Uluslararası Yönetim İktisat ve İşletme Dergisi, Cilt 17, Sayı 4, 2021
International Journal of Management Economics and Business, Vol. 17, No. 4, 2021

Araştırma Makalesi / *Research Article*

GEZGİN SATICI PROBLEMİ İÇİN ARDIŞIK YEREL ARAMA İLE YENİ BİR HİBRİT GENETİK ALGORİTMA ÖNERİSİ

Dr. Osman PALA
Karamanoğlu Mehmetbey Üniversitesi, İİBF, Karaman, *(osmanpala@kmu.edu.tr)*

ÖZET

Birçok rotalama probleminin temelini oluşturan Gezgin Satıcı Problemi, optimizasyon alanında klasikleşmiş bir matematiksel modele sahiptir. Problem klasik yöntemlerle en iyi çözümün bulunmasının zorluğu sebebiyle sıklıkla sezgisel algoritmalar yardımıyla çözülmektedir. Sezgisel algoritmalardan en yaygın kullanıma sahip Genetik Algoritma ile problemde etkili çözümler üretilebilmektedir. Çalışma kapsamında, farklı tipte yerel arama yaklaşımlarıyla hibritleştirilmiş Genetik Algoritma ile problem ele alınmıştır. Çalışmada önerilen, ardışık yerel arama yaklaşımına sahip Genetik Algoritma'da, yerel aramada komşuluk üreten fonksiyonlar olan, yer değiştirme, tersine döndürme ve araya yerleştirmenin birlikte kullanımından sonra 3-opt yerel arama yaklaşımı kullanılmıştır. Algoritmalardan elde edilen çözüm değerlerine bakıldığında önerilen ardışık yerel aramaya sahip yaklaşımın diğerlerine göre daha yüksek başarıma sahip olduğu görülmüştür.

Anahtar Kelimeler: *Gezgin Satıcı Problemi, Genetik Algoritma, Yerel Arama.*

A NEW HYBRID GENETIC ALGORITHM PROPOSAL WITH SEQUENTIAL LOCAL SEARCH FOR TRAVELING SALESMAN PROBLEM

ABSTRACT

The Traveling Salesman Problem, which forms the basis of many routing problems, has a classical mathematical model in the field of optimization. The problem is often solved with the help of heuristic algorithms due to the difficulty of finding the best solution with classical methods. Genetic Algorithm, which is the most widely used heuristic algorithms, can produce effective solutions to the problem. Within the scope of the study, the problem is addressed with Genetic Algorithm hybridized with different types of local search approaches. In the Genetic Algorithm with the sequential local search approach proposed in the study, 3-opt local search approach was used after applying combination of swapping, inversion and insertion, which are neighborhood generating functions in local search. Considering the solution values obtained from the algorithms, it is seen that the proposed approach with sequential local search has a higher success than the others.

Keywords: *Traveling Salesman Problem, Genetic Algorithm, Local Search.*

www.ijmeb.org ISSN:2147-9208 E-ISSN:2147-9194
http://dx.doi.org/10.17130/ijmeb.855044
Başvuru Tarihi: 06.01.2021, Yayına Kabul Tarihi: 29.05.2021

1. Giriş

Gezgin Satıcı Problemi (GSP) en temel rotalama problemi olarak bilinmektedir. Problemin matematiksel modeli üzerinde yapılabilen değişikliklerle, GSP değişik tipte çok sayıda rotalama problemine dönüşebilmektedir. GSP, genel olarak belirli sayıda düğüm noktaları arasında herhangi bir başlangıç düğümünden yola çıkılıp diğer tüm düğümlere sadece bir kez uğranılması ve son olarak başlangıç düğümüne dönülmesi ile bir turun oluştuğu optimizasyon problemi olarak ifade edilebilmektedir. Buradaki temel hedef oluşacak rotanın mümkün olan en kısa şekilde gerçekleşmesidir.

GSP, tanım açısından basit ve anlaşılır gözükse de çözüm yaklaşımı bakımından problemdeki düğüm sayısı çoğaldıkça klasik matematiksel yöntemlerin etkinliği azalmakta ve bu nedenle GSP çözümünde sıklıkla sezgisel algoritmalar tercih edilmektedir (Karagül, 2019:455).

GSP için kullanılan sezgisel algoritmaları inceleyen bazı önemli çalışmalar, Basu & Ghosh, (2008), Ruiz-Vanoye vd. (2012), Raman & Gill (2017) tarafından gerçekleştirilmiştir. Literatürde farklı sezgiseller ile GSP çözümlerine bakıldığında canlı sürüleri ile çözüm sunan sezgisellerden bazıları; doğadaki karınca kolonilerinin besin arayışına dayanan Karınca Kolonisi Optimizasyon Algoritması (Dorigo, 1992), kuş ve balık gibi sürü halinde besin arayan canlıları taklit eden ve Kennedy & Eberhart (1995) tarafından geliştirilen Parçacık Sürüsü Optimizasyon Algoritması (Wang vd., 2003), arıların kolektif biçimde yiyecek aramalarını kullanan Arı Kolonisi Algoritması (Wong vd., 2008), kombinatoryal çözüm uzayına göre uyarlanmış ve Karaboğa (2005) tarafından geliştirilen Yapay Arı Kolonisi Algoritması (Karaboğa & Görkemli, 2011), ateş böcekleri arasındaki mesafenin kesikli değer alacak şekilde uyarlanmış olan ve Yang (2010) tarafından geliştirilen Ateş Böceği Algoritması (Jati, 2011), kanguruların zıplayışlarından esinlenen ve Pollard (1978) tarafından geliştirilen Kanguru Algoritması (Demirtaş & Keskintürk, 2011), arama ve iz sürme biçimleri bulunan ve Chu vd. (2006) tarafından geliştirilen Kedi Sürüsü Algoritması (Bouzidi & Riffi, 2013), GSP için kullanıma çözüm uzayını tam sayılı olarak tanımlayarak uygun hale getirilen ve Yang & Deb (2009) tarafından geliştirilen Guguk Kuşu Algoritması (Quarrab vd., 2014), kombinatoryal hale getirilmiş Penguen Arama Optimizasyon Algoritması (Mzili & Riffi, 2015), simetrik ve asimetrik GSP çözümü için ayrık hale getirilen ve Yang (2010) tarafından geliştirilen Yarasa Algoritması (Osaba vd., 2016), kelebeklerin bulundukları pozisyonu belirleyen kelebek düzeltme oranının parametrik çözümler sonucu belirlendiği ve Wang vd. (2015) tarafından geliştirilen Kral Kelebek Algoritması (Wang vd., 2016), kuşların göç sırasındaki uçuş biçimine odaklanan ve komşuluk operatörü barındıran ve Duman vd. (2012) tarafından geliştirilen Göçmen Kuşlar Algoritması (Tongur & Ülker, 2016), ilk versiyonu Yang vd. (2007) tarafından geliştirilen modifiye edilmiş iki parçalı Kurt Sürüsü Algoritması (Chen vd., 2017) ve aç gözlü yaklaşım ile başarılı iterasyonlarda aramaya devam eden ve Mirjalili & Lewis (2016) tarafından geliştirilen Balina Optimizasyon Algoritması (Gupta vd., 2018), ilk versiyonu Liu vd. (2018) tarafından geliştirilen 2-opt yerel arama ile güçlendirilmiş paralel Aslan Sürüsü Algoritması (Daoqing & Mingyan, 2020), Lin-Kernighan yerel arama destekli yeni bir genelleştirilmiş ayrımlı çaprazlama dayanan bir yaklaşım (Tinós vd., 2020), k-ortalamalar küme yaklaşımı ve çok yönlü mutasyon operatörleri ile iyileştirilmiş ve ilk versiyonu Storn & Price (1997) tarafından geliştirilen Diferansiyel Evrim Algoritması (Ali vd., 2020) şeklindedir.

Öte yandan yine doğada gerçekleşen olaylar ve akıllı yaklaşımlardan esinlenilmiş; toplulukların nesiller boyunca daha iyi bireyler üretmesine dayanan ve Holland (1975) tarafından geliştirilen Genetik Algoritma (GA) (Brady, 1985), algoritma süresince bazı adımları belirli sürelerce yasaklayan Tabu Arama Algoritması (Fiechter, 1994), iterasyonlar boyunca azalan oranda çeşitliliğe imkan tanıyan Benzetimli Tavlama Algoritması (Kirkpatrick vd., 1983), orijinali Geem vd. (2001) tarafından geliştirilen GSP için uyarlanmış Harmoni Arama Algoritması (Bouzidi & Riffi, 2014) ilk versiyonu Zheng (2015) tarafından geliştirilen, operatörleri kombinatoryal arama uzayında çalışacak şekilde adapte edilmiş Su Dalgası Algoritması (Wu vd., 2015) gibi yaklaşımlarla da GSP çözülmüştür.

GSP'nin çözümünde tur oluşturan sürü algoritmaları ile özellikle büyük boyutlara sahip problemlerde iyi amaç değerlerine yakınsama çok uzun sürmektedir (Deng vd., 2015:1). Çalışmada bu nedenle uygun çözümlerden yeni çözümler üreten ve sıklıkla problemin çözümünde tercih edilen GA kullanılmıştır.

GSP'de GA çözüm yaklaşımını kullanan bazı önemli çalışmalar değerlendirildiğinde; Potvin (1996) tarafından problemin çözümünde; kısmen eşlenmiş çaprazlama, döngü çaprazlama, sıralı çaprazlama, sıra temelli çaprazlama ve pozisyon temelli çaprazlama gibi farklı çaprazlama operatörlerinin performansları karşılaştırılmış ve düğüm sıralarını daha az bozan operatörlerin daha yüksek başarıma sahip olduğu sonucu ortaya çıkmıştır. Larranaga vd. (1999) tarafından yapılan çalışmadaysa çaprazlama ve mutasyon yaklaşımları topluca değerlendirilmiş ve kenar rekombinasyonu, sıralı, sıra temelli, pozisyon temelli çaprazlamalar ile yerini alma, araya yerleştirme, tersine döndürmeye dayalı mutasyon operatörlerinin GSP için daha başarılı çözümler ürettiğini ortaya koymuşlardır. Chang vd. (2010) çalışmalarında önerdikleri, popülasyonda dinamik çeşitliliği gözeten GA yaklaşımı ile GSP'de klasik GA ile araştırılmayan çözüm uzaylarına da bakılabildiğini ifade etmişlerdir. Razali & Geraghty (2011) çalışmalarında kendilerinden yeni kromozom üretilecek ebeveynlerin seçim yöntemlerini ele alıp GSP'de performanslarını değerlendirmişlerdir. Pulat & Kocakoç (2017) literatürde yer alan GSP test problemlerinde çaprazlama operatörlerini karşılaştırmışlar ve çalışmada sıralamayı koruyan operatörleri daha başarılı bulmuşlardır. Juneja vd. (2019) GA ile GSP çözümünü ele aldıkları çalışmalarında çözümün kalitesini artırmak için hibrit çaprazlama operatörlerinin çoklu kombinasyonlarının kullanılabileceğini ifade etmişlerdir. Rao & Hegde (2015) yaptıkları detaylı literatür çalışmasında GSP'nin çözümünde yeni önerilen GA yaklaşımlarını incelemişlerdir.

Öte yandan yakın zamanda, Hariyadi vd. (2020) yapay zekâ bakış açısıyla GA'yı GSP çözümünde kullanmışlardır. George & Amudha (2020) çok amaçlı optimizasyon modeli çerçevesinde ele aldıkları GSP'yi GA ile çözmüşlerdir. Ha vd. (2020) yapay uçan kamere yaklaşımı ile hibritleştirdikleri GA ile GSP'ye çözüm önerisi getirmişlerdir. Paul vd. (2020) yeni geliştirdikleri çaprazlama operatörünü GA'da kullanarak GSP'yi çözmüşlerdir. Iqbal vd. (2020) haritalanmış çaprazlama operatörü kullandıkları GA ile GSP çözümünde yeni bir yaklaşım sunmuşlardır.

Sezgisel algoritmalar ile elde edilen GSP sonuç değerlerini iyileştirmek için problem çözüm süreçlerine sıklıkla yerel arama stratejileri dâhil edilmektedir. Ulder vd. (1990) GSP çözümünde yerel arama ile iyileştirilmiş GA'nın klasik GA'ya göre daha iyi sonuçlar verdiğini ifade etmiştir. Freisleben & Merz (1996) yaptıkları çalışmada en yakın komşu ve 3-opt yerel

arama algoritmalarını karşılaştırmış ve en yakın komşu algoritmasının daha düşük başarıya sahip olduğunu ifade etmişlerdir. Lo vd. (2018) belirli nesillerde ve belirli kromozomlara yerel arama operatörleri ile iyileştirme yaptıkları melez GA yaklaşımı ile çoklu GSP çözümünde klasik GA'ya göre daha başarılı sonuçlar elde etmişlerdir.

