MARMARA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ARAÇ ROTALAMA PROBLEMLERİ İÇİN KÜMELEME ALGORİTMALARI İLE VERİ İŞLEME

KEREM BÜYÜKÖZDEMİR

YÜKSEK LİSANS TEZİ
Bilgisayar Mühendisliği
Anabilim Dalı
Bilgisayar Mühendisliği Programı

DANIŞMAN
Dr. Öğr. Üyesi Anıl BAŞ

EŞ-DANIŞMAN
Doç. Dr. Kazım YILDIZ

İSTANBUL, 2023

MARMARA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ARAÇ ROTALAMA PROBLEMLERİ İÇİN KÜMELEME ALGORİTMALARI İLE VERİ İŞLEME

KEREM BÜYÜKÖZDEMİR

523619005

YÜKSEK LİSANS TEZİ
Bilgisayar Mühendisliği
Anabilim Dalı
Bilgisayar Mühendisliği Programı

DANIŞMAN
Dr. Öğr. Üyesi Anıl BAŞ

EŞ-DANIŞMAN
Doç. Dr. Kazım YILDIZ

İSTANBUL, 2023

ÖNSÖZ

Bu çalışmada, bana farklı fikirler sunarak, çalışmamı daha zengin kılmama vesile olan çok değerli danışman hocam sayın Dr. Öğr. Üyesi Anıl Baş'a, pratik tecrübelerini sonuna kadar benimle paylaşan, çalışmamla alakalı veya bunların dışındaki herhangi bir konuda bana desteklerini hiçbir zaman esirgemeyen sayın Doç. Dr. Kazım Yıldız'a, çalışmalarıma fikir olarak desteklerinden dolayı çalışma arkadaşım sayın Kadir İnce'ye teşekkürlerimi ve saygılarımı sunarım.

Son olarak FYL-2022-10457 numaralı projeye desteklerinden dolayı Bilimsel Araştırma Projeleri Birimi'ne Teşekkürlerimi sunarım.

Kerem BÜYÜKÖZDEMİR

İSTANBUL, 2023

İÇİNDEKİLER

ÖZET

ARAÇ ROTALAMA PROBLEMLERİ İÇİN KÜMELEME ALGORİTMALARI İLE VERİ İŞLEME

Günümüzde, gerçekleşen lojistik tedarik operasyonlarında, ürünlerin kabul edilebilir bir zaman aralığında tedarik edilmesi büyük bir önem arz etmektedir. Rotaların zaman maliyeti açısından optimum olması, gerçekleştirilecek olan lojistik operasyonlarının başarışını doğrudan etkilemektedir. Teslimat noktalarına, en kısa ve en az maliyetli rotaların tasarlanması ve oluşturulmasına literatür içerisinde "Araç Rotalama Problemi" (VRP) adı verilmektedir.

Araç rotalama problemi ilk olarak Dantzig ve Ramser tarafından dile getirilmiştir. Sabit bir noktadan başlayarak, uğranması önceden planlanmış tüm noktalara en kısa zaman ve en düşük maliyet ile uğranmasını amaçlayan bir rotalama problemidir. Rotalama problemlerinin, talep miktarı, aracın kapasitesi, zaman aralığı, mevcut araç sayısı, rota uzunluğu gibi kısıtına ve kapsamına göre birçok çeşidi bulunmaktadır. Araç rotalama problemi günümüzde popüler optimizasyon problemleri arasında yer almaktadırlar.

Araç rotalama problemleri için, genel geçer, optimum çözümü bulacak algoritmanın geliştirilmesi amacıyla birçok çalışma yapılmıştır. Ancak tüm araç rotalama problemlerine uyum sağlayabilecek ve en optimum çözümü vermeyi garanti edecek bir matematiksel model geliştirilememiştir. Bunun nedeni araç rotalama problemlerinin karmaşık yapılarıdır. Ayrıca problemlerin kendilerine özgü kısıtları olması nedeniyle, genel bir model oluşturma çabası, günümüze kadar olumlu sonuçlar ortaya koyamamıştır.

Araç rotalama problemlerinin en optimum çözümlerinin bulunması, çözüm uzayının tamamının taranması ve çözümlenmesi ile bulunabilir. Lakin uğranılması gereken noktaların ve kısıtların sayıca fazla olduğu durumlarla araç rotalama problemlerinde sıkça karşılaşılmaktadır. Bu durumlarda en optimum çözümün bulunması makul, kabul edilebilir, zamanlar içerisinde mümkün olamamaktadır. Bu nedenle araç rotalama problemlerine çözüm arayışı aşamasında, sezgisel veya meta-sezgisel yöntemlere başvurulmaktadır.

Sezgisel yöntemler, kısa ve makul hesaplama süresi içerisinde optimum duruma yakın, kaliteli çözümler üreten optimizasyon teknikleridir. Sezgisel algoritmalardan biri olan tasarruf algoritması, her noktanın başlangıç noktasına olan uzaklıkları ile ilişkili olarak rotayı minimize, dolaşılan düğümleri maksimize etmeyi amaçlamaktadır. Tasarruf algoritması çok kısıtlı problemlerde çalışma prensibi ve son seçilen düğüm üzerine ekleme şeklinde iteratif ilerlemesi sebebi ile, çözüm uzayının belirli noktalarına yönelterek kümülatif çözümlemeyi zorlaştırmakta ve makul sonuçlar ortaya koymayı zorlaştırmaktadır. Ancak mevcut noktalar üzerinde, bir gruplama veya kümeleme algoritmasının çalıştırılması, benzer özellikleri taşıyan noktaların bir araya gelmesine sağlayarak, çözüm performansının artırılması ve rotalama maliyetlerinin azalmasına olumlu bir etkide bulunacaktır.

Bu tez çalışmasında, veri setinde önceden belirlenmiş noktalar için, rotalama algoritmalarının çalıştırılmasına ön hazırlık yapılmaktadır. Tanımlanmış noktalar için veri seti içerisindeki tekrar eden noktalar çıkarılmıştır. Elde edilen noktalar için ise, tüm olası ikililer için, uzaklıklar, gerçek hayat verisi üzerinden hesaplanmış ve uzaklıklar matrisi oluşturulmuştur. İlk adım, önerilecek olan algoritmanın çalışması için hazırlık kapsamındadır. Çalışmanın ikinci adımında ise, sezgisel veya meta-sezgisel algoritmalar kullanılarak, araç rotalama problemine optimum ya da optimuma yakın makul kabul edilebilecek seviyede bir çözüm geliştirilmesi amaçlanmaktadır. Çalışmada kullanılacak olan sezgisel algoritma, k-ortalamalar ve bulanık c-ortalamalar gibi kümeleme algoritmalarını bir ön işleme adımı olarak kullanmaktadır. Buradaki temel yaklaşım, algoritma öncesi benzer noktaların bir araya getirilip çözüm performansının artırılmasıdır.

Bu tez çalışmasının sonuçlarına göre, kümeleme algoritmalarının ön işleme adımı olarak kullanılması, sezgisel algoritmanın kümeleme performansını artırmıştır. K-ortalamalar kümeleme algoritması ile birlikte kullanımda, toplam rota uzunluğunda %28'lik bir iyileşme gözlenmektedir. Bulanık C-ortalamalar algoritması ile kullanımda ise, %32'lik bir iyileşme oranı yakalanmıştır.

ABSTRACT

DATA PROCESSING WITH CLUSTERING ALGORITHMS FOR VEHICLE ROUTING PROBLEMS

In logistics supply operations, it is of great importance that the products are supplied at an acceptable time interval. The fact that the routes are optimal in terms of time cost directly affects the success of the logistics operations to be carried out. The design and creation of the shortest and least costly routes to the delivery points is referred to as the "Vehicle Routing Problem" (VRP) in the literature.

The vehicle routing problem was first mentioned by Dantzig and Ramser. It is a routing problem that aims to reach all previously planned points, starting from a fixed point, with the shortest time and lowest cost. There are many types of routing problems depending on the constraint and scope such as the amount of demand, the capacity of the vehicle, the time interval, the number of available vehicles, the length of the route. VRP is one of the popular optimization problems today.

For vehicle routing problems, many studies have been carried out to develop an algorithm that will find the general and optimum solution. However, a mathematical model that can adapt to all vehicle routing problems and guarantee the optimum solution has not been developed. This is because of the complex nature of vehicle routing problems. In addition, since each problem has its own limitations, the effort to create a general model has not yielded positive results until today.

Finding the most optimal solutions for vehicle routing problems can be found by scanning and solving the entire solution space. However, vehicle routing problems are frequently encountered in situations where the number of points and constraints to be visited are high. In these cases, it is not possible to find the optimum solution in a reasonable, acceptable time. For this reason, heuristic or meta-heuristic methods are used in the search for solutions to vehicle routing problems.

Heuristics are optimization techniques that produce close-to-optimal, quality solutions within a short and reasonable computation time.The savings algorithm, one of the heuristic algorithms, aims to minimize the route and maximize the nodes circulated in relation to the distances of each point from the starting point. The saving algorithm complicates the cumulative analysis by directing it to certain points of the solution space and makes it difficult to produce reasonable results due to its working principle in very constrained problems and its iterative progress in the form of addition on the last selected node. However, running a grouping or clustering algorithm on existing points will have a positive effect on increasing solution performance and reducing routing costs by bringing together points with similar characteristics.

In this thesis, preliminary preparation is made for running routing algorithms for predetermined points in the data set. For the defined points, the repeating points in the data set were removed. For the obtained points, for all possible pairs, the distances were calculated over the real-life data and the distance matrix was created.

The first step is in preparation for the proposed algorithm to work. In the second step of the study, it is aimed to develop an optimum or near-optimally reasonable solution to the vehicle routing problem by using heuristic or meta-heuristic algorithms. The heuristic algorithm to be used in the study uses clustering algorithms such as k-means and fuzzy c-means as a preprocessing step. The basic approach here is to bring together similar points before the algorithm and increase the solution performance.

According to this thesis results, the use of clustering algorithms as a preprocessing step has increased the clustering performance of the heuristic algorithm. When used together with the K-means clustering algorithm, a 28% improvement in total route length is observed. When using the fuzzy C-means algorithm, an improvement rate of 32% has been achieved.

SEMBOLLER

s	:	Saniye
m	:	Metre

KISALTMALAR

VRP	: Vehicle Routing Problem (Araç Rotalama Problemi)
CVRP	: Capacitated Vehicle Routing Problem (Kapasite Kısıtlı Araç Rotalama Problemi)
CVRP-SD	: Single Depot Capacitated Vehicle Routing Problem (Tek Depolu Kapasite Kısıtlı Araç Rotalama Problemi)
CVRP-MD	: Multi Depot Capacitated Vehicle Routing Problem (Çok Depolu Kapasite Kısıtlı Araç Rotalama Problemi)
TWVRP	: Time Window Vehicle Routing Problem (Zaman Pencereli Araç Rotalama Problemi)
OVRP	: Open Vehicle Routing Problem (Açık Uçlu Araç Rotalama Problemi)
HVRP	: Heterogeneous Vehicle Routing Problem (Heterojen Araç Rotalama Problemi)
DVRP	: Dynamic Vehicle Routing Problem (Dinamik Araç Rotalama Problemi)
FCM	: Fuzzy C-Means (Bulanık C-Ortalamalar)
GPS	: Global Positioning System (Küresel Konumlama Sistemi)
KML	: Keyhole Markup Language (Anahtar Deliği Biçimlendirme Dili)
OSRM	: Open Source Route Machine (Açık Kaynaklı Rotalama Makinesi)
WPF	: Windows Presentation Foundation (Windows Sunum Yapı Temeli)
JSON	: JavaScript Object Notation (JavaScript Obje Notasyonu)
CSV	: Comma-Separated Values (Virgülle Ayrılmış Değerler)
ARI	: Adjusted Rand Index (Düzeltilmiş Rand İndeksi)
SI	: Silhouette Index (Silhouette İndeksi)

ŞEKİL LİSTESİ

TABLO LİSTESİ

1. GİRİŞ

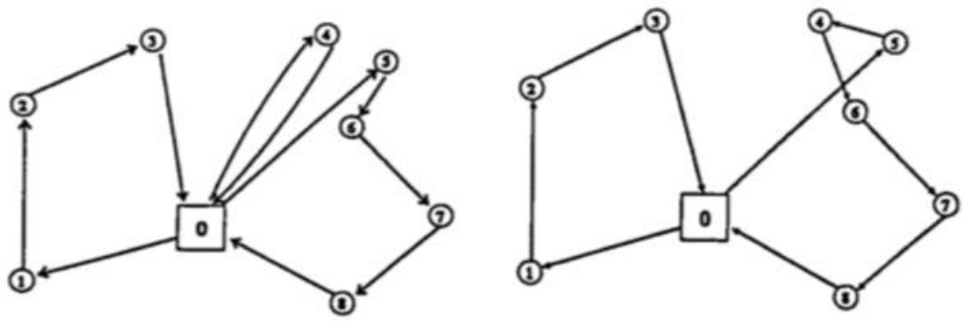
Araç rotalama problemleri, kısıtları ve karmaşıklıkları nedeniyle hala günümüzde çözümü zor problemler arasında yer almaktadır. Rotalamanın doğası gereği uygun çözümün makul süre içerisinde bulunması gerekmektedir. Çünkü rotanın hayata geçirileceği bir zaman aralığı bulunmaktadır. Ancak problemin karmaşıklığının nokta sayısı ve kısıtlara göre artması nedeniyle tüm çözüm uzayının taranıp en optimum çözümün bulunması makul süre içerisinde gerçekleşmeyebilmektedir. Bu gibi durumlara çözüm olması amacıyla, optimum çözüm olmayı garanti etmiyorsa da kaliteli bir çözüm olduğuna inanılan bir sonuç bulabilme amacıyla hareket edilmektedir. Bu durum, sezgisel ve meta-sezgisel algoritmaların rotalama problemlerine çözüm algoritmaları olarak kullanıldığı senaryoları ortaya çıkarmaktadır.

Bu çalışmasında ise yukarıda belirtilen bilgiler ve kapsam ışığında, benzer özellikleri gösteren noktaların algoritma öncesinde kümelenmesi yaklaşımı benimsenmektedir. Bu sayede tasarruf algoritmasının açgözlü yaklaşımından ortaya çıkacak dezavantajların azaltılması amaçlanmaktadır. Varış noktalarının kümelenmesinde literatürde yer alan k-ortalamalar kümeleme algoritması ve bu algoritmaya bulanık bir yaklaşım olan bulanık c-ortalamalar kümeleme algoritmalarının kullanılması öngörülmektedir.

1.1. Araç Rotalama Problemi

Araç rotalama problemi, ilk olarak 1959 yılında Dantzig ve Ramser tarafından literatüre girmiştir. Çalışmalarında benzin istasyonlarına benzin dağıtımı problemini incelemişler ve problemin çözümü için matematiksel model geliştirmişlerdir. Sonrasında 1964 yılında Clarke ve Wright probleme sezgisel bir bakış açısı getirmiş ve ardından sonra araç rotalama problemleri gelişmeye başlamıştır. Araç rotalama problemi, şu ana kadar en çok gelişme gösteren optimizasyon problemlerindendir [3].

Araç rotalama problemi, bir aracın belirli bir başlangıç noktasından belirli bir hedefe ulaşması için en kısa yolu bulmayı amaçlayan problem türüdür. Bu problem, yolculuk süresi, yolculuk maliyeti veya diğer faktörler dahil olmak üzere birçok değişkeni dikkate alarak çözülür. Çözüm algoritmaları genellikle herhangi bir yolun maliyetini temsil eden bir değer üzerinden yolu değerlendirir. Bu değer, genellikle yolun uzunluğu, zamanı veya maliyeti gibi faktörleri içerebilir. Teslimat sistemlerinde, seyahat planlama sistemlerinde veya çevrimiçi rota planlama hizmetlerinde sıkça kullanılmaktadır.
Araç rotalama problemi, lojistik dalındaki, dağıtım alanında yer alan önemli bir problemdir. Klasik araç rotalama problemi bir başlangıç noktası (orijin) barındırır, farklı veya aynı taşıma kapasitesine sahip araçlarla, müşterilere, ürünlerin zamanında taşınması sağlanır. Operasyonun amaç fonksiyonu yani en aza indirgenmeye çalışılan fenomen, araçların oluşturdukları rotaların uzaklıklarıdır [4].



Şekil 1.1 ARP Modeli Örneği [5]

1.1.1. Klasik Araç Rotalama Modelinin Oluşturulması

Klasik bir araç rotalama modeli probleminde, her araç müşterilere sadece 1 kere uğramaktadır. Tek depolu klasik bir araç rotalama problemini şu şekilde modellenmektedir [3],

M: Araç sayısı
N: Müşteri sayısı
D_{ij}: Nokta i ve nokta j arasındaki mesafe
Q_i: Müşteri i'nin talep miktarını temsil etmektedir.

$$X_{ij} : \begin{cases} 1, & eğer\ k\ nolu\ araç\ i\ noktasından\ j\ noktasına\ giderse \\ 0, & eğer\ gitmezse \end{cases} \tag{1.1}$$

$$MinZ = \sum_{i=0}^{N} \sum_{j=0,(j \neq 1)}^{N} \sum_{k=1}^{M} d_{ij}\ x_{ij} \tag{1.2}$$

(1.2) numaralı denklem amaç fonksiyonunu ifade etmektedir. Maliyetlerin en aza indirilmesi adına, amaç fonksiyonunun minimum olması gerekmektedir. Klasik bir depolu araç rotalama probleminin kısıtları ifade edersek,

$$i = 0\ için, \quad \sum_{k=1}^{M} \sum_{j=1}^{N} x_{ijk} = M \tag{1.3}$$

(1.3) numaralı denklem içerisindeki M sayısı araç sayısını ifade etmektedir. Bazı problemlerde araçların hepsi rotalama problemine dahil edilmeyebilir ya da rotalama sonucu bir rota oluşmayabilmektedir. Bu durum için eşitlik ifadesi yerine (≤) işareti bulunacaktır.

$$i \in \{1, \ldots, N\}\ için, \quad \sum_{k=1}^{M} \sum_{j=0,(j \neq 1)}^{N} x_{ijk} = 1 \tag{1.4}$$

(1.4) numaralı denklem i noktasından çıkacak olan toplam araç sayısının 1 olması gerektiği,

$$j \in \{1,\ldots,N\}\ için, \quad \sum_{k=1}^{M} \sum_{j=0,(j\neq 1)}^{N} x_{ijk} = 1 \tag{1.5}$$

(1.5) numaralı denklem j noktasından gelecek olan toplam araç sayısının 1 olması gerektiği belirtmektedir.

