GU J Sci, Part C, 6(4): 774-786 (2018)

Gazi Üniversitesi

Fen Bilimleri Dergisi

PART C: TASARIM VE TEKNOLOJİ

http://dergipark.gov.tr/gujsc

Zaman Pencereli Araç Rotalama Problemi Çözümü İçin Çok Amaçlı Genetik Algoritma Yaklaşımı

Tolunay GÖÇKEN[1,*], Meltem YAKTUBAY[2], Fatih KILIÇ[3]

[1,2] *Endüstri Mühendisliği Bölümü, Adana Bilim ve Teknoloji Üniversitesi, Adana, Türkiye*

[3] *Bilgisayar Mühendisliği Bölümü, Adana Bilim ve Teknoloji Üniversitesi, Adana, Türkiye*

Makale Bilgisi

Başvuru: 22/02/2018
Düzeltme: 14/08/2018
Kabul: 13/09/2018

Anahtar Kelimeler

Araç rotalama
Çok-amaçlı optimizasyon
Genetik algoritma
NSGA-II
Süpürme algoritması

Öz

Bu çalışmada, talepleri bilinen müşterilerin, konumu bilinen bir deponun ve belirli sayıda aynı kapasiteye ve özelliklere sahip özdeş araçların bulunduğu klasik Araç Rotalama Probleminin (ARP) bir çeşidi olan Zaman Pencereli Araç Rotalama Problemi (ZPARP) ele alınmıştır. Müşterilere belirli bir zaman aralığında hizmet verilebilen ZPARP için toplam yolu ve araç sayısını minimize etmek amaç fonksiyonları olarak belirlenmiştir. ZPARP'ye etkin çözümler üretilmesi amacı ile meta-sezgisel bir yöntem olan genetik algoritmanın sezgisel metotlarla melezleştirilmiş bir uyarlaması önerilmiştir. Genetik algoritmanın başlangıç popülasyonu oluşturma aşamasında süpürme algoritması ve en yakın komşu tabanlı bir algoritma kullanılarak üretilen kaliteli çözüm kümeleriyle aramaya başlaması, böylece optimum sonuçlara daha hızlı ulaşılması planlanmıştır. Genetik algoritmada başlangıç popülasyonları oluşturmada farklı sezgisel yöntemlerin kullanılmasının istenilen sonuca bir etkisi olup olmadığı test edilmiştir. Literatürde var olan bir veri problemi çözülmüş, süpürme algoritmasını kullanan genetik algoritma ile daha etkin sonuçlara ulaşıldığı görülmüştür.

A Multi Objective Genetic Algorithm Approach for The Solution of Vehicle Routing Problem with Time Windows

Keywords

Vehicle routing
Multi-objective optimization
Genetic algorithm
NSGA-II
Sweep algorithm

Abstract

In this study, Vehicle Routing Problem with Time Windows (VRPTW) which is a type of classical Vehicle Routing Problem (VRP) includes customers with known demands, a single depot with known location and a certain number of identical vehicles with identical capacities and characteristics, is considered. Minimizing the total distance and the number of vehicles are determined as objective functions for VRPTW which is capable to serve the customers in a prespecified time interval. A hybridized version of genetic algorithm with heuristic methods is proposed to produce effective solutions for VRPTW. By using sweep algorithm and nearest neighbor-based algorithm at initial population generation phase of genetic algorithm, it is planned to begin the search with quality solution sets and in this way, get the optimum solutions faster. It has been tested whether the use of different heuristic methods in generation of initial population in genetic algorithm influences the desired solution. A benchmark problem in the literature has been solved and it is observed that the genetic algorithm beginning with sweep algorithm at initial population generation step reaches more effective solutions.

1. GİRİŞ (INTRODUCTION)

Lojistik yönetimi tedarik zinciri yönetiminin önemli bir parçasıdır. Çeşitli ürünlerin dağıtılması, toplanması veya insanların bir yerden bir yere taşınması gibi her gün karşılaşılan problemlere lojistik yönetimi çerçevesi içerisinde çözüm aranır. Lojistik yönetiminde maliyeti en aza indirmek ve ihtiyaçları hızlı bir şekilde karşılamak asıl amaçtır. Araç Rotalama Problemi (ARP) lojistik alanında yer alan önemli bir yönetim problemidir. ARP, talepleri bilinen müşterilerin talepleri kapasite kısıtlı özdeş araçlarla belirli bir depodan başlanarak ve yine aynı yerde sonlanacak şekilde karşılanırken, araçların kat edeceği toplam mesafeyi minimize eden rotaların belirlenmesi problemidir. Her müşterinin ihtiyacı tek seferde tek bir araç ile karşılanmalı ve aynı rotadaki müşterilerin toplam talep miktarları araç kapasitesini aşmamalıdır.

*İletişim yazarı, e-mail: tgocken@adanabtu.edu.tr

DOI: 10.29109/gujsc.397543

Günlük hayatta karşılaşılan bazı durumlar ve problem yapısındaki bazı farklılıklar ARP'nin çeşitlenmesine neden olmaktadır. Birden fazla depo olması, müşteriler arasındaki öncelik ilişkileri, araçların farklı özelliklere sahip olması, müşteri taleplerinin karakteristik yapısı bu durumlardan bazılarıdır. Müşteriler sadece belli bir zaman dilimi içerisinde servis edilebiliyorlar ise problem Zaman Pencereli Araç Rotalama Problemine (ZPARP) dönüşür.

Kombinatorik bir problem olan ZPARP'nin çözümü büyük veri setlerinde kesin optimizasyon teknikleri kullanıldığında çok uzun sürmektedir. Daha kısa çözüm süreleri amaçlandığında sezgisel yöntemler daha kullanışlı olmaktadır. Bu çalışmada meta-sezgisel bir yöntem olan Genetik Algoritma (GA) kullanılmıştır. GA doğadaki evrim mekanizmasına ve doğal seçim ilkelerine dayanan bir optimizasyon yöntemidir. GA döngüsel yapısı sayesinde başlangıç çözümlerini geliştirerek çözüm uzayındaki daha iyi çözümlere ulaşmaya çalışır. Algoritmanın çözüm kümesindeki her nokta kromozomlarla ifade edilmektedir. GA'nın her bir döngüsünde çözümü temsil eden kromozomlarda seçim, çaprazlama ve mutasyon olarak adlandırılan genetik operatörlerin uygulanması sonucu yeni bireylerin üretilmesi ve bu bireyler arasından güçlü bireylerin seçilmesi ile iyi çözüm kümeleri oluşturulmaktadır. Algoritmanın optimum çözüme daha hızlı ulaşabilmesi için başlangıç popülasyonu oluşturma aşamasında farklı teknikler uygulanarak algoritma melezleştirilebilmektedir. Bu çalışmada GA'da başlangıç popülasyonları oluşturmada farklı sezgisel yöntemlerin kullanılmasının istenilen sonuca bir etkisi olup olmadığı test edilmiş ve algoritma ZPARP literatüründe test problemi olan Solomon veri setine uygulanmıştır.