Çalışmadaki ana amaç GSP özelinde, GA'da değişik yerel arama yöntemlerinin performanslarını değerlendirmek ve bu değerlendirmeler sonucunda farklı yöntemleri ardı ardına birlikte kullanan yeni bir yerel arama yöntemi ile güçlendirilmiş GA metodu önerisi getirmektir. Bu amaçla literatürde yer alan test problemleri üzerinden önerilen yaklaşım ve diğer bilinen yaklaşımlar karşılaştırılmıştır. Önerilen yaklaşımla kıyaslanan yaklaşımlar ise; Tsai vd. (2004) tarafından önerilen ve paralel karınca kolonileri ile en yakın komşuluğa dayanan ACOMAC+NN, Junman & Yi (2012)'nin ortaya koyduğu karınca kolonisi algoritması ve 2-opt yerel aramayı kullanan ACO+2opt ve karınca kolonisi algoritmasında bireysel varyasyon ve tur oluşturma stratejisine sahip, 2-opt yerel aramayı her döngü sonunda kullanan IVRS+2Opt, Othman vd. (2013) tarafından önerilen ve su akışı benzeri algoritma ve 2-opt ile 3-opt yerel arama yaklaşımlarının birlikte kullanıldıkları WFA+2-Opt ve WFA+3-Opt, Wang (2014) tarafından Hamilton yolu ve devresine dayanan iki yerel arama yaklaşımı ve GA'nın birleşimiyle önerilen HGA, Mahi vd. (2015) tarafından GSP için parçacık sürü ve karınca kolonisi optimizasyonlarına yerel arama yaklaşımının eklenmesi ile önerilen PSO-ACO-3opt, Zhou vd. (2015) tarafından ortaya konan kesikli istilacı ot algoritmasının sırasıyla tam 3-opt ve tam 2-opt yerel aramaları ile desteklendiği DIWO, Lin vd. (2016) tarafından GA'nın çaprazlama operatörünün yerel arama ile iyileştirildiği ve çifte köprü rassal mutasyonunu kullanan IHGA, Gülcü vd. (2018) tarafından önerilen paralel karınca kolonisi algoritmasının, belirli bir iterasyon sayısına ulaşıldığında tam 3-opt yerel arama stratejisi uygulanarak güçlendirildiği PACO-3Opt şeklinde olmuştur.

Çalışma kapsamında, önerilen algoritmada amaçlanan iyi sonuçlara ulaşmaya çalışırken aynı zamanda çözümlerdeki çeşitliliği korumak ve bu sayede yerel optimumlara takılmamak olmuştur. Bu nedenle, çalışmada yer alan yerel arama yaklaşımlarında her iki amaca uygun şekilde arama genişliği belirlenmiş ve her bir iterasyonda belirli sayıda çözüme ardışık bir şekilde uygulanmıştır. Çalışma, önerilen algoritmanın yapısının özgünlüğüyle GSP çözümünde yeni bir bakış açısı sağlamıştır.

2. Gezgin Satıcı Probleminin Matematiksel Modeli

Dantzig vd. (1954) tarafından ortaya atılan GSP matematiksel modeli aşağıdaki gibidir;

$$\min \sum_{i \neq j}^{n} d_{ij} x_{ij} \tag{1}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \quad (i = 1,...,n), (j \neq i) \tag{2}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \quad (j = 1,...,n), (j \neq i) \tag{3}$$

$$\sum_{i,j\in S}^{n} x_{ij} \leq |S| - 1, \forall S \subset \{1,...,n\}, |S| \geq 2 (j \neq i) \quad (4)$$

$$x_{ij} = 0 \ veya \ 1 \quad (i, j = 1,..., n; j \neq i) \quad (5)$$

Burada x_{ij} i. düğümden j. düğüme gidilmesi halinde 1, gidilmediğinde 0 değerini alan karar değişkenini, d_{ij} i ve j düğümleri arasındaki mesafeyi, n ise toplam düğüm sayısını belirtmektedir. GSP matematiksel modelindeki Eşitlik 1'de amaç fonksiyonu yer almaktadır. Tüm düğümlere birer kez uğranılmasıyla oluşan turda, sırasıyla uğranılan düğümlerin arasındaki mesafelerin tamamının toplanmasıyla amaç fonksiyonu elde edilmektedir. Eşitlik 2'deki kısıt ise her düğüme sadece bir kez gidilebileceğini ifade etmektedir. Eşitlik 3'deki kısıtlar ise her bir düğümden bir kez ayrılınabileceğini garanti altına almaktadır. Eşitlik 4 ise çözümde oluşabilecek alt turları elemeye yönelik oluşturulan ve çözümün tek bir turdan oluşmasını sağlayan kısıtlamadır. Son olarak Eşitlik 5'te ikili değer alabilen karar değişkenleri yer almaktadır. GSP'de d_{ij} ve d_{ji} mesafeleri birbirlerine eşit olduğunda bir başka ifadeyle düğümler arası geliş ve dönüş uzaklıkları aynı olması durumunda, problem simetrik GSP olarak ifade edilmektedir. Bu durumun sağlanmaması halinde ise, problem asimetrik GSP olarak ifade edilir. Çözüm uzayı, simetrik GSP'ye göre 2 kat daha büyük olan asimetrik GSP'de çözüm elde edilmesi bu durum nedeniyle daha da zor olmaktadır.

3. Genetik Algoritma Yaklaşımı

Holland (1975)'ın ortaya attığı GA'da evrim kavramından yola çıkılarak nesillerin git gide iyileşme süreci optimizasyon problemlerinde uygun çözümlerin iterasyonlar boyunca iyileşmesi için uyarlanmıştır.

GA, Holland (1975)'ın ortaya attığı ve canlıların evrimleşme sürecini taklit eden bir metottur. GA'da problem tipine göre her biri birer uygun çözümü ifade eden bireylerin gösterim şekli değişmektedir. GSP'de bu olası çözümler birer dizilim içermesi nedeniyle sıklıkla Tablo 1'deki kodlama şekli tercih edilmektedir.

Tablo 1: Problem İçin Birey Kodlama Şekli

Birey A	1	3	4	2	5
Birey B	2	4	3	5	1

Tablo 1'de 5 düğümlü problem için uygun çözüm sunan Birey A incelendiğinde gezgin satıcının rotası sırasıyla 1, 3, 4, 2 ve 5 numaralı düğümleri ziyaret edecek şekilde oluşmuştur ve son olarak gezgin satıcı başlangıç düğümü 1'e dönmektedir.

Algoritma için yeni nesillerin kendilerinden türeyeceği, bireylerden oluşan bir başlangıç popülasyonu olmak zorundadır. Başlangıç popülasyonunun çeşitliliği ve boyutu elde edilecek çözümün kalitesini etkilemektedir.

GA yaklaşımında başlangıçtan itibaren bir sonraki iterasyonun neslini meydana getirecek bireylerin oluşması için popülasyondan iki adet birey seçilmesi gerekmektedir. Farklı yolları bulunan seçim işlemi için GA'da sıklıkla kullanılan yaklaşımlar; sıra bazlı seçim, turnuva seçim ve rulet tekerleği seçim şeklindedir.

Seçilmiş olan iki bireyin yeni bireyi oluşturma biçimine GA'da çaprazlama işlemi adı verilmektedir. GA'da sıklıkla kullanılan çaprazlama yaklaşımlarından birkaçı ise; tek noktalı, çok noktalı, pozisyon temelli, sıra temelli şeklindedir. Burada, yeni üretilecek bireyin, üretimde kullanılan bireylerin iyi yönlerini alarak daha iyi uyum değerine sahip olması amaçlanmaktadır. Çaprazlamanın gerçekleşerek eski bireyin popülasyonu terk edip yeni bireyin katılımının olasılığı çaprazlama oranı olarak adlandırılmaktadır. Bu sayede popülasyonun eski nesillerinden kalan bilginin yeni bilgilerle etkileşim halinde kalması amaçlanmaktadır. Öte yandan GA yaklaşımında iyi bireylerin yeni nesillerde de bulunmasını sağlamak için elitizm adı verilen kavram kullanılmaktadır. Popülasyondaki elit bireyler çaprazlama ve mutasyon gibi tüm popülasyondaki bireyleri değiştirebilen işlemlerde herhangi bir değişime uğramadan korunmaktadır.

Çaprazlama ile elde edilen yeni bireyler, düşük olasılıklarla GA yaklaşımında çeşitliliği sağlamak için mutasyona tabi tutulmaktadırlar. GSP için GA'da sıklıkla kullanılan mutasyon yaklaşımları ise rassal iki düğümün yer değiştirmesi ve rassal iki düğümden ikincinin ilk seçilenin önüne eklenerek araya girmesi şeklindedir. Popülasyondaki bir bireyin mutasyona uğraması olasılığına ise mutasyon oranı adı verilmektedir.

GA, önceden belirli bir iterasyon sayısına göre veya algoritmada elde edilen en iyi uyum değerinin değişim hızına göre sonlandırılabilmekte ve problem için sonuçlar elde edilmektedir.

4. Yerel Arama Yaklaşımları

Çalışma kapsamında, GA'da 4 değişik yerel arama yöntemi farklı şekillerde kullanılarak karşılaştırılmıştır. İlgili teknikler, Jahed & Rahbari (2017)'nin çalışmalarında yer verdiği komşuluk oluşturma yapıları; Yer Değiştirme (YD), Tersine Döndürme (TD) ve Araya Yerleştirme (AY) yöntemleri ile Johnson & McGeoch (1997) tarafından GSP için önerilen 3-opt yaklaşımları olmuştur.

Tablo 2'de yer değiştirmenin nasıl yapıldığı görülmektedir. Burada altı çizili düğümler karşılıklı yer değiştirmektedir. Düğüm sayısı n olan birey için YD ile n*(n-1)/2 kadar yeni olası çözüm üretilebilmektedir.

Tablo 2: Yer Değiştirme (YD)

Eski Birey	**1**	2	3	**4**	5
Yeni Birey	**4**	2	3	**1**	5

Tablo 3'te TD'nin gerçekleşme şekli verilmiştir. Altı çizili düğümlerdeki sıralama tam tersi olacak şekilde döndürülmektedir. TD ile n düğüm sayısı için n*(n-1)/2 adet komşuluk oluşturulabilmektedir.

Tablo 3: Tersine Döndürme (TD)

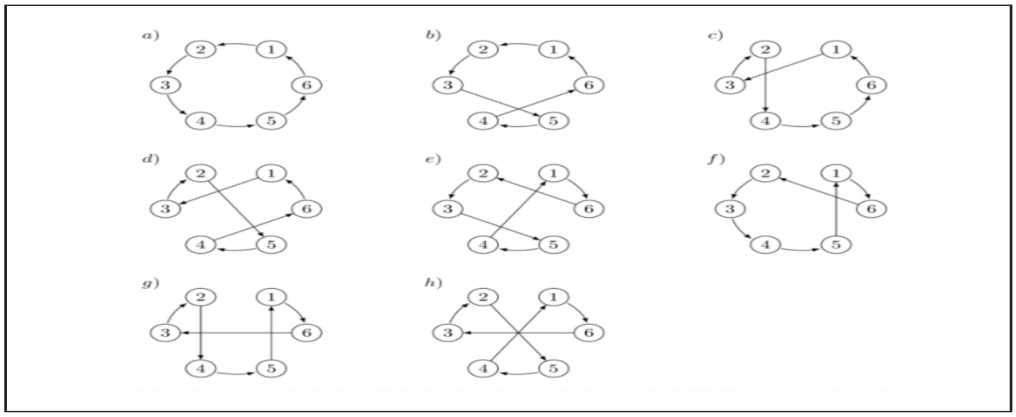
Eski Birey	**1**	**2**	**3**	**4**	5
Yeni Birey	**4**	**3**	**2**	**1**	5

Tablo 4'te AY'nin nasıl yapıldığı verilmiş olup araya yerleştirilecek değer ve aralık altı çizilidir. Bu durumda araya yerleştirilecek düğüm aralık düğümünün hemen yanına eklenerek yeni birey oluşturulur. YD ve TD'ye benzer şekilde AY ile de n düğüm sayısına sahip problemde bir bireyden n*(n-1)/2 adet yeni komşuluk elde edilmektedir.

Tablo 4: Araya Yerleştirme (AY)

Eski Birey	**1**	2	3	**4**	5
Yeni Birey	2	3	**4**	**1**	5

Şekil 1: Tüm Olası 3-Opt Yeniden Bağlanma Durumları



Kaynak: Gazda, M. (2020). Tsp algorithms: 2-opt, 3-opt in python. Erişim Tarihi: 15 Kasım 2020, http://matejgazda.com/.