(1.4) ve (1.5) numaralı denklemler, klasik araç rotalama probleminin karakteristiğini ortaya koymaktadır. Bir talep noktasından, sadece bir araç ve sadece bir defa geçebilir kısıtını açıklamaktadır.

$$k \in \{1,\ldots,M\}\ için, \quad \sum_{k=1}^{M} x_{i0k} \leq 1 \tag{1.6}$$

(1.6) numaralı denklem bir aracın sadece bir rota üzerinde kullanılabileceği veya kullanılmayacağını belirtmektedir.

$$k \in \{1,\ldots,M\}\ için, \quad \sum_{i=1}^{N} Q_i \sum_{j=0,(j\neq 1)}^{N} x_{ijk} \leq C \tag{1.7}$$

(1.7) numaralı denklem araçlar için önerilen rota içerisinde talep toplamının, araç kapasitesi aşmaması gerektiğini belirtmektedir.

Araç rotalama problemleri, birçok farklı varyasyona sahiptir. Bu farklı türler problemin özelliklerine göre değişim göstermektedir. Temel olarak türleri aşağıda sıralandığı gibidir.

1.1.2. Kapasite Kısıtlı VRP (CVRP):

Kapasite kısıtlı araç rotalama probleminde, her araç sabit bir kapasiteye sahiptir. Araçların müşterileri en kısa rota ile ziyaret etmesinde kapasite kısıtını da sağlaması gerekmektedir. Bir aracın ziyaret ettiği tüm müşterilerin toplam talebi, aracın kapasitesini aşmamalıdır. Kapasite kısıtlı araç rotalama problemi, Tek Depo (CVRP-SD) ve birden çok depo (CVRP-MD) olarak iki alt türe ayrılabilir. CVRP-SD'de, araçlar rotalarını aynı depoda başlatıp bitirirler, CVRP-MD'de ise araçlar rotalarını birden fazla depodan başlatıp bitirebilir.

1.1.2.1. Tek Depolu Kapasite Kısıtlı VRP (CVRP-SD):

Tek depolu kapasite kısıtlı araç rotalama problemi (CVRP-SD), araçların tüm rotalarının aynı depoda başlatıp bitirmelerini gerektiren kapasite kısıtlı (CVRP) araç rotalama türüdür. CVRP-SD, NP-zor bir optimizasyon problemi olarak tanımlanmaktadır. Amaç fonksiyonu, tüm müşterilerin taleplerinin, toplam mesafeyi minimize edecek şekilde tek depo üzerinden karşılanmasıdır.

CVRP-SD'nin çözümü için literatürde birçok yöntem kullanılmaktadır. Genetik algoritma, benzetimli tavlama, tabu arama, karınca kolonisi algoritması gibi meta-sezgiseller kullanmaktır. Bu yöntemler, optimum sonucu garanti etmese de genellikle pratik amaçlar için yeterli doğrulukta çözümler sağlamaktadır.

CVRP-SD, nakliye, lojistik, dağıtım ve teslim hizmetleri gibi birçok gerçek dünya uygulamasında karşılaşılan yaygın bir optimizasyon problemidir. CVRP-SD zaman pencereli, çoklu depo gibi çeşitli varyasyonlar ile genişletilebilir.

1.1.2.2. Çok Depolu Kapasite Kısıtlı VRP (CVRP-MD):

Çok depolu kapasite kısıtlı araç rotalama problemi (CVRP-MD), birden fazla deponun bulunduğu sistemde, araçların birden fazla noktadan harekete başlayıp bitirebildiği; müşterilerin taleplerinin kapasite kısıtları dikkate alınarak oluşturulan en uygun rotalar ile karşılanmasını amaçlayan bir araç rotalama problemi (VRP) varyasyonudur.

CVRP-MD, NP-zor bir optimizasyon problemi olarak tanımlanmaktadır. Amaç fonksiyonu, tüm müşterilerin taleplerinin, toplam mesafeyi minimize edecek şekilde birden fazla depo üzerinden karşılanmasıdır.

CVRP-MD'nin çözümü için literatürde birçok yöntem kullanılmaktadır. Bunlar genel araç rotalama probleminin çözümünde kullanılan algoritmalar ile benzerlik göstermektedir. Genetik algoritma, benzetimli tavlama, tabu arama, karınca kolonisi algoritması gibi meta-sezgiseller kullanmaktır.

CVRP-MD'nin kullanımı özellikle farklı depolardan araçların harekete geçirilmesinin maliyetinin farklı olduğu ve hangi rota için hangi deponun kullanılması kararının hem maliyeti hem de kapasitesini etkilediği durumlarda kullanışlıdır.

1.1.3. Zaman Pencereli VRP (TWVRP):

Zaman Penceresi Araç Rotalama Problemi (TWVRP), her müşterinin ziyaret edilebileceği belirli bir zaman aralığı olan bir araç rotalama problemi (VRP) varyasyonudur. TWVRP, NP-zor bir optimizasyon problemi olarak tanımlanır ve genellikle en optimum sonucu bulmak zordur. Amaç, verilen zaman pencereleri içinde müşterileri ziyaret etme tüm araçlar tarafından kat edilen toplam mesafeyi (veya maliyeti) minimize etmektir.

TWVRP, diğer araç rotalama problemleri gibi gerçek dünya uygulamalarında sıklıkla karşılaşılan bir optimizasyon problemidir. TWVRP, ayrıca araç kapasitesi, çoklu depo ve farklı araç türleri gibi ek kısıtları içerebilir. Bu durum, problemi çözmeyi için daha zor hale getirmektedir ancak problemin gerçek dünyaya olan benzerliğini artırmaktadır. TWVRP'nin çözümünde diğer VRP çözümlerine benzer çözüm yöntemleri kullanılmaktadır. Genetik algoritma, benzetimli tavlama, tabu arama, karınca kolonisi algoritması gibi meta-sezgiseller kullanmaktır.

TWVRP, araç hızı ve trafik koşullarını da dikkate alabilmesidir. Araç hızı ve trafik koşulları, müşteriler ve depolar arasındaki yolculuk süresini etkileyebilmektedir ve bu çevresel bilgiler çözümlerin kalitesini arttırmak için kullanılabilmektedir.

1.1.4. Açık Uçlu VRP (OVRP):

Açık araç rotalama problemi (OVRP), müşterilere hizmet veren aracın tekrar başlangıç noktasına dönmesini zorunlu kılmayan araç rotalama problemidir. OVRP'de amaç, verilen müşteri kümesini en iyi şekilde hizmet etmek için gerekli olan en uygun rota ve araç sayısını belirlemektir. Bu problemde, araç kapasiteleri, zaman pencereleri ve diğer kısıtlamalar gibi faktörler de dikkate alınır. OVRP, NP-zor bir problem olduğu için lineer programlama, sezgisel veya meta-sezgisel yöntemlerle çözülür. Bu yaklaşım genellikle nakliye, lojistik ve tedarik zinciri yönetimi gibi alanlarda kullanılır.

1.1.5. Heterojen VRP (HVRP):

Heterojen araç rotalama problemi (HVRP), belirli bir yük miktarını belirli bir nokta serisine teslim etmek ya da toplamak için bir veya daha fazla araç kullanarak, en uygun rotaları bulmak için oluşturulmuş bir araç rotalama problemidir. HVRP, diğer araç rotalama problemlerinin aksine araçların çeşitliliğini de dikkate alır. HVRP, araçların kapasitelerinin, hızlarının veya teslimat yerlerine ilişkin kısıtlamaların farklı olmasına olanak tanımaktadır. HVRP de diğer araç rotalama problemleri gibi NP-Zor bir problemdir. HVRP genellikle lineer programlama, sezgisel ya da meta-sezgisel algoritmalar gibi matematiksel optimizasyon teknikleri kullanılarak çözülür.

1.1.6. Dinamik VRP (DVRP):

Dinamik araç rotalama problemi (DVRP), araç rotalama probleminin tanımına uyacak şekilde en kısa rotaları oluşturmaya amaçlamaktadır. Ancak, DVRP gerçek zamanlı olarak kısıtların değişmesinde adapte olabilmektedir. DVRP, kısıt parametrelerinin gerçek zamanlı olarak değişmesinin akabinde optimum çözümün güncellemesiyle standart araç rotalama problemlerinden ayrılır. Örneğin, bir teslimat kamyonu trafik nedeniyle gecikebilir, bu durumda DVRP yardımıyla rota yeniden hesaplanabilir ve gecikmenin en aza indirilmesi sağlanır. Ayrıca, DVRP, hava koşulları, yol kapatmaları ve talep değişiklikleri gibi diğer dinamik faktörleri de dikkate almaya olanak sağlamaktadır.

DVRP'nin ile rota maliyeti ve teslimat zamanlaması arasındaki denge kurulur. Rotaların maliyeti, yakıt tüketimi, geçiş ücretleri gibi faktörleri içerir. Teslimatların zamanlılığı ise, yükü zamanında teslim etme ihtiyacını ifade eder. DVRP, rotaların toplam maliyetini en aza indirirken teslimatların zamanında yapılmasını sağlayacak optimum çözüm bulmayı hedefler. DVRP genellikle matematiksel optimizasyon teknikleri, gerçek zamanlı izleme ve güncelleme sistemleri kullanılarak çözülür.

1.1.7. Araç Rotalama Problemleri İçin Çözüm Yöntemleri

Araç rotalama problemi, 1950'li yılların sonunda Dantzig ve Ramser tarafından ortaya atılmıştır [1]. Bu tarihten itibaren değişik çözüm yöntemleri ortaya atılmıştır. 1960'lı yıllarda, araç rotalama problemlerinin çözümü için, özellikle büyük veri setlerinin ortaya çıkmasının akabinde sezgisel yöntemlerin kullanılması üzerinde durulmuştur. 1980'li yıllarda yaklaşık 50 müşterili problemler için en optimum sonuç matematiksel programlama yaklaşımı ile çözülmeye başlanmıştır. Sonraki yıllardan itibaren özellikle problemlerin boyutunun büyümesinden dolayı çözüm için meta-sezgisel yöntemlerden faydalanılmaya başlanmıştır.

Tablo 1.1 Araç Rotalamanın Kısaca Tarihçesi [6]

1950'ler	VRP tam sayılı programlama olarak formüle edilmiş ve 10-20 müşterili küçük problemler çözülmüştür.
1960'lar	Rota kurma sezgiselleri sunulmuş ve 30-100 müşterili problemler çözülmüştür.
1970'ler	İki fazlı sezgiseller, interaktif (insan-makina) sezgiseller geliştirilmiş, yaklaşık 50 müşterili problemler optimal metotlarla çözülebilir hale gelmiştir.
1980'ler	Matematiksel programlama esaslı prosedürler literatüre sunulmuştur. Etkileşimli (interaktif: insan-makina) sezgiseller geliştirilmiştir. Optimal yöntemler kullanılarak yaklaşık 50 müşteriye sahip olan bazı problemler çözülmüştür.
1990'lar	Araç rotalama problemlerine meta-sezgiseller uygulanmıştır. 50 – 100 müşteriye sahip bazı problemler optimal olarak çözülmüştür.

1.1.7.1. Kesin Yöntemler

Araç rotalama probleminde, mevcut çözüm uzayının tüm noktalarının içerisindeki en optimum çözümü bulan yöntemler, en iyi çözümü garanti eden yöntemler, kesin yöntemler olarak isimlendirilmiştir. Kesin yöntemler problemin büyük bir ölçekli olması durumunda, çözüm zamanını oldukça uzatabilmektedir. Kesin yöntemler, matematiksel model temelli yöntemlerdir. Tam sayılı model kullanarak, matematiksel model ile çözüm getiren yöntemlere Dal-Sınır (Branch and Bound), Dal-Kesme (Branch and Cut) algoritmaları örnek verilebilir.

1.1.7.2. Sezgisel Yöntemler

Sezgisel algoritmalar, en iyi çözümü bulmayı garanti etmeyen ancak makul bir süre içerisinde kabul edilebilir çözümleri bulmayı amaçlayan algoritmalardır. Bu algoritmalar, gezgin satıcı problemi veya sırt çantası problemi gibi optimizasyon problemleri için sıklıkla kullanılırlar. Bu problemlerde en optimum çözümün bulunması için tüm çözüm uzayının taranması gerekmektedir. Ancak sezgisel algoritmalar sayesinde, makul süre içerisinde optimum çözüme yakın olduğu varsayılan kabul edilebilir çözümler üretilebilmektedir.

Sezgisel yöntemler başlangıç çözümlerinden etkilenilirler. Bu sebeple algoritma, bölgesel optimal çözümün, global optimum çözümden çok uzak olan bir noktada kalabilir. Bu nedenle global optimum çözüme yaklaşabilmek için sezgisel araştırma metotlarında basitlik ve genellik özellikleri korunacak şekilde bazı değişiklikler yapılabilir. Başlangıç çözümlerinin alternatiflerinin artırılması, geçmiş iterasyon özelinde, sürekli benzer alanda arama yapan algoritmanın cezalandırılması gibi yöntemlerde sezgisel optimizasyon yöntemlerinin bölgesel optimum noktalarından kurtulması sağlanabilir.

1.1.7.2.1. Tasarruf Algoritması

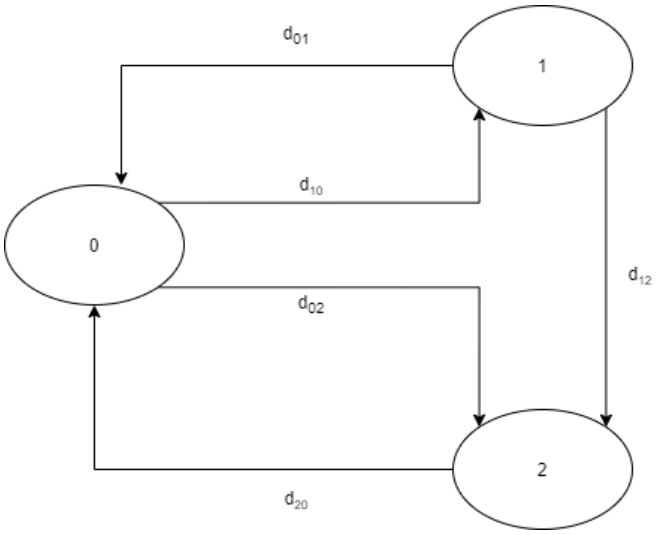
Clarke ve Wright tasarruf algoritması (CWSA), araç rotalama problemi (VRP) için kullanılan bir sezgisel algoritmadır. İlk olarak 1964 yılında J. Clarke ve J. Wright tarafından önerilmiştir [2].

Clarke ve Wright algoritmasının temel fikri, düğüm ikilileri oluşturup, bu iki noktanın bir araya gelmesiyle oluşacak olan yol tasarrufunun maksimize edilmesidir. Öncelikle tüm olası ikililer için tasarruf miktarları hesaplanır. Sonrasında, tüm ikililer tasarruf miktarına büyükten küçüğe sıralanır. Algoritma sonrasında en büyükten başlamak üzere iteratif olarak rotaları birleştirir.

Algoritmanın, paralel tasarruf algoritması ve sıralı tasarruf algoritması olmak üzere iki versiyonu vardır. Tasarrufların hesaplanması için ilk olarak;

$$s_{ij} = d_{i0} + d_{0j} - d_{ij} \tag{1.8}$$

(1.8) numaralı denklem her bir nokta ikilisi için hesaplanan tasarruf miktarını göstermektedir.



Şekil 1.2 Tasarruf Hesabında Kullanılan Yollar

Her bir ikili için, tasarruflar hesaplandıktan sonra, tüm tasarruflar büyükten küçüğe sıralanır,

- Adım 1: En büyük tasarrufu sağlayan ikili seçilip ilk rota oluşturulur.
- Adım 2: Daha önce gidilmemiş en yüksek tasarrufu sağlayan ikili seçilir.
- Adım 3: Eğer herhangi bir rotaya, rotanın başından ya da sonundan bağlanabiliyor ise bağlanır, bağlanamıyor ise yeni bir rota oluşturulur.
- Adım 4: Tüm noktalara uğranana kadar Adım 2 tekrar edilir.

1.2. Kümeleme Algoritmaları

Kümeleme, veri madenciliği ve makine öğreniminde kullanılan, benzer nesneleri birlikte gruplamayı, benzerlikleri az olan nesneleri ise uzaklaştırmayı amaçlayan bir tekniktir. Kümeleme algoritmaları, nesneleri benzerliklerine göre farklı gruplara ayırmak veya kümelemek için kullanılırlar. Kümelemenin ana amacı, benzer elemanları bir araya getirmektir.

Literatürde kümeleme algoritması olarak kullanılan birçok algoritma bulunaktadır. Popüler kümeleme algoritmaları arasında k-ortalamalar, hiyerarşik kümeleme, yoğunluk-tabanlı kümeleme ve model-tabanlı kümeleme bulunur.

K-ortalamalar, en sık kullanılan kümeleme algoritmalarından birisidir. K adet kümeye veriyi bölen bir merkez-tabanlı algoritmadır. Algoritma, her elemanı kendisine en yakın olan küme merkezine göre kümeler. K-ortalamalar yöntemi uygulaması basit ve hızlı olmasından dolayı akademide ve diğer sektörlerde sıkça kullanılmaktadır. K-ortalamalar metodunda başlangıç küme merkezlerinin seçimi tüm iterasyonları etkileyeceği için, başlangıç çözümü tüm çözüme etki etmektedir.

Hiyerarşik kümeleme, veriyi bir küme hiyerarşisi içinde gruplayan başka bir popüler algoritmadır. Algoritma her elemanın ayrı bir küme olduğu varsayımıyla başlar. Sonlandırma koşulu gerçeklesene kadar iteratif olarak kümeler birleştirilir veya bölünür. Hiyerarşik kümeleme, yoğun olarak dendrogramlar oluşturmak için kullanılmaktadır.

Yoğunluk-tabanlı kümeleme, belirli bir alan içindeki elemanların yoğunluğuna dayalı olarak veriyi gruplamayı baz alan kümeleme algoritmasıdır. Yoğunluk tabanlı kümelemede, elemanların yoğun olduğu alanları kümeler olarak tanımlanır. Yoğunluk-tabanlı kümeleme, düzensiz şekilli kümeleri tespit etmek, verideki aykırı değer ve çıkıntıları tanımlamak için kullanılmaktadır. Yoğunluk-tabanlı kümeleme algoritmaları için DBSCAN ve OPTICS örnek olarak verilebilir.

Model-tabanlı kümeleme, veri içerisindeki elemanların benzerlik veya farklılıklarını ortaya çıkarmak için istatistiksel bir model kullanan bir kümeleme algoritmasıdır. Algoritma veriye en uygun olan modeli bulur. Bu modeli kullanarak nesneleri farklı kümelere sınıflandırır.

Kümeleme algoritmaları geniş bir uygulama alanında kullanılmaktadırlar. Kümeleme, veri madenciliği ve makine öğrenimi için değerli bir araçtır. Kümeleme algoritmaları, görüntü ve belge analizi, müşteri segmentasyonu ve anomali tespiti gibi geniş bir uygulama alanında kullanılmaktadırlar. Kümeleme, sınıflandırma ve regresyon gibi diğer makine öğrenimi algoritmaları için ön işleme adımı olarak da kullanılabilir.