2. LİTERATÜR TARAMASI (LITERATURE REVIEW)

ZPARP, kombinatorik bir problemdir. Lenstra ve Kan [1] ARP ve ZPARP'nin zorluğunu analiz etmiş ve çözümü zor (NP-Hard) problemler sınıfında yer aldıkları sonucuna ulaşmışlardır. ZPARP için literatürde çeşitli kesin çözüm algoritmaları, sezgisel ve meta-sezgisel yöntemler bulunmaktadır. Dal-sınır (branch and bound), dal-kesme (branch and cut), sütun yaratma (column generation), dinamik programlama ve Lagrangian ayrıştırma yöntemleri kesin çözüm yöntemleri arasında bulunmaktadır. Müşteri sayısının az olduğu durumlar için kesin çözüm algoritmaları kullanılabilir iken, müşteri sayısı arttıkça gerekli bilgisayar hesaplama süresi üstel olarak arttığı için bu yöntemleri uygulamak mümkün olmamaktadır. Bu yüzden son yıllarda araştırmacılar daha çok sezgisel ve meta-sezgisel yöntemler kullanarak problemi çözmeye çalışmaktadırlar [2]. Tasarruf (saving), süpürme (sweep), iki aşamalı yöntem ve geliştirilmiş petal sezgisel algoritmaları klasik sezgisel yöntemler arasında yer alırken; genetik, tabu arama, benzetimli tavlama, karınca kolonisi, yapay arı kolonisi, parçacık sürüsü ve lokal arama algoritmaları meta-sezgisel yöntemler arasında yer almaktadır [3]. ZPARP için çözüm yöntemlerinin daha ayrıntılı literatür tarama çalışmaları [4], [5] ve [6]'da bulunabilir.

GA, ARP ve ZPARP çözümünde sık kullanılan meta-sezgisel yöntemlerden biridir. GA birçok sezgisel yöntem ile melez bir algoritma oluşturmaya imkân veren bir algoritma çeşididir. Klasik veya meta-sezgisellerle birlikte uygulanabilmektedir. Baker ve Ayechew [7] kapasite kısıtlı ARP için komşu arama algoritması ile birlikte kullanılan melez bir GA geliştirmiştir. Çalışmalarında GA'nın başlangıç popülasyonunu oluşturmak için süpürme ve genelleştirilmiş atama yaklaşımı (generalized assignment approach) algoritmaları kullanılmıştır. Mester ve Bräysy [8], kapasite kısıtlı ARP'nin çözümü için yönlendirilmiş yerel arama (guided local search) ve GA'dan oluşan iki aşamalı tekrarlamalı bir yöntem kullanmıştır. Berger ve Barkaoui [9] ardışık ekleme sezgiselini (sequential insertion heuristic) GA ile birlikte kullanmıştır. Thangiah vd. [10] GIDEON adında biri kümeleme diğeri rotalama işlemi yapan iki ayrı modülden oluşan bir GA sistemi önermişlerdir. Ibrahim vd. [11] En Yakın Komşu sezgiseli ile melezleştirilmiş GA'yı bir şirketin gerçek bir ZPARP olan şişelenmiş su dağıtımı problemine uygulamışlardır. Önerilen algoritma, toplam yol, zaman ve ceza maliyeti bakımından şirketin gerçek hizmet verilerine göre daha iyi sonuçlara ulaşmıştır. Prins [12] ARP için etkili bir evrimsel algoritma önermiştir. Kromozomlarda rota sonlarını belirten bir gösterge kullanmak yerine, kromozomu rotalara ayıran bir bölme algoritması geliştirmiş ve mutasyon aşamasında 9 farklı kural ile yerel arama prosedürünü uygulamıştır. Önerilen algoritma etkili sonuçlara ulaşmıştır. Chang ve Chen [13], Prins'in önerdiği algoritmayı ZPARP'ye uygulamışlardır. Ele aldıkları problemde zaman penceresi olarak sadece servise başlama zamanının belirtildiği ve araçların kapasite sınırı olmadığı varsayılmıştır. Algoritma, 3 farklı popülasyon büyüklüğü ve 3 farklı mutasyon olasılığı belirlenerek 2 farklı veri setinde test edilmiştir. Mutasyon olasılığı

arttıkça, genel olarak, hata oranının ve rota sayısının azaldığı gözlenmiştir. Ayrıca popülasyon büyüklüğü ve mutasyon olasılığı artıkça daha iyi sonuçlar elde edilmiştir.

Ombuki, Ross ve Hanshar [14] ZPARP için Pareto sıralama tekniği kullanarak çok amaçlı GA yaklaşımı sunmuşlardır. Amaç olarak araç sayısını ve toplam maliyeti minimize etmek belirlenmiştir. Pareto değerlendirmesi sayesinde amaçların ağırlıklandırılmasına ve ağırlıklı toplam yönteminin kullanılmasına ihtiyaç kalmamıştır. Bu yaklaşımla etkili sonuçlara ulaşılmıştır. Haddadene, Labadie ve Prodhon [15] evde sağlık bakımı alanında zaman penceresi, senkronizasyon ve öncelik kısıtlı ARP çözümü için melez bir NSGA-II (Non-dominated Sorting Genetic Algorithm II) (Mağlup Olmayan Sıralamalı Genetik Algoritma II) yöntemi önermişlerdir. Problem amaçları seyahat maliyetini minimize etmek ve hasta memnuniyetini maksimize etmek olarak belirlenmiştir. Temel NSGA-II ile melez NSGA-II karşılaştırması yapılmış ve yerel arama tabanlı melez NSGA-II daha iyi sonuçlara ulaşmıştır. Tan, Chew ve Lee [16] bazı sezgiseller ve özelleştirilmiş genetik operatörler ile birleştirilen melez çok amaçlı evrimsel algoritma geliştirmişlerdir. Toplam yolu ve araç sayısını minimize etmeyi amaçlamışlardır. Geliştirilen algoritma iyi sonuçlar üretmiştir. Mungwattana vd. [17] GA, değiştirilmiş ilerletici ekleme sezgisel yöntemi (MPFIH) ve λ-değiştirme yerel arama azaltma yöntemi (λ-LSD) kullanarak esnek ZPARP için bir çözüm yöntemi önermişlerdir. Araç sayısını ve toplam seyahat süresini minimize etmeyi amaçlar olarak belirlemişlerdir. Önerilen algoritma genel olarak etkili sonuçlar sağlamıştır. Ghoseiri ve Ghannadpour [18] hedef programlama ve GA kullanarak ZPARP çözümü için bir model önermişlerdir. Araç filosu büyüklüğünü ve toplam seyahat süresini minimize etmeyi amaçlar olarak belirleyerek problemi çok amaçlı olarak ele almışlardır. Modellerinde bazı sezgisellerle beraber, Pareto sıralama tekniğini ve elitizm stratejisini kullanmışlardır ve oldukça etkili sonuçlara ulaşmışlardır. Najera ve Bullinaria [19] ZPARP için çok amaçlı evrimsel algoritma kullanarak bir çözüm önerisi sunmuşlardır. Önerdikleri algoritmada Pareto sıralama tekniğini ve benzerlik ölçüm metodunu uygulamışlardır. Algoritma yüksek rekabetçi sonuçlara ulaşmıştır.

3. PROBLEM TANIMI (PROBLEM DEFINITION)

ARP bir grup müşteriye hizmet verecek araçlara en uygun rotaların belirlenmesi problemidir. Müşterilerin sayısı, konumları ve talep miktarları belirlidir. Problemde konumu bilinen ve ürünleri toplama merkezi olan bir tane depo bulunur. Her araç, rotasına depodan başlamak ve rota sonunda depoya dönmek zorundadır. Araçlar belirli sayıdadır ve aynı özelliklere sahiptirler. Özdeş araçların kapasite kısıtları bulunmaktadır. Bir aracın takip edeceği rotada hizmet sunacağı müşterilerin talep miktarları toplamı araç kapasitesini aşmamalıdır. Her müşteriye sadece bir araç hizmet sunmalı ve tek seferde talebi karşılamalıdır. Sadece araç kapasite kısıtı bulunan ARP, Kapasite Kısıtlı ARP olarak anılır.