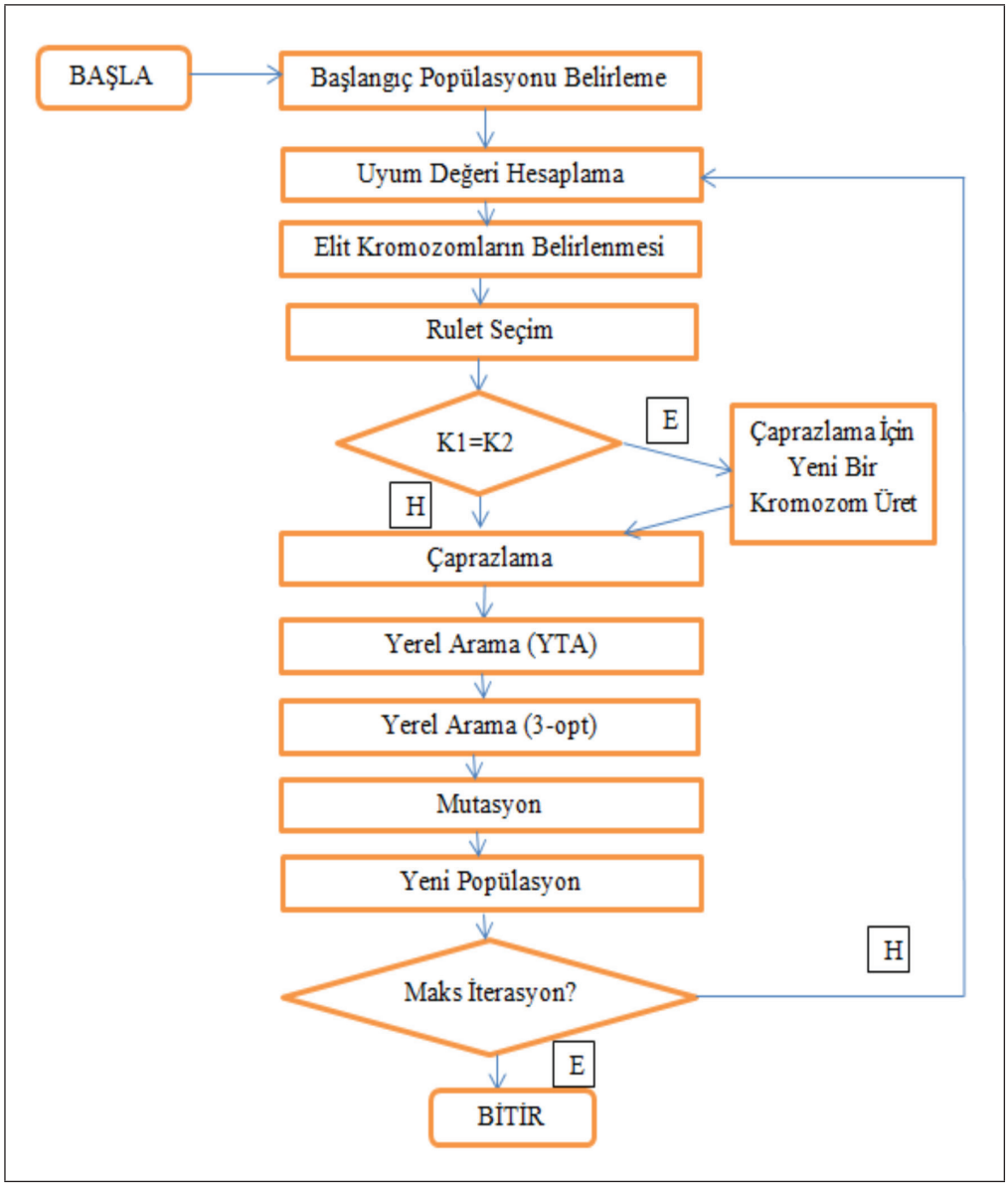
Şekil 1'de 3-opt'ye göre herhangi bir GSP için olası tüm yeniden sekiz adet birleşime yer verilmiştir. 3-opt yaklaşımında düğümler arası üç adet kenarın veya bir başka ifade ile bağlantının silinerek turun toplamda üç ayrı parçaya ayrılması ve sonrasında farklı şekillerde bu parçaların birleştirilmesine dayanmaktadır.

Şekil 1'e bakıldığında altı adet düğüm sıralaması bulunmaktadır. Orijinal tur (a) 1-2-3-4-5-6 şeklinde iken 3-opt'ye uygun olarak 1 ile 2, 3 ile 4 ve 5 ile 6 düğümleri arasındaki bağlantı kesilerek farklı 7 adet bağlantı oluşturulmuştur. Örneğin (e)'de sıralama 1-6-2-3-5-4 olacak biçimde değişmiştir.

5. Önerilen Hibrit Genetik Algoritma

Çalışmada yer alan tüm yerel arama yaklaşımlarını içeren ve çalışma kapsamında GSP için önerilen hibrit genetik algoritmanın akış çizelgesi Şekil 2'deki gibidir. Önerilen yaklaşımdaki tüm adımlar alt başlıklar halinde çalışmada ifade edilmiştir.

Şekil 2: Önerilen Hibrit Genetik Algoritma Akış Şeması



5.1. Başlangıç Popülasyonu Belirleme

Çalışma kapsamında başlangıç popülasyonunu oluşturan ilk nesil bireyler tekdüze dağılım kullanılarak GSP için uygun dizilimde oluşturulmuştur. Popülasyondaki birey sayısı, yapılan denemeler sonucunda problemdeki düğüm sayısının 4 katı olarak kullanılmıştır. Popülasyon büyüklüğü daha az olduğunda algoritmanın optimuma yakınsama oranın düşük

olduğu ve popülasyondaki çeşitliliğin hızlıca azalarak algoritmanın yerel optimumlara takıldığı gözlenmiştir. Popülasyon büyüklüğü belirlenen büyüklükten daha da çok artırıldığında ise bu durumun optimuma yakınsama açısından ek katkısı olmadığı görülmüştür.

5.2. Uyum Değeri Hesaplama

Popülasyonda yer alan tüm bireylerin uyum değerleri Eşitlik 1 ile GSP'deki amaç fonksiyonuna göre hesaplanmaktadır. Her bir bireyin uyum değeri hesaplandıktan sonra popülasyon, uyum değeri açısından en iyiden en kötüye doğru sıralanmaktadır.

5.3. Elit Kromozomların Belirlenmesi

Çalışmada uyum değeri hesaplaması sonucu en iyi %5'lik gruba giren bireyler elit olarak saklanarak, korunmaktadır. Popülasyonun elit bireyleri, çaprazlama ve mutasyon işlemleri sırasında popülasyondaki yerlerini kaybetmemektedir. Fakat çaprazlama işlemi için seçilmesi mümkün olan elit bireyler bu aşamada fayda sağlamaktadır.

5.4. Rulet Tekerleği Seçimi

Çalışma dâhilinde bireylerin seçimi için rulet tekerleği yaklaşımından faydalanılmıştır. Bu yaklaşımda uyum fonksiyon değerleri daha iyi olanlar doğru orantılı bir şekilde daha büyük olasılıkla seçilebilmektedir. Rulet seçimle elde edilen her iki birey de tıpatıp aynı dizilime sahip ise bir tanesi tekdüze dağılımla rassal şekilde üretilen yeni bir bireyle, çaprazlamada kullanılmak için yer değiştirmektedir.

5.5. Çaprazlama

Çalışma kapsamında, Davis (1985)'in ilk defa önerdiği ve aynı zamanda Pulat & Kocakoç (2017)'un da GSP için kullandığı sıra temelli çaprazlama operatöründen Tablo 5'te olduğu gibi faydalanılmıştır.

Tablo 5: Çaprazlama İşlemi

Birey A	2	5	1	3	4
Birey B	5	1	2	4	3
Birey C	_	5	1	_	_
Birey C	2	5	1	4	3

Tablo 5'te ilk birey olan Birey A'da altı çizili değerlerin bulunduğu aralık seçilmiş olsun. Yeni oluşacak Birey C'ye aralık aynı şekilde aktarılır. Bu sayede hem konum hem de sıralama korunmuş olur. Bir sonraki aşamada ise Birey B'nin kesim yerinden itibaren değerler aralığın sonundan başlamak suretiyle Birey C'ye aktarılır. Tur diziliminin korunması için hâlihazırda Birey C'de bulunan değerler aktarma işleminde kullanılmaz. Çalışma kapsamında her bir çaprazlama işlemi için aralık uzunluğu tekdüze dağılıma göre rassal bir şekilde hesaplanmıştır. Ortaya çıkan yeni birey, uyum değerine bakılmaksızın, popülasyona eski bireyin yerine dâhil edilmektedir. Çalışmada denemeler sonucunda çaprazlama oranı, 0.6 olasılıkla çaprazlama gerçekleşecek şekilde ayarlanmıştır.

5.6. Yerel Arama (YTA)

Çalışma kapsamında hibrit GA'da yer alan ilk yerel arama metodunda, komşuluk fonksiyonları YD, TD ve AS birlikte kullanılarak her bir iterasyonda, popülasyonda belirli sayıda kromozoma uygulanmakta ve her bir kromozomdan yeni kromozomlar üretilmesi amaçlanmaktadır. Eğer yeni üretilen kromozomların uyum değerlerinin her biri eski kromozomdan daha kötüyse eski kromozom popülasyonda kalmaya devam etmekte aksi takdirde ise en iyi uyum değerine sahip yeni üretilen kromozom eski kromozomun yerine popülasyona dâhil edilmektedir. Düğüm sayısı n olan birey için YD, TD ve AS yaklaşımlarının tamamı uygulanmakta ve buna göre üretilebilecek tüm komşuluklar yoluyla, toplamda 3*n*(n-1)/2 sayıda yeni kromozom üretilmektedir. Çalışmada bu üç komşuluk üreten yaklaşımın birlikte kullanıldığı yerel arama yaklaşımı YTA olarak adlandırılmıştır. YTA yerel araması için özel tabu listesi bulunmaktadır. Tabu liste A olarak ifade edilen bu yapı sayesinde daha önceden YTA aramasına tabi tutulan bir kromozom tekrardan araştırılmamaktadır. Azalan uyum değerine göre sıralanan popülasyonda en yüksek uyum değerine sahip kromozomdan başlanmak koşuluyla tabu liste A'da bulunan kromozomlar pas geçilerek, toplamda elit kromozom sayısı kadar kromozom YTA aramasına tabi tutulmaktadır. Örneğin popülasyondaki elit kromozom sayısı 5 olsun. Bu elit kromozomlardan bir tanesi tabu liste A'da olduğu takdirde elit kromozomlardan sonra gelen 6. sıradaki kromozom tabu liste A'da yer almıyorsa kendisine YTA uygulanmaktadır. Bu durumda, eğer popülasyonda üst sıralarda bulunan kromozomlar tabu liste A'da bulunduğunda, popülasyonun alt sıralarında bulunan kromozomların da YTA aramasına tabi tutulma olasılıkları ortaya çıkar. Eğer tüm kromozomlar tabu liste A'da ise YTA hiçbir kromozoma uygulanmaz.

5.7. Yerel Arama (3-opt)

Çalışmada önerilen Hibrit GA için kullanılan ikinci yerel arama yaklaşımı ise 3-opt'ye dayanmaktadır. Kullanılan 3-opt aramasına özel tabu liste B bulunmaktadır. Bu sayede daha önce 3-opt aramasına tabi tutulan kromozomlar tekrar aynı yerel aramada kullanılmamaktadır. Popülasyondaki daha önce 3-opt uygulanmamış elit kromozomların tamamına, n düğüm sayılı problem için YTA ile benzer sayıda, rassal biçimde seçilen 3*n*(n-1)/2 adet kadar komşuluk ayrı ayrı oluşturularak 3-opt araması gerçekleşmektedir. Popülasyondaki elit kromozomların tamamı tabu liste B'de ise elitler sonrası en yüksek uyum değerine sahip kromozomun tabu liste B'de olup olmadığına bakılmakta ve eğer tabu liste B'de yoksa onun için 3-opt arama yapılmakta, tabu liste B'de varsa ilgili iterasyonda hiçbir kromozom için 3-opt araması yapılmamaktadır. Gerçekleştirilen 3-opt sonrası eğer uyum değeri açısından daha iyi sonuç elde edilirse en yüksek uyum değerine sahip yeni kromozom, popülasyonda eski kromozomun yerine geçmektedir. Aksi halde bir işlem yapılmamaktadır. Örneğin popülasyondaki elit kromozom sayısı 3 olsun. Bu elit kromozomlardan ancak tamamı tabu liste A'da olduğu takdirde elit kromozomlardan sonra gelen 4. sıradaki kromozoma tabu liste B'de olup olmamasına göre 3-opt arama uygulanabilmektedir. Bu durumun nedeni ise 3-opt ile yapılan aramalarda hızlı optimuma yakınsama nedeniyle elit kromozomların çoğunlukla az oranda değişmesi ve tabu liste B'de olmasıdır. Bu durumda yerel optimumları aşmak için elitlerden sonra gelen en iyi değere sahip ve iterasyonlarda değişme olasılığı ve uyum değeri yüksek olan kromozom üzerinden bu engelin aşılabilmesi hedeflenmektedir.