1.2.1. K-Ortalamalar Algoritması

K-ortalamalar algoritması, veri madenciliğinde küme analizi için kullanılan popüler bir yapay öğrenme algoritmasıdır. Algoritmanın amacı, veride benzer özelliklere sahip gruplar (kümeler) bulmaktır. Adındaki "k", algoritmanın veride oluşturacağı küme sayısını ifade etmektedir.

Algoritma ilk olarak kümelerin temsil edici noktaları olarak kullanılan k merkez noktalarını rastgele seçerek iterasyonları başlatır. Daha sonra, her bir düğüm, merkez noktası en yakın olan kümeye atanır, Her kümenin merkezi, o kümeye atanmış tüm düğümlerin ortalaması olarak yeniden hesaplanır. Bu işlem, küme atamaları artık değişmez ya da maksimum iterasyon sayısına ulaşılana kadar tekrar ettirilir.

K-ortalamalar algoritması matematiksel olarak bir optimizasyon problemi olarak ifade edilebilir, amaç fonksiyonu düğümler ile küme merkezleri arasındaki uzaklıkların minimize edilmesi şeklinde ifade edilebilir. Düğümler ile küme merkezlerinin arasındaki mesafelerin iç küme toplam karelerinin (WCSS) azaltılması amaçlanmaktadır. Optimizasyon problemi WCSS minimize edecek şekilde aşağıdaki gibi yazılabilir:

x: *düğüm*
μ_i: *küme i için merkez noktası*
C: *küme i'ye atanmış düğümler*
k: *küme sayısı*

$$WCSS = \sum_{i=1}^{k} \sum_{x \in Ci} || x - \mu i ||^2 \quad (1.8)$$

(1.8) numaralı denklem k-ortalamalar kümeleme algoritması için minimize edilmeye çalışılan amaç fonksiyonunu göstermektedir.

K-ortalamalar, birçok veri tipinde iyi çalışan basit ve verimli bir algoritmadır. Ancak bazı sınırlamaları bulunmaktadır. Ana sınırlamalardan biri, kümelerin küresel ve eşit büyüklükte olduğunu varsaymasıdır, bu gerçek dünya verilerinde her zaman gerçek olmayabilir. Ayrıca, algoritma başlangıçta merkez noktaların yerleştirilmesine duyarlıdır, bu nedenle başlangıçta merkez noktaları iyi seçilmez ise optimum çözümlere yakın çözümler ortaya çıkarmayabilir.

K-ortalamalar, görüntü işleme, veri kümeleme ve anomali tespiti gibi çeşitli uygulamalarda geniş bir şekilde kullanılmaktadır. Aynı zamanda boyut azaltma ve sınıflandırma gibi diğer yapay öğrenme algoritmaları için bir ön işleme adımı olarak da kullanılabilmektedir.

1.2.2. Bulanık C-Ortalamalar Algoritması

Bulanık C-Ortalamalar (FCM), veriyi kümelendirmek için kullanılan k-ortalamalar algoritmasının bir uzantısıdır. K-ortalamalara ek olarak, her bir elemanın tüm kümelerde bulunabilmesine izin verir. Bulanık C-Ortalamalar algoritması yapı itibariyle k-ortalamalara benzer, ancak her düğüm tek bir kümeye atlamak yerine, her bir düğüne her küme için bir üyelik derecesi atar.

Üyelik dereceleri, üyelik matrisi tarafından temsil edilir, matris içerisindeki her bir üyelik U_{ij} ile gösterilmektedir. U_{ij} düğüm olan x_i'nin küme j'deki üyelik derecesini temsil etmektedir.

Bulanık C-Ortalamalar algoritmasının amaç fonksiyonu aşağıdaki gibidir;

D: *düğüm sayısı*

N: *küme sayısı*

$X_{i:}$ *i'ninci düğüm*

C_j*:* *küme j'nin merkez noktası*

U_{ij}*:* *düğüm xi'nin küme j'de üyelik derecesi*

m: *bulanıklaştırma katsayısı (genellikle 2 alınır)*

$$J_m = \sum_{i=1}^{D} \sum_{j=1}^{N} U_{ij}^m \, ||x_i - c_j||^2 \tag{1.9}$$

(1.9) numaralı denklem bulanık c-ortalamalar kümeleme algoritması için minimize edilmeye çalışılan amaç fonksiyonunu göstermektedir.

Küme merkezlerinin her iterasyonda güncellenmesi gerekmektedir;

$$C_j = \frac{\sum_{i=1}^{D} U_{ij}^m \, x_i}{\sum_{i=1}^{D} U_{ij}^m} \tag{1.10}$$

(1.10) numaralı denklem her bir iterasyonda güncellenen küme merkezlerinin hesaplanmasını göstermektedir.

Her bir iterasyon sonrası üyelik katsayısı şöyle hesaplanır;

$$U_{ij} = \frac{1}{\sum_{i=1}^{D} \left(\frac{||x_i - c_j||}{||x_i - c_k||}\right)^{\frac{2}{m-1}}} \tag{1.11}$$

(1.11) numaralı denklem her bir iterasyonda güncellenen üyelik derecelerinin hesaplanmasını göstermektedir.

Bulanık C-ortalamalar algoritması iterasyonlara, üyelik matrisi ve merkez noktalarını hesaplayarak başlar. İteratif olarak üyelik matrisi ve merkez noktalarını günceller, Amaç fonksiyona artık değişmez ya da maksimum iterasyon sayısına ulaşıldığında algoritma sonlandırılır.

1.2.3. Hiyerarşik Kümeleme

Hiyerarşik Kümeleme, veri madenciliği, makine öğrenmesi ve benzer alanlarda verileri benzer özelliklere sahip gruplara ayırmayı amaçlayan bir kümeleme yöntemidir. Hiyerarşik Kümeleme yöntemi, verileri bir dendrogram (ağaç şeklinde görselleştirilmiş veri yapısı) olarak görselleştirmektedir. Bu dendrogram, veriler arasındaki benzerlikleri ve farklılıkları görsel olarak ortaya koymaktadır.

Hiyerarşik Kümeleme yöntemi, iki farklı alt yaklaşım ile uygulanmaktadır. Bu yaklaşımlar aglomeratif kümeleme ve bölünür kümeleme olarak adlandırılmaktadır. Aglomeratif kümeleme yöntemi, verileri tek tek birleştirerek, sonunda tüm verileri tek bir küme halinde bulundurmayı amaçlamaktadır. Bölünür kümeleme yöntemi ise, tüm verileri başlangıçta tek bir kümede bulunduran ve daha sonra bu kümeyi alt kümelere bölerek verileri gruplandırmayı amaçlar.

Hiyerarşik Kümeleme yöntemi, büyük boyutlar üzerinde etkili bir yöntemdir. Ayrıca, veriler arasındaki benzerlikleri görsel olarak göstermesi, bu verileri daha kolay anlamaya yardımcı olmaktadır. Ancak, hiyerarşik kümeleme yönteminin veriler arasındaki benzerlikleri ölçmek için kullanacağı uzaklık özelliğinin doğru seçilmesi önem arz etmektedir. Özellik seçiminin kümeleme performansına doğrudan etki edeceği unutulmamalıdır.

Hiyerarşik kümeleme yöntemi, sınıflandırma, segmentasyon, veri madenciliği, veri analitiği alanlarında kullanılmaktadır. Özellikle veri miktarı büyük olan durumlarda, Hiyerarşik Kümeleme yöntemi, verileri anlamlı ve kolay bir şekilde gruplandırmaya yardımcı olmaktadır.

1.3. Kümeleme Performansının Değerlendirilmesi

Kümeleme algoritmaları, makine öğrenmesi ve veri madenciliğinde kullanılan, benzer özellikleri taşıyan veri noktalarını bir araya getirmeyi amaçlayan algoritmalardır. Kümeleme algoritmaları, veri içerisindeki desenlerin ortaya çıkarılmasında kullanılabileceği gibi, anomali tespiti için de kullanılabilirler. Kümeleme algoritmalarında sınıflandırma algoritmalarında olduğu gibi bir sınıf etiketi bulunmadığı için, kümeleme performansının değerlendirilmesinde, spesifik olarak bir özellik kullanılamamaktadır. Kümeleme performansının değerlendirilmesinde, kümelenmiş veriler üzerindeki özelliklerin benzerlik ve farklılıklarını baz alan metrikler kullanılmaktadır.

Kümeler içerisinde oluşan benzerlikleri değerlendiren metriklere örnek olarak düzeltilmiş rand indeksi (ARI) ve silhouette indeks (SI) verilebilir. Düzeltilmiş rand indeksi, kümelenmiş verilerin özellikleri ile, kümelerin ortalama özelliklerinin benzerliklerini kıyaslamaktadır. ARI, -1 ile 1 arasında değerler üretmektedir. 1 değeri mükemmel bir eşleşme olduğunu, -1 değeri ise tamamen rastgele bir eşleşme olduğunu göstermektedir.

Bir başka yaygın olarak kullanılan metrik, silhouette indeksidir. Silhouette indeksi, her veri noktasının kendi kümesiyle olan benzerliğini diğer kümelerdeki aynı özellik ile karşılaştırır. Silhouette indeksi, -1 ile 1 arasında değer üretmektedir. 1 değeri veri noktasının kendi kümesiyle iyi eşleştiğini, -1 değeri veri noktasının başka bir kümede daha iyi olacağını göstermektedir.

Bu iki metrik dışında, kümeleme performansını değerlendirmek için kullanılabilecek Davies-Bouldin indeksi ve Calinski-Harabasz indeksi gibi diğer metrikler de bulunmaktadır.

1.3.1. Silhouette İndeksi

Silhouette indeksi, kümeleme algoritmasının performansını değerlendirmek için yaygın olarak kullanılan bir metriktir. Her veri noktasının kendi kümesiyle olan benzerliğini diğer kümelerle karşılaştırır. Silhouette indeksi, -1 ile 1 arasında değişmektedir. Yüksek bir silhouette indeksi, kümelerin iyi tanımlanmış olduğunu ve her kümedeki veri noktalarının birbirlerine benzer olduğunu gösterirken, düşük bir silhouette indeksi kümelerin iyi tanımlanmamış olduğunu ve her kümedeki veri noktalarının birbirlerine benzer olmadığını göstermektedir. Silhouette indeksi, kümeleme sonucundaki tüm veri noktalarının silhouette indeksi puanı ortalaması olarak hesaplanır. Hesaplama formülü aşağıdaki gibidir.

Her kümeleme sonucundaki veri noktası *i* için:

N: *eleman sayısı*

a_i: *i ile aynı kümedeki diğer tüm noktalar arasındaki ortalama uzaklık*

b_i: *i ile başka bir kümedeki tüm noktalar arasındaki en düşük ortalama uzaklık*

s_i: *i noktası için silhouette değeri*

$$s_i = \frac{(b_i - a_i)}{max(a_i, b_i)} \tag{1.12}$$

(1.12) numaralı denklem her bir küme elemanı için silhouette indeksinin hesaplanmasını göstermektedir.

Silhouette indeksi, tüm veri noktaları i için, s(i)'lerin aritmetik ortalaması olarak hesaplanmaktadır.

$$S = \frac{1}{N}\sum_{i=1}^{N} s_i \tag{1.13}$$

(1.13) numaralı denklem kümeleme sonucu oluşan kümeler için ortalama silhouette indeksinin hesaplanmasını göstermektedir.

1.4. Genel Bakış

Birçok rotalama algoritması geliştirilmesine rağmen, rotalama problemlerinde optimum çözümü sağlayacak, kuşku götürmez bir şekilde kabul gören bir algoritma bulunmamaktadır. Bunun temel nedenlerinden birisi, her bir rotalama probleminin kendine özgü kısıt ve koşulları içinde barındırmasıdır.

Geliştirilen her bir algoritma ve yaklaşım, farklı sektörlerdeki diğer çalışmalara fikir verme ve ilham olma adayıdır. Bu alanda yapılan her bir çalışma akademide ve diğer kullanım alanlarında öncü olma adayıdır ve bu nedenle değerlidir.

Sonuçların olumlu ya da olumsuz olması fark etmeksizin, hangi yaklaşımların ne tür veri setleri ile uyumlu veya uyumsuz olduğunu göstermesi nedeniyle çalışmanın başarılı olup olmaması, sadece amaç fonksiyonundaki faydanın artırılması üzerinden değerlendirilmemelidir.

Bu tez çalışmasında gerçek hayat verisi üzerinde, öncelikle verinin kullanımına ilişkin bir dizi temizleme işlemi uygulanmıştır. Sonrasında rotalama algoritması, normal hem de kümelenmiş noktalar kümesi üzerinde çalıştırılıp performans değerleri üzerinde kıyaslama yapılmıştır.

1.4.1. Problemin Tanımı

Maltepe Belediyesi'ne ait çöp konteynırlarının bulunduğu noktalar, araçların üzerine yerleştirilen GPS cihazları üzerinden toplanmıştır [39]. Bu noktaların oluşturduğu veri setinin, öncesinde temizlenmesi ve rotalama algoritmalarında kullanımına uygun bir yapıya büründürülmesi gerekmektedir. Elde edilen noktalar ile, çöp toplama rotasını optimize edecek ve en kısa rota uzunluğunu ortaya çıkaracak rotanın hesaplanması bu tezin çözüm bulmayı hedeflediği problemin temel yapısını ortaya koymaktadır.

1.4.2. Amaç ve Hedef

Gerçek hayat verisi ile elde edilmiş veri setinin öncelikle temizlenip, tekrar eden noktaların kaldırılması, sonrasında noktalar arasındaki uzaklıkların hesaplanıp, kümeleme yöntemleri ile veri setinin belli kısımlara ayrılması ve bunun sonucu olarak da rotalama performansında bir artışın gözlemlenmesi amaçlanmaktadır. Bu sayede Maltepe Belediyesi'nin çöp toplama rotasının iyileştirilmesi, bu sayede zaman ve işgücünden tasarruf edilmesi hedeflenmektedir. Ayrıca karbon salınımındaki azalım da doğaya vermiş olduğumuz zararın azalmasına yardımcı olacaktır.

1.4.3. Ana Katkılar

Sonuçları ile bu çalışma, literatürdeki rotalama algoritmalarına ön işleme adımı olarak kümeleme algoritmalarının kullanılmasına iyi bir örnek oluşturacaktır. Ayrıca gerçek bir veri seti üzerinde çalışılması nedeniyle elde edilen sonuçların, ilgili kurum ile paylaşılması sonucunda işgücü, zaman ve yakıt maliyetlerinden tasarruf edilmesine katkı sağlayacaktır. Oluşturulan veri seti sonrasında farklı rotalama çalışmalarında kullanılabileceği için farklı çalışmalara da ön ayak olabilme potansiyeline sahiptir.

2. LİTERATÜRDEKİ ÇALIŞMALAR

Bu çalışma araç rotalama problemine sezgisel algoritmalar ile bir çözüm geliştirirken öncesinde bir önişleme adımı ile performansını artırmayı amaçlamaktadır. Çalışmada elde bulunan veri seti için, belirli kısıtlar altında optimum araç rotaları oluşturulmaya çalışılacaktır. Araç rotalama problemi, uzun zamandır literatürde yer alan bir problemdir. Bu konu ile ilgili literatürde birçok çalışma bulunmaktadır. Bu çalışmalarda veri setine ve kısıtlara özel çözümler geliştirildiği görülmektedir. Literatürdeki çalışmaları inceleyecek olursak;

Robert Bowerman, Brent Hall ve Paul Calamai 1995 yılında önce grupla sonra rotala prensibi ile çalışan çok amaçlı bir optimizasyon çalışması yapmışlardır. Bu çalışmada okul servis araçlarının rotalama problemi ele alınmıştır [7].

İmdat Kara ve Tolga Bektaş 2006 yılında tam sayılı doğrusal programlama yöntemini kullanarak gezicinin en az sayıda gitmesi gereken düğümü sayısını da hesaba katan yeni bir yöntem geliştirmişlerdir [8].

Willem van der Maden, Richard Eglese ve Deborah Black 2010 yılında farklı bir yaklaşımla yeni bir çalışma ortaya koymuşlardır. Bu çalışmada trafik yoğunluğuna bağlı olarak araç rotalarının değişebildiği bir yöntem oluşturmuşlardır. Bu yaklaşımla toplam seyahati en kısa şekilde oluşturarak sezgisel bir algoritma geliştirmişlerdir [9].

Guy Desaulniers ise 2010 yılında araç rotalarının maliyetini en aza indirecek bir çalışma yürütmüştür. Bu çalışmayı yöneylem araştırması kapsamında ele almış ve çalışmada yeni bir Dal-Değer ve Kesme yöntemi oluşturmuştur. Yöntemi geliştirirken dağıtımlı zaman pencereli araç rotalama problemini ele alarak çalışmıştır [10].

Guy Desaulniers, Claudia Archetti ve Martin Bouchard 2011 yılında, bir önceki yıl yayınlanan makalelerinin üzerine ekleyerek aynı hedefe birden fazla aracın uğrayabileceği bir senaryo üzerinde çalışmışlardır. Yine dağıtımlı zaman pencereli araç rotalama problemini Dal-Değer ve Kesme yöntemi ile ele almışlardır [11].

Suna Çetin, Emre Özkütük ve Cevriye Gencer 2011 yılında heterojen araç filolu eş zamanlı dağıtım-toplamalı araç rotalama problemini ele almış ve çalışmada yeni bir sezgisel algoritma önerisinde bulunmuşlardır. Burada eş zamanlı olarak ifade edilen durum rota boyunca hedeflere hem dağıtım hem de toplama işlemlerinin eş zamanlı olarak yapılmasıdır. Böylece toplam yolculuk maliyetini en aza indirmeyi hedefleyerek her bir hedefe bir kere ziyaret gerçekleşmektedir. Yaptıkları çalışmayı daha sonra özel bir karar destek sistemi olarak sunmuş ve bu çalışmaya da makalelerinde yer vermişlerdir. Bu çalışmada daha önceki çalışmalarla karşılaştırma yapamadıkları için 6 hafta boyunca gerçek bir filo sisteminde deney yapmışlardır. Deney sonucunda 6 haftalık test süresi boyunca 8 araçlık bir filo mesafe olarak 11703 km daha az yol katederek ve 15799,05 dolar daha az maliyetle hedeflerini tamamlamışlardır [12].

Ediz Atmaca 2012 yılında yaptığı çalışmada eş zamanlı dağıtım toplamalı araç rotalama problemini GAMS programı kullanarak çözmüştür [13].