ZPARP'de Kapasite Kısıtlı ARP'de olduğu gibi aynı karakteristik özelliklere sahip araçlar, müşteriler ve bir tane depo bulunmaktadır. Bu problem türünü diğerlerinden ayıran ve problemi daha da güçleştiren kısıt, her müşteri için servise başlanabilecek belirli bir (e_i, l_i) zaman aralığının olmasıdır. Bu zaman aralığı her müşteri için tanımlanmış olan zaman pencere kısıtıdır. Her müşteriye kendi zaman penceresi içerisinde hizmete başlanması gerekmektedir. Her müşteriye dağıtım yapmak ya da hizmet sunmak belirli bir s_i servis süresi kadar zaman gerektirir. Araç i müşterisine uğradıktan sonra s_i hizmet süresi kadar kalacaktır, daha sonra bir sonraki müşteriye ya da depoya hareket edecektir. Servis maliyeti tüm müşterilerin servisi için gerekli araç sayısı ve araçların aldığı toplam mesafe ile ölçülür [20]. Problemdeki amaç maliyeti en küçüklemektir.

ZPARP zaman pencere kısıtının sıkı veya esnek olma durumuna göre ikiye ayrılır. Sıkı zaman pencereli ARP'de müşterilerin zaman aralıkları dışında hizmet verilmemektedir. Araç müşteriye zaman aralığının başlangıcından önce ulaşmışsa beklemek zorundadır, zaman aralığı bitiminden sonra da hizmet verememektedir. Esnek zaman pencereli ARP'de zaman pencere kısıtı ceza maliyeti karşılığında ihlal edilebilmektedir [21]. Bu çalışmada, sıkı zaman pencereli ARP ele alınmıştır.

ZPARP için amaçlar şu şekilde olabilir:

- Toplam yolu minimize etmek,
- Toplam seyahat süresini minimize etmek,
- Araç sayısını minimize etmek,

- Araçların bekleme sürelerini minimize etmek, ve
- Araçların alan kullanımını maksimize etmek.

Bunlardan biri veya birkaçı amaç olarak seçilebilir. Amaçların minimuma ulaşmak olduğu durumlarda birini azaltmaya çalışmanın diğerinde artışa sebep olması birbirleriyle çelişkili olduklarını gösterir. Çok amaçlı optimizasyon yaparken daha etkin sonuçlara ulaşabilmek için seçilen amaçların birbirleriyle çelişkili olması önemlidir. Bu sayede çözüm uzayı araştırılırken amaçların eş zamanlı olarak optimize edilmesi anlamlı olur. Bu çalışmada toplam yolu ve araç sayısını minimize etmek amaç fonksiyonları olarak belirlenmiştir ve problem çok amaçlı optimizasyon problemine dönüştürülmüştür.

Problem kısıtları ise şu şekildedir:

- Rotalar deponun zaman penceresi içerisinde depoda başlayıp yine depoda bitmelidir.
- Her müşteri tek bir araçla tek seferde hizmet görmelidir.
- Aynı rotadaki müşterilerin toplam talep miktarları araç kapasitesini aşmamalıdır.
- Müşteriler zaman pencereleri içerisinde hizmet görmelidir. Erken gelen araçlar müşterinin hazır olma zamanını beklemek durumundadır.

ZPARP'nin matematiksel modelini oluşturmada kullanılan karar değişkenleri, parametreler ve kümeler aşağıdaki gibidir.

Karar değişkenleri:

x_{ijk} $\begin{cases} 1, \text{k. araç i müşterisinden j müşterisine gidiyorsa} \\ 0, \text{aksi takdirde} \end{cases}$

t_i i müşterisine varış zamanı

w_i i müşterisinde bekleme süresi

Parametreler:

d_{ij} i müşterisi ve j müşterisi arasındaki mesafe

s_i i müşterisinin servis süresi

e_j j müşterisinin zaman aralığının başlangıç zamanı

l_j j müşterisinin zaman aralığının bitiş zamanı

l_0 Deponun zaman aralığının bitiş zamanı (Bir aracın maksimum seyahat süresi)

m_i i müşterisinin talep miktarı

Q Özdeş araçların kapasitesi

Kümeler:

C $\{1, 2, \dots, n\}$ Müşteri kümesi

N $\{0, 1, 2, \dots, n\}$ Düğüm kümesi (0. düğüm depoyu ifade etmektedir.)

V $\{1, 2, \dots, v\}$ Araç kümesi

ZPARP'nin matematiksel modeli aşağıdaki gibidir:

Amaç fonksiyonları:

Minimum

$$\sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{k=1}^{v} d_{ij} * x_{ijk} \quad (1)$$

$$\sum_{k=1}^{v} \sum_{j=1}^{n} x_{0jk} \quad (2)$$

Şu kısıtlara göre:

$$\sum_{k=1}^{v}\sum_{j=1}^{n} x_{0jk} \leq v \quad (3)$$

$$\sum_{j=1}^{n} x_{0jk} = \sum_{j=1}^{n} x_{j0k} \leq 1, \quad \forall k \in V \quad (4)$$

$$\sum_{k=1}^{v}\sum_{i=0\, i\neq j}^{n} x_{ijk} = 1, \quad \forall j \in C \quad (5)$$

$$\sum_{k=1}^{v}\sum_{j=0, j\neq i}^{n} x_{ijk} = 1, \quad \forall i \in C \quad (6)$$

$$\sum_{i=1}^{n}\left(m_i * \sum_{j=0, j\neq i}^{n} x_{ijk}\right) \leq Q, \quad \forall k \in V \quad (7)$$

$$\sum_{i=0}^{n}\sum_{j=0, j\neq i}^{n} x_{ijk}\left(d_{ij} + s_i + w_i\right) \leq l_0, \quad \forall k \in V \quad (8)$$

$$t_0 = w_0 = s_0 = 0 \quad (9)$$

$$t_i + w_i + s_i + d_{ij} = t_j, \quad \forall (i,j) \in N, i \neq j, if\ x_{ijk} = 1 \quad (10)$$

