GEZGİN SATICI PROBLEMİNİN GENETİK ALGORİTMALARLA ÇÖZÜMÜNDE BAŞLANGIÇ POPÜLASYONUN BELİRLENMESİ[1]

Araş.Gör.Meryem PULAT
Dokuz Eylül Üniversitesi, İktisadi ve İdari Bilimler Fakültesi, Ekonometri Bölümü.
meryem.pulat@deu.edu.tr

Prof.Dr.İpek DEVECİ KOCAKOÇ
Dokuz Eylül Üniversitesi, İktisadi ve İdari Bilimler Fakültesi, Ekonometri Bölümü.
ipek.deveci@deu.edu.tr

ÖZET: Gezgin satıcı probleminde aralarındaki uzaklıkları bilinen şehirlerin her birinden yalnız bir kez geçen en az maliyetli turu bulmayı hedeflemektedir. Gezgin satıcı problemi tanımlanması kolay olmasına rağmen optimal çözümü elde etmek çok zordur ve NP-zor problemdir. Bu problemin temel zorluğu olası tur sayısının şehir sayısı arttıkça büyük oranda artmasıdır bu da problemi kesin yöntemlerle çözümünü imkânsız hale getirmektedir bu yüzden problemi çözebilen farklı yöntemler öne sürülmüştür. Bu yöntemlerden biri de genetik algoritmalardır. Genetik algoritmalar özellikle geleneksel optimizasyon yöntemlerinin daha az etkin olduğu zor optimizasyon problemlerini çözmek için uygundur. Genetik Algoritmanın çözüm performansını önemli ölçüde etkileyen başlangıç popülasyonun nasıl oluşturulacağı ve popülasyon büyüklüğünün belirlenmesidir. Başlangıç popülasyonu çoğunlukla rasgele seçilir ama genetik algoritmaların performansını geliştirmek için farklı sezgisellerde kullanılmaktadır. Çalışmada başlangıç popülasyonu en yakın komşuluk sezgiseli ve rasgele bir şekilde oluşturularak farklı popülasyon büyüklükleri de dikkate alınarak karşılaştırılmıştır.

Anahtar Kelimeler: Gezgin Satıcı Problemi, Genetik Algoritmalar, En Yakın Komşuluk, Başlangıç Popülasyonu.

DETERMINING THE INITIAL POPULATION OF SOLVING THE TRAVELING SALESMAN PROBLEM WITH GENETIC ALGORITHMS

ABSTRACT: The traveler is aiming to find the least costly tour in the traveling salesman problem, which is only one time out of each of the known cities. Although it is easy to identify the traveling salesman problem, obtaining the optimal solution is very difficult and NP-hard problem. The basic difficulty of this problem is that the number of possible tours increases in large numbers as the number of cities increases, which makes the problem impossible to solve with definite methods, so different methods have been proposed to solve the problem. One of these methods is genetic algorithms. Genetic algorithms are particularly suited to solve difficult optimization problems where traditional optimization methods are less effective. It is determine how to create and the size of the initial population that significantly affect the performance of the Genetic Algorithm solution. The initial population is often randomly selected but is used in different heuristics to improve the performance of genetic algorithms. In the study, the initial population was created with the nearest neighbour intentionally and randomly, and the different population sizes were considered and compared.

Key Words: Traveling Salesman Problem, Genetic Algorithms, Nearest Neighbour, Initial Population.

[1] Bu makale 20-21 Mayıs 2017 tarihinde İstanbul'da düzenlenen International Congress of Management Economy and Policy isimli kongrede bildiri olarak sunulmuştur.

GİRİŞ

Gezgin satıcı probleminde aralarındaki uzaklıkları bilinen şehirlerin her birinden yalnız bir kez geçen en kısa yolu veya en az maliyetli turu bulmayı hedeflemektedir. Gezgin satıcı problemi tanımlanması kolay olmasına rağmen optimal çözümü elde etmek çok zordur ve literatürde NP-zor (Non polynomial-hard) problemler arasında yer almaktadır. Bu problemin temel zorluğu olası tur sayısının şehir sayısı arttıkça büyük oranda artmasıdır bu da problemi kesin yöntemlerle kısa sürede çözümünü imkânsız hale getirmektedir bu yüzden problemi kısa sürede çözebilen farklı yöntemler öne sürülmüştür. Bu yöntemlerden biri de genetik algoritmalardır.

Genetik algoritmalar rastlantısal arama tekniklerini kullanarak çözüm bulmaya çalışan parametre kodlama esasına dayanan sezgisel bir arama tekniğidir. Bu yöntem optimal çözümü garanti edemez ancak kısa sürede makul bir çözüm elde edebilmektedir. Genetik algoritmalar özellikle geleneksel optimizasyon yöntemlerinin daha az etkin olduğu zor optimizasyon problemlerini çözmek için uygundur. Genetik Algoritmanın çözüm performansını önemli ölçüde etkileyen başlangıç popülasyonun nasıl oluşturulacağı ve popülasyon büyüklüğünün belirlenmesidir. Başlangıç popülasyonu çoğunlukla rasgele seçilir ama genetik algoritmaların performansını geliştirmek için farklı sezgisellerde kullanılmaktadır.

Çalışmanın amacı, gezgin satıcı problemini çözmek için kullanılan genetik algoritmaların performansını önemli ölçüde etkileyen başlangıç popülasyonunu en yakın komşuluk sezgiseli ve rasgele bir şekilde oluşturularak farklı popülasyon büyüklükleri de dikkate alınarak karşılaştırılmıştır. Başlangıç popülasyonunun nasıl oluşturulacağı ve hangi popülasyon büyüklüğünün daha iyi sonuç verdiği belirlenmiştir. Ayrıca en uygun nesil büyüklüğünün ne olması gerektiği ve literatürde var olan üç operatörde dikkate alınarak hangi tür çaprazlamanın daha iyi sonuç verdiği belirlenmiştir.

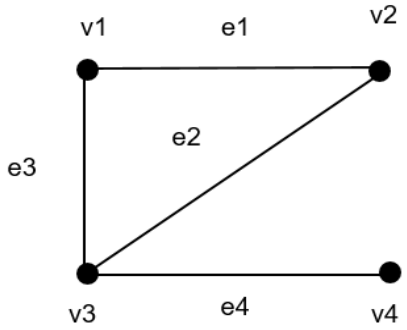
Çalışmada, birinci bölümde; gezgin satıcı problemi tanımı, özellikleri, çözüm yöntemleri açıklanmıştır. İkinci bölümde; genetik algoritma özellikleri, parametreleri ve operatörleri açıklanmıştır. Üçüncü bölümde; gezgin satıcı probleminin genetik algoritmalarla çözümü ele alınmıştır. Son bölümde de TSPLIB'den alınan üç farklı boyuttaki problem kullanılarak MATLAB programında çözüm yapılmıştır.

1. GEZGİN SATICI PROBLEMİ

Gezgin satıcı problemi (TSP-Traveling Salesman Problem) aralarındaki uzaklıkları bilinen n adet noktanın (şehir veya düğüm) her birinden yalnız bir kez geçen en kısa yolu veya en az maliyetli turu bulmayı hedeflemektedir (Cevre, Özkan, & Uğur, 2007:1).

Grafik teorisi ile gezgin satıcı problemi şu şekilde ifade edebilir. Bir grafik $G(V,E)$ V köşe ve E kenar sayısının kümesini içerir. Bir kenar $e \in E$ iki köşe $v1, v2 \in V$ birbirine bağlar ve $e(v1,v2)$ olarak gösterilebilir. $e(v1,v2)$ kenarın yön bilgisi yoksa yönsüz bir grafiktir. $e1 = (v1,v2) \neq (v2,v1) = e2$ ise G yönlü bir grafiktir. $e(v1,v2)$ ve $e'(v1,v2)$ aynı köşeyi paylaşıyorsa yani $v1$ ve $v2$ köşelerini bağlayan iki kenar varsa e ve e' nün paralel olduğunu

söylenebilmektedir. Paralel kenarı olmayan grafikler basit grafikler olarak adlandırılmaktadır. Hamilton yolu her bir noktayı tam olarak bir kez ziyaret eden yönsüz grafikte basit bir yoldur. Hamilton döngüsü ise; her köşeyi tam olarak bir kez ziyaret eden ve başlangıç köşesine dönen yönsüz bir grafikte basit bir döngüdür. Köşeler şehirleri kenarlar şehirler arasındaki mesafeyi temsil ederse; TSP yönsüz basit bir grafikte minimum mesafe ile bir Hamilton döngüsü bulma olarak yorumlanmaktadır (Yu ve Gen, 2010: 276).



Şekil 1: Yönsüz basit grafik Yu & Gen, 2010: 176

Gezgin satıcı problemi ilk olarak 1930'lu yılların başında Karl Menger tarafından matematiksel olarak tanımlanmış kombinasyonel optimizasyon (combinatorial optimizaton) problemleri arasında yer almaktadır. Tanımlanması kolay olmasına rağmen çözümü zordur ve NP-zor (Nonpolinomially-hard) problemler arasında yer almaktadır (Çolak, 2010: 424).

Gezgin satıcı problemi araştırmacı ve akademisyenler tarafından üzerinde en çok çalışılan kombinasyonel optimizasyon problemleri arasında yer almaktadır (Çolak, 2010: 424). TSP'nin araştırmacının ilgisini çekmesi ve aktif bir araştırma alanı olmaya devam etmesinin başlıca dört sebebi vardır (Allaoua & Brahim, 2015: 110; Gopal, vd., 2015:151):

I. Gezgin satıcı probleminin tanımlanması kolay olmasına rağmen çözümü zordur NP-zor (NP-hard) bir problemdir.

II. NP-zor problemler problem boyutu büyük olduğunda problemi kısa sürede kesin olarak çözmek için etkin bir yol bulunamamıştır.

III. Çok sayıda gerçek dünya problemi TSP ile modellenebilir.

IV. Farklı sonuçları karşılaştırarak çözüm kalitesini ölçmeyi sağlayan TSPLIB kütüphanesi vardır.

TSP 'nin Tam Sayılı Doğrusal Programlama Formülasyonu aşağıdaki gibidir (Laporte, 1992: 233):

Minimize: $\sum_{i\neq j}^{n} c_{ij}\ x_{ij}$.(1)

Kısıtlar:

$\sum_{j=1}^{n} x_{ij} = 1$ i=1,2,…,n (2)

$\sum_{i=1}^{n} x_{ij} = 1$ j=1,2,…,n (3)

$\sum_{i,j\in S}^{n} x_{ij} \leq |S| - 1$

$S \subset V\ , 2 \leq |S| \leq n-2$ (4)

$x_{ij} \in \{0,1\}$

$i, j = 1, \ldots, n$ i≠j (5)

Yukarıdaki eşitlikte x_{ij} karar değişkenleri, c_{ij} şehir i den şehir j ye seyahat maliyeti ve n şehir sayısını göstermektedir. (1) numaralı eşitlik amaç fonksiyonunu verir. Amaç fonksiyonu optimal turun maliyetini tanımlamaktadır. (2) numaralı kısıt her köşeye bir kez girileceğini; (3) numaralı kısıt her köşeden bir kez ayrılacağını (4) numaralı kısıt alt tur eleme kısıtıdır diğer bir deyişle oluşabilecek alt turlardan kurtulmaya yöneliktir. Uygun çözümleri tek turla sınırlandırmaktadır. (5) numaralı kısıt ikili (binary) koşuludur.

Gezgin satıcı probleminde maliyet matrisinin yapısı dikkate alındığında TSP simetrik gezgin satıcı problemi (STSP) ve asimetrik gezgin satıcı problemi (ATSP) olarak iki gruba ayrılmaktadır. STSP de tüm i,j için $c_{ij} = c_{ji}$ olur. Eğer $c_{ij} \neq c_{ji}$ olduğunda ATSP dir. Asimetrik Gezgin Satıcı Probleminde şehir i den şehir j ye gitmenin maliyeti farklıdır.