Çağrı Koç ve İsmail Karaoğlan 2012 yılında yaptıkları çalışmada çok kullanımlı ve zaman pencereli araç rotalama problemi üzerinde çalışmışlardır. Bu problem klasik araç rotalama problemlerinden farklı olarak araçların birden fazla rotada kullanılabilmesini hedeflenmektedir. Aslında pratik olarak çok fazla karşılaşılan bir sorun olmasına karşın literatürde bu konuyla ilgili çalışmaların azlığına dikkat çekmiş ve bu alanda yeni bir matematiksel model önerisinde bulunmuşlardır. Karma tam sayılı doğrusal programlama yöntemi ile de çözüm geliştirmişlerdir [14].

Emmanouil Zachariadis, Christos Tarantilis ve Chris Kiranoudis 2012 yılında yaptıkları çalışmada yepyeni bir lojistik problemini ele alarak palet yükleme araçlarının rotalama problemlerini ele almışlardır. Bu problemde yükleme kısıtları mevcut olup, yüklemenin doğrudan ilgili araca değil de bir palet aracılığı ile yapılması ele alınmıştır. Ancak bu problemin optimal çözümünün elde edilmesi zor olacağından yasak arama sezgisel metodu ile çözüm elde edilmiştir [15].

Chungmok Lee ve arkadaşları 2012 yılında yaptıkları çalışmada belirsizlik ortamında problemin daha robust bir şekilde çözümü için önermelerde bulunmuşlardır. Bu çalışmayı yaparken müşterilere teslim tarihleri ve araç kapasitesi dikkate alınmıştır. Bunun yanı sıra herhangi iki müşteri arasındaki mesafe ve müşterilerin talepleri belirsiz olarak ele alınmıştır. Çıktı olarak minimum seyahat mesafesiyle belirli sayıdaki müşterinin gereksinimlerini karşılamak amaçlanmıştır. Verilerin belirsizliğini Dantzig-Wolfe ayrıştırma metodu kullanılmıştır. Veri belirsizliği olan alt problemi çözmek için dinamik bir programlama algoritması önerilmiştir [16].

Pieter Vansteenwegen ve arkadaşları yine 2012 yılında yaptıkları çalışmada otel seçimli gezgin satıcı problemi ele alınmıştır. Bu problemde sezgisel yöntem kullanılmıştır. Bu çalışmada literatürde yer almayan yeni bir öneri geliştirilmiştir. Matematiksel bir formülasyon sunulmuş, ilgili optimizasyon problemleriyle farkı açıklanmış ve bu problemi doğası gereği daha zor yapan durumlar belirtilmiştir. Literatürde bazı tipik komşulukların yanı sıra bu problem için özel olarak tasarlanmış birkaç komşuluk arama operatöründen oluşan bir iyileştirme prosedürü ve iki yapıcı başlatma prosedürü kullanan basit ama etkili bir yöntem önerisi geliştirilmiştir. Yeni geliştirilen buluşsal yöntemin performansı CPLEX (10.0) ile karşılaştırılmıştır [17].

Jane Wang ve arkadaşları 2014 yılında yaptıkları çalışmada araçların bir çizelgeleme periyodu boyunca birden fazla sefer yapmasına izin verilen ve her müşteriye belirli bir zaman aralığında hizmet verilmesi gereken çoklu seferler ve zaman pencereleri ile araç rotalama problemini incelemiştir. Bu çalışma özellikle son zamanlarda hayatımızın vazgeçilmez parçalarından olan Getir, Trendyol, Yemeksepeti gibi uygulamaların filolarının planlanması için özel bir öneme sahiptir. Problem çözümünün çok katmanlı yapı karakteristiğinin bir sonucu olarak, bir havuzu doldurmak için önce çeşitli rotaların inşa edildiği, ardından rotalardan bazılarının seçildiği ve araç çalışma programlarını oluşturmak için birleştirildiği, havuz tabanlı bir meta-sezgisel yöntem önerilmiştir [18].

Seda Hezer ve Yakup Kara 2013 yılında yaptıkları çalışmada eş zamanlı dağıtımlı ve toplamalı araç rotalama problemi ele alınmıştır. Bu problemde her müşteri hem dağıtım hem de toplama talebinde bulunmaktadır ve müşterilere eş zamanlı olarak hizmet sağlanmaktadır. Ele alınan problem çözümü zor olan ve oldukça karmaşık kombinasyonel optimizasyon problemi olarak literatürde yer almaktadır. Ve bu problemin ele alındığı pek çok çalışmada metasezgisel metotlar üzerinde durulmuştur. Hezer ve Kara'nın çalışmalarında da son yıllarda yeni bir yaklaşım olarak ele alınan Bakteriyel Besin Arama Optimizasyonu Algoritması tabanlı bir sezgisel çözüm önerisi önerilmiştir. Yapılan çalışmalarda bazı test problemlerinde daha iyi sonuçlar elde edildiği gösterilmiştir [19].

Miguel Martínez Carrasco ve Carlos Amaya 2012 yılında yaptıkları çalışmada zaman pencereli, çoklu yolculuklu, sınırlı sayıda araçlı ve dairesel nesneler için yükleme kısıtlamalı bir araç rotalama problemi üzerinde çalışmışlardır. Bu problem aslında günümüzde de sıklıkla karşılaşılan eve teslim yapan şirketlerin karşılaştığı gerçekçi bir problemdir. Bu problem iki aşamalı olarak ele alınmıştır. İlk adımda geliştirilmiş bir sıralı ekleme algoritması olan Solomon I1 modifikasyonu ile bir başlangıç çözümü elde edilmiştir, ikinci adımda ise tabu arama yöntemi kullanılmıştır [20].

Mehmet Eryavuz ve Cevriye Gencer yayınladıkları çalışmalarında personeli taşıyan servis araçlarının rotalarının belirlenmesinde tasarruf ve rassal tasarruf algoritmalarını kullanmışlardır. Elde edilen sonuçların daha da iyileştirilmesi adına ise 2-opt algoritması ve op-opt algoritması ile rotalar üzerinde tekrar bir algoritma çalıştırılmıştır. Elde edilen sonuçlara göre mevcut durumdaki rotaya göre %28'lik bir iyileşme sağlandığı saptanmıştır [21].

Barrie M. Baker ve Mossie A. Ayechew araç rotalama problemine çözüm çalışmalarında öneri olarak genetik algoritma temelli melez modeller geliştirmişlerdir. Benzetimli tavlama, komşuluk algoritması gibi algoritmalar ile melezlenerek geliştirilen modelleri birbirleri ile kıyaslayıp, zaman ve başarı performanslarını karşılaştırmışlardır [22].

Teemu Nuortio ve arkadaşları katı atık toplama rotalarının ve planlamalarının iyileştirilmesini konu alan çalışmalarında Finlandiya gerçek hayat vakası üzerine çalışmışlardır. Algoritma olarak ise rehberli değişken komşuluk sezgisel algoritmasını kullanmışlardır. Elde ettikleri deneysel sonuçlar, mevcut uygulamaya kıyasla önemli tasarrufların elde edilebileceğini göstermiştir [23].

Alp Üstündağ ve Emre Çevikcan çalışmalarında atık toplama sistemlerine, atık kutuları üzerine RFID entegrasyonu önermişlerdir. Bu sayede atık kutularının atık miktarları ve kesin konumlarını saklamayı amaçlayan çalışma, uygun rotaların belirlenmesinde ise Dash Optimization XPress Solver kullanmaktadır. Oluşturulan optimizasyon modeli ile 15 adet çöp kutusu içeren sayısal bir örnek çözülmüştür. Sonuçlar, her bir çöp kutusunun yeri bilgisinin atık miktarı bilgisi kadar kritik olduğunu göstermektedir [24].

Tiago R. M. Freitas ve arkadaşları GPS verilerinin işlenmesi rotalar üzerinde yanlış bilgilerin düzeltilmesini ve yeni özelliklerinin belirlenmesini amaçlamışlardır. Çalışmalarında önerdikleri yaklaşımla; kavşakların, yolların yönlerinin düzeltilmesinde çok fazlı bir belirleme algoritma önermişlerdir [25].

Sevgi Erdoğan ve Elise Miller-Hooks çalışmalarında yeşil araç rotalama problemine çözüm getirmek için kümeleme temelli ve tasarruf algoritmasını baz alan melez bir algoritma önermişlerdir. Alternatif yakıtlar ile çalışan araç filolarının rotalarını belirlemeyi amaçlayan algoritmalarında, tasarruf algoritmasını modifiye etmişler ve yoğunluk tabanlı kümeleme algoritması ile beraber kullanmışlardır. Sonuçlarda önerilen algoritmanın temel çözümlerine göre iyi çözümler oluşturduğu gözlemlenmiştir [26].

Hakan Güvez ve arkadaşları tıbbı atık toplama sektöründe faaliyet gösteren bir işletmenin atıklarını toplaması üzerine yaptıkları çalışmalarında, rotaların belirlenmesi için model temelli bir yaklaşım önermişlerdir. Tam sayılı matematiksel programlama modelini baz alarak geliştirilen model, en küçük rota maliyetinin oluşmasını amaçlamaktadır. Birden fazla kısıt bulunduran model işletme verileri üzerinde çalıştırılmıştır. Modelin uygulanmasıyla mevcut sistem karşılaştırılmış ve önerilen modelin firmanın bir aylık toplam yol mesafesini %20,63 oranında bir iyileşme sağladığı belirtilmiştir [27].

Zafer Bozyer ve arkadaşları, kapasite kısıtlı araç rotalama problemine yönelik önce noktaları kümeleyen, ardından küme içerisinde rotalama yapan bir sezgisel yöntem önermiştir. Kümeleme algoritması olarak bulanık c-ortalamalar yöntemi kullanılmıştır. Rotalama için ise tabu arama yönteminden yararlanılmıştır. Çalışılan veri seti üzerindeki çözümlere göre, 17 rotada en iyi çözüm bulunmuştur. 7 rota için ise %0,8 ve %10,7 oranları arasında en iyi çözümden uzak sonuçlar oluşturulmuştur [28].

Harun Reşit Yazgan ve arkadaşları çalışmalarında Clarke ve Wright tasarruf algoritması temelli bir melez algoritma kullanmışlardır. Talep ve kapasite kısıtlarını içeren araç rotalama probleminin çözümünde iki basamaklı melez bir algoritma kullanmayı önermişlerdir. Farklı boyutlardaki veri setleri için k-en yakın komşular algoritması, tasarruf algoritması ve kendilerinin geliştirdiği melez algoritma ile çözmüşlerdir. Elde ettikleri sonuçları ANOVA testi ile değerlendirmişlerdir. Önerdikleri algoritmanın diğer iki algoritmadan daha çözüm verdiğini gözlemlemişlerdir [29].

Swapan Das ve Bidyut Kr. Bhattacharyya belediyelerin katı atık toplama ve taşıma yollarının optimizasyonunu konu alan çalışmalarında, katı atıkların öncelikle merkezi toplama istasyonlarına, sonrasında ise transfer merkezlerine aktarıldığını belirtiliyor. Bu rotalamanın çözümünde ise sezgisel bir çözüm modeli önermişlerdir. Model tam sayılı programlama temellerine dayanarak formüle edilmiştir. Sonuçlara göre atık toplama yolunu %30 oranından fazla azaltabilmektedir. Elde ettikleri sonuçlar performansın önemli ölçüde iyileştirilebileceğini göstermektedir [30].

Russel Aziz ve arkadaşları GPS verilerini işleyerek araçların trafik yoğunluğuna maruz kaldığı noktaları belirlemeyi amaçlamışlardır. 10 dakikada bir gelen GPS verileri ile bu süre içerisinde 830 metreden daha az mesafe almış kayıtlar ile yaptıkları çalışmalarının son adımı olarak ise, ilgili noktaları kümeleyip, yoğunluk noktalarını belirlemeye çalışmışlardır. Bu sonuçlar ile dinamik araç planlama problemine çözüm yeni bir yaklaşım göstermişlerdir [31].

Piotr Nowakowski ve arkadaşları atık toplama planlaması yeni bir sistem önermişlerdir. Bu sistem öncelikle kullanıcılara sunulan arayüz ile, belli atık noktalarının sisteme tanımlanmasını içermektedir. Sisteme tanımlanan noktalar yapay zekâ destekli algoritmalar ile uygun rotalara atanarak rota planlaması yapılmaktadır. Çalışmalarında heterojen yapılı bir araç filosunun zaman kısıtla araç rotalama problemine çözüm geliştirmişlerdir. Amaç fonksiyonu olarak en az sayıda aracın rotalama atanması belirlenmiştir. Çalışma simüle edilmiş tavlama, tabu arama, aç gözlü algoritma, arı kolonisi algoritmasını içermektedir [32].

Emir Žunić ve arkadaşları GPS verilerini kullanarak araç rotalama algoritmalarının performansını iyileştirmeyi amaçlamışlardır. Rotalar üzerindeki fazla zaman geçirmeyi belirlemek amacıyla gps dataları üzerinden bazı varsayımlar ile yoğunluk noktalarını belirlemişlerdir. Verilere göre hızın sıfıra eşit olduğu durumlarda, trafik ışıklarının uzunluklarının da 40-50 saniye arası olarak varsayılmış, daha fazla duraklamalar yoğunluk noktası olarak belirlenmiştir. Bu yoğunlukları zamanlara dahil ettikleri tahminlerinde ise hata paylarının iki kattan fazla azaldığını gözlemlemişlerdir [33].
Onur Çetin ve Necdet Özçakar akaryakıt dağıtım sistemleri için araç rotalama problemine çözüm aradıkları çalışmalarında tasarruf algoritması, genetik algoritma ve tabu arama algoritmalarını içiren bir çözüm modeli önermişlerdir. Melez algoritma, öncelikle tasarruf algoritması kullanılarak rotaların oluşturulması, sonrasında rotaların kendi içlerinde optimizasyonu için genetik algoritma temelli bir tabu arama algoritmasını kullanmaktadır. Test verileri üzerindeki sonuçlara göre, melez algoritmanın makul seviyede kabul edilebilir çözümler ürettiği belirtilmiştir [34].

Büşra Özoğlu ve arkadaşları, çalışmalarında araçlar ve beraberlerinde yardımcı olarak kullanılan insansız hava araçlarının rotalarının belirlenmesini konu edinmişlerdir. Araç rotalarının belirlenmesinde tasarruf algoritması ve genetik algoritmayı melez kullanan bir sezgisel model önermektedirler. Önerilen yaklaşım ile kamyon ya da insansız hava aracının veyahut her ikisinin de talep noktalarına atanmasını amaçlamaktadır. Önerilen yaklaşımda rotalar tasarruf algoritması ile oluşturulduktan sonra, genetik algoritma ile iyileştirilmiştir [35].

Mahammad A. Hannan ve arkadaşları statik verilerle sabit rota optimizasyonu ve gerçek zamanlı verilerle değişken rota optimizasyonu bir arada ele almaktadır. Çalışmanın amacı, statik varsayımlar yerine dinamik rota optimizasyonu ile verimliliği artırmak, toplam toplama maliyetlerinden tasarruf etmektir. Yalnızca %70 den fazlasının dolu olduğu atık kutularının rotalamaya dahil edildiği model rotalama performansını iyileştirmiştir. Elde ettikleri sonuçlara göre toplama verimliliği %26,08 artmıştır. Bununla birlikte, maliyet tasarrufu ise %44,44 olmuştur [36].

Emir Žunić ve arkadaşları çalışmalarında büyük dağıtım firmalarındaki rota verilerindeki anormallikleri belirlemek ve rotaları iyileştirmek için gps verilerindeki anormalliklerin tespitlerinden yararlanmışlardır. Belirli kısıtlara uyan kayıtlar için yoğunluk noktaları olarak belirlenmiştir. Ayrıca bu geliştirilen algoritma ile noktaların gerçek konumlarından sapmalarının da önüne geçilmesi amaçlanmaktadır [37].

Merve Özalp ve Selçuk Alp kargo firmasının uygun dağıtım rotasının belirlenmesini konu alan çalışmalarında melez bir algoritma kullanmayı önermişlerdir. Önerdikleri melez sezgisel yöntemde karınca kolonisi optimizasyonu ve genetik algoritma bir arada kullanılmaktadır. İlgili çalışmada bir kargo firması için 5 araç kullanılarak bir günlük uygun dağıtım rotası belirlenmeye çalışılmıştır. Rotalar karınca kolonisi optimizasyonu algoritması ile oluşturulmuş, sonrasında genetik algoritması çaprazlaması ile rotalar kendi içlerinde optimize edilmeye çalışılmıştır. Çaprazlamalar sonucunda çözülen modellerle ilk çözüme göre %7,06 oranında bir iyileşme sağlanmıştır [38].

Hatice Şahin Akdaş ve arkadaşları, 2021 yılında yayınladıkları, katı atık toplamasına ilişkin çalışmalarında, Maltepe Belediyesi tarafından sağlanan gerçek hayat verisi ile araç rotalama problemi üzerinde çalışmışlardır. Karınca kolonisi algoritması ile kümeleme algoritmalarını beraber kullandıkları çalışmalarının sonuçlarına göre, ilk rota uzaklığından %13 daha düşük rota uzaklığı maliyetinde rotalar oluşturmayı başarmışlardır [39].

Ruixue Gu ve arkadaşları, dronların birden fazla ziyaretlerine olanak sağlayacak, araç rotalama problemine çözüm çalışmalarında, hibrit bir yöntem önermişlerdir. Önerilen çözüm yöntemi, hiyerarşik bir yapıda çözüm değerlendirme kriterleri kullanarak, her bir ziyaretin doğruluğunu ve kapsamını hesaplamaktadır. Hibrid algoritma, bir çevrimiçi aşama ve bir çevrimdışı aşama içermektedir. Çevrimiçi aşama, kısmi bir çözümü üretmek için genetik algoritma kullanmaktadır. Çevrimdışı aşama ise, daha iyi bir çözüm bulmak için tabu arama algoritması kullanmaktadır. Bu aşamalar birleştirildiğinde, hibrid algoritma, diğer yöntemlere göre daha iyi sonuçlar vermektedir [40].

Zheng Zhang ve arkadaşları, 2023 yılında yayınladıkları çalışmalarında, stokastik müşteri taleplerinin bulunduğu yükleme kısıtlı araç rotalama problemi için kendileri tarafından modifiye edilen tabu arama algoritması geliştirilmiştir. Geliştirilen algoritma, tabu arama yöntemini ve adaptif yöntemi birleştirerek en iyi rotaları bulmayı amaçlamaktadır. Adaptif yöntem, algoritmanın daha verimli olması için parametreleri otomatik olarak güncellemektedir. Önerilen algoritma, müşteri taleplerini tahmin etmek için bir regresyon modeli kullanmaktadır. Elde edilen sonuçlar karşılaştırıldığında, önerilen algoritmanın, diğer yöntemlere göre daha iyi sonuç verdiği gözlemlenmektedir [41].