$$w_i = max\{e_i - t_i, 0\}, \quad \forall i \in C \quad (11)$$

$$e_j \leq \left(t_j + w_j\right) \leq l_j, \quad \forall j \in C \quad (12)$$

$$x_{ijk} \in \{0,1\}, \quad \forall k \in V, \forall (i,j) \in N \quad (13)$$

$$t_i \geq 0, \quad \forall i \in C \quad (14)$$

$$w_i \geq 0, \quad \forall i \in C \quad (15)$$

Amaç fonksiyonlarından Eş. 1 toplam yolu ve Eş. 2 kullanılan araç sayısını minimize etmeyi amaçlar. Eş. 3 kısıtı depodan çıkabilecek maksimum araç sayısının v olmasını sağlar ve böylece tüm araçların kullanım zorunluluğu ortadan kalkar. Eş. 4 kısıtı her rotanın depodan başlayacağını ve depoda biteceğini belirtir. Eş. 5 ve Eş. 6 kısıtları her müşteriye sadece tek bir aracın servis sunmasını sağlar. Eş. 7 kısıtı bir rotadaki müşterilerin toplam talep miktarlarının araç kapasite sınırını aşmasına engel olur. Eş. 8 -12 problemin zaman pencere kısıtını sağlar. Eş. 8 kısıtı maksimum seyahat süresi kısıtıdır. Eş. 9, Eş. 10'da i düğümünün 0 olması durumunda depo için kullanılacak karar değişkenlerinin ve bir parametresinin değerlerini gösterir. Eş. 10, müşteri i'den müşteri j'ye gidiliyorsa, müşteri j'ye varış zamanının müşteri i'ye varış zamanından i'deki bekleme süresi, servis süresi ve i'den j'ye gidiş süresi kadar sonra olduğunu ifade eder. Araçların 1 birim mesafeyi 1 birim zamanda gittikleri varsayıldığından elde edilen mesafe değeri süre olarak da kullanılabilir. Eş. 11 kısıtı eğer araç müşterinin hazır olma zamanından önce gelmişse bekleme süresinin hesaplanmasını sağlar. Eş. 12 kısıtı müşteriye sunulacak hizmetin ilgili müşterinin zaman aralığı içinde başlamasını sağlar. Eş. 13 -15 kısıtları karar değişkenlerinin alabileceği değer kümelerini belirtir. Bu model ZPARP için uygulanabilir çözümlere ulaşılmasını sağlar.

4. ÖNERİLEN ALGORİTMA (PROPOSED ALGORITHM)

4.1. Başlangıç Algoritmaları (Initial Algorithms)

4.1.1. Algoritma I (Algorithm I)

GA'nın ilk aşaması, her bireyin bir çözümü temsil ettiği bir çözüm kümesi olan başlangıç popülasyonunu oluşturmaktır. Bu çalışmada kullanılan ilk başlangıç popülasyonu oluşturma algoritması En Yakın Komşu algoritması temel alınarak oluşturulmuştur. En Yakın Komşu algoritması bir rotaya eklenmiş en son müşterinin ardına bu müşteriye mesafe açısından en yakın olan başka bir müşteriyi eklemektedir. Ancak problemde zaman pencere kısıtı olduğundan rotaya atanacak müşterinin seçimine sadece mesafeye bakarak karar vermek istenilen çözüme ulaşmada yeterince etkili olamamaktadır. Bu nedenle depodan başlayarak bir rotaya atanacak ilk müşterinin seçiminde müşterinin depoya uzaklığı ile o müşterinin servise hazır olma süresi karşılaştırılır, ilgili müşteriye bu değerlerden büyük olanı seçim değeri olarak atanır. Bir birim uzaklık, bir birim zaman diliminde gidildiği varsayıldığı için karşılaştırma rahatlıkla yapılabilir. Bütün müşteriler arasından seçim değeri minimum olan müşteri ilk rotanın ilk müşterisi olarak seçilir. Bir rotaya eklenmek üzere seçilen her müşteri, müşteri listesinden kaldırılır ve araç kapasitesinin doluluğu hesaplanır. Sıradaki müşterinin seçimi için rotalanan son müşterinin müşteri listesindeki rotalanmayan müşterilere uzaklıkları hesaplanır. Servis süresi dikkate alınarak olası müşterilerin servise başlama süreleri hesaplanır ve zaman pencere kısıtını sağlayan müşteriler arasında bir değerlendirme yapılır. Müşteriler arasında seçim

değeri küçük olanın rotaya alınma olasılığı yüksek olacak şekilde rastgele bir seçim yapılır. Seçilen müşteri rotaya eklenir ve müşteri listesinden kaldırılır. Araç kapasitesi dolana kadar müşteriler aynı şekilde rotaya alınmaya devam edilir. Araç kapasitesinin aşıldığı durumda aracın depoya dönmesi için rota sonuna depo eklenir ve yeni bir rotaya başlanır. Rota çözümleri rota kümesine eklenir. Müşteri listesinde müşteri kalmayınca rotalarda iyileştirme yapılır. En Yakın Komşu tabanlı olarak oluşturulan Algoritma 1'in aşamaları Tablo 1'de gösterilmiştir.

Tablo 1. *En Yakın Komşu Tabanlı Algoritma*

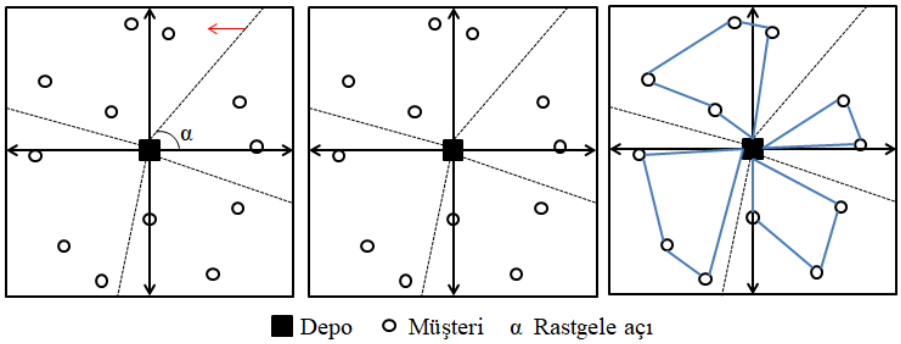
Algoritma 1 Aşamaları
Girdi: MüsteriBilgileri, KapasiteSınırı, AracSayısı.
Çıktı: RotaKümesi.
1: RotaKümesi ← ∅
2: rota ← Depoyu kuyruğa ekle // *Rota çözümleri depo ile başlar.*
3: müsteriListesi ← Müşteri listesi oluştur
4: uygunMüsteriListesi ← UzaklıkHesapla (müsteriListesi, Depo, KapasiteSınırı)
5: Döngü müsteriListesi ≠ ∅ **sağlandığı sürece**
6: eklenenDüğüm ← MüşteriSeç(uygunMüsteriListesi) // *Olasılık tabanlı seçim uygulanır.*
7: rota ← eklenenDüğüm'ü kuyruğa ekle
8: müsteriListesi.kaldır(eklenenDüğüm)
9: uygunMüsteriListesi ← UzaklıkHesapla (müsteriListesi, eklenenDüğüm, KapasiteSınırı)
10: **Eğer** uygunMüsteriListesi = ∅
11: rota ← Depoyu kuyruğa ekle // *Rota çözümleri depo ile biter.*
12: RotaKümesi ← Rotayı çözüm kümesine ekle
13: rota ← Boş Rota oluştur ve Depoyu kuyruğa ekle
14: uygunMüsteriListesi ← UzaklıkHesapla (müsteriListesi, Depo)
15: Döngü sonu
16: RotaKümesi ← İyileştir (RotaKümesi, KapasiteSınırı, AracSayısı)

