, C., Jäger, G., Richter, D., and Molitor, P., 2009, Effective tour searching for TSP by contraction of pseudo backbone edges, *In Algorithmic Aspects in Information and Management: 5th International Conference*, AAIM 2009, San Francisco, CA, USA, June 15-17, 2009. Proceedings 5 (pp. 175-187). Springer Berlin Heidelberg.

Fığlalı, A., 2008, Optimizasyona Giriş, *Optimizasyon Seminerleri Dizisi*, (2-27).

Ford, J. K. B., 2018, Killer Whale, *Encyclopedia of Marine Mammals*, 531–537

Ghani, J. A., Choudhury, I. A., and Hassan, H. H., 2004, Application of Taguchi method in the optimization of end milling parameters, *Journal of materials processing technology*, 145(1), 84-92.

Gündüz, M., Kiran, M. S., and Özceylan, E., 2015, A hierarchic approach based on swarm intelligence to solve the traveling salesman problem, *Turkish Journal of Electrical Engineering and Computer Sciences*, 23(1), 103-117.

Held, M., and Karp, R. M., 1962, A dynamic programming approach to sequencing problems, *Journal of the Society for Industrial and Applied mathematics*, 10(1), 196-210.

Helsgaun, K., 2006, An effective implementation of K-opt moves for the Lin-Kernighan TSP heuristic (Doctoral dissertation, Roskilde University), *Department of Computer Science*.

Helsgaun, K., 2009, General k-opt submoves for the Lin–Kernighan TSP heuristic, *Mathematical Programming Computation*, 1, 119-163.

İlhan, İ., and Gökmen, G., 2022, A list-based simulated annealing algorithm with crossover operator for the traveling salesman problem, *Neural Computing and Applications*, 1-26.

Jiang, Y., Wu, Q., Zhu, S., and Zhang, L., 2022, Orca predation algorithm: A novel bio-inspired algorithm for global optimization problems, *Expert Systems with Applications*, 188, 116026.

Kasat, R. B., and Gupta, S. K. (2003). Multi-objective optimization of an industrial fluidized-bed catalytic cracking unit (FCCU) using genetic algorithm (GA) with the jumping genes operator, *Computers & Chemical Engineering*, 27(12), 1785-1800.

Khan, I., and Maiti, M. K., 2019, A swap sequence based artificial bee colony algorithm for traveling salesman problem, *Swarm and evolutionary computation*, 44, 428-438.

Kiran, M. S., Hakli, H., Gunduz, M., and Uguz, H., 2015, Artificial bee colony algorithm with variable search strategy for continuous optimization, *Information Sciences*, 300, 140-157.

Lawler, E. L., and Wood, D. E., 1966, Branch-and-bound methods: A survey, *Operations research*, 14(4), 699-719.

Miliotis, P., 1978, Using cutting planes to solve the symmetric travelling salesman problem, *Mathematical programming*, 15, 177-188.

Mozdgir, A., Mahdavi, I., Badeleh, I. S., and Solimanpur, M., 2013, Using the Taguchi method to optimize the differential evolution algorithm parameters for minimizing the workload smoothness index in simple assembly line balancing, *Mathematical and Computer Modelling*, 57(1-2), 137-151.

Osaba, E., Yang, X. S., Diaz, F., Lopez-Garcia, P., and Carballedo, R., 2016, An improved discrete bat algorithm for symmetric and asymmetric traveling salesman problems, *Engineering Applications of Artificial Intelligence*, 48, 59-71.

Padberg, M., and Rinaldi, G., 1991, A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems, *SIAM review*, 33(1), 60-100.

Pan, Q. K., Tasgetiren, M. F., Suganthan, P. N., and Chua, T. J., 2011, A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem, *Information sciences*, 181(12), 2455-2468.

Panwar, K., and Deep, K., 2021, Discrete Grey Wolf Optimizer for symmetric travelling salesman problem, *Applied Soft Computing*, 105, 107298.

Pfluger, P., 1968, Diskussion der Modellwahl am Beispiel des Traveling-Salesman Problems, *Einführung in die Methode Branch and Bound*, 88-106.

Pulat, A. G. M., and Kocakoç, İ. D., 2017, Gezgin Satıcı Probleminin Genetik Algoritmalarla Çözümünde Başlangıç Popülasyonun Belirlenmesi1. Joeep:, *Journal of Emerging Economics and Policy*.

Ramteke, M., Ghune, N., and Trivedi, V., 2015, Simulated binary jumping gene: a step towards enhancing the performance of real-coded genetic algorithm, *Information Sciences*, 325, 429-454.

Richard Hamming – Vikipedi, 2023, https://tr.wikipedia.org/wiki/Richard_Hamming [Ziyaret Tarihi: 17 Şubat 2023].

Saji, Y., and Barkatou, M., 2021, A discrete bat algorithm based on Lévy flights for Euclidean traveling salesman problem, *Expert Systems with Applications*, 172, 114639.

Symmetric TSPs., 2007, http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/STSP.html [Ziyaret Tarihi: 20 Şubat 2023].

Taguchi, G., 1986, Introduction to quality engineering: designing quality into products and processes.

Yan, X., Zhang, C., Luo, W., Li, W., Chen, W., and Liu, H., 2012, Solve traveling salesman problem using particle swarm optimization algorithm, *International Journal of Computer Science Issues (IJCSI)*, 9(6), 264.

Yang, X. S., 2010, A new metaheuristic bat-inspired algorithm, *Nature inspired cooperative strategies for optimization (NICSO 2010)*, 65-74.

Zhang, C. and Ding, S., 2021, A stochastic configuration network based on chaotic sparrow search algorithm, *Knowledge-Based Systems*, 220, 106924.

Zhang, P., Wang, J., Tian, Z., Sun, S., Li, J., and Yang, J., 2022, A genetic algorithm with jumping gene and heuristic operators for traveling salesman problem, *Applied Soft Computing*, 127, 109339.

Zhang, Z., and Han, Y., 2022, Discrete sparrow search algorithm for symmetric traveling salesman problem, *Applied Soft Computing*, 118, 108469.