Gezgin satıcı probleminin temel zorluğu olası tur sayısının büyüklüğüdür. N şehirli STSP $(n-1)!/2$ olası çözüm varken ; ATSP için $(n-1)!$ olası çözüm vardır (Ahmed, 2010: 96).

Tablo 1: STSP ve ATSP için Olası Tur Sayısı

Şehir (Düğüm, Nokta) sayısı	ATSP için Olası tur sayısı $(\boldsymbol{n-1})!$	STSP için Olası tur sayısı $(\boldsymbol{n-1})!/\boldsymbol{2}$
13	479001600	239500800
15	87178291200	43589145600
22	$5{,}11\times10^{19}$	$2{,}55\times10^{19}$
51	$3{,}04\times10^{64}$	$1{,}52\times10^{64}$
76	$2{,}48\times10^{109}$	$1{,}24\times10^{109}$
101	$9{,}33\times10^{157}$	$4{,}66\times10^{157}$

Tablo 1'de görüldüğü gibi şehir sayısı çok küçük olduğunda bile olası tur sayısı çok büyüktür ve şehir sayısı arttıkça olası tur sayı da üstel olarak artmaktadır.

1.1. Gezgin Satıcı Probleminin Çözüm Yöntemleri

Gezgin satıcı problemi kombinasyonel optimizasyon problemidir ve problemin NP-zor olduğu bilinmektedir. Problem boyutu arttıkça olası tur sayısı arttığından kesin olarak kısa sürede çözmek zor hatta imkânsız hale gelmektedir. TSP çözmek için kesin (exact) ve sezgisel (heuristic) algoritmalar geliştirilmiştir (Potvin, 1996: 339).

➢ *Kesin (Exact) Yöntemler:* Problemin kesin optimum çözümünü veren yöntemlere kesin (exact) yöntemler denir. Bu yöntemler TSP'nin doğrusal programlama formülasyonundan türetilmiştir. TSP çözmenin kesin yolunu basitçe şu şekilde ifade edilebilmektedir: öncelikle mümkün çözümlerin hepsi listelenir amaç fonksiyonu değerine göre değerlendirir ve en iyisini seçilir. Ancak problem boyutu çok küçük olduğunda bile olası tur sayısı çok sayıda olmaktadır. Aynı zamanda çok bellek alanı ve çok yüksek hesaplama zamanı gerekmektedir. Bu yüzden bu yöntemler orta büyüklükteki problemlerde bile etkin değildir. Küçük boyutlu problemlerin çözümünde kullanılmaktadır (Ahmed, 2010: 97). Kesin yöntemlere örnek olarak şunlar verilebilir:

- Dinamik Programlama
- Dal ve Sınır (Branch and Bound)
- Dal Kesme (Branch-Cut)

➢ *Sezgisel Yöntemler:* Kesin (Exact) yöntemlerin etkinliği küçük boyutlu problemler ile sınırlıdır. Aynı zamanda TSP NP-zor bir problem olduğundan aramayı yönlendirip tüm arama uzayında aramadan kaçınmak için sezgisel yaklaşımlara ihtiyaç vardır (Ghoseiri & Sarhadi, 2007: 903). Sezgisel yöntemler kısa sürede çözüm bulabilmesine rağmen optimal çözümü garanti edememektedir.

Sezgisel algoritmalar genel olarak turu oluşturan sezgiseller (tour construction), turu geliştiren (tour improvement) sezgiseller ve melez yöntemler olmak üzere üçe ayrılmaktadır.

- *Tur Oluşturan Sezgiseller:* Tüm turu oluşturan algoritmalarda bir çözüm bulduğunda durmaktadır ve hiçbir zaman onu iyileştirmemektedir (Matai, vd., 2010: 13). Bu yöntemlere örnek olarak; en yakın komşuluk, açgözlü (greedy) sezgiseli, ekleme sezgiseli verilebilir.
- *Tur Geliştiren Sezgiseller:* Herhangi bir tur yapı sezgiseli kullanarak turu oluşturduktan sonra oluşturulan turun kalitesinin daha da arttırmak için iyileştirme sezgiseli uygulanabilir (Matai vd., 2010: 14). Tur geliştiren sezgiseller 2-opt,3-opt gibi yerel optimizasyon algoritmalarının yanında; GA, Tavlama Benzetimi, Karınca Kolonisi, Tabu Arama gibi yöntemlerdir.
- *Melez Yöntemler:* Tur oluşturma ve tur geliştirme tekniklerine dayalıdır. Çeşitli yöntemlerin kombinasyonu ile yapılabilmektedir.

2. GENETİK ALGORİTMALAR

Genetik algoritmalar (GA), genetik ve evrim ilkelerine dayanan olasılıksal, sağlam (robust) ve sezgisel arama algoritmasıdır (Javidi & Fard, 2015: 27). Goldberg (1989)'in tanımına göre; genetik algoritmalar rastlantısal arama tekniğini kullanarak çözüm bulmaya çalışan parametre kodlama esasına dayanan sezgisel bir arama tekniğidir (Elmas, 2016: 379). Evrime dayalı algoritmaların bir türüdür. Bir arama algoritmasında probleme ait olası çözümlerin bir dizisi mevcuttur ve belirli bir zamanda en iyi olası çözüm bulunur. Evrimsel arama algoritması bazı geleneksel algoritmalardan farkı popülasyona dayalı olmasıdır. Evrimsel çok sayıda bireyin ardışık nesillere adaptasyonu, evrimsel algoritmada direk aramadan daha etkili çalışır.

GA, John Holland tarafından 1960 yılında ortaya atılmış ve Holland öğrencileri ve arkadaşları tarafından 1960 ve 1970'lerde Michigan üniversitesinde geliştirilmiştir (Melanie, 1999: 2). 1975 yılında Holland'ın "Adaptation in natural and artificial system" isimli kitabında doğal evrim prensibinin optimizasyon problemlerine nasıl uygulanacağını açıklanmıştır (Sivanandam & Deepa, 2008: 15).

2.1. Genetik Algoritmaların Optimizasyondaki Yeri

Birçok gerçek hayat problemi sürekli doğrusal olmayan optimizasyon problemi olarak modellenebilir ve global optimum çözümü aranır. Doğrusal olmayan optimizasyon problemlerini çözmek için var olan optimizasyon teknikleri 2 grupta sınıflandırılabilir (Deep & Thakur, 2007: 211):

- Deterministik Optimizasyon Teknikleri
- Stokastik Optimizasyon Teknikleri

Deterministik Optimizasyon Teknikleri: Tek çözüm ile çalışır. Arama işlemi tahmini (arama uzayında rasgele seçilir) bir çözüm ile başlar ve eğer bu tahmini çözüm global minimuma yeterince yakın seçilmezse lokal minimum çözüme yakalanabilir (Deep & Thakur, 2007: 212).

Stokastik Optimizasyon Teknikleri: Tavlama Benzetimi, evrimsel algoritmalar gibi tekniklere dayanır. Bunların arasında evrimsel algoritmalar yeteneklidir.

Stokastik optimizasyon yöntemlerinden biri olan GA geleneksel yöntemlerle çözümü zor ya da hemen hemen imkânsız olan problemlerin çözümüne kullanılan sayısal optimizasyon yöntemlerinden biridir. Karmaşık, süreksiz, gürültü (noisy) içeren, kısıt sayısı fazla, amaç fonksiyonu kurulamayan, kesin çözüm yöntemi olmayan problemlerin çözümünde kullanılabilmektedir (Paksoy & Uzun, 2015: 347).

Genetik algoritmanın geleneksel optimizasyon ve arama tekniklerinden farkı şu şekildedir (Sivanandam & Deepa, 2008: 34):

1. Parametrelerin kendisi ile değil parametrelerin kodlanmış şekliyle işlem yapmaktadır.

2. Çoğu optimizasyon tekniği tek noktadan arama yaparken GA noktaların popülasyonu üzerine işlem yapar. Arama yaparken tek noktadan ziyade çözümlerin popülasyonu kullanmak optimuma ulaşma şansını artırır ve lokalden kaçmayı sağlar.

3. GA değerlendirmede amaç fonksiyonu bilgisini kullanır bunun dışında yardımcı bilgi gerektirmez.

4. Olasılıksal geçiş kurallarını kullanır.

Genetik Algoritmanın Avantajları (Sivanandam & Deepa, 2008: 34):

1.Çözüm uzayı daha geniştir.
2.Global optimumu keşfetmek daha kolaydır.
3.Fonksiyon evrimlerini kullanır.
4.Farklı problemler için kolaylıkla değiştirilebilir.
5.Büyük anlaşılması zor arama uzayını kolaylıkla anlamaktadır.
6.Uygunluk yüzeyi (Fitness Landscape) karmaşıktır.
7.Yanıt yüzey (Response surface) hakkında herhangi bir bilgi gerektirmez.
8.Lokal optimuma düşme konusunda dirençlidir.
9. Büyük boyutlu problemlere uygulanabilir.

2.2. Genetik Algoritmaların Tasarımı

Basit GA yapısı evrimsel programlama yapısına benzerdir. t iterasyonu boyunca potansiyel çözümlerin popülasyonunu sürdürmektedir. Çözüm uygunluk değerine göre değerlendirilir. Yeni popülasyon t+1 iterasyonda daha uygun bireylerin seçimi ile oluşmaktadır. Bu yeni popülasyondaki bireyler çaprazlama ve mutasyon ile değişime uğrar yeni çözümler oluşmaktadır (Michalewicz, 1992: 17).

Genetik algoritma da izlenecek adımlar şu şekildedir (Elmas, 2016: 399):

Adım1: Toplumda bulunan birey sayısını belirlemekle başlanır. Birey sayısı ile ilgili kesin bir sayı yoktur. Probleme bağlı olarak değişmektedir gözlem ve inceleme ile belirlenir.

Adım2: Kromozom (diziler/bireyler) ne kadar iyi olduğuna uygunluk değerinin hesaplanmasıyla karar verilir.

Adım3: Yeni popülasyon daha uygun bireylerin seçimi ile oluşur. Bu seçim için orantılı rulet tekerleği, sıraya dayalı rulet tekerleği ve turnuva seçimi gibi farklı seçim yöntemleri kullanılmaktadır.

Adım4: Bu yeni popülasyondaki bireyler çaprazlama ve mutasyon ile değişime uğrar ve yeni çözümler oluşur.

***Adım5*:** GA belirli bir nesil sayısı veya durdurma kriteri sağlanana kadar defalarca çalıştırılır. GA işleyişi sonucunda en iyi kromozom (dizi, birey) çözüm olarak alınır.

2.3. Genetik Algoritmanın Parametreleri

Popülasyon büyüklüğü, çaprazlama oranı ve mutasyon oranı genetik algoritmanın parametreleridir. GA parametreleri GA'nın performansı üzerinde çok önemli bir etkiye sahiptir. GA performansı bu parametrelere bağlıdır. Bu yüzden bu parametrelerin seçimi dikkatle yapılmalıdır. Bu parametrelerin seçimine karar verirken bu değerler hakkında bir kesinlik mevcut değildir probleme bağlı olarak değişmektedir. Bu yüzden deneme yoluyla amaca uygun olarak belirlenmektedir.

➢ *Popülasyon Büyüklüğü:*En önemli kararlardan biri kullanıcı tarafından belirlenen popülasyon büyüklüğüdür. Probleme bağlı olarak değişmektedir. Popülasyon büyüklüğü GA 'nın genel performansını ve GA'nın etkinliğini etkilemektedir. Algoritmanın performansı iki şekilde etkilenmektedir (Grefenstette, 1986: 124):

I. Çok küçük popülasyon çoğu hiperdüzlem için yetersiz örneklem büyüklüğü sağlar buda aramanın belirli bir alt optimal noktaya sürükleyecektir.
II. Çok büyük popülasyon çok sayıda hiperdüzlemin temsilini içermektedir. Aynı zamanda alt optimal çözüme yakınsamayı da engellemektedir. Ancak büyük popülasyon nesil gelişimi için daha fazla süreye ihtiyaç duymaktadır buda hesaplama zamanını arttırmaktadır bu da istenmeyen bir durumdur.