4.1.2. Algoritma II (Algorithm II)

1974 yılında Gillett ve Miller [22] tarafından önerilen Süpürme algoritması bu çalışmada başlangıç popülasyonu oluşturmada ikinci algoritma olarak kullanılmıştır. Süpürme algoritmasında (x, y) koordinatları bilinen müşterilerin polar açıları, orijin kabul edilen depoya göre Eş. 16'da gösterildiği gibi hesaplanır:

$$An(i) = arctan\left[\frac{(y(i)-y(0))}{(x(i)-x(0))}\right] \tag{16}$$

Hesaplanan polar açılar küçükten büyüğe doğru sıralanır. Şekil 1'de gösterildiği gibi, orijinden başlayan ve rastgele belirlenen bir açıdan çıkarılan ışın saat yönünün tersi yönünde döndürülerek süpürme işlemi gerçekleştirilir. Bu esnada ışının üzerinden geçtiği müşteriler sıra takip edilerek bir araca atanır. Atama işlemi araç kapasitesi dolana kadar devam eder, kapasite kısıtı aşıldığı durumlarda atamaya kullanılmamış bir araç ile devam edilir. Bu şekilde müşteriler kümelenmiş olur. Kümelenen müşterilere depo dahil edilerek rotalama işlemi yapılır. Rotaya ilk olarak depo atanır ve aynı kümedeki müşterilerden depoya uzaklığı en kısa olan rotaya eklenir. Rotaya eklenen müşterinin rotalanmayan müşterilere uzaklığı hesaplanır ve zaman kısıtını ihlal etmediği sürece en yakın olan rotaya alınır. Bu şekilde tüm kümelerdeki müşteriler için birer rota oluşturulur. Rotalanamayan müşterilerin olması durumunda, mevcut rotalara kısıtların ihlali olmayacak şekilde eklenebilecekler eklenir. Uygun bir yer bulunmazsa yeni bir rota oluşturulur. Tüm rotalarda iyileştirme yapılır. Süpürme Algoritması tabanlı oluşturulan Algoritma 2'nin aşamaları Tablo 2'de gösterilmiştir.



Şekil 1. *Süpürme Algoritması*

Tablo 2. *Süpürme Tabanlı Algoritma*

Algoritma 2 Aşamaları
Girdi: MüsteriBilgileri, KapasiteSınırı, AracSayısı.
Çıktı: RotaKümesi**.**
1: RotaKümesi ← ∅
2: KullanılmayanMüsteri ← ∅
3: müsteriListesi ← Müşterilerin depoya göre polar açılarını hesapla ve sırala
4: rastAci ← RastgeleSayiSec (0, 359)
5: hamRotaKümesi ← MüsteriGrupla (müsteriListesi, rastAci, KapasiteSınırı)
6: Her bir müşteri kümesi için mstKume ← hamRotaKümesi // *Kümeden bir eleman çekilir.*
7: rota ← Depoyu kuyruğa ekle // *Rota çözümleri depo ile başlar.*
8: eklenenDüğüm ← Depo
9: uygunMüsteriListesi ← UzaklıkHesapla (mstKume, eklenenDüğüm)
10: **Döngü** mstKume ≠ ∅ **sağlandığı sürece**
11: eklenenDüğüm ← MüsteriSeç (uygunMüsteriListesi, eklenenDüğüm) // *En yakın müşteri seçilir.*
12: rota ← eklenenDüğüm'ü kuyruğa ekle
13: mstKume.kaldır (eklenenDüğüm)
14: uygunMüsteriListesi ← UzaklıkHesapla (mstKume, eklenenDüğüm)
15: **Eğer** uygunMüsteriListesi = ∅ **ve** mstKume ≠ ∅
16: KullanılmayanMüsteri ← mstKume
17: mstKume ← ∅
18: **Döngü sonu**
19: rota ← Depoyu kuyruğa ekle // *Rota çözümleri depo ile biter.*
20: RotaKümesi ← rotayı çözüm kümesine ekle
21: Döngü sonu
22: RotaKümesi ← İyileştir (RotaKümesi, KullanılmayanMüsteri, KapasiteSınırı, AracSayısı)

4.2. Genetik Algoritma (Genetic Algorithm)

GA'da çözümü aranan problem için olası tüm çözümler birey (kromozom) olarak adlandırılır ve her bir birey çözümü oluşturan parametrelerden oluşmaktadır. Bireylerden oluşan çözüm kümesi popülasyonu oluşturmaktadır. Başlangıç popülasyonunun üretilmesinden sonra her bir çözümün amaç fonksiyon değeri hesaplanmaktadır. Amaç fonksiyon değeri olası çözümün istenen çözümü ne kadar karşıladığının bir ölçüsüdür. Amaç fonksiyon değeri problemde belirlenen ölçütü sağlıyorsa algoritma sonlandırılır, aksi halde seçim, çaprazlama ve mutasyon işlemleri uygulanarak optimizasyon ölçütü sağlanana veya önceden belirlenmiş olan bir iterasyon sayısına ulaşıncaya kadar algoritma devam ettirilir [23].

Kılıç ve Gök [24] 2015'teki çalışmalarında geliştirdikleri GA'yı toplu taşıma araçlarının rotalarının belirlenmesi probleminde kullanmışlardır. Bu çalışmada ise geliştirilen GA ZPARP'ye çözüm üretmek için kullanılmıştır. ZPARP çözümü için uyarlanan GA'nın temel adımları Tablo 3'te verilmektedir.

Tablo 3. *Genetik Algoritma*

Genetik Algoritma Aşamaları
Girdi: JenerasyonSayısı, PopülasyonBüyüklüğü, BaşlangıçAlgoritması, $P_{mutasyon}$, $P_{çaprazlama}$, MüsteriBilgileri, KapasiteSınırı, AracSayısı. **Çıktı:** Popülasyon. **1:** Popülasyon ← BaşlangıçProsedürü (PopülasyonBüyüklüğü, BaşlangıçAlgoritması, VeriDosyası) **2:** İterasyon ← 0 **3: Döngü** İterasyon ≤ JenerasyonSayısı **sağlandığı sürece** **4:** Yavrular ← EbeveynSeçimi (Popülasyon) **5:** ÇaprazlamaİşlemiI (Yavrular, $P_{çaprazlama}$) **6:** MutasyonİşlemiI (Yavrular, $P_{mutasyon}$) **7:** Popülasyon ← NSGA-II (Popülasyon + Yavrular, PopülasyonBüyüklüğü) **8:** İterasyon ← İterasyon + 1 **9: Döngü sonu**

Popülasyondaki her birey permütasyon kodlaması kullanılarak gösterilmiştir. Permütasyon kodlamada müşteri numaraları araç tarafından servis edilme sırasına göre dizilmektedir. Başlangıç popülasyonları önceki bölümde anlatılan algoritmalar kullanılarak üretilmiştir. Yavruların üretimi için ebeveyn seçim aşamasında bireylerin baskınlık durumları dikkate alınmıştır. Çok amaçlı optimizasyon yapıldığı için baskınlık sıralamasına Pareto optimal cepheler kullanılarak karar verilmiştir. Baskın olma durumlarına göre sıralanan bireyler ardışık olarak ikişerli eşleştirilmiştir. Çaprazlama işlemi için çift noktalı çaprazlama yöntemi kullanılmıştır. Genetik çeşitliliği korumak ve çözümlerin yerel optimuma düşmesini engellemek adına komşu çözümleri bulmak amacıyla mutasyon işlemi uygulanmıştır. Mutasyon operatörü olarak yer değiştirme yöntemi şu şekilde gerçekleştirilmiştir. Üretilen yavru bireyden rastgele bir rota, rotadan da rastgele bir müşteri seçilip kaldırılır. Çözümden çıkarılan müşteri amaç fonksiyonunu minimum yapacak şekilde en uygun yere yerleştirilir. Son olarak bir sonraki popülasyonu oluşturacak bireylerin seçimi için NSGA-II kullanılmıştır. NSGA-II için DEAP Kütüphanesinden yararlanılmıştır [25, 26, 27].