5.8. Mutasyon

Çalışma kapsamında iki nokta seçilip ilk noktayı ikinci noktanın önüne getirerek sıralamayı daha az bozan araya girme yaklaşımı ile mutasyon yeni bireylere Tablo 6'daki gibi gerçekleştirilmiştir. Burada rassal biçimde seçili olan 3. ve 4. sıralar mutasyona tabi tutulacak olsun. İlgili sıralardaki düğümlerden ilk değer ikincinin önüne getirilerek Birey C'den yeni Birey D türetilmiş olur. Mutasyon oranı, denemeler sonucunda 0.3 olarak belirlenmiştir.

Tablo 6: Mutasyon İşlemi

Birey C	4	3	5	1	2
Birey D	4	3	1	5	2

5.9. Yeni Popülasyon

Mutasyon işleminden sonra ilgili iterasyonda yeni popülasyon oluşumu tamamlanmaktadır. Önceden belirlenmiş sayıda iterasyon gerçekleşmediğinde tekrardan "Uyum Değeri Hesaplama" adımına dönülmekte ve sıradaki adımlar tekrarlanmaktadır. İterasyon sayısı maksimum iterasyon sayısına ulaştığında ise algoritma sonlandırılmaktadır. Çalışma kapsamında maksimum iterasyon sayısı denemeler yoluyla 200 olarak belirlenmiştir. Daha az iterasyon sayısında daha kötü uyum değerleri elde edilirken, iterasyon sayısı 200'ü geçtikten sonra ise uyum değerinde önemli iyileşme gözlenmemektedir.

5.10. Çalışmadaki Hibrit Algoritma Yaklaşımları

Çalışmada karşılaştırılan 3 adet hibrit GA metodu bulunmaktadır. Bunlardan ilki sadece yerel arama safhası YTA'yı içeren ve GA-YTA olarak isimlendirilen yaklaşımdır. İkinci hibrit GA ise sadece yerel arama safhası 3-opt'u barındıran ve GA-3-opt şeklinde adlandırılan yaklaşımdır. Çalışmada önerilen ve Şekil 2'de akış çizelgesi bulunan hibrit GA ise YTA ve 3-opt yaklaşımlarını ardışık şekilde kullanan ve GA-YTA-3-opt olarak anılan yaklaşımdır. Algoritmalar MATLAB ortamında kodlanmış ve GA-YTA-3-opt kodları verilmiştir (Bknz. Ek 1-8).

6. Bulgular

Çalışmada Reinelt (1991)'ın oluşturduğu tsplib kütüphanesinden 10 adet çeşitli boyutlarda GSP örneklerinin kullanımı ile önerilen hibrit yaklaşımın etkinliğini değerlendirmek amaçlanmıştır. Tüm yöntemlerin MATLAB ortamında, Intel(R) Core(TM) i5-3210M CPU @ 2.50GHz, 2501 Mhz, 2 Çekirdek, 4 Mantıksal İşlemciye ve toplamda 8 GB Ram'e sahip bilgisayarda, problemlere göre 10'ar kez ayrı ayrı çalıştırılmasıyla elde edilen sonuçlar Tablo 7'de verilmiştir. Örnek problemlerin isimlerinin sonunda yer alan rakamsal değerler örnekte kaç adet düğüm noktası olduğunu ifade etmekte olup 5 örnek 100'ün altında düğüm sayısına sahipken diğer 5'i de 100 veya daha fazla düğüme sahiptir.

Tablo 7: Çalışmada Kullanılan Yöntemlerin Karşılaştırılması

Problem	Optimum	Bulunan En İyi Değer			Bulunan Ortalama Değer			Süre (Saniye)		
		GA-YTA	GA-3-opt	GA-YTA-3-opt	GA-YTA	GA-3-opt	GA-YTA-3-opt	GA-YTA	GA-3-opt	GA-YTA-3-opt
eil51	426	435	427	**426**	436.7	430.3	**428.1**	**17**	43	48
berlin52	7542	7542	7542	**7542**	7668.2	7560.9	**7555**	**19**	54	43
st70	675	675	675	**675**	683.6	678.9	**677.9**	**73**	96	104
eil76	538	545	545	**538**	554.5	551.3	**544.2**	**101**	118	117
rat99	1211	1223	1214	**1211**	1241.5	1237.4	**1224.8**	360	**236**	306
kroa100	21282	21292	21282	**21282**	21559	21389	**21321**	348	**217**	282
eil101	629	647	637	**629**	654.1	647.8	**641.8**	343	**262**	289
lin105	14379	14379	14379	**14379**	14524	14436	**14398**	515	**258**	314
kroa150	26524	26945	26784	**26707**	27430.5	27004	**26859**	2415	**835**	1067
ch150	6528	6642	6564	**6528**	6712.2	6692.9	**6575.8**	2178	**798**	1268

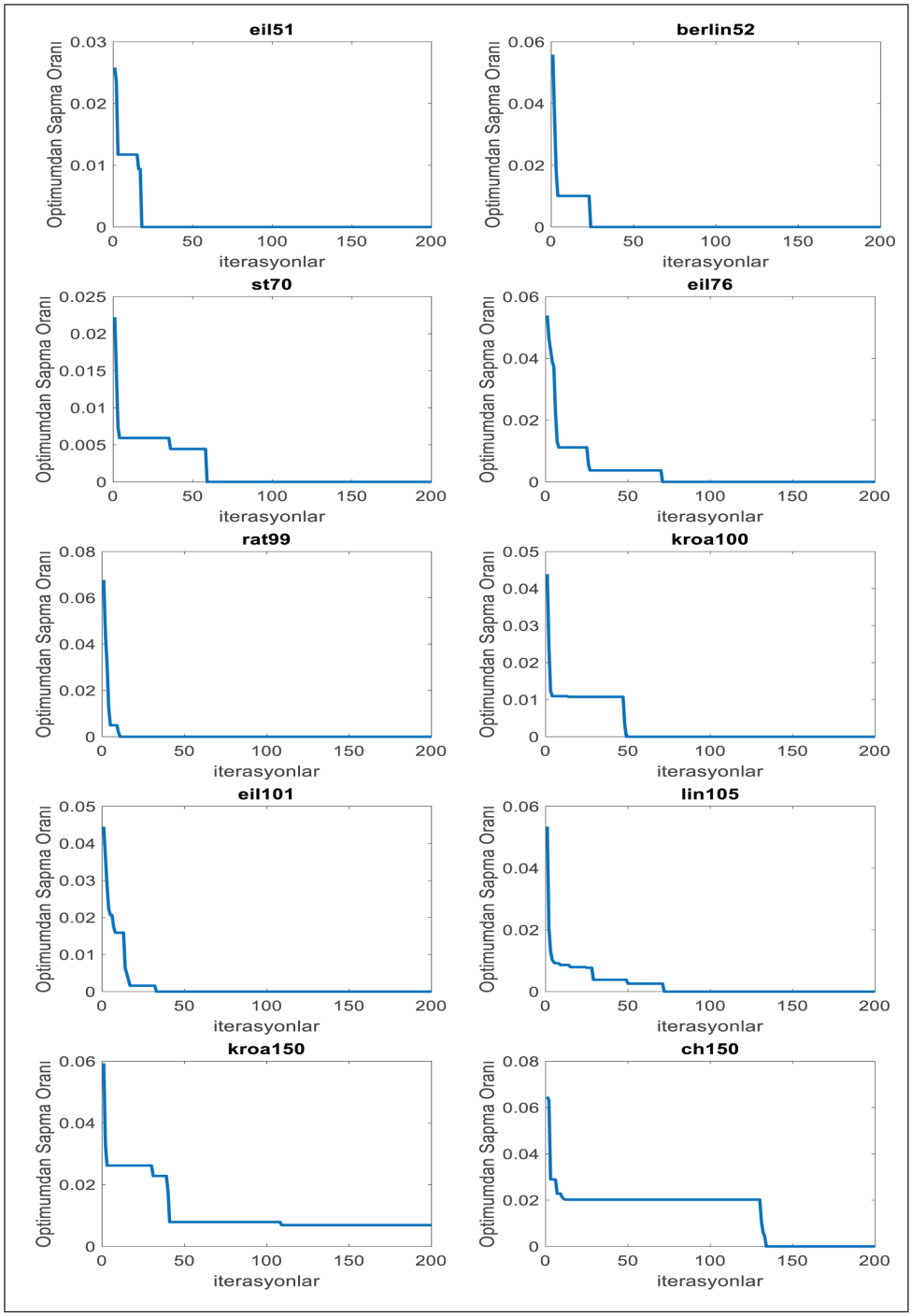
Tablo 7'deki sonuçlara göre önerilen GA-YTA-3-opt yaklaşımı tüm örnek problemler için bulunan en iyi değer bakımından diğerlerinden üstün veya eşit performansa sahip olduğu görülmektedir. Ayrıca GA-YTA-3-opt ile örnek problemlerden 9'unda optimum değer elde edilmişken kroa150 probleminde de optimuma oldukça yakın sonuçlar alınmıştır. 10 adet denemenin ortalama sonuçlarına bakıldığında ise benzer şekilde GA-YTA-3-opt yaklaşımı diğer iki hibrit yaklaşımdan tüm örneklerde daha iyi sonuçlar elde etmeyi başarmıştır. Denemelerin ortalama çalışma süreleri bakımından ise GA-YTA örnekteki düğüm sayısı 100'ün altında olduğunda diğerlerinden daha hızlı çalışırken, düğüm sayısı 100 civarı ve üstüne çıktığında ise diğerlerinden çok daha uzun çalışma sürelerine sahip olmuştur. Buna neden olarak ise ilgili yaklaşımda iterasyon süresi boyunca çeşitliliğin daha yüksek olduğu, diğer algoritmalarda ise belirli iterasyon sayısı sonrası algoritmanın çeşitliliğinin azalışa geçtiği ve uygulanan yerel aramalarda yer alan tabu liste yaklaşımının payı olduğu görülmektedir. Diğer iki yaklaşım olan GA-YTA-3-opt ve GA-3-opt'un ise benzer sürelere sahip olduğu görülmüştür. Genel olarak elde edilen sonuçlar incelendiğinde çalışmada önerilen GA-YTA-3-opt yaklaşımının daha yüksek performansa sahip olduğu ortaya çıkmıştır.

Şekil 3'te ise önerilen GA-YTA-3-opt yaklaşımının TSP örnekleri bakımından eğri gelişim grafikleri mevcuttur. Algoritma ile elde edilen en iyi çözümlere göre oluşturulan eğri gelişimlerinde bilinen optimumdan sapma oranı kullanılarak, algoritmanın optimuma yakınsama hızı ortaya konmuştur. Optimumdan sapma oranı Eşitlik 6'daki gibi hesaplanmaktadır.