Bu yüzden popülasyon büyüklüğü (P_s) seçimi dikkatle yapılmalıdır.

➢ *Çaprazlama Oranı:* Çaprazlamanın amacı mevcut iyi kromozomları birleştirerek daha iyi kromozom oluşturmaktır. Temel parametrelerden biri olan çaprazlama olasılığı (P_c). ne sıklıkla çaprazlamanın yapılacağını tanımlayan bir parametredir (Javidi & diğerleri, 2015: 31). Bu oran büyükse; popülasyon değişkenliği hızlı bir şekilde gerçekleştirir. Bu oran düşükse; aramanın çok yavaş gerçekleşmesine sebep olmaktadır (Elmas, 2016: 394).

➢ *Mutasyon Oranı:* Mutasyon yapılmasının nedeni; birbirini izleyen daha uygun bireylerin atılmasından gelmektedir. Kromozomlarda rasgele değişiklikler yapılarak GAs arama uzayında yeni kısımlara ulaşılmasını sağlar. Aynı zamanda önemli özelliklerin erken kaybolmamasının ve böylece eşleşme havuzunda çeşitliliğin korunmasın sağlamaktadır (Ahmed, 2010: 101). Mutasyon oranı (P_m) ne sıklıkla mutasyonun yapılacağını tanımlayan bir parametredir. Eğer bu oran çok yüksek ise GA rasgele aramaya dönüşmektedir; çok düşük ise yerel optimum tuzağına düşmesine sebep olmaktadır (Pham ,vd., 1997: 157).

3. GEZGİN SATICI PROBLEMİNİN GENETİK ALGORİTMALARLA ÇÖZÜMÜ

Genetik algoritmaların, kombinasyonel optimizasyon problemlerine uygulamaları ile ilgili çalışmalar mevcuttur. En yoğun yapılan çalışmalardan biri de gezgin satıcı problemleridir. Gezgin satıcı probleminde amaç, kat edilen toplam mesafeyi minimize eden bir yolculuk planı oluşturmaktır. Birçok problem tipi gezgin satıcı problemi gibi modellenebilmektedir. Aynı zamanda gezgin satıcı probleminde değişken sayısı arttıkça çözüm zamanı üstel olarak artar.

Bu da problemin klasik yöntemlerle kısa sürede çözümünü zor hatta imkânsız hale getirmektedir. GA kombinasyonel optimizasyon problemlerini klasik yöntemlere göre daha kısa sürede çözmektedir ve optimale yakın kabul edilebilir bir çözüm elde edebilmektedir (Emel & Taşkın, 2002: 145).

3.1. Kodlama

Birey genlerinin temsil (gösterim) sürecidir. Bireyin nasıl kodlanacağı ilgilenilen probleme göre değişmektedir. Gezgin satıcı problemini genetik algoritmalar kullanarak çözmek için pek çok farklı gösterim (temsil) kullanılmaktadır. Bunlar ikili(binary) gösterim, yol(path) gösterimi, komşuluk(adjacency) gösterimi, sıralı(ordinal) gösterim ve matris(matrix) gösterimi.

- *Yol (Path) Gösterimi:* Gezgin Satıcı Problemi bir sıralama problemidir. TSP için çözüm problemdeki düğümlerin sayısı uzunluğunda kromozomlar ile temsil edilir. Kromozomdaki her geni hiçbir düğüm iki kez görünmeyecek şeklinde düğüm etiketi alır. TSP de tur gösteriminde yol gösterimi kullanılmaktadır (Gopal, vd., 2015:153). Yol gösteriminde; ziyaret edilmesi gereken n şehir n elemanlı listeye göre sıraya konur. İ'inci şehir listenin j'inci elemanı ise şehir i ziyaret edilecek j'inci şehirdir. Genetik algoritmalar ile TSP çözümü bu gösterim kullanılarak yapılmaktadır (Larranaga ,vd., 1999: 134)

Yedi şehirli bir örneği (olay/durum) ele alırsak düğümlerin etiketleri $\{1,2,3,4,5,6,7\}$ şeklindedir. Tur ise; $\{2 \rightarrow 1 \rightarrow 5 \rightarrow 3 \rightarrow 4 \rightarrow 7 \rightarrow 6 \rightarrow 1\}$ olarak ya da $(2\ 1\ 5\ 3\ 4\ 7\ 6\ 1)$ olarak gösterilebilir.

Diğer gösterim türlerinin tercih edilmemesinin nedenleri: ikili gösterimde çok küçük problem durumunda kullanılabilir ancak problem büyüdükçe her şehir $[\log_2 n]$ bit dizisi olarak kodlandığından gösterim turları yönetemeyecek büyüklükte olmaktadır ve klasik operatörler geçerli çocuk turuyla sonuçlanamaz tamir algoritması kullanmayı gerektirir. Komşuluk gösteriminde bu gösterime özgü çok az sayıda çaprazlama operatörü geliştirilmiş. Ancak bütün geliştirilen çaprazlama operatörleri düşük performans göstermiştir. Oluşturulan çocuk ebeveynden yeterli bilgiyi alamamaktadır o yüzden tercih edilmemektedir. Sıralı gösterimde klasik operatörler kullanılabilmektedir tamir algoritması gerektirmeden ancak kötü sonuçlar verdiği için tercih edilmemektedir. Matris gösteriminde ise temel zorluk geçerli çocuğa yol açan operatörleri tanımlamaktadır. Çocuğun geçerli bir tur olduğunu garanti etmek için tamir algoritması gereklidir. TSP'nin GA çözümünde gösterim olarak yol gösterimi kullanılmaktadır ve bu gösterime ilişkin çok sayıda operatör geliştirilmiştir. (Larranaga, vd., 1999: 166).

3.2. Başlangıç Popülasyonunun Oluşturulması

Gen havuzu olarak adlandırılan dizilerin başlangıç popülasyonu oluşturulur ve ardışık nesillere daha iyi popülasyon oluşturmak için seçim, çaprazlama ve mutasyon operatörü uygulanır (Gopal, vd., 2015: 152).

Başlangıç popülasyonu çoğunlukla rasgele seçilir ama genetik algoritmaların performansını geliştirmek için farklı sezgisellerde kullanılmaktadır. Çalışmada başlangıç popülasyonu rasgele ve en yakın komşuluk sezgiseli kullanılarak oluşturulup karşılaştırılmıştır. İlk popülasyon tüm arama alanını keşfedebilmek için mümkün olduğunca geniş bir gen havuzuna sahip olmalıdır. Gen havuzu yeterince büyük olmadığı takdirde popülasyon çeşitlilikten yoksun olmaktadır, algoritma arama alanının küçük bir kısmını araştırmaktadır ve asla global optimal çözümü bulamayabilir. Bu yüzden büyük popülasyon çok yararlıdır. Ama daha çok hesaplama maliyeti bellek ve zaman gerektirir. (Sivanandam & Deepa, 2008: 42)

- *En Yakın Komşuluk Sezgiseli:* Mevcut şehre en yakın şehri sırasıyla düşünerek TSP için bir çözüm oluşturmaktadır. Bu yöntemde izlenecek adımlar (Yu & Gen, 2010: 278):

  Adım 1: Herhangi bir şehirden başla ve onu mevcut şehir olarak kabul et.
  Adım 2: Mevcut şehre en yakın ziyaret edilmeyen şehri seç ve onu mevcut şehir yap.
  Adım 3: Tüm şehirler ziyaret edilene kadar adım 2'yi tekrarla ve sonra başlangıç şehrine dön.

3.3. Uygunluk Fonksiyonu

Genetik biliminde bireyin çevre şartına ne kadar dayanıklı olduğunu belirten uygunluk değeri GA ile çözümün kalitesini belirler ve uygunluk fonksiyonu bulunur. Kromozomların çözümünde gösterdikleri başarı derecesini belirleyen bir değerlendirme işlevidir. Hangi kromozomların (diziler) bir sonraki nesle taşınacağı ve hangi kromozomların yok olacağı uygunluk değerinin büyüklüğüne göre karar verilmektedir (Elmas, 2016: 389). Bireyin uygunluğu onun fenotipi için amaç fonksiyonu değeridir. Eğer x ile fenotip değer temsil ediliyorsa; f(x) ile uygunluk fonksiyonu gösterilebilir. Başlangıç popülasyonu oluşturulduktan sonra yapılacak ilk işlem bu popülasyonda yer alan kromozomların uygunluk değerlerinin hesaplanmasıdır (Çolak, 2010: 427). Çözülen her problem için uygunluk fonksiyonu geliştirilmelidir.

Kromozomun uygunluk değeri genetik algoritmalar yaklaşımına göre maksimize edilmelidir. TSP de amaç en düşük maliyetle turu bulmaktır bu en düşük maliyetli tur en kaliteli çözümü temsil etmektedir. Bu yüzden dönüşümle uygunluk fonksiyonu maksimize edilmelidir. Dönüşüm şu şekilde gerçekleştirilebilir (Bhattacharyya & Bandyopadhyay, 2008: 7):

$$f(x) = \frac{1}{d}$$

d: Kromozom ile temsil edilen turun maliyeti (uzaklık/uzunluk)

Eğer şehirler 2D koordinat sisteminde x ve y koordinatları ile temsil edilirse o zaman onların arasındaki mesafe şöyle hesaplanabilir (Agarwal & Kanchan, 2013:36):

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

3.4. Seçim

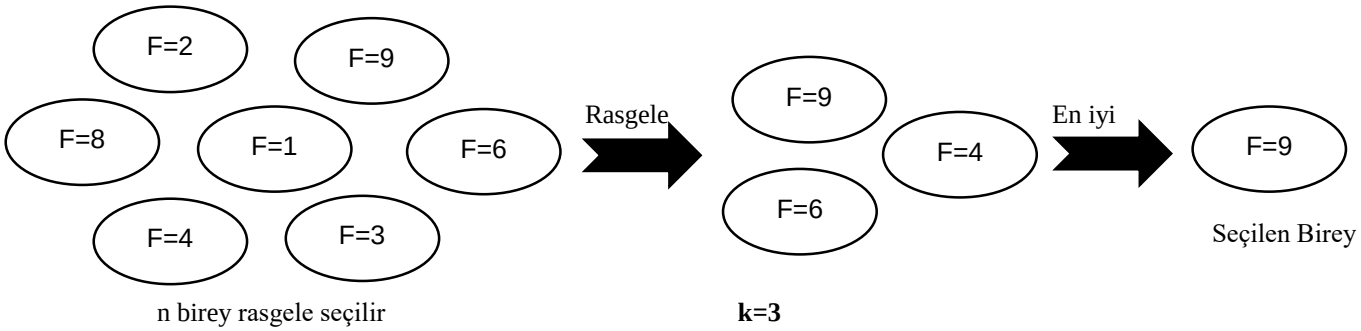
Seçme işlemi tabiatta çevre tarafından, yapay sistemlerde ise amaç fonksiyonu ve diğer kalite değerlendirme işlemleri tarafından kontrol edilir. Darwin 'in teorisine göre; en iyi olanlar çocuğu oluşturmak için hayatta kalır. Bireylerin bir jenerasyondan diğer kuşağa geçerken kalite değerlerine göre daha fazla kopya edilme şansları vardır. Böylece, daha kaliteli çözümler popülasyonda barınırken, kalitesiz olanların yavaş yavaş kaybolması sağlanmaktadır (Karaboğa, 2014: 81). Hangi dizilerin bir sonraki nesle aktarılacağı farklı seçim mekanizmaları ile sağlanır. Bunlara örnek olarak *orantılı rulet tekerleği*, *sıraya dayalı rulet tekerleği* ve *turnuva seçimi* verilebilir. Çalışmada *turnuva seçim yöntemi* kullanılmıştır.