Yong Wang ve arkadaşları, araç rotalama problemini zaman kısıtlı olarak ele almışlardır. Çok depolu araç rotalama problemi için kümeleme tabanlı genetik bir algoritma önerilmektedir. Çözüm yöntemi iki aşamalı bir şekilde çalışmaktadır. İlk aşama, müşterileri ve depoları kümeler halinde gruplandırmakta ve her bir küme için bir alt problem oluşturmaktadır. İkinci aşamada ise, genetik algoritma kullanılarak her bir alt problem için ayrı ayrı çözümler üretilmektedir. Bu çözümler daha sonra birleştirilerek ana problemin çözümü elde edilmektedir. Önerilen çözüm yöntemi, daha önceki çalışmalarda kullanılan yöntemlerden daha iyi sonuçlar vermektedir. Ayrıca, kümeleme yöntemi sayesinde daha büyük boyutlu problemlerde de uygulanabilir olmaktadır [42].

3. MATERYAL VE YÖNTEM

3.1. Veri seti

Bu tez çalışmasında kullanılacak olan veri seti, Maltepe Belediyesi tarafından sağlanmış olup, Maltepe Belediyesi sorumluluk alanı içerisinde bulunan çöp konteynırlarının lokasyonlarını, çalışan çöp kamyonu detaylarını içeren dosyalar şeklinde paylaşılmıştır.

Veriler Maltepe Belediyesine ait mahallelere göre bölünmüş olarak 18 adet KML dosyası şeklindedir. KML dosya tipi, iki veya üç boyutlu nokta, şekil ya da coğrafi haritaları görselleştirmede kullanılan genel bir dosya tipidir. Maltepe Belediyesi tarafından paylaşılan dosyaların listesi Şekil 3.1'de gösterildiği gibidir.

altayçeşme mah.kml	27/06/2022 13:45	KML	162 KB
altıntepe mah.kml	27/06/2022 13:45	KML	122 KB
aydınevler mah.kml	27/06/2022 13:46	KML	106 KB
bağlarbaşı mah.kml	27/06/2022 13:45	KML	237 KB
başıbüyük mah.kml	27/06/2022 13:46	KML	55 KB
büyükbakkalköy mah.kml	27/06/2022 13:46	KML	110 KB
Cevizli.kml	27/06/2022 13:45	KML	239 KB
cinar mah.kml	27/06/2022 13:45	KML	93 KB
esenkent.kml	27/06/2022 13:45	KML	96 KB
feyzullah mah..kml	27/06/2022 13:45	KML	113 KB
fındıklı mah.kml	27/06/2022 13:46	KML	119 KB
girne mah.kml	27/06/2022 13:46	KML	59 KB
gülensu mah.kml	27/06/2022 13:45	KML	57 KB
gülsuyu mah.kml	27/06/2022 13:45	KML	84 KB
idealtepe mah.kml	27/06/2022 13:45	KML	129 KB
küçükyalı mah.kml	27/06/2022 13:45	KML	155 KB
yalı mahlesi.kml	27/06/2022 13:45	KML	78 KB
zümrütevler mah..kml	27/06/2022 13:45	KML	224 KB

Şekil 3.1 Belediye Tarafından Paylaşılan KML Dosyaları

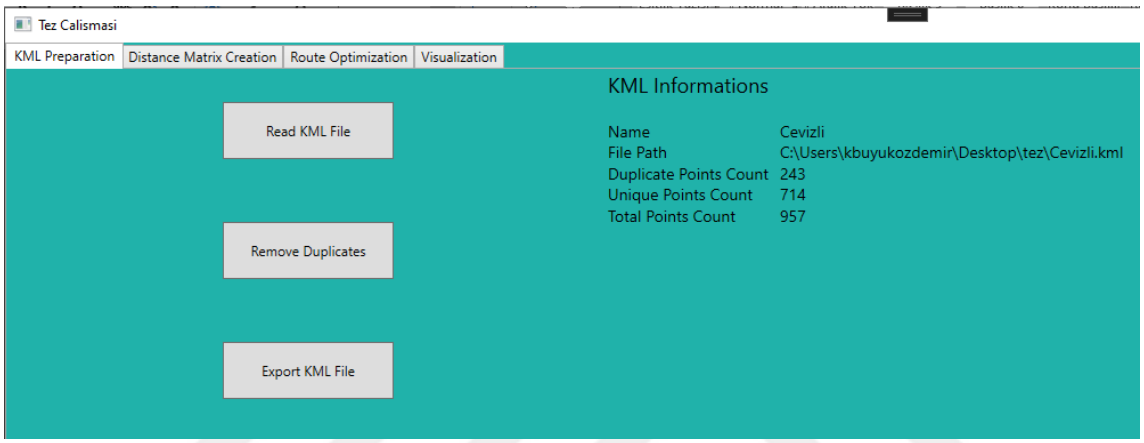
KML dosyalarının içeriğinde, mahallerin özelinde bulunan çöp kutularının koordinat bilgileri bulunmaktadır. Geliştirilen yazılım aynı zamanda, KML dosyalarından tüm noktaları içeren, birleştirilmiş, CSV uzantılı dosyanın oluşturulmasında da kullanılacaktır. Bu nedenle, KML uzantılı dosyanın, doğru şekilde işlenmesi önem arz etmektedir. KML dosyasının doğru şekilde içeri alınabilmesi için, SharpKml isimli C# paketi kullanılmıştır. KML dosyalarının içerikleri Şekil 3.2'de gösterilmektedir.

```
<Document>
    <Style id="icon-503-BCA920">
        <IconStyle>
            <color>FFBCA920</color>
            <Icon><href>http://www.gstatic.com/mapspro/images/stock/503-wht-blank_maps.png</href></Icon>
        </IconStyle>
    </Style>
    <Placemark>
        <name><![CDATA[m 1]]></name>
        <styleUrl>#icon-503-BCA920</styleUrl>
        <TimeStamp><when>2021-12-08T01:25</when></TimeStamp>
        <Point>
            <coordinates>29.13796976,40.93407879,0</coordinates>
        </Point>
    </Placemark>
    <Placemark>
        <name><![CDATA[m 2]]></name>
        <styleUrl>#icon-503-BCA920</styleUrl>
        <TimeStamp><when>2021-12-08T01:25</when></TimeStamp>
        <Point>
            <coordinates>29.13772367,40.93415376,0</coordinates>
        </Point>
    </Placemark>
    <Placemark>
        <name><![CDATA[m 3]]></name>
        <styleUrl>#icon-503-BCA920</styleUrl>
        <TimeStamp><when>2021-12-08T01:25</when></TimeStamp>
        <Point>
            <coordinates>29.13707223,40.93449798,0</coordinates>
        </Point>
    </Placemark>
```

Şekil 3.2 KML Dosyasının İçeriği

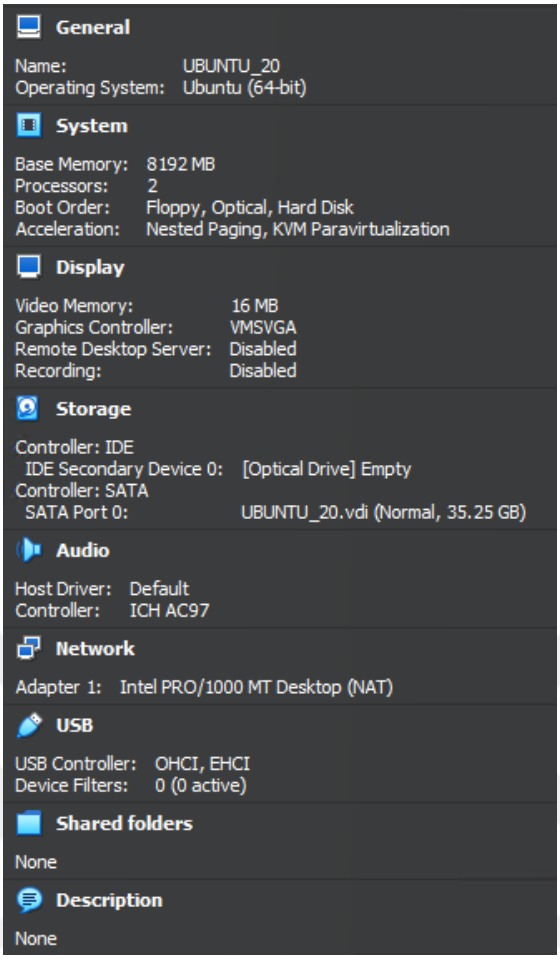
3.2. Verinin Temizlenmesi ve İlk Hazırlık

Veri seti içerisindeki noktalar incelendiğinde, bazı noktaların birden fazla işaretlendiği görülmüştür. Rotalama ve kümeleme algoritmalarının doğru çalışabilmesi adına bu noktaların temizlenmesi gerekmektedir. Bu temizleme işlemi için C# diliyle WPF çerçevesi kullanılarak bir yazılım geliştirilmiştir. Bu yazılım ile KML dosyalarının sadece benzersiz noktaları içeren haliyle yeniden KML formatında dışarıya aktarılmıştır. Hazırlık için geliştirilen yazılım Şekil 3.3'te gösterilmiştir.



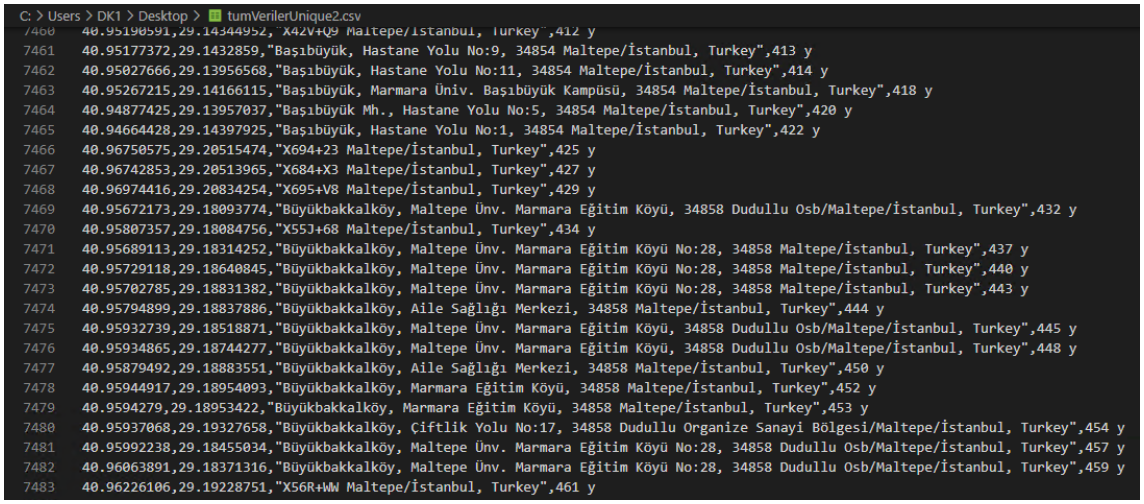
Şekil 3.3 KML Ön Hazırlığı için Geliştirilen Program

Rotalama algoritmaların çalışabilmesi için noktaların birbirine olan uzaklıkların bilinmesi gerekmektedir. Elimizdeki noktaların, birbirilerine olan uzaklıkları koordinatları üzerinden teorik olarak hesaplanabilmektedir. Ancak gerçek hayat verisi ile çalıştığımız için, noktaların teorik uzaklıklarının yerine gerçek uzaklıklarının hesaplanmasının gerekli olduğu görülmüştür. Bu hesaplamanın yapılabilmesi için açık kaynak olan, olarak GitHub platformu üzerinden kaynak kodlarını paylaşan, Open Source Routing Machine (OSRM)'nin arkayüz uygulama programlama arayüzü kullanılmıştır. Paylaşılan platform üzerinden ilgili arayüz temin edilmiş Oracle VM VirtualBox yazılımı üzerinde sanal olarak hazırlanmış olan işletim sisteminde kurulmuştur. Sanal bilgisayar "Ubuntu 22.04.1 LTS" işletim sistemini kullanmaktadır. Sanal bilgisayar kendine ayrılmış 8192 MB geçici bellek, Intel® Xeon E-2224 işlemci üzerinden kendine ayrılmış 2 adet çekirdek, depolama için ise dinamik olarak ayrılmış 35.25 GB alan kullanmaktadır. Sanal makine özellikleri Şekil 3.4'te gösterilmiştir.



Şekil 3.4 Sanal Makine Özellikleri

Rotalama öncesinde tüm noktalar kümeleneceği için tüm noktaların birbiri ile aynı kümeye düşme ihtimali bulunmaktadır. Bu nedenle tüm noktaların birbirleri ile olan uzaklıklarının belirlenmesi gerekmektedir. Bunun için belediye tarafından paylaşılan tüm KML dosyaları, benzersiz olan noktalarının ortaya çıkarılmasının ardından birleştirilmiştir. Birleştirilen noktalar CSV formatında dışarı aktarılmıştır. Ayrıca başlangıç noktası olan Maltepe Belediyesi Temizlik İşleri Müdürlüğü Temizlik İşleri Şefliğinin mevcut noktalar içerisindeki yeri belirlenmiştir. Algoritmaların çalışmasına kolaylık olması adına ilk kayıt olarak yeni oluşturulan CSV dosyasında ilk sıraya alınmıştır. Oluşturulan CSV dosyasında 7483 nokta kaydı bulunmaktadır. Bu kayıtlardan ilk sırada olan kayıt başlangıç noktasını tanımlamaktadır. Birleştirilen CSV dosyalarının örnek görüntüsü Şekil 3.5'te gösterilmiştir.



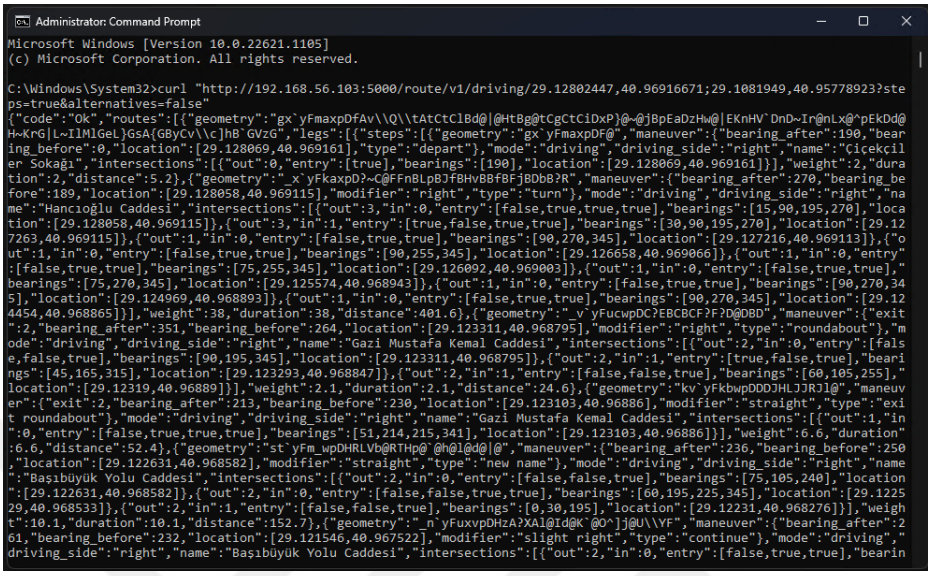
Şekil 3.5 Birleştirilen CSV Dosyasının Örnek Görüntüsü

Sanal makine üzerinde çalıştırılan OSRM arkayüz yazılımını gösteren örnek, Şekil 3.6 gösterilmiştir.

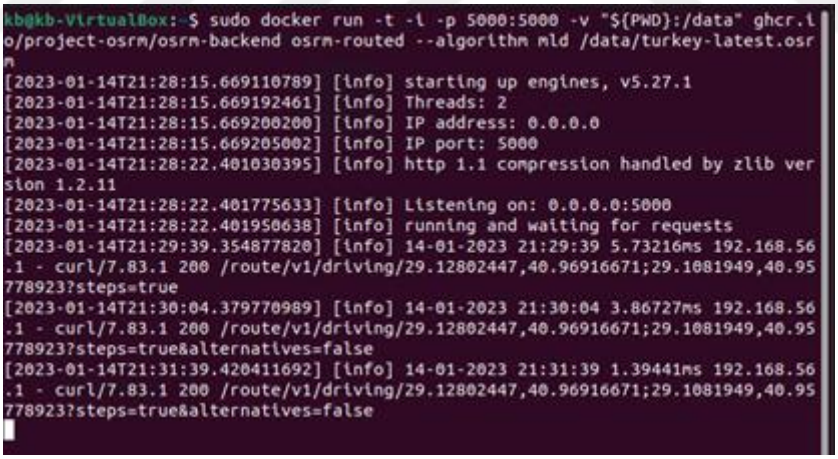
Şekil 3.6 Sanal Makine Üzerinde Çalıştırılan OSRM Arkayüz Yazılımı

Geliştirilen yazılım üzerinden 7483 * 7483 = 55987806 ikili için uzaklık hesaplanmıştır. Sonucunda ise 7483x7483 boyutunda bir uzaklıklar matrisi elde edilmiştir. Örnek istek aşağıdaki gibidir. Şekil 3.7 örnek uzaklık isteği ve OSRM arkayüz yazılımının cevabını göstermektedir. Şekil 3.8 ise OSRM arkayüz yazılımının gelen isteği karşılamasını göstermektedir.

curl
“http://192.168.56.103:5000/route/v1/driving/29.12802447,40.96916671;29.1081949,40.95778923?steps=true&alternatives=false”