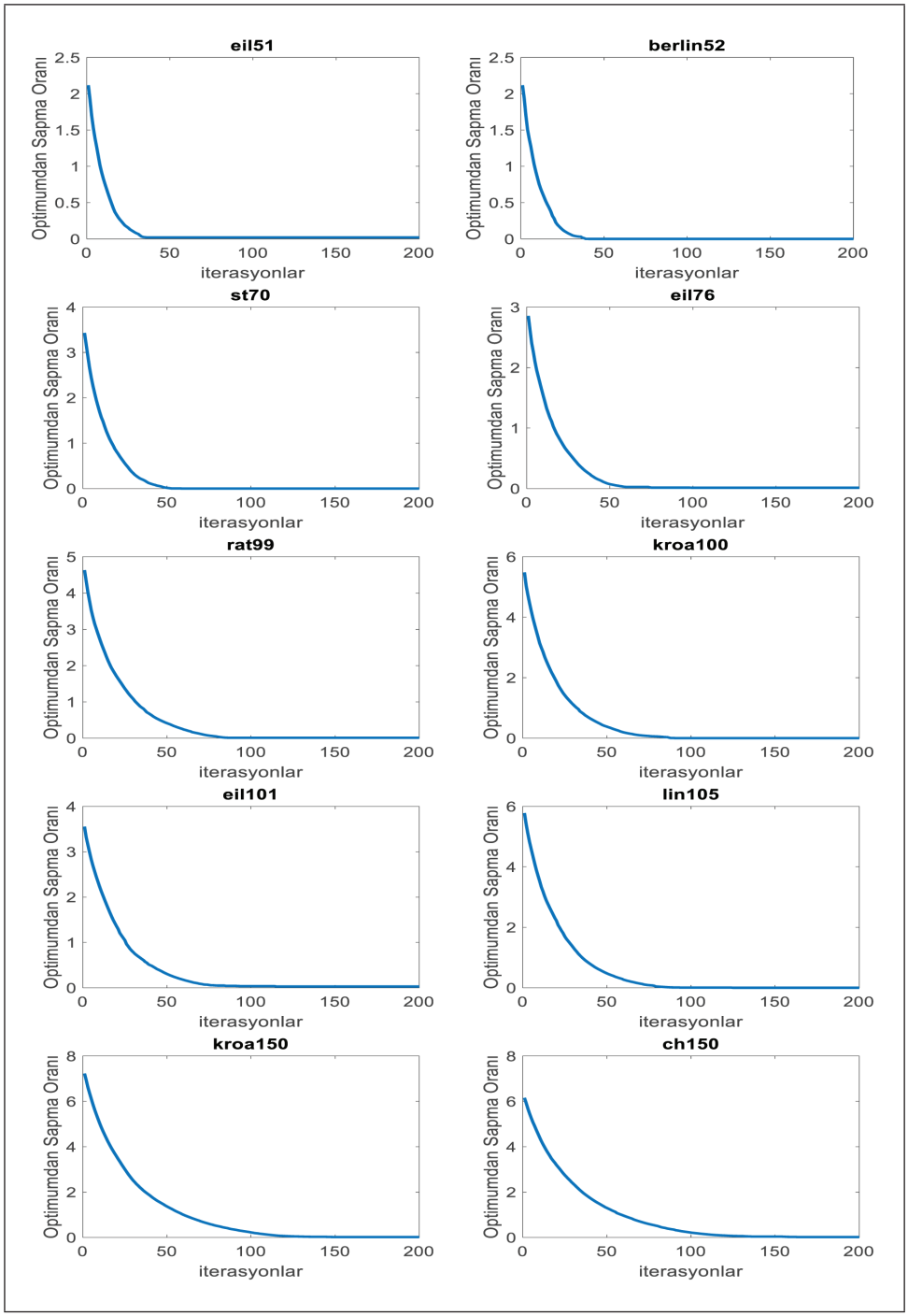
$$Optimumdan\ Sapma\ Oranı = \frac{Elde\ Edilen\ Uyum\ Degeri}{Optimum\ Deger} - 1 \quad (6)$$

GSP örneklerinde en kısa tur elde edilmeye çalışıldığı için, eğer elde edilen sonuçta optimumdan sapma varsa daha büyük bir uyum değerinin bulunduğu bilinmektedir. Örneğin optimumdan sapma oranı 5 olduğunda, elde edilen uyum değerinin optimum değerden %500 daha fazla olduğunu, bir başka ifadeyle optimumdan sapma yüzdesinin %500 olduğunu ifade etmektedir. Benzer şekilde, optimumdan sapma oranı 0.1, 0.01 ve 0 olduğunda elde edilen uyum değerinin sırasıyla, %10, %1 ve %0 optimum sapma yüzdesine sahip olduğu anlaşılmaktadır. Burada görüldüğü üzere düğüm sayısı 100'den az olan örnekler için genel olarak 50. iterasyon ve öncesinde en iyi sonuca yakınsama sağlanırken, daha büyük düğüm sayısına sahip problemlerde ise yakınsama 100. iterasyon civarında sağlanabilmektedir. GA-YTA-3-opt ile algoritmanın daha başlangıcında çok hızlı bir yakınsama olduğu gözlenmiştir. Örnekler değerlendirildiğinde optimumdan sapma oranı en fazla 0.06 olup, ilk iterasyonlardan itibaren en yüksek %6'lık optimumdan sapma yüzdesi bulunduğu görülmektedir. GA-YTA-3-opt ile yakınsamanın hızlı ve sıçramalı olduğu görülmüştür.

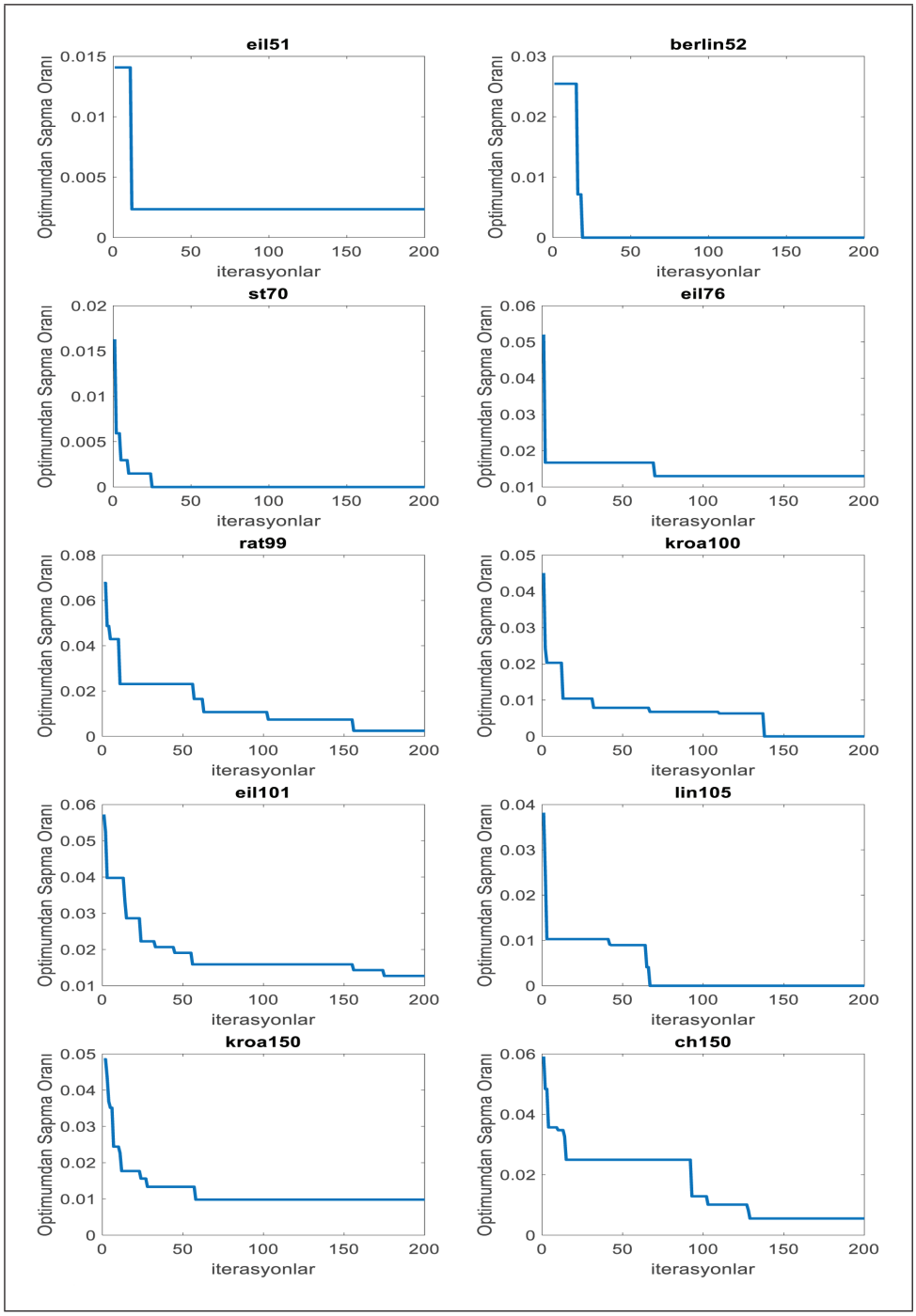
Şekil 3: TSP Örneklerinde GA-YTA-3-opt İçin Eğri Gelişimi



Şekil 4: TSP Örneklerinde GA-YTA İçin Eğri Gelişimi



Şekil 5: TSP Örneklerinde GA-3-opt İçin Eğri Gelişimi



Şekil 4'te GA-YTA yaklaşımının TSP örnekleri bakımından eğri gelişim grafikleri mevcuttur. Burada görüldüğü üzere düğüm sayısı 150 olan örnekler için genel olarak 100. iterasyon sonrasında en iyi sonuca yakınsama sağlanırken, daha küçük düğüm sayısına sahip problemlerde ise yakınsama 100. iterasyon öncesinde sağlanabilmektedir. GA-YTA ile algoritmanın başlangıcında optimuma yakınsama hızının diğer yöntemlere göre düşük olduğu gözlenmiştir. Örnekler değerlendirildiğinde optimumdan sapma oranı başlangıçta 2 ile 7 olup ilk iterasyonlarda optimumdan sapma yüzdelerinin de % 200 ile % 700 aralığında değiştiği görülmektedir. Şekil 4'teki sonuçlarda GA-YTA'nın örneklerden berlin52, st70 ve lin105 dışında istenen optimumdan sapma oranı 0'a ulaşamadığı gözlenmektedir. GA-YTA ile yakınsamanın yavaş fakat sürekli olduğu görülmektedir.

Şekil 5'te GA-3-opt yaklaşımının TSP örnekleri bakımından eğri gelişim grafikleri mevcuttur. Burada görüldüğü üzere düğüm sayısı arttıkça en iyi sonuca ulaşılan iterasyon sayısı da büyümektedir. GA-3-opt ile algoritmanın başlangıcında optimuma yakınsama hızının oldukça hızlı olduğu gözlenmiştir. Örnekler değerlendirildiğinde optimumdan sapma oranı en fazla 0.07 olup, ilk iterasyonlardan itibaren en yüksek %7'lik optimumdan sapma yüzdesi bulunduğu görülmektedir. GA-3-opt ile yakınsamanın hızlı ve sıçramalı olduğu görülmüştür. Şekil 5'teki sonuçlarda GA-3-opt'un örneklerden berlin52, st70, kroa100 ve lin105 dışında istenen optimumdan sapma oranı 0'a ulaşamadığı gözlenmektedir.

Genel olarak üç hibrit GA'nın eğrisel gelişimlerine bakıldığında GA-YTA'nın diğerlerinden oldukça farklı şekilde, optimizasyon sürecinin başlarında oldukça kötü uyum değerleri ile başladığı ve algoritma çalışma süresince yavaş ve istikrarlı bir şekilde optimuma yakınsadığı fakat 100. iterasyon civarı sonrasında yakınsama hızının neredeyse tükendiği görülmüştür. GA-YTA-3-opt ve GA-3-opt açısından eğrisel gelişim benzer olarak çok hızlı yakınsama ve sonrasında belirli anlarda iyileşme görülmektedir. Fakat bu iki algoritmadan GA-YTA-3-opt, diğerinden farklı olarak takıldığı yerel optimum noktalarını YTA araması sayesinde aşabilmektedir. Örneğin, GA-YTA-3-opt ile, 10 problemin 9'unda optimum değere ulaşılabilirken GA-3-opt ile 4 problemde, GA-YTA ile sadece 3 problemde optimum değer elde edilebilmiştir.

Çalışma kapsamında önerilen GA-YTA-3-opt yaklaşımının performansı literatürdeki önemli ve yerel arama stratejileri kullanan hibrit yaklaşımlarla ortalama elde edilen sonuçlar optimumdan sapma yüzdesi bakımından karşılaştırılmıştır. Marinakis vd. (2011:4694) tarafından önerilen optimumdan sapma yüzdesi aşağıdaki eşitlik yardımıyla hesaplanmaktadır;

$$Optimumdan\ Sapma\ Yüzdesi = \frac{Ortalama\ Uyum\ Degeri - Optimum\ Deger}{Optimum\ Deger} x100 \quad (7)$$

Tablo 8'deki karşılaştırma sonuçlarına bakıldığında önerilen GA-YTA-3-opt ile eil51 için literatürdeki 10 yöntemin 5'inden, berlin52 için 9 yöntemin 2'sinden, st70 için 4 yöntemin 1'inden, eil76 için 7 yöntemin 2'sinden, rat99 için 4 yöntemin 2'sinden, kroa100 için 9 yöntemin 6'sından, eil101 için 8 yöntemin 4'ünden, lin105 için 5 yöntemin 2'sinden, kroa150 için 5 yöntemin 1'inden, ch150 için 5 yöntemin 2'sinden daha iyi sonuç elde edilmiştir. Öte yandan önerilen GA-YTA-3-opt yöntemiyle elde edilen ortalama değerlerin optimumdan sapma yüzdesi en çok % 2 civarı olarak gerçekleşmiş ve buna göre önerilen yaklaşımın

yakınsama oranı açısından da yüksek performansa sahip olduğu ortaya çıkmıştır. Tablo 8'deki sonuçlar topluca değerlendirildiğinde GA-YTA-3-opt yönteminin literatürdeki yöntemlerle GSP açısından rekabet edebilir düzeyde olduğu görülmüştür.