NSGA-II hızlı ve elitist (seçkinci) çok amaçlı bir optimizasyon çözüm yöntemidir. NSGA-II ile bir popülasyonun oluşturabileceği Pareto cephelerin tüm noktaları belirlenebilmekte ve tüm bireyler baskınlıklarına, başka bir deyişle mağlup olmamalarına, göre sıralanabilmektedir. Bir birey başka bir bireye şu durumlarda baskın olur: Her bir amaç fonksiyonu değeri diğerinin değerinden daha kötü değildir. Ve en az bir amaç fonksiyonu değeri diğerinden daha iyidir [28]. Bir en küçükleme probleminde a bireyinin b bireyine baskın olması, başka bir deyişle a bireyinin b bireyine mağlup olmaması, aşağıda gösterilen durumlarda sağlanır:

$$a \leq b \ (a \ baskın \ b) \Leftrightarrow \forall_i : a_i \leq b_i \ , \exists_i : a_{i_0} < b_{i_0}$$

Popülasyondaki bütün bireyler her bir amaç fonksiyonu için birbirleriyle karşılaştırılır. Her bireyin 1) mağlup olduğu birey sayısı bir sayaç yardımı ile hesaplanır ve 2) mağlup ettiği bireyler kümesi oluşturulur. Sayaç değeri 0 olan bireyler ilk cephede (first nondominated front) yer alırlar. İlk cephedeki bireylerin rütbeleri (rank) 1 olur. Daha sonra rütbeleri 1 olan bireylerin mağlup ettiği bireyler kümesindeki her elemanın sayaç değeri 1 azaltılır. Sayaç değeri 0 olan bireyler ikinci cepheyi oluştururlar ve rütbeleri 2 olur. Bu işlem bütün cepheler tanımlanıncaya kadar devam eder. Düşük rütbelerin önemi fazladır ve bireylerin sıralaması düşük rütbeliden yüksek rütbeliye doğru olur.

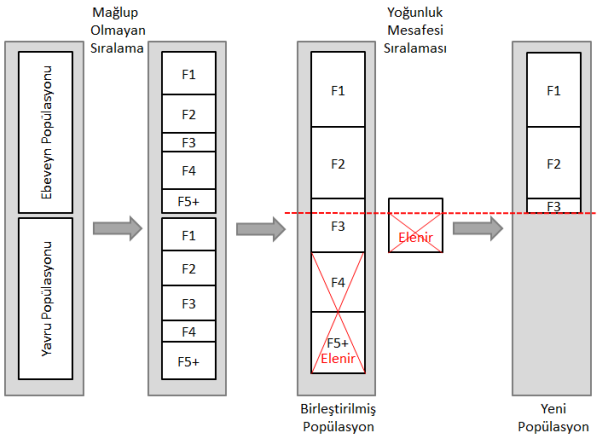
Bireyler aynı cephede yer alıyorsa, bireylerin önem sırasını belirlemek için ikinci bir ölçüte ihtiyaç duyulmaktadır. Bu nedenle yoğunluk mesafesi veya yığılma uzaklığı (Crowding Distance) olarak ifade edilen bir kavram kullanılmaktadır. Yoğunluk mesafesi, ilgili bireyin her amaç için komşuluklarına olan uzaklığın toplamıdır ve Eş. 17'de gösterildiği gibi hesaplanır:

$$CD_i = \sum_{m=1}^{n} \frac{|f_{i+1}^m - f_{i-1}^m|}{f_{max}^m - f_{min}^m} \tag{17}$$

CD: yoğunluk mesafesini, n: toplam amaç sayısını, f: amaç fonksiyon değerini ifade etmektedir. f_{max}^m ve f_{min}^m m. amaç fonksiyonunun ilgili cephedeki en büyük ve en küçük olduğu değerlerdir. Amaç

fonksiyonunun en büyük ve en küçük olduğu değerler ilgili cephenin sınır bireylerine aittir ve sınır bireylerinin sonsuz yoğunluk mesafesine sahip olduğu kabul edilir. Sınır bireyler arasında kalan bireylerin yoğunluk mesafeleri hesaplanır. Küçük yoğunluk mesafesine sahip bireyler diğer çözümlere daha yakındır. Çözümler arasında karşılaştırma yapılırken yüksek mesafeye sahip olanlar, başka bir deyişle yoğunluktan uzak olanlar, tercih edilir.

Bir sonraki popülasyonu oluşturacak bireylerin seçimi için Şekil 2'de gösterildiği gibi o döngüdeki popülasyon ve oluşturulan yavru popülasyonu birleştirilir. Bu sayede bir popülasyondaki en iyi bireylerin korunarak sonraki nesle aktarılması, yani elitizm (seçkincilik) sağlanır. Her popülasyonda N adet birey bulunur ve tekrar N adet birey seçilecektir. 2N adet birey mağlup olmayan sıralama yardımıyla baskınlıklarına göre sınıflandırılır. Oluşturulan cephelere göre rütbesi 1 olan bireylerden başlanarak seçme işlemi yapılır. 1., 2., 3., 4. cepheler şeklinde ilerleyerek bireyler sırayla popülasyona alınır. Toplam birey sayısı popülasyon büyüklüğünü aştığı durumda alınan son cephedeki bireylerde yoğunluk mesafesi karşılaştırması yapılır. Buna göre ilgili cephedeki bireylerin yoğunluk mesafesi değerleri büyükten küçüğe doğru sıralanır ve yeni popülasyonu dolduracak sayıdaki bireyler yeni popülasyona seçilir. Bu şekilde bir sonraki popülasyon oluşturulmuş olup tekrar GA prosedürü işletilir.



Şekil 2. *NSGA-II elitist seçim prosedürü*

5. SAYISAL SONUÇLAR (NUMERICAL RESULTS)

Önerilen algoritma Solomon'un C1, R1 ve RC1 veri setlerinin ilk beş problemine uygulanarak test edilmiştir [29]. Veri setleri müşterilerin coğrafik alandaki dağılım şekillerine göre farklılaşmaktadır. C1 setinde müşteriler depo etrafında kümelenmiş olarak, R1 setinde rastgele şekilde ve RC1 setinde yarı kümelenmiş yarı rastgele şekilde bulunmaktadır. Kullanılan veri setleri kısa zaman çizelgelerine sahiptir, araçların maksimum seyahat süreleri fazla uzun değildir. Problemlerde bir depo ve 100 müşteri bulunmaktadır. Müşteriler ve depoların konumları (x, y) koordinatları ile belirtilmiştir. Birbirleri arasındaki mesafeler Öklid uzaklığı ile hesaplanır ve elde edilen değer mesafe birimidir. Araçların 1 birim mesafeyi 1 birim zamanda gittikleri varsayıldığından elde edilen mesafe sonucu süre olarak da kullanılabilir. Müşterilerin talep miktarları veri setinde mevcuttur. Toplam 25 adet özdeş araç bulunmaktadır ve araçların kapasiteleri 200 birimdir. Her müşterinin servise en erken başlanılabileceği hazır olma zamanı ile en geç başlanılabileceği son zamanları ve servis süreleri belirtilmiştir.