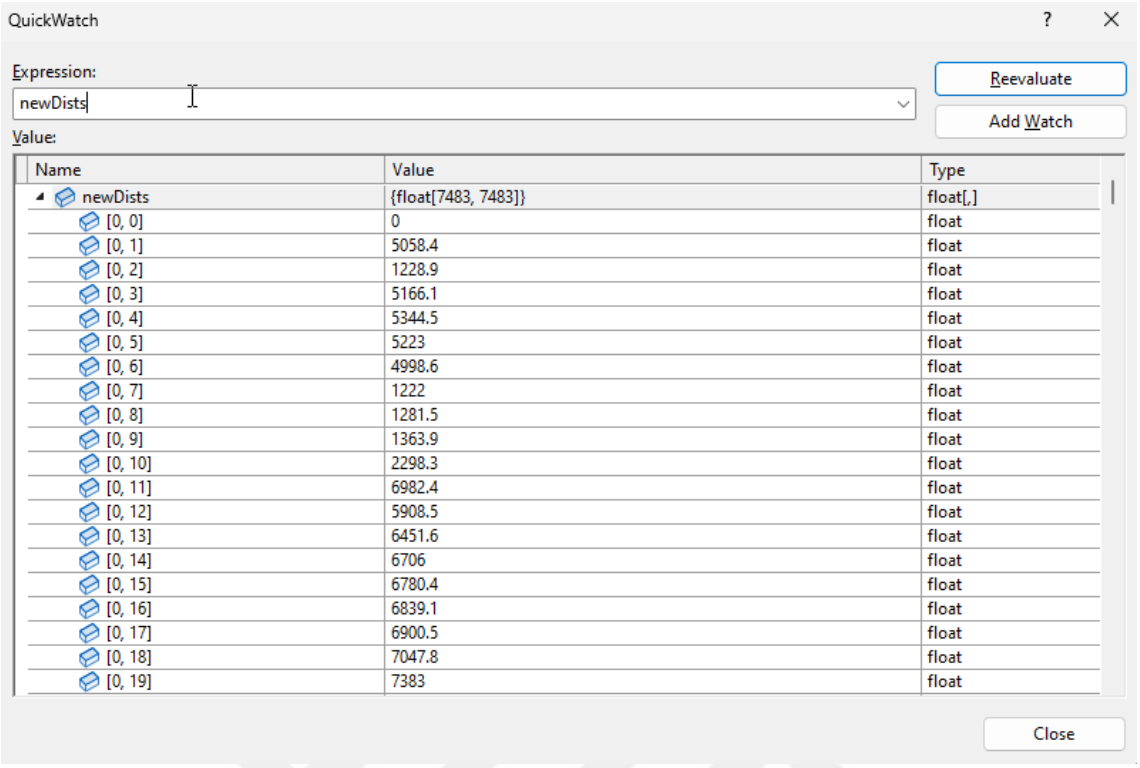


Şekil 3.7 Örnek Uzaklık İsteği ve OSRM Arkayüz Yazılımının Cevabı



Şekil 3.8 OSRM Arkayüz Yazılımına İstek Geldiğindeki Durumu

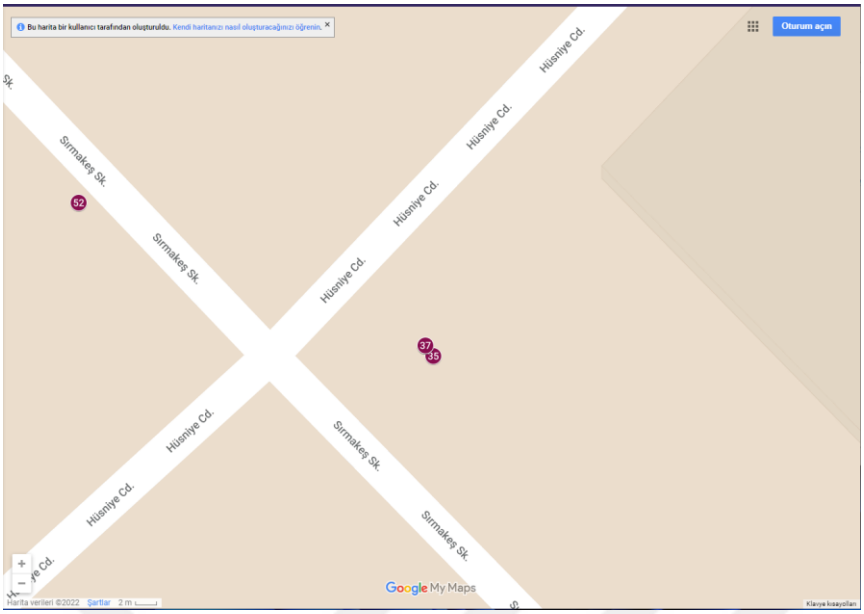
Oluşturulan uzaklıklar matrisi, C# dili kullanılarak, JSON formatında dışarı aktarılmıştır. Şekil 3.9 oluşturulan uzaklıklar matrisini göstermektedir. Şekil 3.10 ise matris içerisindeki toplam ikili sayısını ortaya çıkarmaktadır.



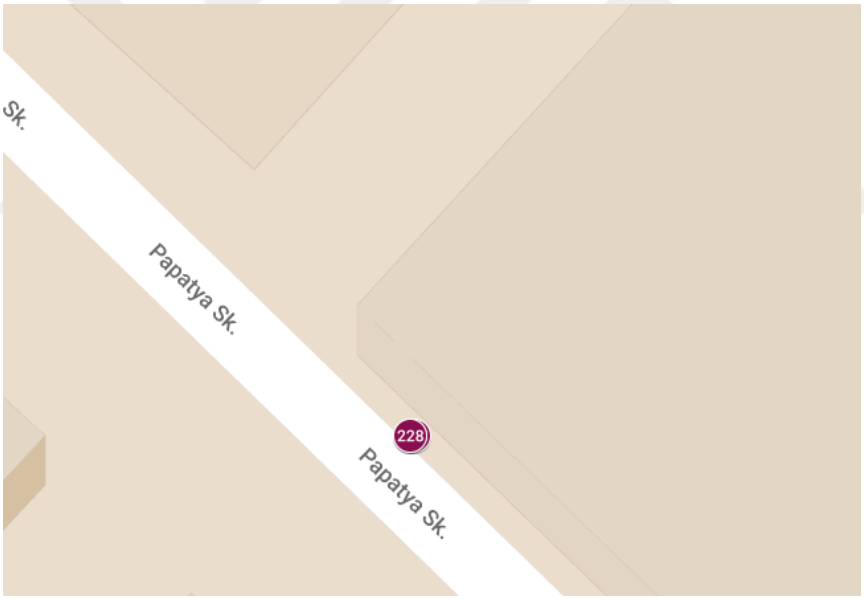
Şekil 3.9 C# Yazılımında Uzaklıklar Matrisinin Okunması

Şekil 3.10 C# Yazılımındaki Uzaklık Matrisinin Uzunluğu

Çıkarılan uzaklıklar matrisi incelendiğinde, bazı noktalar arasındaki uzaklıkların 0 olarak hesaplandığı görülmüştür. Bu durumun neden olduğu araştırılmıştır. Uzaklıkları 0 olan noktaların, birbirlerine çok yakın ve birbirleri arasında herhangi bir yol bulunmayan noktalar olduğu anlaşılmıştır. Uzaklığı 0 hesaplanan noktalar Şekil 3.11 ve Şekil 3.12'de görülmektedir.



Şekil 3.11 Uzaklıkları Sıfır Hesaplanan Noktalar — 1



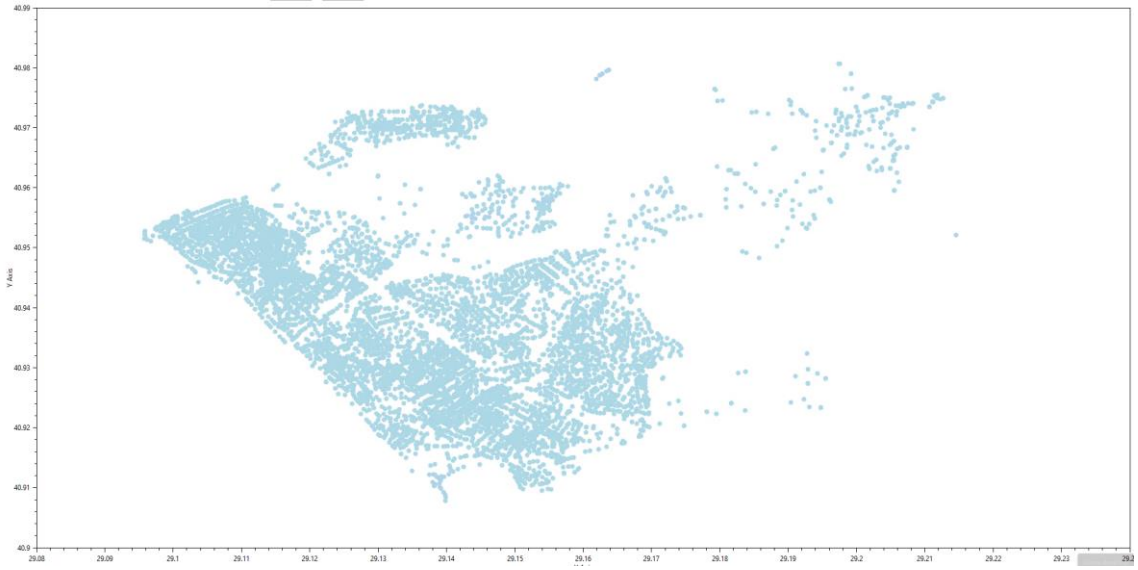
Şekil 3.12 Uzaklıkları Sıfır Hesaplanan Noktalar — 2

Bu noktalar için, verilerin işlenmeden öncesinde rotalama algoritmalarının çalışmalarını olumsuz etkilememeleri adına, 1 değeri atanmıştır. Değeri atan kod parçacığı Şekil 3.13'te görülmektedir.

```
var jsonArray = JsonSerializer.Deserialize<JsonArray>(File.ReadAllText("distancesF.JSON"));
for (int i = 0; i < nodeCount; i++)
{
    var jsonNode = jsonArray[i].AsArray();
    for (int j = 0; j < nodeCount; j++)
    {
        var v = (long)(float)jsonNode[j].AsValue();
        if (i != j && v == 0)
        {
            v = 1;
        }
        DistanceMatrix[i, j] = v;
    }
}
```

Şekil 3.13 Uzaklıkları Düzenlemek için Yazılan Kod Parçacığı

Tüm bu aşamalardan sonra ilgili noktalar ve uzaklıklar matrisi, algoritmalar üzerinde çalıştırılmak üzere hazırlığı tamamlanmıştır. Tüm noktaların koordinat düzlemi üzerindeki lokasyonları Şekil 3.14'te gösterilmiştir.



Şekil 3.14 Veri Seti İçerisinde Tüm Noktalar

3.3. Varsayımlar

Rotalama problemleri, içerisinde çokça değişken ve parametre barındırmaktadır. Elde edilen veri seti içerisinden bazı gerekli parametreler bilinmemekte, bazıları ise mantıklı şekilde çıkarılamamaktadır. Bu nedenle ihtiyaç olan değişken ve koşullar için, problemin çerçevesini çizme adına varsayımlar üzerinden ilerlenmiştir.

Bu kısımda, tez çalışmasında içerisindeki varsayılan kısıt ve kabuller bulunmaktadır.

- Tüm çöp kutularının birbiri ile eş olduğu ve aynı hacim ve doluluk oranına sahip olduğu kabul edilmiştir.
- Belediye bünyesindeki tüm araçların kapasitelerinin birbiri ile aynı olduğu kabul edilmiştir.
- En iyi rotanın bulunmasında parametre olarak "uzaklık" olarak ele alınmıştır. Zaman kısıtı rotalamaya dahil edilmemiştir.
- Çalışma içerisindeki tüm uzaklıklar "metre" cinsinden ele alınmıştır.
- Rotalama çalışmalarında, iki rota arasındaki uzaklıkların hesabında, oluşturulan rota üzerindeki, duba, kısıtlı dönüş açısı gibi geometrik kısıtlar göz ardı edilmiştir.
- Belediye tarafından paylaşılan KML dosyaların içerisinde rota varsayılan rota olarak kabul edilmiştir. Rotalama algoritmaların performans değerlendirmesi bu kabuller üzerinden değerlendirilmiştir. Bu durumda oluşan, varsayılan değerler Tablo 3.1'de verilmiştir.

Tablo 3.1 Varsayılan Rota Uzaklıkları

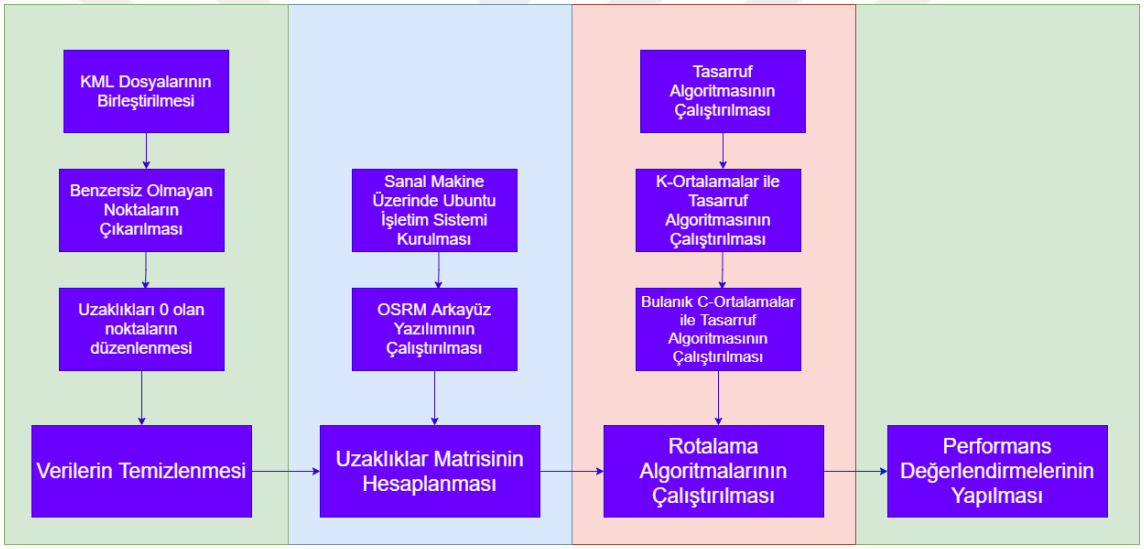
Mahalle	Rota Uzaklığı
Altayçeşme Mahallesi	39579 m
Altıntepe Mahallesi	57095 m
Aydınevler Mahallesi	40383 m
Bağlarbaşı Mahallesi	78002 m
Başıbüyük Mahallesi	32851 m
Büyükbakkalköy Mahallesi	84556 m
Cevizli Mahallesi	88525 m
Çınar Mahallesi	27327 m
Esenkent Mahallesi	61350 m
Feyzullah Mahallesi	27310 m
Fındıklı Mahallesi	56935 m
Girne Mahallesi	23757 m
Gülensu Mahallesi	26749 m
Gülsuyu Mahallesi	34638 m
İdealtepe Mahallesi	39755 m
Küçükyalı Mahallesi	43977 m
Yalı Mahallesi	32938 m
Zümrütevler Mahallesi	89580 m
TOPLAM	***885307 m***

3.4. Kullanılacak Yöntemler ve İzlenecek Yol

Bu çalışmada, performans değerlendirmelerini yapmak ve sonuçları kıyaslayabilmek adına, öncelikle tüm noktaların birleştirildiği veri seti kullanılarak, yeniden rotalama yapılacaktır. Varsayılan çözüm ile kıyaslamadaki çözümün doğru kıyaslanabilmesi adına, mevcut rota sayısı kadar rota oluşması sağlanacaktır.

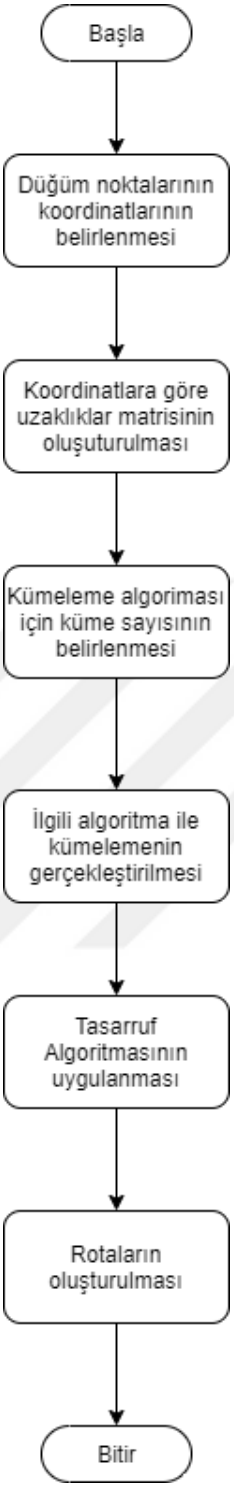
İkinci aşama olarak kümeleme algoritması çalıştırılacak ve noktalar tüm noktalar üzerinden değil, bu kümeler üzerinden bir rotalama algoritmasına dahil edileceklerdir. Kıyaslamanın bozulmaması adına oluşturulacak küme sayısı da 18 olarak belirlenmiştir.

Çalışmanın mevcut ilerleyişi Şekil 3.15'te gösterilmiştir.



Şekil 3.15 Mevcut Çalışmanın Akış Şeması

Bu çalışmada öne sürülen, önce kümele sonra rotala algoritmalarının çalışma akışını gösteren akış şeması Şekil 3.16'da gösterilmiştir.



Şekil 3.16 Önce Kümele Sonra Rotala Yaklaşımı Akış Şeması

3.5. Küme Sayısının Belirlenmesi

Çalışmanın bu kısmında, elde bulunan veri seti için, algoritmalarda kullanılmak üzere küme sayısının belirlenmesi üzerinde çalışılmıştır.

Birbirine benzerliği fazla olan kümeler oluşturmak rotalamanın performansını artıracağı için, silhouette indeksi ile oluşturulan kümelerin, kümeleme performansları ölçülmüştür. İndeks hesaplaması için MATLAB yazılımı kullanılmıştır. Şekil 3.17 kullanılan kod parçacığını göstermektedir.

```
load('points.mat');
evaluation = evalclusters(points,"kmeans","silhouette","Distance","sqEuclidean","KList",2:50);
```

Şekil 3.17 Silhouette İndeksi ile Küme Sayısının Belirlenmesi

Veri seti ile 2'den 50'ye kadar küme sayısı için silhouette indeksi değeri hesaplanmıştır. Silhouette indeksi değerleri Tablo 3.2'de gösterilmiştir.

Tablo 3.2 Küme Sayıları için Silhouette İndeksi Değerleri

Küme Sayısı	Silhouette Değeri
2	0,61103
3	0,67405
4	0,58681
5	0,62634
6	0,59998
7	0,59758
8	0,57807
9	0,58896
10	0,56949
11	0,56763
12	0,55928
13	0,56497
14	0,55311
15	0,54922
16	0,55469
17	0,54080
18	0,55103
19	0,53951
20	0,54565
21	0,54996
22	0,56053
23	0,55332
24	0,54233
25	0,53547

26	0,54070
27	0,53666
28	0,54305
29	0,54533
30	0,52463
31	0,54336
32	0,54248
33	0,54140
34	0,53320
35	0,54677
36	0,52970
37	0,54496
38	0,54952
39	0,54979
40	0,54465
41	0,53929
42	0,53648
43	0,54137
44	0,53662
45	0,53848
46	0,53612
47	0,53092
48	0,53919
49	0,53635
50	0,53166

Değerlerden de görüleceği üzere en optimum küme sayısı 3 olarak hesaplanmıştır. Ancak, çalışmalarda bu küme sayısının kullanılması uygun değildir. Gerçek hayat verisi üzerinde 18 mahalleden oluşan bir yapıyı 3 kümeye bölen bir çözümün, gerçek hayata yansıtılması pek olası değildir. Bir rota üzerinde çokça noktanın bulunması, o rota üzerindeki çöp kutularına erişim süresini uzatacaktır. Bu nedenle bir rotanın tam turunu tamamlama süresi, makul sürenin dışına çıkabilir. Bu çalışmada, rotalar üzerindeki gezme süresi ve süre kısıtları bilinmediği için, rota sayısının dramatik şekilde azaltılması uygun görülmemiştir.