Tablo 8: GA-YTA-3-opt Yönteminin Literatürdeki Yöntemlerle Karşılaştırılması

Yöntem	Örnek	eil51	berlin52	st70	eil76	rat99	kroa100	eil101	lin105	kroa150	ch150
	Opt.[1]	426	7542	675	538	1211	21282	629	14379	26524	6528
GA-YTA-3-opt	Ort.[2]	428.1	7555	677.9	544.2	1224.8	21321	641.8	14398	26859	6575.8
	Sap.[3]	0.493	0.172	0.43	1.152	1.140	0.183	2.035	0.132	1.263	0.732
Gülcü vd. (2018) PACO-3Opt	Ort.	426.35	7542	677.85	539.85	1217.1	21326.8	630.55	14393		6601.4
	Sap.	0.082	0	0.422	0.345	0.504	0.211	0.246	0.097		1.124
Mahi vd. (2015) PSO-ACO-3Opt	Ort.	426.45	7543.2	678.2	538.3	1227.4	21445.1	632.7	14379.2		6563.9
	Sap.	0.106	0.016	0.474	0.056	1.354	0.766	0.588	0.001		0.551
Junman & Yi (2012) IVRS+2opt	Ort.	431.1	7547.23				21498.6	648.67			
	Sap.	1.197	0.069				1.018	3.127			
ACOMAC+ NN (2004)	Ort.	430.68			555.9		21433.3				
	Sap.	1.099			3.327		0.711				
Junman & Yi (2012) ACO+2opt	Ort.	439.25	7556.58				23441.8	672.37			
	Sap.	3.110	0.193				10.148	6.895			
Othman vd. (2013) WFA+2-Opt	Ort.	426.65	7542		541.22		21282	639.87	14379	26710.3	6572.1
	Sap.	0.153	0		0.599		0	1.728	0	0.702	0.676
Othman vd. (2013) WFA+3-Opt	Ort.	426.6	7542		539.44		21282.8	633.5	14459.4	26705.9	6700.1
	Sap.	0.141	0		0.268		0.004	0.715	0.559	0.686	2.636
Lin vd. (2016) IHGA	Ort.	440	7559		559	1257		661		27297	
	Sap.	3.286	0.225		3.903	3.799		5.087		2.914	
Wang (2014) HGA	Ort.	429.19	7544.37	677.39	546.06	1221.95	21312.5	644.82	14422.9	26597.8	6557.7
	Sap.	0.749	0.031	0.354	1.498	0.904	0.143	2.515	0.305	0.278	0.455
Zhou vd. (2015) DIWO	Ort.	428.98	7544.36	677.12			21290			26730.4	
	Sap.	0.6999	0.0313	0.3125			0.0375			0.778	

Not: (1)'deki opt. Problemin optimum sonuç değerini, (2)'deki ort. Algoritmalar ile elde edilen ortalama sonuç değerini ve (3)'deki sap. ise ilgili ortalama değerinin optimumdan sapma yüzdesini ifade etmektedir.

5. Sonuç ve Öneriler

GSP birçok alanda karşılaşılan ve çözüme kavuşturulması bir hayli zor olan çok sayıda rotalama probleminin temel hali olup sıklıkla sezgisel metotlar ile çözülebilmektedir. Problemin boyutu büyüdükçe çözüm aşamasında kullanılması kaçınılmaz hale gelen sezgisel metotlar ile optimum değere yakınsayan çözümler elde edilebilmektedir. Birçok optimizasyon probleminde olduğu gibi GSP için de sıklıkla tercih edilen GA oldukça etkili sonuçlar üretebilen bir sezgisel algoritmadır. GA'ya diğer sezgisel algoritmalara da yapıldığı gibi farklı yaklaşımlar ekleyerek etkinliğini artırmaya çalışmak sık görülen bir durumdur. Yerel arama yaklaşımları ile GA'da elde edilen iyi sonuçların etrafında detaylı arama yapılması, hibrit yöntemlere önemli bir örnektir. Çalışma kapsamında YD, TD ve AS tabanlı GA-YTA, 3-opt tabanlı GA-3-opt ve önerilen karma yaklaşım GA-YTA-3-opt literatürde yer alan GSP örnekleri üzerinden başarımları kıyaslanmıştır. Sonuçlar değerlendirildiğinde önerilen algoritma GA-YTA-3-opt genel olarak diğer yaklaşımlara göre daha yüksek başarıma sahip olduğu görülmektedir. Önerilen GA-YTA-3-opt, yerel arama YTA ile arama bölgelerinde çeşitlilik sağlarken aynı zamanda 3-opt ile de belirli bölgelerde yoğun arama yapan yapısıyla, GA-YTA'daki yoğun arama yapamama sorunu ile GA-3-opt'deki çeşitlilik azlığından yerel optimum takılma problemlerini içermemekte ve ardışık yerel arama yapısı ile bazı durumlarda yerel aramaların ilkinden elde edilen çıktının ikinci yerel aramaya girdi olabilmesiyle, GSP için etkili sonuçlar üretebilmektedir. Öte yandan test örnekleri üzerinden yapılan kıyaslama ile GA-YTA-3-opt yönteminin literatürdeki algoritmalarla rekabet edebildiği gözlenmiştir. Bu açıdan da önerilen yaklaşımın GSP örnekleri için kabul edilebilir sonuçlar üreterek başarılı olduğu ortaya konmuştur. Gelecekteki çalışmalar için GSP'ye özgü üretilecek yerel arama yaklaşımları veya iyileştirme metotlarının bir arada kullanılarak, hibritleştirilen GA'nın başarımının ve etkinliğinin artacağı öngörülmektedir.

Çıkar Çatışmaları ve Katkı Beyanı

Çalışmada, sonuçları veya yorumları etkileyebilecek herhangi bir maddi hata veya herhangi bir kurum ya da kişiler ile çıkar çatışması olmadığını beyan ederim. Yazar, çalışmanın tümüne tek başına katkı sağlamıştır.

Kaynakça

Ali, I. M., Essam, D. & Kasmarik, K. (2020). A novel design of differential evolution for solving discrete traveling salesman problems. Swarm and Evolutionary Computation, 52, 100607.

Basu, S. & Ghosh, D. (2008). A review of the tabu search literature on traveling salesman problems. IIMA Working Papers, Indian Institute of Management, No: 2008-10-01, Ahmedabad.

Bouzidi, A. & Riffi, M. E. (2013). Discrete cat swarm optimization to resolve the traveling salesman problem. International Journal of Advanced Research in Computer Science and Software Engineering, 3(9), 13-18.

Bouzidi, M. & Riffi, M. E. (2014). Adaptation of the harmony search algorithm to solve the travelling salesman problem. Journal of Theoretical & Applied Information Technology, 62(1), 157-160.

Brady, R. M. (1985). Optimization strategies gleaned from biological evolution. Nature, 317, 804–806

Chang, P. C., Huang, W. H. & Ting, C. J. (2010). Dynamic diversity control in genetic algorithm for mining unsearched solution space in tsp problems. Expert Systems with Applications, 37(3), 1863-1878.

Chen, Y., Jia, Z., Ai, X., Yang, D. & Yu, J. (2017). A modified two-part wolf pack search algorithm for the multiple traveling salesmen problem. Applied Soft Computing, 61, 714-725.

Chu, S. C., Tsai, P. W. & Pan, J. S. (2006). Cat swarm optimization. 9th Pacific Rim International Conference on Artificial Intelligence, August 7-11, Guilin, China, Proceedings, 854-858.

Dantzig, G., Fulkerson, R. & Johnson, S. (1954). Solution of a large-scale traveling-salesman problem. Journal of the Operations Research Society of America, 2(4), 393-410.

Daoqing, Z. & Mingyan, J. (2020). Parallel discrete lion swarm optimization algorithm for solving traveling salesman problem. Journal of Systems Engineering and Electronics, 31(4), 751-760.

Davis, L. (1985). Applying adaptive algorithms to epistatic domains. Proceedings of the International Joint Conference on Artificial Intelligence, August 18-23, Los Angeles CA, 162–164.

Demirtaş, Y. E. & Keskintürk, T. (2011). Kanguru algoritması ve gezgin satıcı problemine uygulanması. İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi, 10(19), 51-63.

Deng, Y., Liu, Y. & Zhou, D. (2015). An improved genetic algorithm with initial population strategy for symmetric tsp. Mathematical Problems in Engineering, 2015, 1-6.

Dorigo, M. (1992). Optimization, learning and natural algorithms (Unpublished PhD Thesis). Politecnico di Milano, Italy.

Duman, E., Uysal, M. & Alkaya, A. F. (2012). Migrating birds optimization: A new metaheuristic approach and its performance on quadratic assignment problem. Information Sciences, 217, 65-77.

Fiechter, C. N. (1994). A parallel tabu search algorithm for large traveling salesman problems. Discrete Applied Mathematics, 51(3), 243-267.

Freisleben, B. & Merz, P. (1996). A genetic local search algorithm for solving symmetric and asymmetric traveling salesman problems. IEEE International Conference on Evolutionary Computation, May 20-22, Nagoya, Japan, Proceedings, 616-621.

Gazda, M. (2020). Tsp algorithms: 2-opt, 3-opt in python. Erişim Tarihi: 15 Kasım 2020, http://matejgazda.com/.

Geem, Z. W., Kim, J. H. & Loganathan, G. V. (2001). A new heuristic optimization algorithm: Harmony search. Simulation, 76(2), 60-68.

George, T. & Amudha, T. (2020). Genetic algorithm based multi-objective optimization framework to solve traveling salesman problem. In H. Sharma, K. Govindan, R. C. Poonia, S. Kumar & W. M. El-Medany (Eds.), Advances in computing and intelligent systems, Proceedings of ICACM 2019 (pp. 141-151), Singapore: Springer.

Gülcü, Ş., Mahi, M., Baykan, Ö. K. & Kodaz, H. (2018). A parallel cooperative hybrid method based on ant colony optimization and 3-Opt algorithm for solving traveling salesman problem. Soft Computing, 22(5), 1669-1685.

Gupta, R., Shrivastava, N., Jain, M., Singh, V. & Rani, A. (2018). Greedy woa for travelling salesman problem. Second International Conference, ICACDS 2018, April 20-21, Dehradun, India, Advances in Computing and Data Sciences, Revised Selected Papers, Part I, 321-330.

Ha, Q. M., Deville, Y., Pham, Q. D. & Hà, M. H. (2020). A hybrid genetic algorithm for the traveling salesman problem with drone. Journal of Heuristics, 26(2), 219-247.

Hariyadi, P. M., Nguyen, P. T., Iswanto, I. & Sudrajat, D. (2020). Traveling salesman problem solution using genetic algorithm. Journal of Critical Reviews, 7(1), 56-61.

Holland, J. H. (1975). Adaptation in natural and artificial systems. Ann Arbor: University of Michigan Press.

Iqbal, Z., Bashir, N., Hussain, A. & Cheema, S. A. (2020). A novel completely mapped crossover operator for genetic algorithm to facilitate the traveling salesman problem. Computational and Mathematical Methods, 2(6), 1-13.

Jahed, A. & Rahbari, M. (2017). Comparison of three neighbor generation structures by simulated annealing method to solve quadratic assignment problem. 10th International Conference of Iranian Operations Research Society (ICIORS 2017), May 3-5, Babolsar, Iran.

Jati, G. K. (2011). Evolutionary discrete firefly algorithm for travelling salesman problem. ICAIS: International Conference on Adaptive and Intelligent Systems, September 6-8, Klagenfurt, Austria, Proceedings, 393-403.

Johnson D. S. & McGeoch L. A. (1997). The traveling salesman problem: A case study in local optimization. In E. H. L. Aarts & J. K. Lenstra (Eds.), Local search in combinatorial optimization (pp. 215-310), New York: John Wiley & Sons.

Juneja, S. S., Saraswat, P., Singh, K., Sharma, J., Majumdar, R. & Chowdhary, S. (2019). Travelling salesman problem optimization using genetic algorithm. 2019 Amity International Conference on Artificial Intelligence (AICAI), February 4-6, Dubai, UAE, Proceedings, 264-268.

Jun-man, K. & Yi, Z. (2012) Application of an improved ant colony optimization on generalized traveling salesman problem. Energy Procedia, 17(2012), 319–325.

Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization (Technical Report). Erciyes University, Engineering Faculty, Computer Engineering Department.

Karaboga, D. & Gorkemli, B. (2011). A combinatorial artificial bee colony algorithm for traveling salesman problem. 2011 International Symposium on Innovations in Intelligent Systems and Applications (INISTA), June 15-18, Istanbul, Turkey, Proceedings, 50-53.

Karagül, K. (2019). Prüfer-Karagül algoritması: Gezgin satıcı problemi için yeni bir yaklaşım. Mehmet Akif Ersoy Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi, 6(2), 452-470.