Her bir problem seti için başlangıç popülasyonu oluşturma algoritmaları 1 ve 2, GA'ya uygulanarak 5'er adet bağımsız test çalıştırılıp karşılaştırma yapılmıştır. Önerilen uygulama Python dili kullanılarak geliştirilmiştir. Uygulama testi için i7 işlemci teknolojisine, 2.6 GHz Turbo işlemci hızına ve 8 GB ram kapasitesine sahip bir bilgisayar kullanılmıştır. Uygulama için gerekli parametreler; popülasyon büyüklüğü, jenerasyon sayısı, maksimum araç sayısı, maksimum araç kapasitesi ve çaprazlama olasılığı ile mutasyon olasılığı parametreleri Tablo 4'te verilmiştir. Olasılık parametreleri literatürde kullanılan değerler temel alınarak belirlenmiştir.

Tablo 4. *Algoritma parametreleri*

PopülasyonBüyüklüğü	200	**JenerasyonSayısı**	100
$\mathbf{P_{çaprazlama}}$	0.7	$\mathbf{P_{mutasyon}}$	0.1
Mak. Araç Sayısı	25	**Mak. Kapasite**	200

Tablo 5'te, kullanılan veri setindeki her problem için, algoritma 1 ve 2'yi kullanan GA'ların kat edilen toplam yola göre elde edilen en iyi çözümlerinin karşılaştırılması yapılmıştır. Bu karşılaştırmada ölçüt olarak en iyi çözümlerin toplam yol ve araç sayısı değerleri kullanılmıştır. Tablo 5'e göre algoritma 2'yi kullanan GA, algoritma 1'i kullanana göre bir araç daha az çözüm üretmeyi başarmıştır. Toplam yol açısından algoritma 2'yi kullanılan GA ile C1 setinde %33'e, R1 setinde %18'e ve RC1 setinde %14'e kadar daha iyi sonuçlara ulaşılmıştır. Ayrıca RC102 probleminin literatürdeki en iyi sonucu olan 1554,75 birim toplam yol sonucu algoritma 2'yi kullanan GA ile aşılmıştır. Ancak literatürdeki bu sonuç 12 adet araç ile bulunmuştur. Toplam yol sonuçlarının genel toplamına göre algoritma 2'yi kullanan GA diğerine göre daha iyi performansa sahiptir.

Tablo 5. *Algoritma 1 ve 2'yi kullanan GA'ların kat edilen toplam yola göre elde edilen en iyi çözümlerinin karşılaştırılması*

Veri Seti	**Algoritma 1'i kullanan GA**		**Algoritma 2'yi kullanan GA**		**Karşılaştırma Oranı**	
	Toplam Yol	**Araç sayısı**	**Toplam Yol**	**Araç sayısı**	**Toplam Yol**	**Araç sayısı**
C101	1168,052	12	892,6307	11	23,58%	8,33%
C102	1146,565	11	954,622	11	16,74%	0,00%
C103	1315,369	12	1079,744	11	17,91%	8,33%
C104	1377,93	11	1187,427	11	13,83%	0,00%
C105	1260,394	12	836,7574	10	33,61%	16,67%
R101	2180,591	22	1768,728	20	18,89%	9,09%
R102	1783,374	17	1668,935	19	6,42%	-11,76%
R103	1528,757	15	1486,471	16	2,77%	-6,67%
R104	1151,352	11	1202,48	13	-4,44%	-18,18%
R105	1720,662	17	1555,377	16	9,61%	5,88%
RC101	1890,23	17	1823,87	17	3,51%	0,00%
RC102	1800,582	15	**1541,996**	**14**	14,36%	6,67%
RC103	1536,823	13	1516,749	14	1,31%	-7,69%
RC104	1340,015	11	1295,02	12	3,36%	-9,09%
RC105	1831,024	16	1780,302	16	2,77%	0,00%

Tablo 6 iki algoritmanın araç sayısı açısından elde edilen en iyi çözümlerinin karşılaştırmasını sunmaktadır. Tablo 6 yapısal ve sayısal olarak Tablo 5 ile oldukça benzerdir. Ancak algoritma 1'i kullanan GA C101 ve R105 problemlerinde, algoritma 2'yi kullanan GA C103 ve R104 problemlerinde daha az araç ile çözüm elde etmeyi başarmıştır. Bu tabloda bu sonuçlar koyu renkte belirtilmiştir. Toplam yol sonuçlarının genel toplamına göre algoritma 2'yi kullanan GA diğerine göre daha iyi performansa sahiptir.

Tablo 6. *Algoritma 1 ve 2'yi kullanan GA'ların araç sayısına göre elde edilen en iyi çözümlerinin karşılaştırılması*

Veri Seti	Algoritma 1'i kullanan GA		Algoritma 2'yi kullanan GA		Karşılaştırma Oranı	
	Toplam Yol	Araç sayısı	Toplam Yol	Araç sayısı	Toplam Yol	Araç sayısı
C101	**1201,514**	**11**	892,6307	11	25,71%	0,00%
C102	1146,565	11	954,622	11	16,74%	0,00%
C103	1315,369	12	**1094,279**	**10**	16,81%	16,67%
C104	1377,93	11	1187,427	11	13,83%	0,00%
C105	1260,394	12	836,7574	10	33,61%	16,67%
R101	2180,591	22	1768,728	20	18,89%	9,09%
R102	1783,374	17	1668,935	19	6,42%	-11,76%
R103	1528,757	15	1486,471	16	2,77%	-6,67%
R104	1151,352	11	**1225,663**	**12**	-6,45%	-9,09%
R105	**1746,448**	**16**	1555,377	16	10,94%	0,00%
RC101	1890,23	17	1823,87	17	3,51%	0,00%
RC102	1800,582	15	1541,996	14	14,36%	6,67%
RC103	1536,823	13	1516,749	14	1,31%	-7,69%
RC104	1340,015	11	1295,02	12	3,36%	-9,09%
RC105	1831,024	16	1780,302	16	2,77%	0,00%

Algoritma 1 ve 2'yi kullanan GA'ların kat edilen toplam yola göre elde edilen en iyi sonuçları arasında istatistiksel olarak anlamlı bir farklılığın olduğunu kanıtlamak amacıyla Minitab paket programı kullanılarak Bağımlı Örneklem t-Testi uygulanmıştır. Algoritmalar aynı problemlere uygulandıkları için algoritma sonuçlarının bağımlı değişkenler oldukları kabul edilmiştir. Parametrik testlerde değişken normal dağılım göstermelidir. Bu nedenle algoritma sonuçları arasındaki fark değerlerine Normallik testleri yapılmış ve normal dağılıma uydukları kanıtlanmıştır. Şekil 3'te algoritma sonuçlarının Bağımlı Örneklem t-Testi analiz sonucu görülmektedir. Buna göre, %99 güven aralığında, algoritma 1'i kullanan GA sonuçları algoritma 2'yi kullanan GA sonuçlarına göre ileri düzeyde yüksek bulunmuştur.