Rotaların performanslarının değerlendirilmesinde, oluşturulan rota uzunlukları kullanılmaktadır. Rotaların en büyük maliyetlerinden birini başlangıç konumundan, ilk çöp kutusuna gitme uzaklığı oluşturmaktadır. Rota sayısının az olduğu durumlarda, bu maliyetten kaçınıldığı için optimum rota uzaklıkları daha düşük çıkabilmektedir. Bu nedenle rotalama performansının da doğru değerlendirebilmesi adına, ilk rota sayısına benzer sayıda rota oluşturulmasının, daha uygun olduğu varsayılmıştır. İndeks değerleri incelendiğinde, küme sayısının, özellikle ilk rota sayısına yakın olan küme sayılarının silhouette indeksi değerlinin birbirine çok yakın olduğu görülmektedir. Ayrıca kümeleme algoritmalarının başlangıç çözümlerine bağlı yapıları gereği, aradaki farklar, rasgele şekilde değişmektedir. Bu nedenle, küme sayılarının silhouette indeksi perspektifinden bir üstünlükleri bulunmamaktadır. Tüm bu sebepler neticesinde, küme sayısının ilk rota sayısı olan 18 olarak seçilmesi kararlaştırılmıştır. Bu sayede tüm çözümler benzer sayıda başlangıç rota maliyetine sahip olacaktır. Böylece önerilen algoritmanın performansı daha anlamlı bir şekilde gözlemlenecektir.

3.6. Tüm Noktalar Üzerinden Rotalama Algoritmalarının Çalıştırılması

Çalışmanın bu kısmında, tüm noktalar üzerinden tasarruf rotalama algoritması çalıştırılmış ve sonuçları paylaşılmıştır.

3.6.1. Tasarruf Algoritmasının Çalıştırılması

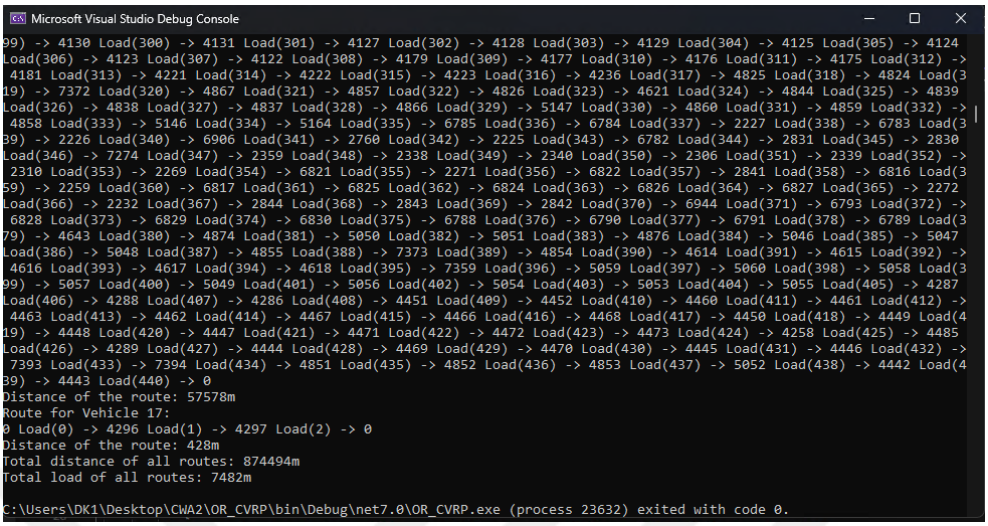
Tasarruf rotalama algoritmasının çalıştırılmasında "OR-Tools-Google Optimization Tools" kütüphanesinden faydalanılmıştır. C# diliyle geliştirilen konsol uygulamasında kısıtlar 18 adet rota oluşturulacak şekilde ayarlanmıştır. Algoritmanın çalışma gösteren kod parçacığı Şekil 3.18'de gösterilmiştir. Geliştirilen C# konsol uygulamasında çalıştırılan tasarruf algoritması, sonuçlar konsol ekranına yazdırılmaktadır. Ayrıca oluşan rotanın çizdirilmesi için benzer bir C# uygulaması geliştirilmiştir.

```
// Setting first solution heuristic.
RoutingSearchParameters searchParameters =
    operations_research_constraint_solver.DefaultRoutingSearchParameters();
searchParameters.FirstSolutionStrategy = FirstSolutionStrategy.Types.Value.Savings;

// Solve the problem.
Assignment solution = routing.SolveWithParameters(searchParameters);
```

Şekil 3.18 OR-Tools Örnek Kod Parçacığı

Algoritmanın çalıştırılması sonucunda oluşan rotalar ve uzaklıklar aşağıdaki Tablo 3.3'te gösterilmiştir. Tasarruf algoritmasının örnek çıktısı ise Şekil 3.19'daki gibidir.

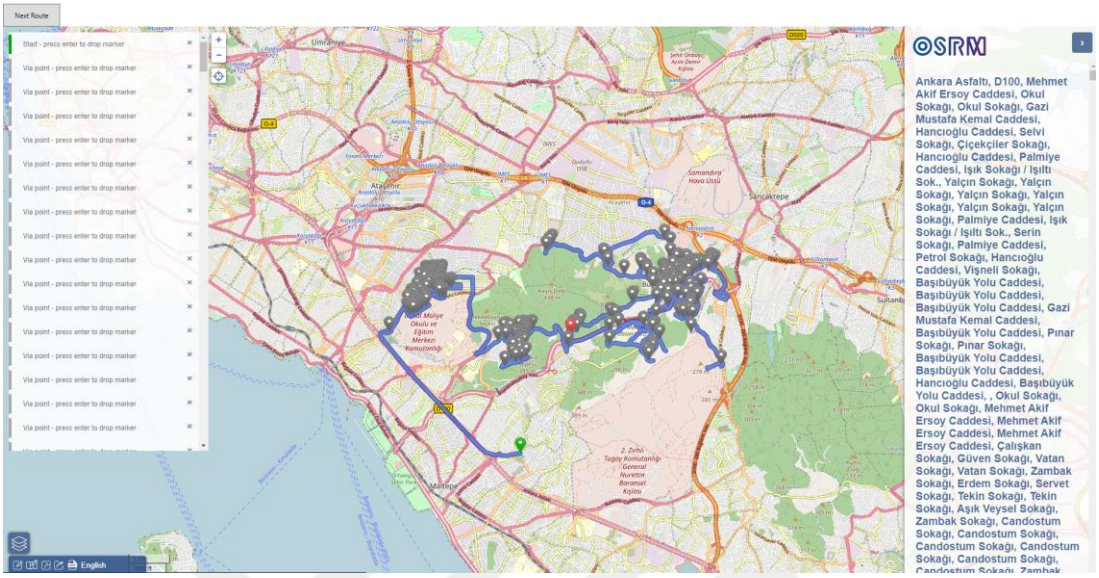


Şekil 3.19 Tasarruf Algoritmasının Çalıştırılması Konsol Çıktısı

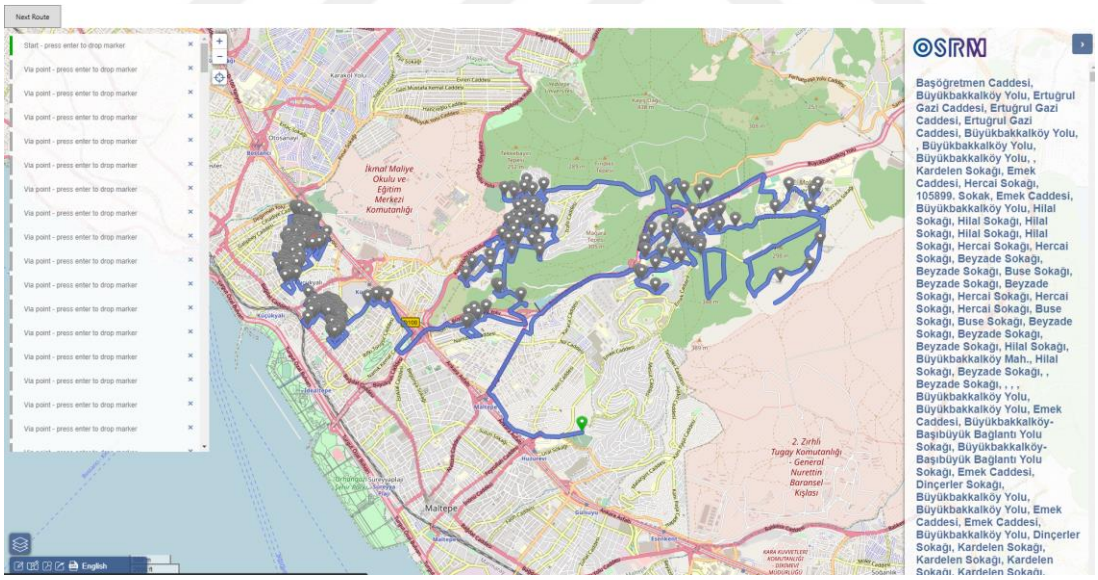
Tablo 3.3 Tasarruf Algoritması Rota Uzaklıkları

Rota İsmi	Rota Uzaklığı
Rota 1	79699 m
Rota 2	52096 m
Rota 3	42470 m
Rota 4	65356 m
Rota 5	79437 m
Rota 6	42149 m
Rota 7	47892 m
Rota 8	47221 m
Rota 9	42020 m
Rota 10	67795 m
Rota 11	48735 m
Rota 12	35699 m
Rota 13	37863 m
Rota 14	40352 m
Rota 15	44490 m
Rota 16	46849 m
Rota 17	36261 m
Rota 18	17972 m
TOPLAM	***874356 m***

Tablo 3.3'te görüldüğü üzere toplamda 18 rota oluşturulmuş ve bu 18 rota için toplam uzaklık miktarı 874356 metre olmuştur. Örnek rota görselleri Şekil 3.20 ve Şekil 3.21'de gösterilmiştir.



Şekil 3.20 Tasarruf Algoritması Sonucunda Oluşan Rota — 1



Şekil 3.21 Tasarruf Algoritması Sonucunda Oluşan Rota — 2

Oluşan rotalarda görüldüğü üzere, tasarruf algoritmasının yapısı gereği, başlangıç noktasına en uzak ve birbirlerine en yakın ikilileri seçerek rotalar oluşturmaktadır. Bu durum, aynı bölgede olmamalarına karşın tasarruf miktarları üst sırada olduğu durumlarda, farklı bölgelerdeki noktaların bir araya gelmesine neden olmaktadır.

Algoritma bu haliyle bile varsayılan çözümden daha iyi sonuç vermiştir. Ancak bu sonuç anlamlı bir seviyede değildir. Algoritmanın performansını artırmak için, tüm noktalar önce kümelenip sonrasında tasarruf algoritması çalıştırılacaktır.

3.7. Kümele Algoritmaları ile Beraber Rotalama

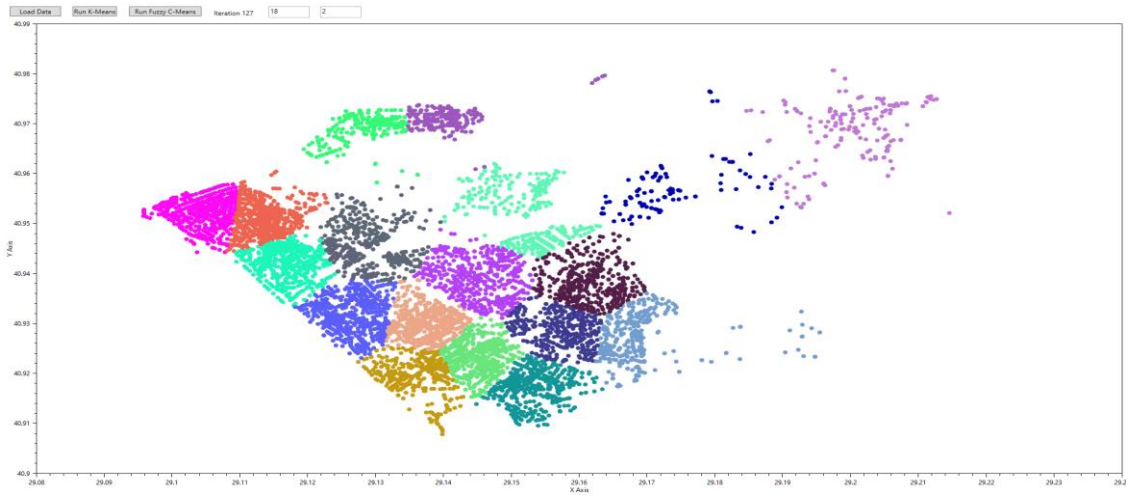
Algoritma performansının yükseltilmesi ve konum olarak benzer noktaların bir araya getirilmesini adına, tüm noktalar üzerinde kümele algoritmaları çalıştırılacaktır. Bu algoritmalar, K-Ortalamalar kümeleme algoritması ve Bulanık C-Ortalamalar kümeleme algoritmasıdır. Bu algoritmaların çalıştırılmasın sonrasında, her bir küme için tasarruf algoritması çalıştırılacak ve kümeler için ayrı ayrı rotalar oluşturulması sağlanacaktır.

3.7.1. K-Ortalamalar Kümelemesi ile Algoritmanın Uygulanması

K-ortalamalar algoritması kümeleme algoritmaları içerisinde en yaygın kullanılan algoritmalardan birisidir. Her bir kümenin küme merkezleri, içlerinde bulunan elemanların özelliklerinin ortalaması şeklinde hesaplanmaktadır. Her bir nokta kendisine en yakın kümeye atanıp sonrasında, bu işlem kümelerde herhangi bir değişiklik olmayıncaya kadar devam ettirilmektedir.

Bu çalışmada K-ortalamalar kümeleme algoritması 7482 nokta için çalıştırılmış ve 18 küme oluşturulması sağlanmıştır. Küme algoritmasının çalıştırılması için C# üzerinde WPF uygulaması geliştirilmiştir.

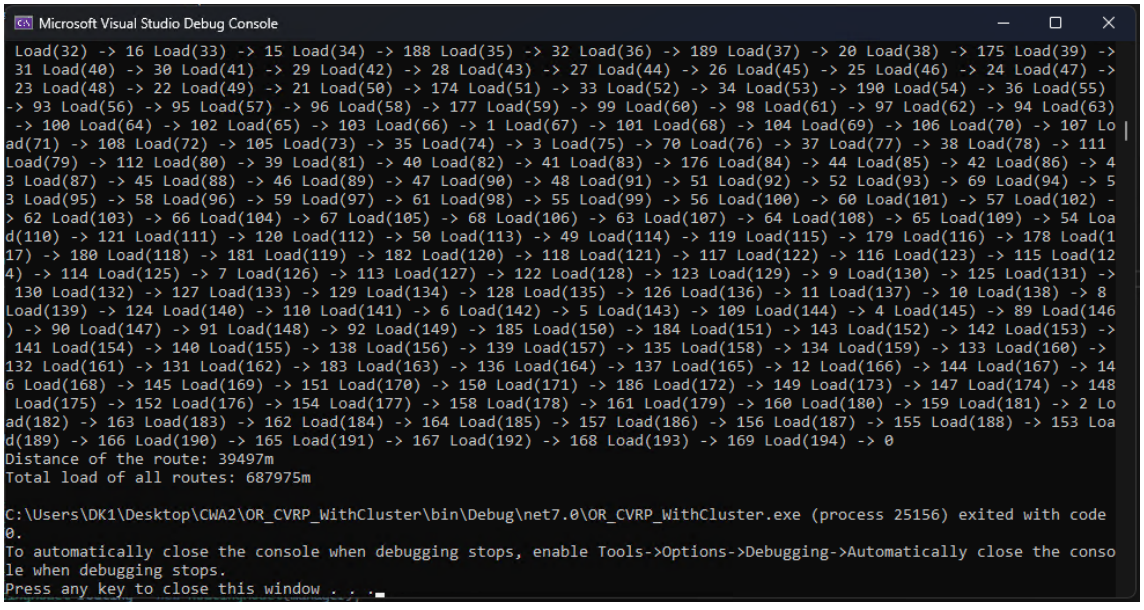
Algoritmanın çalıştırılması sonucunda, algoritma 47 iterasyon sürmüştür. Durma koşulu olan değişikliğin olmaması akabinde 18 ayrı küme oluşturulmuştur. Oluşturulan kümeler Şekil 3.22'de gösterilmiştir.



Şekil 3.22 K-Ortalamalar Kümelemesi Sonucu Oluşan Kümeler

Oluşan kümeler, JSON formatındaki dosyalar ile dışarı aktarılmıştır.

Geliştirilen C# konsol uygulaması ile tasarruf algoritması, her bir oluşturulan küme için ayrı ayrı çalıştırılmıştır. Sonuçlara göre, önce K-Ortalamalar kümeleme algoritması ardından tasarruf algoritmasının çalıştırıldığı durumda, toplam 687975 m rota uzunluğu hesaplamıştır. Rotaların nokta sayılarını ve toplam uzaklıklarını içeren Tablo 3.4'te gösterilmiştir. K-ortalamalar sonrasında çalıştırılan tasarruf algoritmasının örnek konsol çıktısı Şekil 3.23'deki gibidir.