Kennedy, J. & Eberhart, R. (1995). Particle swarm optimization. ICNN'95-International Conference on Neural Networks, November 27-December 1, Perth, Western Australia, Proceedings, Vol. 4, 1942-1948.

Kirkpatrick, S., Gelatt, C. D. & Vecchi, M. P. (1983). Optimization by simulated annealing. Science, 220(4598), 671-680.

Larranaga, P., Kuijpers, C. M. H., Murga, R. H., Inza, I. & Dizdarevic, S. (1999). Genetic algorithms for the travelling salesman problem: A review of representations and operators. Artificial Intelligence Review, 13(2), 129-170.

Lin, B. L., Sun, X. & Salous, S. (2016). Solving travelling salesman problem with an improved hybrid genetic algorithm. Journal of Computer and Communications, 4(15), 98-106.

Liu, S. J., Yang, Y. & Zhou, Y. Q. (2018). A swarm intelligence algorithm-lion swarm optimization. Pattern Recognition and Artificial Intelligence, 31(5), 431-441.

Lo, K. M., Yi, W. Y., Wong, P. K., Leung, K. S., Leung, Y. & Mak, S. T. (2018). A genetic algorithm with new local operators for multiple traveling salesman problems. International Journal of Computational Intelligence Systems, 11(1), 692-705.

Mahi, M., Baykan, Ö. K. & Kodaz, H. (2015). A new hybrid method based on particle swarm optimization, ant colony optimization and 3-opt algorithms for traveling salesman problem. Applied Soft Computing, 30, 484-490.

Marinakis, Y., Marinaki, M. & Dounias, G. (2011). Honey bees mating optimization algorithm for the Euclidean traveling salesman problem. Information Sciences, 181(20), 4684-4698.

Mirjalili, S. & Lewis, A. (2016). The whale optimization algorithm. Advances in Engineering Software, 95, 51-67.

Mzili, I. & Riffi, M. E. (2015). Discrete penguins search optimization algorithm to solve the traveling salesman problem. Journal of Theoretical & Applied Information Technology, 72(3), 331–336.

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

Othman, Z. A., Srour, A. I., Hamdan, A. R. & Ling, P. Y. (2013). Performance water flow-like algorithm for tsp by improving its local search. International Journal of Advancements in Computing Technology, 5(14), 126-137.

Ouaarab, A., Ahiod, B. & Yang, X. S. (2014). Discrete cuckoo search algorithm for the travelling salesman problem. Neural Computing and Applications, 24(7-8), 1659-1669.

Paul, P. V., Ganeshkumar, C., Dhavachelvan, P. & Baskaran, R. (2020). A novel ODV crossover operator-based genetic algorithms for traveling salesman problem. Soft Computing, 1-31.

Pollard, J. M. (1978). Monte Carlo methods for index computation *(modp)*. Mathematics of Computation, 32(143), 918-924.

Potvin, J. Y. (1996). Genetic algorithms for the traveling salesman problem. Annals of Operations Research, 63(3), 337-370.

Pulat, M. & Kocakoç, İ. D. (2017). Gezgin satıcı probleminin çözümünde kullanılan genetik algoritmanın parametrelerinin incelenmesi. Uluslararası İktisadi ve İdari İncelemeler Dergisi, Özel Sayı, 21-36.

Raman, V. & Gill, N. S. (2017). Review of different heuristic algorithms for solving travelling salesman problem. International Journal of Advanced Research in Computer Science, 8(5), 423-425.

Rao, A. & Hegde, S. K. (2015). Literature survey on travelling salesman problem using genetic algorithms. International Journal of Advanced Research in Eduation Technology (IJARET), 2(1), 42-45.

Razali, N. M. & Geraghty, J. (2011). Genetic algorithm performance with different selection strategies in solving tsp. International Conference of Computational Intelligence and Intelligent Systems (ICCIIS'11), July 6-8, London, UK, Proceedings, 2(1), 1-6.

Reinelt, G. (1991). TSPLIB—A traveling salesman problem library. ORSA Journal on Computing, 3(4), 376-384.

Ruiz-Vanoye, J. A., Díaz-Parra, O., Cocón, F., Soto, A., Arias, M. D. L. Á. B., Verduzco-Reyes, G. & Alberto-Lira, R. (2012). Meta-heuristics algorithms based on the grouping of animals by social behavior for the traveling salesman problem. International Journal of Combinatorial Optimization Problems and Informatics, 3(3), 104-123.

Storn, R. & Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization, 11(4), 341-359.

Tinós, R., Whitley, D. & Ochoa, G. (2020). A new generalized partition crossover for the traveling salesman problem: Tunneling between local optima. Evolutionary Computation, 28(2), 255-288.

Tongur, V. & Ülker, E. (2016). The analysis of migrating birds optimization algorithm with neighborhood operator on traveling salesman problem. The 19th Asia Pasific Symposium, November 22-25, Bangkok, Thailand, Proceedings, 227-237.

Tsai, C. F., Tsai, C. W. & Tseng, C. C. (2004). A new hybrid heuristic approach for solving large traveling salesman problem. Information Sciences, 166(1-4), 67-81.

Ulder, N. L., Aarts, E. H., Bandelt, H. J., Van Laarhoven P. J. & Pesch, E. (1990). Genetic local search algorithms for the traveling salesman problem. PPSN: International Conference on Parallel Problem Solving from Nature, October 1-3, Dortmund, FRG, Prooceedings, 109-116.

Wang, G. G., Hao, G. S., Cheng, S. & Qin, Q. (2016). A discrete monarch butterfly optimization for Chinese tsp problem. The 7th International Conference on Swarm Intelligence (ICSI 2016), June 25-30, Bali, Indonesia, Proceedings, 165-173.

Wang, G. G., Zhao, X. & Deb, S. (2015). A novel monarch butterfly optimization with greedy strategy and self-adaptive. International Conference on Soft Computing and Machine Intelligence (ISCMI), November 23-24, Hong Kong, Proceedings, 45-50.

Wang, K. P., Huang, L., Zhou, C. G. & Pang, W. (2003). Particle swarm optimization for traveling salesman problem. 2003 International Conference On Machine Learning and Cybernetics (ICMLC), November 5, Xi'an, China, Proceedings, Vol. 3, 1583-1585.

Wang, Y. (2014). The hybrid genetic algorithm with two local optimization strategies for traveling salesman problem. Computers & Industrial Engineering, 70, 124-133.

Wong, L. P., Low, M. Y. H. & Chong, C. S. (2008). A bee colony optimization algorithm for traveling salesman problem. Asia International Conference on Modelling & Simulation, May 13-15, Kuala Lumpur, Malaysia, Proceedings, 818-823.

Wu, X. B., Liao, J. & Wang, Z. C. (2015). Water wave optimization for the traveling salesman problem. ICIC: International Conference on Intelligent Computing, August 20-23, Fuzhou, China, Proceedings, 137-146.

Yang, C., Tu, X. & Chen, J. (2007). Algorithm of marriage in honey bees optimization based on the wolf pack search. International Conference on Intelligent Pervasive Computing (IPC), October 11-13, Jeju Island, Korea, Proceedings, 462-467.

Yang, X. S. (2010). A new metaheuristic bat-inspired algorithm. In J. R. González, D. A. Pelta, C. Cruz, G. Terrazas & N. Krasnogor (Eds.), Nature inspired cooperative strategies for optimization (NICSO 2010) (pp. 65-74), Berlin: Springer.

Yang, X. S. & Deb, S. (2009). Cuckoo search via Lévy flights. World Congress on Nature & Biologically Inspired Computing (NaBIC), December 9-11, Coimbatore, India, Proceedings, 210-214.

Yang, X. S. (2010). Firefly algorithm, stochastic test functions and design optimization. International Journal of Bio-Inspired Computation, 2(2), 78-84.

Zheng, Y. J. (2015). Water wave optimization: A new nature-inspired metaheuristic. Computers & Operations Research, 55, 1-11.

Zhou, Y., Luo, Q., Chen, H., He, A. & Wu, J. (2015). A discrete invasive weed optimization algorithm for solving traveling salesman problem. Neurocomputing, 151, 1227-1236.

Ek

Ek 1: GA-YTA-3-opt Ana Fonksiyon

```
clc;clear;
tic
ctime = cputime;
it=10;
bah=zeros(152,it);
graf=zeros(it,200);
for ate=1:it
clc;clearvars -except ate bah it ctime graf
[hes,say,a,c,e,elor,cor,mor,d,populasyon,maliyet,maksiter]=tspbilgi;
[itereniyi,I]=min(maliyet);%min yönlü
itereniyipop=populasyon(I,:);%globest
[maliyet,indexmal]=sort(maliyet);
populasyon=populasyon(indexmal,:);
tabulisto=zeros(1,size(d,2));
tabulistopt=zeros(1,size(d,2));
for ik=1:maksiter
 %ÇAPRAZLAMA
[hes,populasyon]=caprazlama3(hes,elor,cor,populasyon,maliyet);
%YEREL ARAMA
[hes,populasyon,tabulisto]=yerelarama1(hes,elor,populasyon,d,tabulisto);
[populasyon,tabulistopt]=yerelaramaopt3i(elor,say,a,c,e,populasyon,d,tabulistopt);
%MUTASYON
[populasyon]=mutasyoninver(elor,mor,populasyon);
[maliyet]=maliyetfonksiyonu(d,populasyon);
 [maliyet,indexmal]=sort(maliyet);
populasyon=populasyon(indexmal,:);
[itereniyi(ik),I]=min(maliyet);
itereniyipop(ik,:)=populasyon(I,:);
disp(ik)
disp(min(maliyet))
disp(ate)
disp(hes)
end
graf(ate,:)=itereniyi;
[best,besti]=min(itereniyi);
besto=itereniyipop(besti,:);
bah(:,ate)=[best,besti,besto]';
end
toc
etim = cputime-ctime
X=mean(graf,1);
plot(X,'LineWidth',3)
```

Ek 2: GA-YTA-3-opt Problem Tanımlama

```
function[hes,say,a,c,e,elor,cor,mor,d,populasyon,maliyet,maksiter]=tspbilgi
tmp=load('ch150.mat','d');
d=tmp.('d');
cor=0.6;
mor=0.3;
npopulasyon =size(d,1)*4
elor=round(npopulasyon/20);
dpopulasyon = size(d,1);
maksiter=200;
populasyon=zeros(npopulasyon,dpopulasyon);
for i=1:npopulasyon
populasyon(i,:)=randperm(size(d,1));
end
n=size(populasyon,1);
maliyet=zeros(n,1);
nn=size(populasyon,2);
populasyonmal=horzcat(populasyon,populasyon(:,1));
for k=1:n
    for i=1:nn
    maliyet(k)= maliyet(k) + d(populasyonmal(k,i),populasyonmal(k,i+1));
    end
   end
say=0;
for ai=1:nn-4
    for ci=3:nn-2
        for ei=5:nn
            if ai+2<=ci && ci+2<=ei
                say=say+1;
                a(say)=ai;
                c(say)=ci;
                e(say)=ei;
            end
        end
    end
end
hes=npopulasyon;
end
```

Ek 3: GA-YTA-3-opt Çaprazlama Fonksiyonu

```
function [hes,populasyon]=caprazlama3(hes,elor,cor,populasyon,maliyet)
nn=size(populasyon,2);
n=size(populasyon,1);
for i=1:n-elor
```

```
rassal=rand;
    if rassal <= cor
        [secilen]=rulet2(maliyet);
        ortak=0;
        if nnz(populasyon(secilen(1,1),:)-populasyon(secilen(1,2),:))==0
        secilenek=randperm(size(populasyon,2));
        ortak=1;
        end
        aralik=randi(nn-1);
        baslangic=randi(nn-aralik);
        sabitler=ones(1,nn).*(-1); sabitler(1,baslangic+1:baslangic+aralik)=populasyon(secile
       n(1,1),baslangic+1:baslangic+aralik);
        if ortak==0
        pop2=populasyon(secilen(1,2),:);
        elseif ortak==1
        pop2=secilenek;
        end pop3=[pop2(1,baslangic+1+aralik:end),pop2(1,1:baslangic+aralik)];
        kalan2=setdiff(pop3,sabitler,'stable');
        kalan=[kalan2,zeros(1,aralik)];
        kalan=circshift(kalan,[0,baslangic+aralik]);
        for ii=1:nn
            if sabitler(1,ii)==-1
                sabitler(1,ii)=kalan(1,ii);
            end
        end
        populasyon(i+elor,:)=sabitler(1,:);
        hes=hes+1;
    end
end
end
```