- *Turnuva Seçimi:* Popülasyon içerisinden rasgele k (3,5,7) adet birey (kromozom) alınır ve daha sonra bu kromozomlar kıyaslanarak aralarında uygunluk değeri yüksek olan kromozom popülasyona aktarılmak üzere seçilir (Cevre & diğerleri., 2007: 3).

Turnuva seçimi kolaylık ve uygunluğu açısından en yaygı olandır. Turnuva seçiminin aşağıdaki özellikleri onu kullanışlı hale getirir özellikle de çok amaçlı optimizasyon durumunda (Yu & Gen, 2010: 74):

- Turnuva seçimi sadece yerel bilgiyi kullanır.
- Turnuva seçiminin uygulaması kolaydır ve zaman karmaşıklığı azdır.
- Turnuva seçimi paralel olarak kolaylıkla uygulanabilir.

Sıralı ölçekleme ile kıyaslandığında sıralı ölçekleme tüm popülasyonu sıralamayı gerektirir bu da zaman alıcıdır aynı zamanda hesaplama açısından da turnuva seçimi etkilidir (Melanie, 1998: 127). Paralel olarak uygulama açısından bakıldığında ise; diğer yöntemleri paralelleştirmek daha zordur. Çünkü; bir miktar paralel bilgiye ihtiyaç duyarlar. Orantılı seçim, fonksiyon değerlerinin toplamın gerektirirken; sıralı seçim, global sıralamayı elde edebilmek için tüm bireylere ve bunların fonksiyon değerlerine erişmeyi gerektirir (Goldberg & Deb, 1991: 82).



Şekil 2: Turnuva Seçimi Razali & Geraghty, 2011: 3.

Şekil 2'de gösterilen şekilde popülasyon içerisinden 3 adet birey rasgele seçilmiştir yani; turnuva boyutu üç alınmıştır. Üç birey birbiriyle yarışarak en yüksek uygunluğa sahip birey

seçilmektedir. Rasgele seçilen 3 birey arasından en yüksek uygunluk değeri olarak 9 olduğu görülmektedir. Bu değere sahip olan birey seçilmiştir.

3.5. Çaprazlama

İki ebeveyn çözümü alarak onlardan çocuk üretme sürecidir. Seçim işleminden sonra çaprazlama ile popülasyon daha iyi bireyler ile zenginleştirilmiştir. Farklı çözümler arasında bilgi alışverişi sağlar (Michalewicz, 1992: 17). Çaprazlama üç adımda gerçekleştirilen bir rekombinasyon operatörüdür (Sivanandam & Deepa, 2008: 50).

I. Üreme (reproduction) operatörü çiftleştirmek için iki bireyin dizisinden bir kısmını rasgele seçilir.
II. Çaprazlama konumu dizi (kromozom) uzunluğu boyunca rasgele seçilir.
III. Son olarak, pozisyon değerleri çaprazlama takiben iki dizi arasında değiştirilir.

Bir önceki nesilden daha iyi nitelikler içeren yeni kromozomlar yaratmak amacıyla çaprazlama operatörleri kullanılmaktadır. GA Holland tarafından ortaya atıldıktan sonra probleme bağlı olarak birçok çaprazlama operatörü araştırmacılar tarafından ortaya atılmıştır. Çünkü GA performansı etkileyen en önemli operatörlerden biridir (Esmkhan & Zamanifar, 2012:1). Çaprazlama operatörlerine örnek olarak tek noktalı çaprazlama, iki noktalı çaprazlama, çok noktalı çaprazlama, pozisyona dayalı çaprazlama, sıraya dayalı çaprazlama örnek olarak verilebilir. Bunlardan temel olan iki tanesi açıklanmıştır.

Gezgin satıcı problemine klasik GA kullanılan çaprazlama operatörleri uyguladığında geçerli olmayan turlar oluşturduğu görülmektedir.

Bu problem için özel operatörler geliştirilmeli ve kullanılmalıdır. Çok sayıda çaprazlama operatörü araştırmacılar tarafından ortaya atılmıştır. PMX (Partially mapped crossover), CX (Cycle crossover), OX (Order crossover), ERX (Edge recombination crossover), EERX (Enhanced edge recombinaton crossover), PMX (Partially mapped crossover), OX2 (Order based crossover), POS (Position based crossover), MPX (Maximal preservative crossover), VR (Voting recombination crossover), GX (Greedy crossover), HX (Heuristic crossover), SCX (Sequential constructive crossover) TSP için geliştirilmiş operatörlerinden bir kısmıdır tamamı (Pulat & Deveci Kocakoç, 2016: 10-17) bu çalışmada ele alınmıştır. Uygulamada kullanılan CX (Dairesel çaprazlama-Cycle crossover), OX (Sıralı çaprazlama-order crossover) ve EERX (Gelişmiş kenar birleştirme çaprazlama-Enhanced edge recombination crossover) çaprazlama operatörleri aşağıda açıklanmıştır:

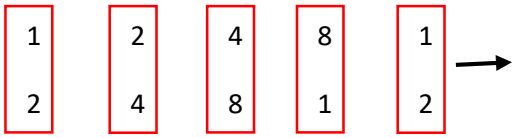
3.5.1 CX (Dairesel Çaprazlama-Cycle Crossver):

Oliver ve arkadaşları (1987) tarafından önerilmiştir. Şehirlerin alt kümelerine odaklanır. Her iki ailenin pozisyonunda aynı alt kümede yer alıyorsa, bu şehirler ilk aileden çocuğa aynı pozisyonda kopyalanır, kalan pozisyonlar ikinci ebeveyndeki şehirler ile doldurulmaktadır (Potvin, 1996: 352).

Ebeveyn A : 1 2 3 4 5 6 7 8

Ebeveyn B : 2 4 6 8 7 5 3 1

Alt Kümeler:



Tekrardan 1-2 geldi alt kümeler bunlar alt kümeleri aynı pozisyona kopyalanır kalan pozisyonları diğer ebeveynden alınmaktadır.

Çaprazlama Sonrası:

Çocuk A: 1 2 6 4 7 5 3 8

Çocuk B: 2 4 3 8 5 6 7 1

Bu şekilde her bir şehrin konumu iki ebeveynden birinden gelmektedir. Her iki ebeveynde de elemanların ortalama yarısının pozisyonu korunmaktadır.

3.5.2 OX (Sıralı Çaprazlama-Order Crossver):

Davis (1985) tarafından önerilmiştir. Şehirlerin konumu değil sırası önemlidir. Ebeveynler alt turu seçer ve diğer ailedeki şehirlerin sırasını koruyarak oluşturur. İlk olarak kesim noktaları rasgele seçilir. Burada, ilk kesim noktasının 2. ve 3. bit arasında ve ikincisinin 5. ve 6. bit arasında seçildiğinde;

Ebeveyn A: 1 2 I 3 4 5 I 6 7 8

Ebeveyn B: 2 4 I 6 8 7 I 5 3 1

Kesim noktaları arasındaki kısım çocuğa kopyalanır.

Çocuk A: – – I 3 4 5 I – – –

Çocuk B: – – I 6 8 7 I – – –

Sonra ikinci kesim noktasından başlayarak zaten mevcut olan şehirleri atlayarak diğer ebeveynde göründükleri sırada çocuğa kopyalanır dizin sonuna ulaşıldığında ilk pozisyondan devam edilmektedir.

Çaprazlama Sonrası:

Çocuk A: 7 1 I 3 4 5 I 2 6 8 (2 4 6 8 7 5 3 1)

Çocuk B: 4 5 I 6 8 7 I 1 2 3 (1 2 3 4 5 6 7 8)

3.5.3 EERX (Gelişmiş Kenar Birleştirme Çaprazlama-Enhanced Edge Recombination Crossover):

Starkweather, Mcdaniel, Mathias, ve Darrell (1991) tarafından önerilmiştir.

EERX operatörü şu şekilde çalışır (Starkweather ,vd., 1991: 70):

Adım1: Turdaki her şehir için bir kenar listesi oluşturulur. Bir şehrin komşu şehirleri her iki ebeveynde mevcutsa bu negatif işaret (-) kullanılarak gösterilir. İki ebeveyn turundan başlangıç şehri rasgele seçilir bu mevcut şehir olarak alınır.

Adım2: Mevcut şehir tüm kenar listesinden kaldırılır.

Adım3: Bir sonraki şehri seçme işlemi şu şekilde yapılmaktadır: Bu operatörü ERX' den ayıran kısımdır aynı zamanda mevcut şehrin kenar listesindeki negatif işaretli şehirler ilk önceliğe sahiptir ancak hiç negatif işaretli şehir yoksa kendi kenar listesinde en az bağlantıya sahip şehir ikinci önceliğe sahiptir. Eğer aynı sayıda bağlantılı şehir varsa bu durumda seçim rasgele yapılır. Şehri seç adım 2'ye git.

Adım4: Ziyaret edilmemiş şehir kalmadıysa dur. Aksi takdirde ziyaret edilmemiş bir şehri rasgele seç ve Adım 2'ye git.

Ebeveyn A: (1 2 3 4 5 6)

Ebeveyn B: (2 4 3 1 5 6)

Tablo 2: EERX Turlar için Kenar Haritası

Şehir	Kenar Listesi
1	**2, 6, 3, 5**
2	**1, 3, 4, 6**
3	**2,- 4, 1**
4	**- 3, 5, 2**
5	**4,- 6, 1**
6	**1,- 5, 2**

Başlangıç şehri olarak [2] seç, şehir 2'yi tüm kenar listesinden kaldır. Bir sonraki adım şehir 2'nin kenar listesinde (1,3,4,6) hiç negatif işaretli şehir olmadığından ikinci öncelik olan en az bağlantıya sahip şehir seçilmeliydi burada (3,4,6) her üçüde iki bağlantıya sahip bu durumda seçim rasgele yapılmaktadır. Şehir 6'yı seç [2 6]. Şehir 6'yı tüm kenar listesinden kaldır. Şehir 6'nın kenar listesine bakıldığında (1,-5) var negatif işaretli eleman olduğundan şehir 5 seçilir [2 6 5]. Şehir 5'i tüm kenar listesinden kaldır. Şehir 5'in kenar listesine bakıldığında (1,4) var her ikisi de tek bağlantıya sahip olduğundan seçim rasgele yapılır şehir 1'i seç [2 6 5 1]. Şehir 1'i tüm kenar listesinden kaldır. Şehir 1'in kenar listesine bakıldığında sadece (3) olduğundan

şehir 3 seçilir [2 6 5 1 3]. Şehir 3'ü tüm kenar listesinden kaldır. Şehir 3'ün kenar listesine bakıldığında sadece (4) olduğundan şehir 4 seçilir [2 6 5 1 3 4]. Ziyaret edilmemiş şehir kalmadığından çaprazlama işlemi sona ermiştir. Çaprazlama sonrası tur şu şekildedir:

[2 6 5 1 3 4]

3.6. Mutasyon

Diğer önemli genetik algoritma operatörü mutasyon operatörüdür. Çaprazlamadan sonra diziler mutasyona tabi tutulur. Mutasyon geleneksel basit bir arama operatörü olarak kabul edilmektedir ve popülasyonda genetik çeşitliliği korumak için bir arka plan operatör olarak görülmektedir. Mutasyon algoritmanın yerel minimuma düşmesini engeller ve çaprazlamanın sebep olduğu genetik materyale kaybına karşı korumaktadır böylece popülasyondaki çeşitlilik korunmaktadır (Sivanandam & Deepa, 2008: 57). Aynı zamanda erken yakınsamayı önlemektedir. GA performansını etkileyen operatörlerden biridir. Problem türüne bağlı olarak farklı mutasyon operatörleri ortaya atılmıştır.