Paired T-Test and CI: Algoritma1+GA toplam yol; Algoritma2+GA toplam yol

```
Paired T for Algoritma1+GA toplam yol - Algoritma2+GA toplam yol

                          N    Mean  StDev  SE Mean
Algoritma1+GA toplam yol  15  1535,4  318,1     82,1
Algoritma2+GA toplam yol  15  1372,7  333,6     86,1
Difference                15   162,7  140,8     36,4


99% lower bound for mean difference: 67,3
T-Test of mean difference = 0 (vs > 0): T-Value = 4,48  P-Value = 0,000
```

Şekil 3. *Minitab çıktısı*

6. SONUÇ (CONCLUSION)

ZPARP belirli zaman aralıklarında hizmet görmesi gereken müşterilere hizmet veren araçların bir depodan başlayan ve orada biten rotalarının çizilmesi problemidir. Bu çalışmada toplam yolu ve araç sayısını minimize etmek amaç fonksiyonları olarak belirlenmiş ve ZPARP için çok amaçlı GA çözümü önerilmiştir. GA'da bireylerin değerlendirilmesi, sıralanması ve seçilmesi işlemlerinde çok amaçlı çözüm yöntemlerinden NSGA-II kullanılmıştır. Önerilen GA başlangıç popülasyonu oluşturma aşamasında yapısal sezgisel yöntemlerden biri olan süpürme algoritması kullanılarak melezleştirilmiştir. En yakın komşu algoritması temel alınarak oluşturulan algoritma 1'i kullanan GA ve süpürme algoritması temel

alınarak oluşturulan algoritma 2'yi kullanan GA literatürdeki bir veri problemine uygulanmış ve sonuçları karşılaştırılmıştır. Algoritma 2'yi kullanan GA ile toplam yola göre C1 setinde %33'e, R1 setinde %18'e ve RC1 setinde %14'e kadar daha iyi sonuçlara ulaşılmıştır. Araç sayısı kriterine göre sonuçlar karşılaştırıldığında anlamlı bir fark görülmemekle beraber, algoritma 2'yi kullanan GA daha az araç sayısıyla çözüme ulaşmıştır. Süpürme algoritmasıyla melezleştirilen GA Bağımlı Örneklem t-Testi analizi sonuçlarına göre daha iyi performansa sahiptir.

Gelecek çalışmalarda bu çalışmada kullanılan çaprazlama ve mutasyon operatörleri geliştirilerek ve daha etkili parametreler belirlenerek test problemleri çözülüp literatürde bulunan sonuçlarla karşılaştırılabilir veya farklı amaç fonksiyonları belirlenerek alternatif bir çalışma yapılabilir.

KAYNAKLAR (REFERENCES)

[1] J. K. Lenstra, A. H. G. Kan, Complexity of vehicle routing and scheduling problems. Networks, 11:2 (1981) 221-227.

[2] S. Çolak, H. Güler, Dağıtım rotaları optimizasyonu için meta sezgisel bir yaklaşım. Gazi Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi, 11 (2009) 171-189.

[3] Y. Şahin, A. Eroğlu, Kapasite kısıtlı araç rotalama problemi için metasezgisel yöntemler: Bilimsel yazın taraması. Süleyman Demirel Üniversitesi İ.İ.B.F. Dergisi, 19:4 (2014) 337-355.

[4] G. Laporte, The vehicle routing problem: An overview of exact and approximate algorithms. European journal of operational research, 59:3 (1992) 345-358.

[5] N. A. El-Sherbeny, Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods. Journal of King Saud University-Science, 22:3 (2010) 123-131.

[6] P. Toth, D. Vigo (Eds.), Vehicle routing: problems, methods, and applications, Society for Industrial and Applied Mathematics, 2014.

[7] B. M. Baker, M. A. Ayechew, A genetic algorithm for the vehicle routing problem. Computers & Operations Research, 30 (2003) 787-800.

[8] D. Mester, O. Braysy, Active-guided evolution strategies for large-scale capacitated vehicle routing problems. Computers & Operations Research, 34:10 (2007) 2964-2975.

[9] J. Berger, M. Barkaoui, A parallel hybrid genetic algorithm for the vehicle routing problem with time windows. Computers & Operations Research, 31:12 (2004) 2037-2053.

[10] S. R. Thangiah, K E. Nygard, P. L. Juell, Gideon: A Genetic Algorithm System for Vehicle Routing With Time Windows, 7th IEEE Conference on Artificial Intelligence Applications, Miami Beach-Florida, 322–325, 24-28 Şubat, 1991.

[11] M. F. Ibrahim, I. Masudin, T. E. Saputro, A hybrid genetic algorithm implementation for vehicle routing problem with time windows. Jurnal Ilmiah Teknik Industri, 14:2 (2016) 196-204.

[12] C. Prins, A simple and effective evolutionary algorithm for the vehicle routing problem. Computers & Operations Research, 31:12 (2004) 1985-2002.

[13] Y. Chang, L. Chen, Solve the vehicle routing problem with time windows via a genetic algorithm. Discrete and continuous dynamical systems supplement, (2007) 240-249.

[14] B. Ombuki, B. J. Ross, F. Hanshar, Multi-objective genetic algorithms for vehicle routing problem with time windows. Applied Intelligence, 24:1 (2006) 17-30.

[15] S. A. Haddadene, N. Labadie, C. Prodhon, NSGAII enhanced with a local search for the vehicle routing problem with time windows and synchronization constraints. IFAC-PapersOnLine, 49:12 (2016) 1198-1203.

[16] K. C. Tan, Y. H. Chew, L. H. Lee, A hybrid multiobjective evolutionary algorithm for solving vehicle routing problem with time windows. Computational Optimization and Applications, 34:1 (2006) 115-151.

[17] A. Mungwattana, T. Manisri, K. Charoenpol, G. K. Janssens, A solution for the bi-objective vehicle routing problem with the windows using local search and genetic algorithms. International Journal for Traffic and Transport Engineering, 6:2 (2016) 149-158.

[18] K. Ghoseiri, S. F. Ghannadpour, Multi-objective vehicle routing problem with time windows using goal programming and genetic algorithm. Applied Soft Computing, 10:4 (2010) 1096-1107.

[19] A. Garcia-Najera, J. A. Bullinaria, An improved multi-objective evolutionary algorithm for the vehicle routing problem with time windows. Computers & Operations Research, 38:1 (2011) 287-300.

[20] W. K. Ho, J. C. Ang, A. Lim, A hybrid search algorithm for the vehicle routing problem with time windows. Int. J. Artif. Intell. Tools, 10:3 (2001) 431-449.

[21] M. M. Solomon, J. Desrosiers, Survey Paper—Time Window Constrained Routing and Scheduling Problems. Transportation Science 22:1 (1988) 1-13.

[22] B. E. Gillett, L. R. Miller, A heuristic algorithm for the vehicle dispatch problem. Oper. Res., 22 (1974) 340-349.

[23] U. Tül, A. Tuncer, Genetik Algoritma ile Akıllı Test Sayfası Oluşturma. GU J Sci, Part C, 5:4 (2017) 27-34.

[24] F. Kılıç, M. Gök, A Benchmark Proposal for Route-Planning Of Urban Bus Service, 4th Eastern European Regional Conference on the Engineering of Computer Based Systems, Bruno-Çek Cumhuriyeti, 134-137, 27-28 Ağustos, 2015.

[25] K. Deb, A. Pratap, S. Agarwal, T. A. M. T. A. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput., 6:2 (2002) 182-197.

[26] K. Deb, Multi-Objective Optimization Using Evolutionary Algorithms. Vol. 16, John Wiley & Sons, 2001.

[27] F. A. Fortin, D. Rainville, M. A. G. Gardner, M. Parizeau, C. Gagné, DEAP: Evolutionary algorithms made easy. The Journal of Machine Learning Research, 13:1 (2012) 2171-2175.

[28] A. Seshadri, A Fast Elitist Multiobjective Genetic Algorithm: NSGA-II. MATLAB Central, 182, 2006.

[29] M. M. Solomon, Algorithms for the vehicle routing and scheduling problems with time window constraints. Oper. Res., 35:2 (1987) 254-265.