Ek 4: GA-YTA-3-opt Yerel Arama YTA Fonksiyonu

```
function [hes,populasyon,tabulisto]=yerelarama1(hes,elor,populasyon,d,tabulisto)
rassay=rand;
if rassay<1
deg=0;
ij=0;
mdeg=elor;
popy=zeros(1,1);
for i=1:size(populasyon,1)
    if deg==mdeg
        break
    end
    if ismember(populasyon(i,:),tabulisto,'rows').'==0
```

```
            ij=ij+1;
            popy(ij,1)=i;
            tabulisto=[tabulisto;populasyon(i,:)];
            deg=deg+1;
        else
            deg=deg+1;
        end
end
mdeg=ij;
if mdeg > 0
cont=0;
gbestoy=zeros(mdeg,size(d,1));
for ix=1:mdeg
gbestoy(ix,:)=populasyon(popy(ix,1),:);
end
gbestol=zeros(mdeg,size(d,1)+1);
for ix=1:mdeg
gbestol(ix,:)=horzcat(gbestoy(ix,:),gbestoy(ix,1));
end
gbesty=zeros(mdeg,1);
for ix=1:mdeg
      s=0;
for j=1:size(d,1)
s=s+d(gbestol(ix,j),gbestol(ix,j+1));
end
gbesty(ix,1)=s;
end
n=size(populasyon,2);
sizeii=(3*n*(n-1)/2);
hes=hes+sizeii*mdeg;
for ses=1:mdeg
gloaday=zeros(sizeii,n);
ii=0;
for jj=1:n
      for kk=1:n
            if jj<kk
                  ii=ii+1;
                  gloada=gbestoy(ses,:);
                  jk=[jj kk];
                  kj=fliplr(jk);
                  gloada(1,[jk])=gbestoy(ses,[kj]);
                  gloaday(ii,:)=gloada;
            end
      end
end
for jj=1:n
```

```
    for kk=1:n
        if jj<kk
            ii=ii+1;
            gloada=gbestoy(ses,:);
            jk=[jj:kk];
            kj=fliplr(jk);
            gloada(1,[jk])=gbestoy(ses,[kj]);
            gloaday(ii,:)=gloada;
        end
    end
end
for jj=1:n
    for kk=1:n
        if jj<kk
            ii=ii+1;
            gloada=gbestoy(ses,:);
            ins=gloada(jj);
            gloada(jj)=[];
            idx = kk+1;
            gloada= [gloada(1:length(gloada) < idx), ins, gloada(1:length(gloada) >= idx)];
            gloaday(ii,:)=gloada;
        end
    end
end
gloadaymal=horzcat(gloaday,gloaday(:,1));
fi=zeros(ii,1);
for i=1:ii
    s=0;
    for j=1:n
        s=s+d(gloadaymal(i,j),gloadaymal(i,j+1));
    end
    fi(i,1)=s;
end
fmin=min(fi);
if fmin<gbesty(ses)
    [~,gbesti]=min(fi);
    gbestoy(ses,:)=gloaday(gbesti,:);
    populasyon(popy(ses,1),:)=gbestoy(ses,:);
    cont=cont+1;
end
end
if cont==0
    tabulisto=[tabulisto;gbestoy];
end
end
end
end
```

Ek 5: GA-YTA-3-opt Yerel Arama 3opt Fonksiyonu

```
function [populasyon,tabulistopt]=yerelaramaopt3i(elor,say,a,c,e,populasyon,d,tabulistopt)
rassay=rand;
if rassay<1
deg=0;
ij=0;
mdeg=elor;
popy=zeros(1,1);
for i=1:elor
    if deg==mdeg
        break
    elseif ismember(populasyon(i,:),tabulistopt,'rows').'==0
        ij=ij+1;
        popy(ij,1)=i;
        tabulistopt=[tabulistopt;populasyon(i,:)];
        deg=deg+1;
    end
end
mdeg=ij;
if mdeg >0
gbestoy=zeros(mdeg,size(d,1));
for ix=1:mdeg
gbestoy(ix,:)=populasyon(popy(ix,1),:);
end
gbestol=zeros(mdeg,size(d,1)+1);
for ix=1:mdeg
gbestol(ix,:)=horzcat(gbestoy(ix,:),gbestoy(ix,1));
end
gbesty=zeros(mdeg,1);
for ix=1:mdeg
    s=0;
for j=1:size(d,1)
s=s+d(gbestol(ix,j),gbestol(ix,j+1));
end
gbesty(ix,1)=s;
end
boyut=size(populasyon,2);
for ses=1:mdeg
cont=0;
sayi=0;
for i =1:say
    if e(i)<boyut
eskiler=[d(gbestoy(1,a(i)),gbestoy(1,a(i)+1)),d(gbestoy(1,c(i)),gbestoy(1,c(i)+1)),d(gbestoy(1
,e(i)),gbestoy(1,e(i)+1))];
```

```
yeniler=[d(gbestoy(1,a(i)),gbestoy(1,c(i))),d(gbestoy(1,a(i)),gbestoy(1,c(i)+1)),d(gbestoy(1,a(
i)),gbestoy(1,e(i))),d(gbestoy(1,a(i)+1),gbestoy(1,c(i)+1)),d(gbestoy(1,a(i)+1),gbestoy(1,e(i)+
1)),d(gbestoy(1,a(i)+1),gbestoy(1,e(i))),d(gbestoy(1,c(i)),gbestoy(1,e(i)+1)),d(gbestoy(1,c(i)),
gbestoy(1,e(i))),d(gbestoy(1,c(i)+1),gbestoy(1,a(i)+1)),d(gbestoy(1,c(i)+1),gbestoy(1,e(i)+1))
,d(gbestoy(1,e(i)),gbestoy(1,a(i)+1)),d(gbestoy(1,e(i)),gbestoy(1,c(i)))];
    else
eskiler=[d(gbestoy(1,a(i)),gbestoy(1,a(i)+1)),d(gbestoy(1,c(i)),gbestoy(1,c(i)+1)),d(gbestoy(1
,e(i)),gbestoy(1,1))]; yeniler=[d(gbestoy(1,a(i)),gbestoy(1,c(i))),d(gbestoy(1,a(i)),gbestoy(1,c(
i)+1)),d(gbestoy(1,a(i)),gbestoy(1,e(i))),d(gbestoy(1,a(i)+1),gbestoy(1,c(i)+1)),d(gbestoy(1,a(
i)+1),gbestoy(1,1)),d(gbestoy(1,a(i)+1),gbestoy(1,e(i))),d(gbestoy(1,c(i)),gbestoy(1,1)),d(gbes
toy(1,c(i)),gbestoy(1,e(i))),d(gbestoy(1,c(i)+1),gbestoy(1,a(i)+1)),d(gbestoy(1,c(i)+1),gbestoy
(1,1)),d(gbestoy(1,e(i)),gbestoy(1,a(i)+1)),d(gbestoy(1,e(i)),gbestoy(1,c(i)))];
    end
    orteski=mean(eskiler);
    ortyeni=mean(yeniler);
    if orteski>ortyeni
        sayi=sayi+1;
        ai(sayi)=a(i);
        ci(sayi)=c(i);
        ei(sayi)=e(i);
    end
end
for i=1:sayi
if cont==1
        break
else
p1=gbestoy(1,1:a(i));
p2=gbestoy(1,a(i)+1:c(i));
p3=gbestoy(1,c(i)+1:e(i));
p4=gbestoy(1,e(i)+1:end);
s1=[p1,fliplr(p2),p3,p4];
s2=[p1,p2,fliplr(p3),p4];
s3=[p1,fliplr(p3),fliplr(p2),p4];
s4=[p1,p3,fliplr(p2),p4];
s5=[p1,fliplr(p3),p2,p4];
s6=[p1,fliplr(p2),fliplr(p3),p4];
s7=[p1,p3,p2,p4];
gloaday=[s1;s2;s3;s4;s5;s6;s7];
gloadaymal=horzcat(gloaday,gloaday(:,1));
fi=zeros(size(gloadaymal,1),1);
for ix=1:size(fi,1)
    cost=0;
    for j=1:boyut
        cost=cost+d(gloadaymal(ix,j),gloadaymal(ix,j+1));
    end
```

```
    fi(ix,1)=cost;
end
fmin=min(fi);
if fmin<gbesty(ses)
    [~,gbesti]=min(fi);
    gbestoy(ses,:)=gloaday(gbesti,:);
    populasyon(popy(ses,1),:)=gbestoy(ses,:);
    cont=cont+1;
    gbesty(ses)=fmin;
end
end
end
end
end
end
end
```

Ek 6: GA-YTA-3-opt Mutasyon Fonksiyonu

```
function [populasyon]=mutasyoninver(elor,mor,populasyon)
boyut=size(populasyon,2);
n=size(populasyon,1);
for iko=1:n-elor
    rassal=rand;
    if rassal<= mor
        r1=randperm(boyut-3,1);
        r2=randi([r1+2,boyut-1],1);
        rr=randperm(2,1);
        if rr==1;
            insloc=r1;
            idx=r2;
        else
            insloc=r2;
            idx=r1;
        end
        dbi=populasyon(iko+elor,:);
        ins1=dbi(insloc);
        dbi(insloc)=[];
        dbia= [dbi(1:length(dbi)<idx),ins1,dbi(1:length(dbi) >= idx)];
        populasyon(iko+elor,:)=dbia;
    end
end
end
```

Ek 7: GA-YTA-3-opt Uyum Fonksiyonu

```
function [maliyet]=maliyetfonksiyonu(d,populasyon)
n=size(populasyon,1);
maliyet=zeros(n,1);
nn=size(populasyon,2); populasyonmal=horzcat(populasyon,populasyon(:,1));
for k=1:n
    for i=1:nn
    maliyet(k)= maliyet(k) + d(populasyonmal(k,i),populasyonmal(k,i+1));
    end
end
end
```

Ek 8: GA-YTA-3-opt Rulet Seçim Tekerleği Fonksiyonu

```
function [secilen]=rulet2(maliyet)
n=size(maliyet,1);
etoplam=sum(1./maliyet(1:n));
eolasilik=(1./maliyet(1:n))./(etoplam);
ekumolasilik=0;
ras=rand;
for k=1:n
    ekumolasilik=ekumolasilik+eolasilik(k);
    if ras<=ekumolasilik
        secilen1=k;
        break
    end
end
ras=rand;
ekumolasilik=0;
for k=1:n
    ekumolasilik=ekumolasilik+eolasilik(k);
    if ras<=ekumolasilik
        secilen2=k;
        break
    end
end
if secilen1==secilen2 && secilen1==1
    secilen2=secilen2+1;
elseif secilen1==secilen2
    secilen2=secilen2-1;
end
secilen=[secilen1 secilen2];
end
```

EXTENDED SUMMARY

Purpose

The main purpose of the study is to evaluate the performances of different local search methods in GA, and as a result of these evaluations, to propose a new local search method and a strengthened GA method that uses different methods one after the other. For this purpose, the proposed approach and other known approaches were compared over the test problems in the literature. Within the scope of the study, the aim of the proposed algorithm was to maintain the diversity in the solutions while trying to achieve good results, and thus not to be stuck with local optima. For this reason, the search width was determined in accordance with both purposes in the local search approaches in the study and it was applied to a certain number of solutions sequentially in each iteration. The study provided a new perspective on the GSP solution with the originality of the structure of the proposed algorithm.

Methodology

Based on the concept of evolution, Holland's (1975) GA is adapted for the improvement of the appropriate solutions in the optimization problems of the generational improvement process over the course of iterations. In the GA approach, from the beginning, two individuals must be selected from the population in order to create the individuals that will form the next iteration. The approaches frequently used in GA for the selection process are; turn-based selection, tournament selection and roulette wheel selection. The way in which two selected individuals create the new individual is called the crossover operation in GA. New individuals obtained by crossover are mutated to ensure diversity in the GA approach with low probability. The GA can be terminated according to a predetermined number of iterations or the rate of change of the best fit value obtained in the algorithm, and results are obtained for the problem. Within the scope of the study, the problem is addressed with Genetic Algorithm hybridized with different types of local search approaches. In the Genetic Algorithm with the sequential local search approach proposed in the study, 3-opt local search approach was used after applying combination of swapping, inversion and insertion, which are neighborhood generating functions in local search.

Results and Conclusion

The performances of YD, TD and AS-based GA-YTA, 3-opt-based GA-3-opt and the proposed mixed approach GA-YTA-3-opt were compared over the GSP examples. When the results are evaluated, it is seen that the proposed algorithm GA-YTA-3-opt has higher performance than other approaches in general. While the proposed GA-YTA-3-opt method provides diversity in the search regions with local search YTA, it also makes detailed searches in certain regions with 3-opt. It does not include the problem of getting stuck in the local optimum and can produce effective results for the GSP, with the sequential local search structure and in some cases the output from the first of the local searches can be input to the second local search. On the other hand, it has been observed that the GA-YTA-3-opt method can compete with the algorithms in the literature with the comparison made on the test samples. In this respect, it has been shown that the proposed approach is successful by producing acceptable results for GSP samples. For future studies, it is predicted that the performance and efficiency of hybridized GA will increase by using local search approaches or improvement methods to be produced specific to GSP.