Gezgin satıcı problemini çözmek için çok fazla sayıda çaprazlama operatörü geliştirilmesine rağmen mutasyon ile ilgili çok az sayıda operatör önerilmiştir. Bunun nedenleri arasında mutasyon oranının çok düşük alınması çaprazlama oranına göre yani çok küçük değişimler meydana getirmektedir. TSP özgü önerilen mutasyon operatörüne örnek olarak DM (Displacement mutation), EM (Exchange mutation), SIM (Simple inversion mutation), SM (Scrable mutation), IVM (Inversion mutation) verilebilir. Uygulamada kullanılan yer değiştirme mutasyonu (Swap mutation) aşağıda açıklanmıştır.

- *Yer Değiştirme Mutasyonu (Swap Mutation):* İki şehir rasgele bir şekilde seçilir ve karşılıklı yer değiştirilir. Bu mutasyon operatörü orijinal mutasyon felsefesine çok yakındır çünkü turda sadece çok küçük değişiklikler yapmaktadır (Potvin, 1996: 361).

Mutasyon Öncesi: 2 6 5 1 3 4

Mutasyon Sonrası: 2 3 5 1 6 4

4. UYGULAMA

Bu çalışmanın amacı, gezgin satıcı problemini çözmek için kullanılan genetik algoritmaların performansını önemli ölçüde etkileyen başlangıç popülasyonun nasıl oluşturulacağı ve büyüklüğünün belirlenmesidir. Başlangıç popülasyonu çoğunlukla rasgele seçilir ama genetik algoritmaların performansını geliştirmek için farklı sezgisellerde kullanılmaktadır. Çalışmada başlangıç popülasyonu en yakın komşuluk sezgiseli ve rasgele bir şekilde oluşturularak farklı popülasyon büyüklükleri dikkate alınarak karşılaştırılmıştır. Çalışmada kullanılan veriler TSPLIB'den alınmıştır ve Matlab R2015b programında kodlanmıştır.

TSPLIB 1990'da Gernard Reinelt tarafından kurulmuş gezgin satıcı problemi (ve benzer problemler) ait 14 şehirden 85900 şehir boyutuna kadar 100 üzerinde örnek bulunmaktadır. Bu

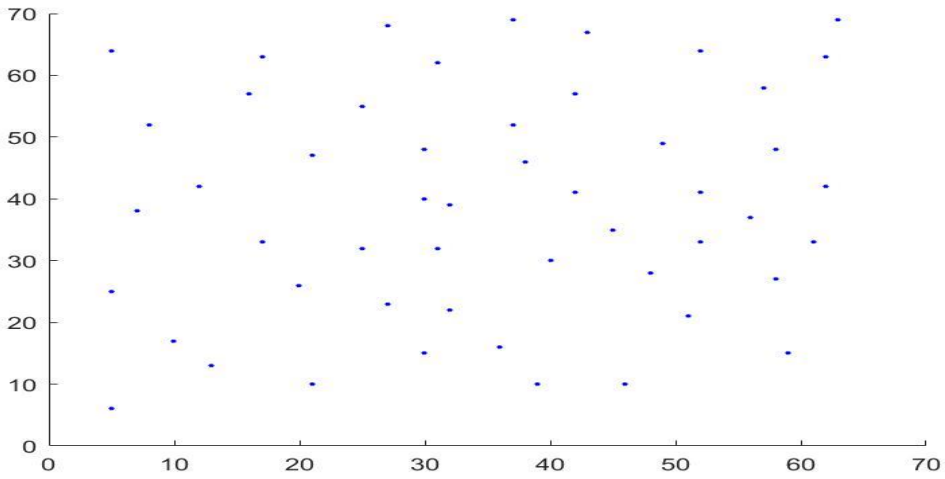
örneklerin optimal çözüm değerleri de yer almaktadır. Optimal çözüm değerlerinin var olması elde edilen değerlerin kıyaslanabilmesini ve optimal değere ne kadar yaklaşıldığının görülmesini sağlamaktadır. TSP ile ilgili yapılan çalışmalarda veri olarak çoğunlukla TSPLIB' den alınan veriler kullanılmaktadır nadiren rasgele bir şekilde şehirleri oluşturup performans analizi yapılmaktadır. Çalışmada farklı boyuttaki problem üzerinde parametrelerin etkisini görebilmek için küçük, orta ve büyük olmak üzere üç farklı boyutta veri seçilmiştir. Çalışmada 'eil51','eil76' ve 'eil101' verileri üzerinde inceleme yapılmıştır. Buradaki sayılar şehir sayısını göstermektedir. Bu veriler için TSPLIB de verilen optimal çözüm değerleri sırasıyla şu şekildedir: 426, 538 ve 629.

İlk olarak başlangıç popülasyonu rasgele bir şekilde oluşturulup; farklı popülasyon büyüklükleri, farklı nesil sayıları ve üç farklı çaprazlama operatörü kullanılarak hesaplama zamanı da dikkate alınarak problem için uygun değerler belirlenmiştir. Çaprazlama operatörü olarak OX, CX ve EEAX kullanılmıştır. Bu üç operatör sırasıyla şehirlerin sırasını, konumunu ve kenarlarını korumayı amaçlamaktadır. Bu üç operatörün seçilmesinin nedeni; hangi korumanın daha iyi sonuç verdiğini görebilmektir. Tüm bireyler çaprazlamaya tabi tutulmuştur. Mutasyon operatörü olarak yer değiştirme mutasyon kullanılmıştır. Mutasyon oranı 0.01 alınmıştır. Bu oranın farklı değerlerinin sonuca etkisine bakılarak uygun mutasyon oranı belirlenmiştir. Seçim yöntemi olarak turnuva seçimi kullanılmış, turnuvaya uğrayacak kromozom sayısı k farklı denemeler sonucunda 5 olarak belirlenmiştir.

İkinci olarak başlangıç popülasyonu en yakın komşuluk sezgiseli kullanılarak oluşturulduktan sonra rasgele bir şekilde oluşturulan başlangıç popülasyonun da olduğu gibi geri kalan işlemler aynı şekilde gerçekleştirilmiştir. Daha sonra her iki başlangıç popülasyonu oluşturma yöntemi kıyaslanarak başlangıç popülasyonun hangi yöntemle oluşturulacağına karar verilmiştir.

4.1. Eil51 için İnceleme

Verilerin ilk hali şu şekildedir:



Şekil 3: Eil51 için veriler

➢ 51 şehirli problemde başlangıç popülasyonu *rasgele* bir şekilde oluşturulup; popülasyon büyüklüğü [20-2000] arasında farklı popülasyon büyüklüğü ile [100-5000] arasında farklı nesil sayısı ve üç farklı çaprazlama operatörü (OX, CX ve EEAX) hesaplama zamanı (saniye) da dikkate alınarak sonuçlar hesaplanmıştır. Eil51 için 15 tekrar sonucu elde edilen değerlerin ortalaması şu şekildedir:

Tablo 3: Eil51 için G, Ps ve Çaprazlama Operatörlerinin Başlangıç Popülasyonunun Rasgele Oluşturulduğundaki Sonuçlar

Nesil sayısı (G)	Popülasyon büyüklüğü (P_s)	CX	Hesaplama Zamanı (saniye)	OX	Hesaplama Zamanı (saniye)	EERX	Hesaplama Zamanı (saniye)
G=100	P_s =20	1.295e+03	0,32	1.1656e+03	0,53	1.28153+03	4,84
	P_s =40	1.2876e+03	0,40	1.0388e+03	0,79	1.1935e+03	8,37
	P_s =100	1.0908e+03	0,90	835.5029	1,40	1.1432e+03	19,75
	P_s =200	849,7556	1,33	743,3046	2,30	933,0166	40,98
	P_s =1000	726,3030	6,38	596,2653	10,13	886,0516	176,83
	P_s =2000	712,0663	13,13	535,1638	18,55	800,4647	382,74
G=500	P_s =20	1.099e+03	0,70	1.0700e+03	1,51	983,2480	22,19
	P_s =40	866,91	1,32	868,90	2,46	903,3075	39,19
	P_s =100	761,9483	3,69	667,3529	5,15	762,7603	95,84
	P_s =200	679,8641	6,91	601,6579	9,12	745,2821	189,42
	P_s =1000	610,4046	41,47	506,5712	43,13	679,7071	961,16
	P_s =2000	596,5823	107,57	**470,5779**	**92,96**	569,0102	2038,48
G=1000	P_s =20	976,8140	1,22	916,2646	2,71	962,4758	43,34
	P_s =40	846,9770	2,29	724,9945	4,72	932,0957	80,80
	P_s =100	689,4872	7,24	689,5191	9,91	754,7727	218,82
	P_s =200	673,0655	13,40	595,9367	17,81	592,0308	430,87
	P_s =1000	542,6829	84,68	501,8290	86,88	548,5948	2059,93
	P_s =2000	561,3299	189,92	494,2154	182,59	522,4539	3412,03
G=5000	P_s =20	734,8760	5,67	669,7487	12,52	662,6057	186,57
	P_s =40	671,4780	10,62	644,8321	21,97	583,0058	317,93
	P_s =100	606,0973	36,34	592,4110	48,08	557,0619	809,96
	P_s =200	555,624	73,57	530,5728	91,03	556,4395	1576,82
	P_s =1000	539,6234	543,99	510,2691	496,26	610,1566	8188,93
	P_s =2000	511,1601	1214,32	477,7440	2357,83	604,5395	16106,87

Tablo 3'e bakıldığında belirli nesil sayısında popülasyon büyüklüğünü arttırdıkça sonuçlar iyileşmektedir. Popülasyon büyüklüğünün 2000 ile sınırlandırılmasının nedeni daha önce değinildiği gibi popülasyon büyüklüğü çok büyük olduğunda hesaplama zamanını artırmaktadır bu da istenmeyen bir durumdur. 2000'den sonra algoritmanın çalışma zamanı büyük oranda arttığından 2000'den sonrası dikkate alınmadı aynı şekilde maksimum nesil sayısı 5000'den

fazla incelenmesinin sebebi nesil sayısı da çalışma zamanı üzerinde olumsuz etkiye sahiptir ve 5000' den sonra daha büyük nesil sayısı sonuç değerlerini değiştirmemiştir.

Belirli bir nesil sayısında popülasyon büyüklüğünü her arttırıldığında sonuç tüm çaprazlama operatörlerinde daha da iyileşmiştir. Aynı zamanda nesil sayısını arttırıldığında da bir önceki nesil değerlerine bakıldığında nesil sayısını arttırılması olumlu etkiye sahiptir. Daha küçük bir değerden başlayıp popülasyon büyüklüğü arttıkça daha iyiye gitmektedir. Ancak hesaplama zamanı da artmaktadır.

G=100 ve P_s=20 bakıldığında değeri 1.295e+03; G=500 ve P_s=20 olduğunda değeri 1.099e+03 yani daha düşük bir değer elde edilmiştir böylece daha iyi bir değerle başlayıp popülasyon büyüklüğünün artırılmasıyla önemli oranda iyileşme göstermektedir. Çaprazlama operatörlerini kıyasladığında en iyi değerler OX ile elde edilmiştir daha sonra CX ve en kötüsü ise EERX. Hesaplama zamanı açısından baktığımızda ise; OX, CX daha büyük hesaplama zamanına sahip ama arlarında çok büyük farklılık bulunmamaktadır. Ayrıca, P_s=2000 olduğu durumda OX; CX den daha kısa sürmektedir. EERX hesaplama zamanı açısından kötü performansa sahiptir.

Eil51 için en iyi popülasyon değeri nesil sayısı ve çaprazlama operatörü şu şekildedir.

G= 500

P_s=2000 ve

OX operatörleri ile en iyi değer 470.5779 olarak elde edilmiştir.

- 51 şehirli problemde başlangıç popülasyonu *en yakın komşuluk sezgiseli* ile oluşturulup; popülasyon büyüklüğü [20-2000] arasında farklı popülasyon büyüklüğü ile [100-5000] arasında farklı nesil sayısı ve üç farklı çaprazlama operatörü (OX, CX ve EEAX) hesaplama zamanı (saniye) da dikkate alınarak sonuçlar hesaplanmıştır. Eil51 için 15 tekrar sonucu elde edilen değerlerin ortalaması şu şekildedir:

Tablo 4: Eil51 için G, Ps ve Çaprazlama Operatörlerinin Başlangıç Popülasyonunun En Yakın Komşuluk Sezgiseli ie Oluşturulduğundaki Sonuçları

Nesil sayısı (G)	Popülasyon büyüklüğü (P_S)	CX	Hesaplama Zamanı (saniye)	OX	Hesaplama Zamanı (saniye)	EERX	Hesaplama Zamanı (saniye)
G=100	P_S =20	562,5358	0,27	544,9796	0,60	565,6101	3,52
	P_S =40	553,1187	0,43	542,9809	0,69	560,5374	6,86
	P_S =100	560,8414	1,14	529,5886	1,18	555,1215	16,60
	P_S =200	546,8842	1,81	525,8089	2,01	545,4876	32,47
	P_S =1000	528,0577	11,82	494,1491	9,06	527,2362	158,41
	P_S =2000	519,4636	32,65	491,6080	19,46	493,2827	328,60
G=500	P_S =20	555,9036	0,90	535,3246	1,59	562,4611	17,38
	P_S =40	544,8516	2,64	503,8325	2,54	559,5374	33,89
	P_S =100	528,5809	6,58	509,1363	5,16	535,9904	81,44
	P_S =200	522,1803	9,88	502,9264	9,31	524,9827	162,59
	P_S =1000	493,9065	59,85	485,3694	45,73	497,3474	757,41
	P_S =2000	499,0217	177,39	479,5519	98,20	489,3468	1509,68
G=1000	P_S =20	540,6834	1,35	535,5380	2,89	538,3267	32,75
	P_S =40	532,9093	2,33	503,7230	4,88	530,9193	62,99
	P_S =100	518,3735	8,45	487,6658	10,22	519,9131	155,29
	P_S =200	511,2961	15,37	485,7942	18,73	514,9502	301,74
	P_S =1000	504,6697	116,46	479,8622	92,67	489,3468	1627,76
	P_S =2000	503,5949	414,42	479,5487	196,62	489,3468	3200,85
G=5000	P_S =20	529,4	9,66	505,3773	13,16	517,0883	173,13
	P_S =40	515,4277	16,01	491,3615	23,09	495,8959	340,46
	P_S =100	489,3468	43,57	488,9915	49,01	503,5949	816,07
	P_S =200	477,3942	83,93	475,9989	92,40	494,0631	1506,71
	P_S =1000	492,5889	508,81	475,9449	472,48	489,3408	9547,76
	P_S =2000	480,8778	1819,87	**476,3189**	**976,13**	482,6375	16268,11

Tablo 4'e bakıldığında belirli nesil sayısında popülasyon büyüklüğünü arttırdıkça sonuçlar iyileşmektedir. Belirli bir nesil sayısında popülasyon büyüklüğünü her arttırıldığında sonuç tüm çaprazlama operatörlerinde daha da iyileşmiştir. Aynı zamanda nesil sayısını arttırıldığında da bir önceki nesil değerlerine bakıldığında nesil sayısını arttırılması olumlu etkiye sahiptir. Daha küçük bir değerden başlayıp popülasyon büyüklüğü arttıkça daha iyiye gitmektedir. Ancak hesaplama zamanı da artmaktadır.

G=100 ve P_s=20 bakıldığında değeri 544,9796; G=500 ve P_s=20 olduğunda değeri 535,32461 yani daha düşük bir değer elde edilmiştir böylece daha iyi bir değerle başlayıp popülasyon büyüklüğünün artırılmasıyla önemli oranda iyileşme göstermektedir. Çaprazlama operatörlerini kıyasladığında en iyi değerler OX ile elde edilmiştir daha sonra CX ve en kötüsü ise EERX. Hesaplama zamanı açısından baktığımızda ise; CX, OX daha büyük hesaplama zamanına sahip ama aralarında çok büyük farklılık bulunmamaktadır. EERX hesaplama zamanı açısından kötü performansa sahiptir.

Eil51 için en iyi popülasyon değeri nesil sayısı ve çaprazlama operatörü şu şekildedir.

G= 5000

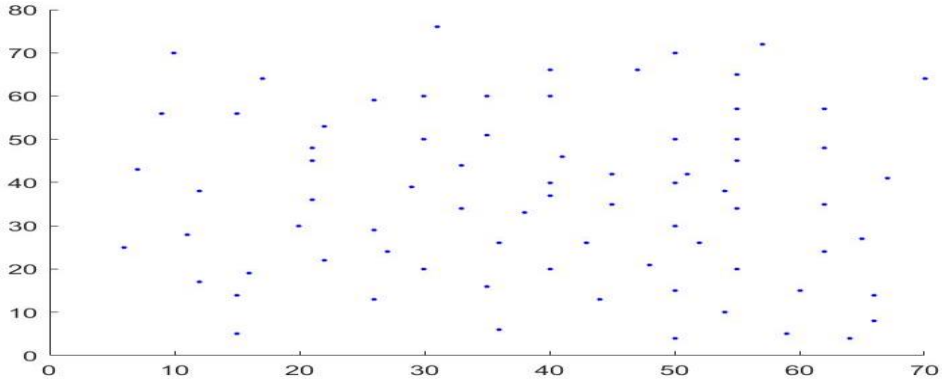
P_s=2000 ve

OX operatörleri ile en iyi değer 476,3189 olarak elde edilmiştir.

Tablo 3 ve Tablo 4 incelendiğinde; başlangıç popülasyonu rasgele oluşturulduğu durumda popülasyonun büyüklüğünü arttırmak çözüm sonucunda büyük bir iyileşme sağlarken; başlangıç popülasyonun en yakın komşuluk sezgiseli ile oluşturulduğu durumda çok büyük iyileşme sağlanmamaktadır. Fakat popülasyon büyüklüğününün arttırılması daha da iyileştirmiştir. Bir diğer durum başlangıç popülasyonun en yakın komşuluk sezgiseli ile oluşturulduğunda çözüm daha iyi bir değerden başlamıştır (tablo 3 de G=100 ve P_s=20 bakıldığında değeri 1.295e+03; tablo4 de G=100 ve P_s=20 olduğunda değeri 544,9796 veya tablo 3 de G=1000 ve P_s=40 bakıldığında değeri 724,9945; tablo4 de G=1000 ve P_s=40 olduğunda değeri 503,7230) yani daha düşük bir değer elde edilmiştir. Çoğunlukla en yakın komşuluk sezgiseli ile oluşturulduğunda çözüm değerleri daha iyidir. Aynı zamanda çaprazlama operatörleri arasında çözüm sonuçları açısından büyük farklılıklar bulunmamaktadır sonuçlar birbirine çok yakındır. Ancak en iyi çözüm değeri başlangıç popülasyonunun rasgele oluşturulduğunda elde edilmiştir.

4.2. Eil76 için İnceleme

Verilerin ilk hali şu şekildedir:



Şekil 3: Eil76 için veriler

➢ 76 şehirli problemde başlangıç popülasyonu *rasgele* bir şekilde oluşturulup; popülasyon büyüklüğü [20-2000] arasında farklı popülasyon büyüklüğü ile [100-5000] arasında farklı nesil sayısı ve üç farklı çaprazlama operatörü (OX, CX ve EEAX) hesaplama zamanı (saniye) da dikkate alınarak sonuçlar hesaplanmıştır. Eil76 için 15 tekrar sonucu elde edilen değerlerin ortalaması şu şekildedir:

Tablo 5: Eil76 için G, Ps ve Çaprazlama Operatörlerinin Başlangıç Popülasyonunun Rasgele Oluşturulduğundaki Sonuçları

Nesil sayısı (G)	Popülasyon büyüklüğü (P_s)	CX	Hesaplama Zamanı (saniye)	OX	Hesaplama Zamanı (saniye)	EEAX	Hesaplama Zamanı (saniye)
G=100	P_s =20	1.9789e+03	0,37	2.0092e+03	0,63	2.0954+03	7,60
	P_s =40	1.9242e+03	0,52	1.6187e+03	0,94	1.18439e+03	12,78
	P_s =100	1.6307e+03	1,13	1.3083e+03	1,68	1.7505e+03	29,05
	P_s =200	1.6287e+0.3	1,95	1.2715e+03	2,80	1.6139e+03	59,20
	P_s =1000	1.2027e+03	8,01	962,9679	21,51	1.4006e+03	287,73
	P_s =2000	1.1865e+03	19,80	926.2154	29,67	1.2974+03	553,38
G=500	P_s =20	1.6272e+03	1,00	1.6857e+03	2,30	1.8924e+03	23,98
	P_s =40	1.4087e+03	1,67	1.3318e+03	3,38	1.5190e+03	46,20
	P_s =100	1.2476e+03	5,19	1.1996e+03	7,17	1.3662e+03	113,54
	P_s =200	1.1073+03	10,25	1.0705e+03	10,43	1.2034e+03	229,18
	P_s =1000	931,5209	51,48	744,6683	52,18	926,7359	1177,89
	P_s =2000	792,56668	143,71	724,6413	117,21	888,6731	2469,11
G=1000	P_s =20	1,5909e+03	1,87	1.4780e+0.3	3,44	1.4754e+03	54,40
	P_s =40	1.2166e+03	3,14	1.2303e+03	6,12	1.2805e+03	102,90
	P_s =100	1.1889e+03	11,17	940,5132	12,83	1,1481e+03	249,18
	P_s =200	934,3077	20,81	965,6038	24,74	976,9619	521,90
	P_s =1000	867,1732	125,45	747,2906	126,71	804,0556	2308,62
	P_s =2000	797,9674	407,24	702,16669	250,44	760,5126	4845,81
G=5000	P_s =20	1.1291e+03	8,35	1.1269e+03	16,08	1.0907e+03	251,98
	P_s =40	978,7012	16,29	891,5406	30,95	978,2829	496,59
	P_s =100	883,4997	56,10	841,3365	68,48	951,0521	1258,60
	P_s =200	812,6075	114,22	800,9275	128,01	887,4929	2453,33
	P_s =1000	733,3307	675,91	697,5895	592,36	781,0212	12071,29
	P_s =2000	714,2675	2301,14	**581,0146**	**1189,92**	751,0521	23164,77

Popülasyon büyüklüğünün artırılması önemli oranda iyileşme göstermektedir. Çaprazlama operatörlerini kıyasladığında en iyi değerler OX ile elde edilmiştir daha sonra CX ve en kötüsü ise EERX. Hesaplama zamanı açısından baktığımızda ise; OX, CX daha büyük hesaplama zamanına sahip ama aralarında çok büyük farklılık bulunmamaktadır. Ayrıca, P_s=2000 olduğu durumda OX; CX den daha kısa sürmektedir. EERX hesaplama zamanı açısından kötü performansa sahiptir.

Eil76 için en iyi popülasyon değeri nesil sayısı ve çaprazlama operatörü şu şekildedir.

G= 5000, P_s=2000 ve OX operatörleri ile en iyi değer 581,0146 olarak elde edilmiştir.

➢ 76 şehirli problemde başlangıç popülasyonu *en yakın komşuluk sezgiseli* ile oluşturulup; popülasyon büyüklüğü [20-2000] arasında farklı popülasyon büyüklüğü ile [100-5000] arasında farklı nesil sayısı ve üç farklı çaprazlama operatörü (OX, CX ve EEAX) hesaplama zamanı (saniye) da dikkate alınarak sonuçlar hesaplanmıştır. Eil76 için 15 tekrar sonucu elde edilen değerlerin ortalaması şu şekildedir:

Tablo 6: Eil76 için G, Ps ve Çaprazlama Operatörlerinin Başlangıç Popülasyonunun En Yakın Komşuluk Sezgiseli ie Oluşturulduğundaki Sonuçları

Nesil sayısı (G)	Popülasyon büyüklüğü (P_S)	CX	Hesaplama Zamanı (saniye)	OX	Hesaplama Zamanı (saniye)	EERX	Hesaplama Zamanı (saniye)
G=100	P_S =20	617,8481	0,38	617,7675	0,48	617,8481	6,60
	P_S =40	617,0582	0,42	617,4292	0,67	617,8481	9,51
	P_S =100	616,7762	1,24	617,3553	1,27	616,7799	25,07
	P_S =200	616,7675	2,16	616,6283	2,14	615,4580	51,85
	P_S =1000	612,5134	13,75	614,3889	10,26	614,4344	229,46
	P_S =2000	612,1196	51,86	613,1807	21,18	613,0376	491,01
G=500	P_S =20	616,8965	0,81	617,4160	1,61	617,8481	25,22
	P_S =40	616,4344	1,69	617,3387	2,79	617,7786	47,95
	P_S =100	615,8325	5,29	616,0938	5,70	616,7799	130,57
	P_S =200	615,6834	10,78	615,2329	10,17	616,7324	238,57
	P_S =1000	611,0137	77,90	611,7639	49,54	611,2302	1130,95
	P_S =2000	611,2716	218,90	610,0906	105,40	610,4202	2589,88
G=1000	P_S =20	615,6691	1,52	617,2677	3,06	617,8481	48,13
	P_S =40	615,1819	2,66	615,7241	5,01	616,7324	103,94
	P_S =100	614,4787	10,49	614,5905	10,36	614,8938	257,75
	P_S =200	612,0170	20,32	613,2939	19,45	611,2522	522,68
	P_S =1000	611,9774	149,18	609,7198	93,66	610,4202	2493,72
	P_S =2000	610,8171	410,65	606,6711	209,42	613,0156	5010,23
G=5000	P_S =20	614,4332	6,40	614,4074	14,31	614,6004	255,15
	P_S =40	611,6495	12,29	611,5987	24,65	612,8253	509,22
	P_S =100	609,3498	45,28	609,2089	51,54	610,4202	1185,41
	P_S =200	608,1707	85,89	608,1350	97,40	610,4202	2309,29
	P_S =1000	607,7856	678,33	607,0358	497,40	610,4202	12374,75

	P_s =2000	606,6711	2259,79	**606,6011**	**958,24**	613,0156	24633,75

Tablo 6'ya bakıldığında Tablo 5'deki gibi yorumlanabilmektedir.

Eil76 için en iyi popülasyon değeri nesil sayısı ve çaprazlama operatörü şu şekildedir.

G= 5000

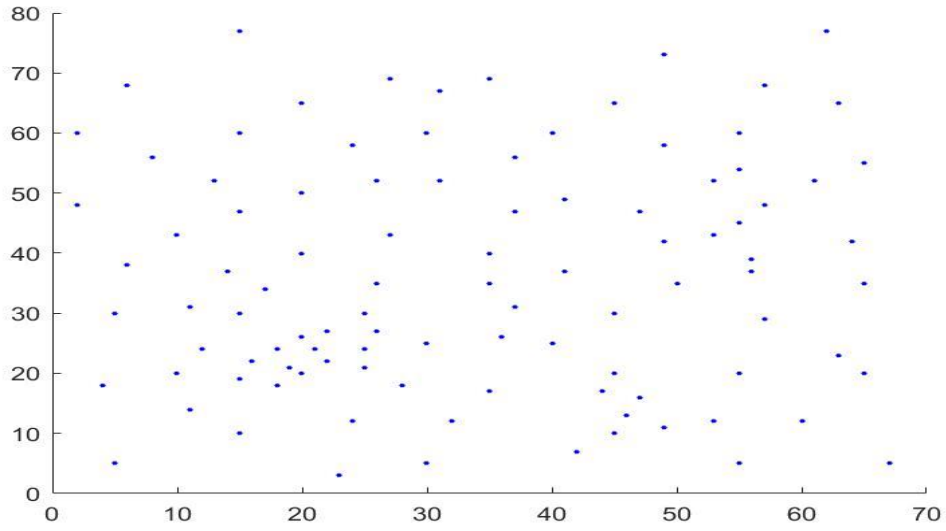
P_s=2000 ve

OX operatörleri ile en iyi değer 606,6011olarak elde edilmiştir.

Tablo 5 ve Tablo 6 incelendiğinde Eil51 de olduğu gibi; başlangıç popülasyonun en yakın komşuluk sezgiseli ile oluşturulduğu durumda çok büyük iyileşme sağlanmamaktadır. Bir diğer durum başlangıç popülasyonun en yakın komşuluk sezgiseli ile oluşturulduğunda çözüm daha iyi bir değerden başlamıştır (tablo 5'te G=100 ve P_s=20 bakıldığında değeri 2.0092e+03; tablo6 da G=100 ve P_s=20 olduğunda değeri 617,7675veya tablo 5 'te G=1000 ve P_s=40 bakıldığında değeri 1.2303e+03; tablo6'da G=1000 ve P_s=40 olduğunda değeri 615,7241) yani daha düşük bir değer elde edilmiştir. Çoğunlukla en yakın komşuluk sezgiseli ile oluşturulduğunda çözüm değerleri daha iyidir. Aynı zamanda çaprazlama operatörleri arasında çözüm sonuçları açısından farklılıklar küçük boyuttaki probleme göre daha da azalmıştır ve çok küçük farklılık bulunmamaktadır sonuçlar birbirine çok yakındır. Ancak orta boyuttaki problemde de En iyi çözüm rasgele oluşturulan başlangıç çözümü ile elde edilmiştir.

4.3. Eil101 için İnceleme

Verilerin ilk hali şu şekildedir:



Şekil 4: Eil101 için veriler

➢ 101 şehirli problemde başlangıç popülasyonu *rasgele* bir şekilde oluşturulup; popülasyon büyüklüğü [20-2000] arasında farklı popülasyon büyüklüğü ile [100-5000] arasında farklı nesil sayısı ve üç farklı çaprazlama operatörü (OX, CX ve EEAX) hesaplama zamanı (saniye) da dikkate alınarak sonuçlar hesaplanmıştır. Eil101 için 15 tekrar sonucu elde edilen değerlerin ortalaması şu şekildedir:

Tablo 7: Eil76 için G, Ps ve Çaprazlama Operatörlerinin Başlangıç Popülasyonunun Rasgele Oluşturulduğundaki Sonuçları

Nesil sayısı (G)	Popülasyon büyüklüğü (P_s)	CX	Hesaplama Zamanı (saniye)	OX	Hesaplama Zamanı (saniye)	EERX	Hesaplama Zamanı (saniye)
G=100	P_s =20	2.9768e+03	0,39	2.5605e+03	0,57	2.7159+03	7,87
	P_s =40	2.4448e+03	0,55	2.4596e+03	0,80	2.7038e+03	16,33
	P_s =100	2.2639e+03	1,17	2.2316e+03	1,42	2.5494e+03	36,27
	P_s =200	2.0526e+03	2,02	1.9552e+03	2,40	2.4341e+03	71,93
	P_s =1000	2.0526e+03	9,37	1.4870e+0	10,98	2.0464e+03	371,32
	P_s =2000	1.8496e+03	22,71	1.2927e+03	22,70	2.0412e+03	720,13
G=500	P_s =20	2.1701e+03	1,09	2.4724e+03	1,59	2.4180e+03	42,09
	P_s =40	2.1349e+03	1,86	1.8850e+03	2,71	2.2954e+03	93,38
	P_s =100	1.6844e+03	5,34	1.6653e+03	5,63	1.8255e+03	216,96
	P_s =200	1.5566e+03	10,81	1.3852e+03	10,52	1.6542e+03	365,14
	P_s =1000	1.2085e+03	65,91	1.0708e+03	52,86	1.3701e+03	2052,38
	P_s =2000	1.2095e+03	139,41	2.2676e+03	116,07	1.19683+03	3426,66
G=1000	P_s =20	2.1044e+03	1,89	1.7992e+0.3	3,14	2.1916e+03	87,29
	P_s =40	1.9742e+03	3,57	1.2303e+03	5,42	1.9687e+03	165,38
	P_s =100	1.5542e+03	10,25	1.5053e+03	11,34	1.4972e+03	418,05
	P_s =200	1.3868e+03	21,38	1.2709e+03	21,72	1.4782e+03	763,97
	P_s =1000	1.2387e+03	122,74	967,372	118,34	1.2347e+03	3446,91
	P_s =2000	963,4249	373,80	938,6020	277,46	1.1160e+03	6398,00
G=5000	P_s =20	1.4877e+03	9,92	1.5279e+03	16,51	1.7115e+03	417,54
	P_s =40	1.1619e+03	17,61	1.3268e+03	30,79	1.3990e+03	896,96
	P_s =100	1.1762e+03	58,45	1.1392e+03	60,32	1.1687e+03	1972,71
	P_s =200	1.0875e+03	127,19	1.0674e+03	96,45	1.0695e+03	3364,765
	P_s =1000	987,6611	972,76	957,9026	498,33	986,7768	17705,66
	P_s =2000	898,7513	2613,41	**705,6579**	**1301,51**	935,6677	32358,59

Popülasyon büyüklüğünün artırılması önemli oranda iyileşme göstermektedir. Çaprazlama operatörlerini kıyasladığında en iyi değerler OX ile elde edilmiştir daha sonra CX ve en kötüsü ise EERX. Hesaplama zamanı açısından baktığımızda ise; OX, CX daha büyük hesaplama zamanına sahip ama aralarında çok büyük farklılık bulunmamaktadır. Ayrıca, genellikle popülasyon büyüklüğü 40'dan büyük olduğu OX; CX den daha kısa sürmektedir. EERX hesaplama zamanı açısından kötü performansa sahiptir.

Eil101 için en iyi popülasyon değeri nesil sayısı ve çaprazlama operatörü şu şekildedir.

G= 5000

P_s=2000 ve

OX operatörleri ile en iyi değer 705,6579 olarak elde edilmiştir.

➢ 101 şehirli problemde başlangıç popülasyonu *en yakın komşuluk sezgiseli* ile oluşturduğunda 15 tekrar sonucu elde edilen değerlerin ortalaması şu şekildedir:

Tablo 8: Eil101 için G, Ps ve Çaprazlama Operatörlerinin Başlangıç Popülasyonunun En Yakın Komşuluk Sezgiseli ile Oluşturulduğundaki Sonuçları

Nesil sayısı (G)	Popülasyon büyüklüğü (P_S)	CX	Hesaplama Zamanı (saniye)	OX	Hesaplama Zamanı (saniye)	EERX	Hesaplama Zamanı (saniye)
G=100	P_s =20	805,8881	0,43	808,4630	0,66	808,1597	9,01
	P_s =40	799,0969	0,60	808,4630	0,80	808,4630	16,58
	P_s =100	801,3857	0,82	793,8954	1,46	808,4630	41,01
	P_s =200	799,5302	2,99	783,0499	2,44	799,2484	84,77
	P_s =1000	790,9032	16,85	762,0977	10,62	790,7641	410,86
	P_s =2000	776,1284	45,05	749,9463	21,46	775,0867	803,06
G=500	P_s =20	802,1398	0,95	804,6383	1,57	808,4630	41,96
	P_s =40	801,1842	1,34	782,6641	2,76	802,8709	69,53
	P_s =100	795,4483	3,32	765,3031	5,59	789,0394	182,30
	P_s =200	774,5397	10,61	763,9129	10,83	790,3554	381,32
	P_s =1000	754,6259	102,01	741,0075	53,42	760,5867	1957,14
	P_s =2000	749,5623	242,28	738,7894	112,26	751,8846	4123,72
G=1000	P_s =20	797,9957	1,98	796,6898	3,15	806,6426	88,91
	P_s =40	793,7347	3,49	782,9990	5,43	804,7630	163,15
	P_s =100	786,9237	8,01	761,2956	11,42	791,9019	409,49
	P_s =200	766,2893	26,14	757,3606	21,77	767,2206	823,73
	P_s =1000	747,3717	167,69	740,0190	104,67	750,1173	3428,49
	P_s =2000	747,8578	479,97	736,9343	219,03	745,5536	6449,21
G=5000	P_s =20	789,8961	7,96	765,2615	14,76	794,6483	346,95
	P_s =40	770,7267	14,90	758,9986	26,14	793,0187	682,50
	P_s =100	756,2056	43,14	744,9367	55,60	757,3530	1562,09
	P_s =200	746,4059	109,67	740,5450	104,39	753,7869	3049,47
	P_s =1000	741,7751	782,54	736,1389	534,50	747,8162	16916,89

	P_S =2000	739,1149	2589,05	**736,1348**	**1332,96**	749,4179	33290,08

Eil101 için en iyi popülasyon değeri nesil sayısı ve çaprazlama operatörü şu şekildedir.

G= 5000

P_s=2000 ve

OX operatörleri ile en iyi değer 736,1348 olarak elde edilmiştir.

Tablo 7 ve Tablo 8 incelendiğinde diğer iki boyutta da olduğu gibi; başlangıç popülasyonun arttırılması en yakın komşuluk sezgiseli ile oluşturulduğu durumda çok büyük iyileşme sağlanmamaktadır. Bir diğer durum başlangıç popülasyonun en yakın komşuluk sezgiseli ile oluşturulduğunda çözüm daha iyi bir değerden başlamıştır. Aynı zamanda çaprazlama operatörleri arasında çözüm sonuçları açısından farklılıklar orta boyuttaki probleme göre daha da azalmıştır ve çok küçük farklılık bulunmamaktadır sonuçlar birbirine çok yakındır. Büyük boyuttaki problemde de en iyi çözüm rasgele oluşturulan başlangıç çözümü ile elde edilmiştir.

5. SONUÇ VE ÖNERİLER

Uygulamada üç farklı boyuttaki problemde başlangıç popülasyonu en yakın komşuluk sezgiseli ve rasgele bir şekilde oluşturularak farklı popülasyon büyüklükleri dikkate alınarak karşılaştırılmıştır. Başlangıç popülasyonu rasgele ve en yakın komşuluk sezgiseli ile oluşturulmuş, seçim yöntemi olarak turnuva seçimi tercih edilmiş, OX, CX ve EEAX çaprazlama operatörü ve yer değiştirme mutasyon operatörü kullanılmıştır. Farklı boyutlardaki problemler için en iyi sonucu veren popülasyon oluşturma şekli, çaprazlama operatörü, P_s ve G değerleri belirlenmiştir.

Elde edilen sonuçlar şu şekildedir:

Tablo 9: Eil51, Eil76 ve Eil101 için Çözüm Sonuçları

Başlangıç Popülasyonu	Problem	P_S	G	Çaprazlama operatörü	Çözüm Değeri	Hesaplama Zamanı (saniye)
Rasgele	Eil51	2000	500	OX	470,5779	92,96
	Eil76	2000	5000	OX	581,0146	1189,92
	Eil101	2000	5000	OX	705,6579	1301,51
En Yakın Komşuluk	Eil51	2000	5000	OX	476,3189	976,13
	Eil76	2000	5000	OX	606,6011	958,24
	Eil101	2000	5000	OX	736,1348	1332,96

Her üç şehir boyutunda da aynı çaprazlama operatörü ve P_S değeri iyi çözüm elde edilmiştir ancak nesil sayısında farklılık meydana gelmiştir Eil51 için 500 iken diğerlerinde 2000 ile çözüm elde edilmiştir. Başlangıç popülasyonu rasgele oluşturulduğunda en iyi çözüm değerini vermiştir. Her üç şehir boyutu için de en yakın komşuluk sezgiseli ve rasgele oluşturulan başlangıç popülasyonuna ilişkin sonuçlar karşılaştırıldığında; başlangıç popülasyonun en yakın komşuluk sezgiseli ile oluşturulduğu durumda çözüm daha iyi bir değerden başlamıştır. Tüm nesil sayısı başlangıç popülasyon büyüklüklerine baktığımızda çoğunlukla en yakın komşuluk sezgiseli ile oluşturulduğunda çözüm değerleri daha iyidir. Aynı zamanda çaprazlama operatörleri arasında çözüm sonuçları büyük farklılıklar bulunmamaktadır sonuçlar birbirine çok yakındır. Ancak en iyi çözüm değeri başlangıç popülasyonunun rasgele oluşturulduğunda elde edilmiştir.

Popülasyon' un sonuçlar üzerinde önemli etkiye sahip olduğunu görmekteyiz. Sonuçları daha da iyileştirmek için başlangıç popülasyonunu farklı yöntemlerle oluşturduktan sonra GA kullanılması bir yaklaşım olabilir. Popülasyon büyüklüğü arttırıldığında sonuçlarda önemli iyileşmeler gözlenmiştir. Ancak bu hesaplama zamanın da arttırmaktadır bu da istenmeyen bir durumdur. Amaç kısa sürede sonuç elde etmek olduğundan popülasyon büyüklüğü algoritmanın çalışma zamanı amaca bağlı olarak bir yerde durdurulmalıdır ve bu büyüklük dikkatle belirlenmelidir.

Bir diğer dikkat edilmesi gereken husus kullanılan çaprazlama operatörüdür. Uygulamada daha önce önerilmiş olan üç operatörü kullanılmıştır ve burada gözlemlenilen durum çaprazlama operatörlerin çözüm sonucunu önemli ölçüde etkilediği çözüm sonucunu iyileştirdiği gözlenmektedir. Kullandığımız operatörlerden OX ele aldığımız üç problem boyutunda en iyi sonucu vermektedir. OX operatöründe; şehirlerin sırası korunmaktadır. Bu yüzden şehirlerin sırasını koruyan operatörlerin geliştirilmesi çözümü daha da iyileştirebilir.

KAYNAKÇA:

Agarwal, T., & Kanchan. (2013). Using New Variation Crossover Operator of Genetic Algorithm for Solving the Traveling Salesmen Problem. *International Journal of Computer Science ve Information Technology*, *3*(1), 35–37

Cevre, U., Özkan, B., & Uğur, A. (2007). Gezgin Satıcı Probleminin Genetik Algoritmalarla Eniyilemesi ve Etkileşimli Olarak İnternet Üzerinde Görselleştirilmesi. *XI I."Türkiye'de İnternet "Konferansı ,Ankara*.

Ahmed, Z. H. (2010). Genetic Algorithm for the Traveling Salesman Problem using Sequential Constructive Crossover Operator. *International Journal of Biometrics ve Bioinformatics (IJBB)*, *3*(6), 96–105.

Bhattacharyya, M., & Bandyopadhyay, A. K. (2008). Comparative Study of Some Solution Methods for Traveling Salesman Problem Using Genetic Algorithms. *Cybernetics and Systems*, *40*(1), 1–24.

Çolak, S. (2010). Genetik Algoritmalar Yardımı ile Gezgin Satıcı Probleminin Çözümü Üzerine Bir Uygulama. *Ç.Ü. Sosyal Bilimler Enstitüsü Dergisi*, *19*(3), 423–438.

Davis, L. (1985). Applying Adaptive Algorithms to Epistatic Domains. *Proceedings of the 9th International Joint Conference on Artificial Intelligence - Volume 1*, *1*, 162–164.

Deep, K., & Thakur, M. (2007). A new mutation operator for real coded genetic algorithms. *Applied Mathematics and Computation*, *193*(1), 211–230.

Elmas, Ç. (2016). *Yapay Zeka Uygulamaları*. Ankara: Seçkin Yayıncılık.

Emel, G. G., & Taşkın, Ç. (2002). Genetik Algoritmalar ve Uygulama Alanları. *Uludağ Üniversitesi İktisadi ve İdari Bilimler Fakültesi*, *21*(1), 129–152.

Ghoseiri, K., & Sarhadi, H. (2007). A 2opt-DPX Genetic Local Search for Solving Symmetric Traveling Salesman Problem, 903–906.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. USA: Addison-Wesley Publishing Company.

Goldberg, D. E., & Deb, K. (1991). A Comparative Analysis of Selection Schemes Used in Genetic Algorithms. *Foundations of Genetic Algorithms*, *1*, 69–93.

Gopal, G., Kumar, R., Jawa, I., & Kumar, N. (2015). Enhanced Order Crossover for Permutation Problems. *International Journal of Innovative Research in Science, Engineering and Technology*, *4*(2), 151–157.

Grefenstette, J. J. (1986). Optimization of Control Parameters for Genetic Algorithms. *IEEE Transactions on Systems, Man and Cybernetics*, *16*(1), 122–128. https://doi.org/10.1109/TSMC.1986.289288

Javidi, M. M., Fard, R. H., & Jampour, M. (2015). Research in Random Parameters of Genetic Algorithm and Its Application on TSP and Optimization Problems. *Walailak Journal of Science and Techonology*, *12*(1), 27–34.

Laporte, G. (1992). The Traveling Salesman Problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, *59*, 231–247.

Larranaga, P., Kuıjpers, C. M. H., Murga, R. H., Inza, I., & Dızdarevıc, S. (1999). Genetic Algorithms for the Travelling Salesman Problem: A Review of Representations and Operators. *Artificial Intelligence Review*, *13*(2), 129–170.

Matai, R., Singh, S. P., & Mittal, M. L. (2010). Traveling Salesman Problem: An Overview of Applications, Formulations, and Solution Approaches. *Traveling Salesman Problem,Theory and Aplications*, 1–24.

Melanie, M. (1998). *An introduction to genetic algorithms*. Boston: MIT press.

Michalewicz, Z. (1992). *Genetic algorithms + data structures = Evolution programs*. Berlin: Springer-Vergal.

Paksoy, S., & Uzun, A. (2015). Genetik Algoritma ile Kaynak Kısıtlı Proje Çizelgeleme. *Ç.Ü. Sosyal Bilimler Enstitüsü Dergisi*, *17*(2).

Pham, D. T., ve Karaboga, D. (1997). Genetic algorithms with variable mutation rates : application to fuzzy logic controller design. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, *211*(2), 157–167.

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

Pulat, M., & Deveci Kocakoç, İ. (2016). Gezgin Satıcı Probleminin Genetik Algoritmalar Kullanarak

Çözümünde Çaprazlama Operatörlerinin Örnek Olaylar Bazlı İncelenmesi. *Yöneylem Araştırması Endüstri Mühendisliği 36.Ulusal Kongresi* (ss. 1–27). İzmir.

Razali, N. M., & Geraghty, J. (2011). Genetic Algorithm Performance with Different Selection Strategies in Solving TSP.

Sivanandam, S. N., & Deepa, S. N. (2008). *Introduction to Genetic Algorithms*. Berlin: Springer Science & Business Media.

Starkweather, T., Mcdaniel, S., Mathias, K., & Darrell, W. (1991). A Comparison of Genetic Sequencing Operators. *Proceedings of the Fourth International Conference on Genetic Algorithms*, 69–76.

Yu, X., & Gen, M. (2010). *Introduction to Evolutionary Algorithms*. New York: Springer Science & Business Media.