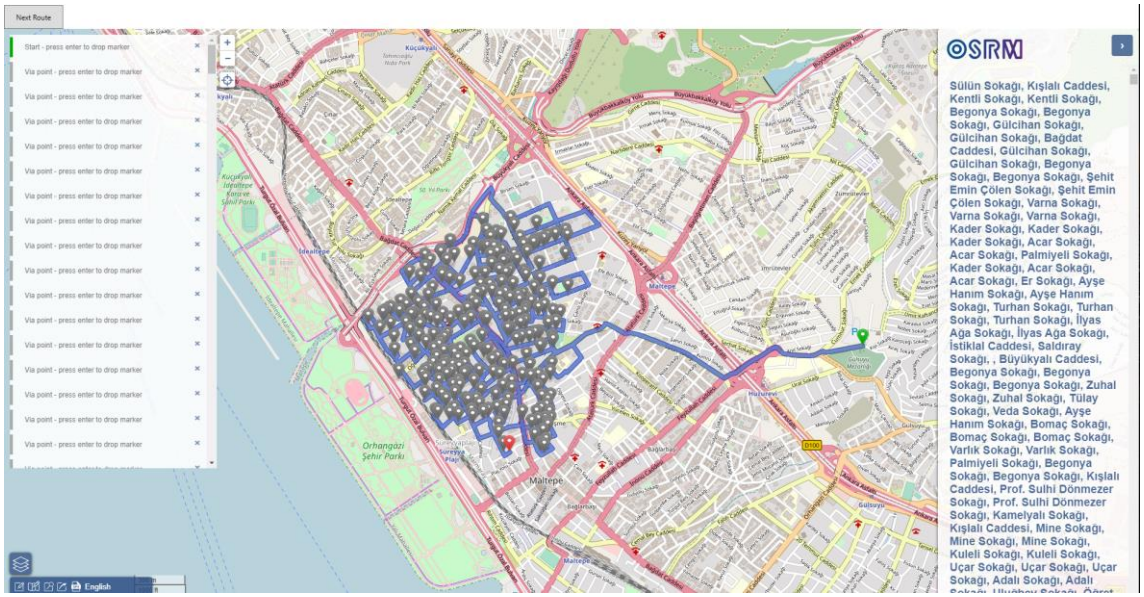


Şekil 3.23 K-Ortalamalar Sonrasında Çalıştırılan Tasarruf Algoritması Konsol Çıktıları

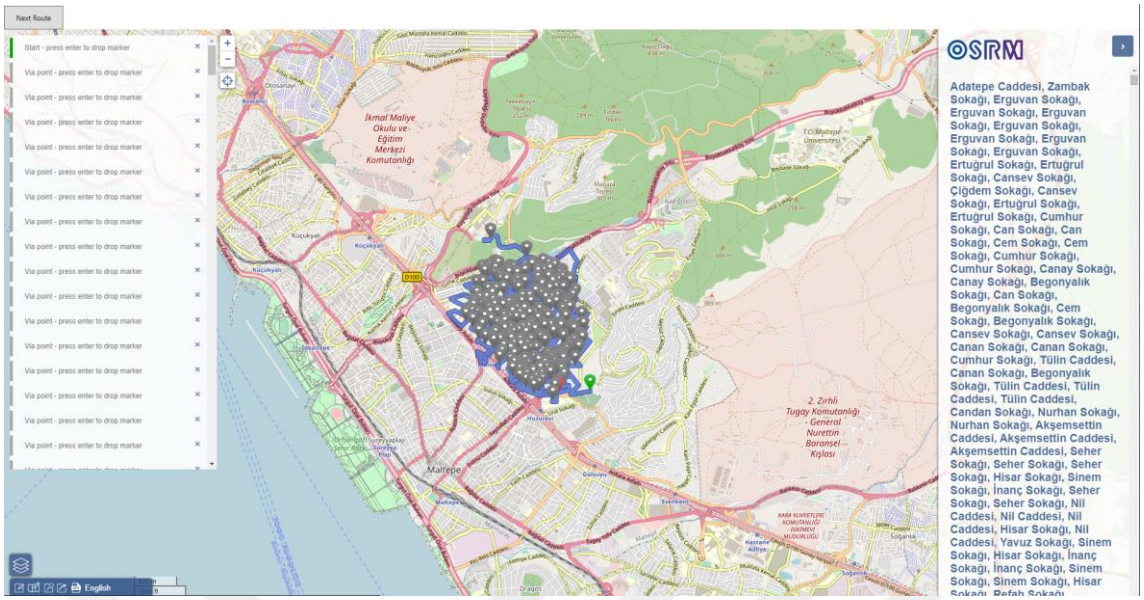
Tablo 3.4 K-Ortamalar Sonrasında Tasarruf Algoritması Rota Uzaklıkları

Rota İsmi	Nokta Sayısı	Rota Uzaklığı
Rota 1	583	41108 m
Rota 2	481	37127 m
Rota 3	114	33840 m
Rota 4	390	38163 m
Rota 5	214	35032 m
Rota 6	224	36148 m
Rota 7	340	36951 m
Rota 8	444	46293 m
Rota 9	359	43077 m
Rota 10	609	49696 m
Rota 11	657	46900 m
Rota 12	598	40763 m
Rota 13	550	33455 m
Rota 14	368	30478 m
Rota 15	536	33479 m
Rota 16	456	37768 m
Rota 17	364	28200 m
Rota 18	194	39479 m
TOPLAM	***7482***	***687957 m***

Oluşturulan rotalardan ilk ikisine ait gösterim Şekil 3.24 ve Şekil 3.25'de gösterilmiştir.



Şekil 3.24 K-Ortamalar Sonrasında Tasarruf Algoritması Rota — 1



Şekil 3.25 K-Ortamalar Sonrasında Tasarruf Algoritması Rota — 2

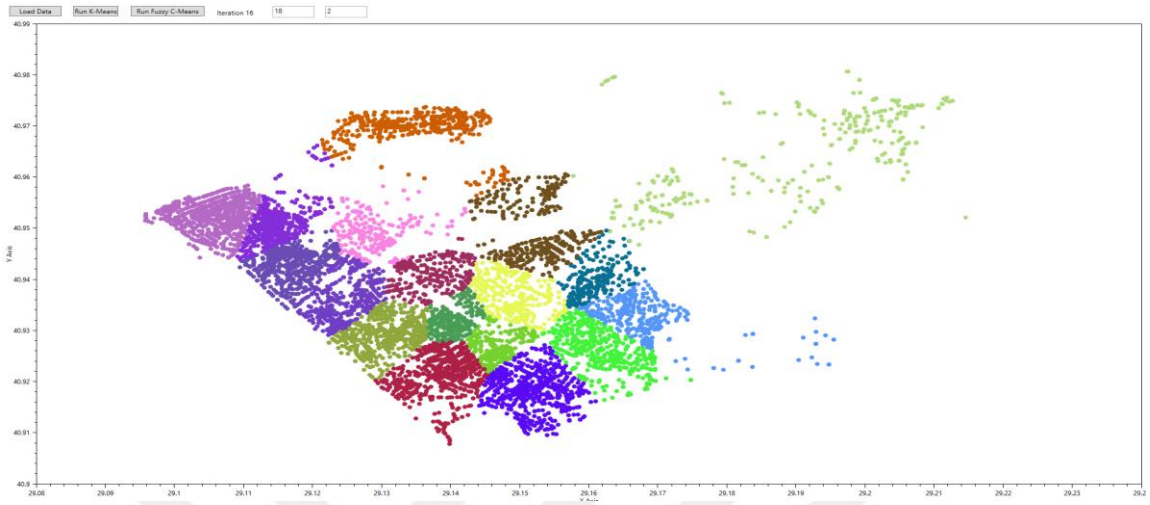
Rota 1 ve Rota 2 için paylaşılan şekillerde de görüldüğü üzere, kümeleme algoritması sonrasında oluşan rotalar, gerçek hayat verisine daha uygun şekilde tutarlı ve birbirlerine yakın bölgelerde bulunmaktadır. Bu tutarlılık benzer şekilde, toplam mesafenin azalmasında da anlaşılmaktadır.

3.7.2. Bulanık C-Ortalamalar Kümelemesi ile Algoritmanın Uygulanması

Bulanık C-ortalamalar algoritması kümeleme algoritmaları içerisinde en yaygın kullanılan algoritmalardan birisidir. Çalışma yapısı olarak K-ortalamalar algoritmasına benzemektedir. Farklı olarak aslında her bir elemen, tüm kümelerde yer almaktadır. Elemanların kümelerde bulunma üyeliklerini gösteren üyelik katsayısı bunu belirtmektedir. Her bir kümenin küme merkezleri, içlerinde bulunan elemanların özelliklerinin ortalaması üyelik katsayıları ile orantılanarak hesaplanmaktadır. Durma koşulu olarak değişikliğin çok az olduğu bir alt sınır ve iterasyon sayısı belirlenebilmektedir.

Bu çalışmada Bulanık C-ortalamalar kümeleme algoritması 7482 nokta için çalıştırılmış ve 18 küme oluşturulması sağlanmıştır. Durma koşulu epsilon $10^{-5,}$ iterasyon sayısı ise 200 olarak belirlenmiştir. M katsayısı ise 2 olarak varsayılan değerinde bırakılmıştır. Küme algoritmasının çalıştırılması için C# üzerinde WPF uygulaması geliştirilmiştir.

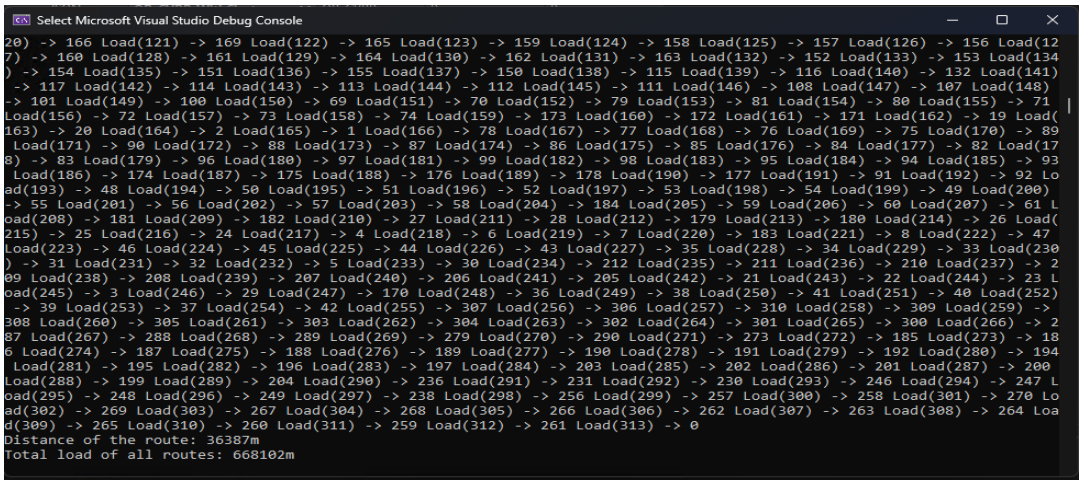
Algoritmanın çalıştırılması sonucunda, algoritma 16 iterasyon sürmüştür. Durma koşulu olan değişikliğin olmaması akabinde 18 ayrı küme oluşturulmuştur. Oluşturulan kümeler Şekil 3.26'da gösterilmiştir.



Şekil 3.26 Bulanık C-Ortalamalar Kümelemesi Sonucu Oluşan Kümeler

Oluşan kümeler, JSON formatındaki dosyalar ile dışarı aktarılmıştır.

Geliştirilen C# konsol uygulaması ile tasarruf algoritması, her bir oluşturulan küme için ayrı ayrı çalıştırılmıştır. Sonuçlara göre, önce Bulanık C-Ortalamalar kümeleme algoritması ardından tasarruf algoritmasının çalıştırıldığı durumda, toplam 668102 m rota uzunluğu hesaplamıştır. Rotaların nokta sayılarını ve toplam uzaklıklarını içeren Tablo 3.5'te gösterilmiştir. Bulanık C-ortalamalar sonrasında çalıştırılan tasarruf algoritmasının örnek konsol çıktısı Şekil 3.27' deki gibidir.

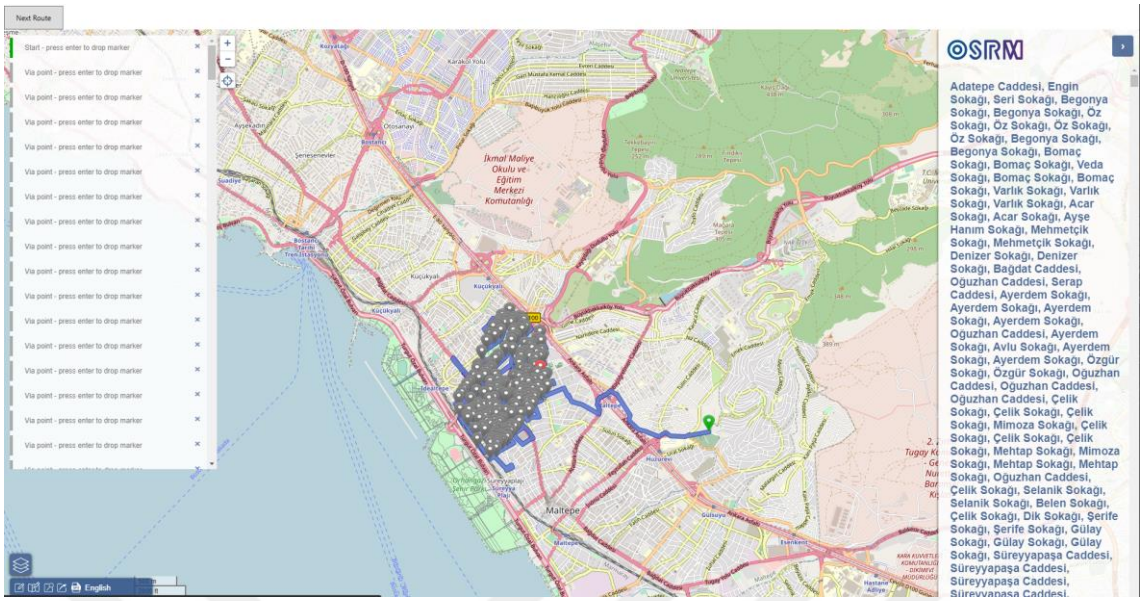


Şekil 3.27 Bulanık C-Ortalamalar Sonrasında Çalıştırılan Tasarruf Algoritması Konsol Çıktıları

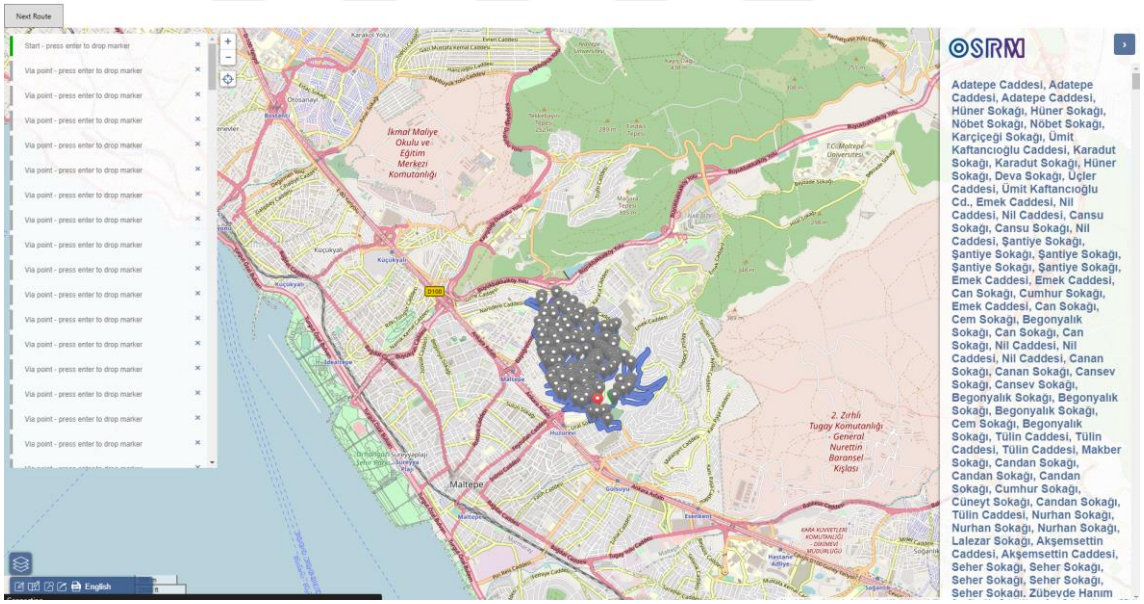
Tablo 3.5 Bulanık C-Ortalamalar Sonrasında Tasarruf Algoritması Rota Uzaklıkları

Rota İsmi	Nokta Sayısı	Rota Uzaklığı
Rota 1	418	33901 m
Rota 2	310	22100 m
Rota 3	321	61837 m
Rota 4	209	23201 m
Rota 5	447	55326 m
Rota 6	524	43491 m
Rota 7	376	36141 m
Rota 8	256	36089 m
Rota 9	444	40397 m
Rota 10	761	54894 m
Rota 11	565	40292 m
Rota 12	515	39264 m
Rota 13	261	15218 m
Rota 14	257	22529 m
Rota 15	662	44991 m
Rota 16	228	14259 m
Rota 17	614	47785 m
Rota 18	314	36387 m
TOPLAM	***7482***	***668102 m***

Oluşturulan rotalardan ilk ikisine ait gösterim Şekil 3.28 ve Şekil 3.29'da gösterilmiştir. Rota 1 ve Rota 2 için paylaşılan şekillerde de görüldüğü üzere, k-ortalamalar kümeleme algoritmasına benzer şekilde, kümeleme algoritması sonrasında oluşan rotalar, gerçek hayat verisine daha uygun şekilde tutarlı ve birbirlerine yakın bölgelerde bulunmaktadır. Bu tutarlılık benzer şekilde, toplam mesafenin azalmasında da anlaşılmaktadır.



Şekil 3.28 Bulanık C-Ortalamalar Sonrasında Tasarruf Algoritması Rota — 1



Şekil 3.29 Bulanık C-Ortalamalar Sonrasında Tasarruf Algoritması Rota — 2

4. BULGULAR VE TARTIŞMA

Araç rotalama problemi, her problemin kendine has kısıtlar ve koşullar içermesi nedeniyle genel geçer bir çözüm metoduna sahip olamamaktadır. Bu nedenle, probleminin çözümüne yönelik çalışmalarda; melez, birbirinin ön adımı veya son adımı olarak farklı algoritmaların kullanıldığı görülmektedir.

Bu çalışmadaki temel motivasyon ise, sezgisel bir rotalama algoritması olan tasarruf algoritmasının kümeleme algoritma ile beraber kullanılması ve rotalama performansında artış görünmesinin gözlenmesidir.

Veri seti olarak gerçek hayat verisi olarak Maltepe Belediyesi Temizlik İşleri Müdürlüğü tarafında sağlanıştır. Maltepe Belediyesine ait 18 Mahallenin çöp konteynırlarına ait lokasyonları içermektedir [39]. Veri seti öncelikle ön hazırlık adımından geçirilerek temizlenmiş. Sonrasında noktaların birbirleriyle olan uzaklıkları hesaplanmıştır. Bu kısımda gerçek hayat verisi kullanılması nedeniyle, gerçek uzaklıkların kullanılması gerekmiştir. Bu durum, OSRM'in arkayüz yazılımının sanal bilgisayar üzerinde çalıştırılması ile hesaplanmıştır.

Hazırlanan uzaklıklar matrisi, öncelikle belediyenin paylaşmış olduğu rota için kullanılmış ve hesaplanan uzaklık, ilk rota uzaklığı olarak kabul edilmiştir.

Sonrasında, tüm noktalar birleştirilerek 7482 nokta için, toplamda 18 rota oluşturacak kısıtlama ile tasarruf algoritması çalıştırılmıştır. Buradaki tasarruf algoritmasının kümeleme yapmadan önceki performansının saptanmasıdır.

Ardından, veri seti üzerinde tasarruf algoritması, iki farklı rotalama algoritmasının ön işleme adımı olarak kullanılması ile birlikte çalıştırılmıştır. Buradaki amaç, tasarruf algoritmasının tasarruflarının fazla olması nedeniyle, birbirinden uzak bölgelerdeki noktaları bir araya getirmesinin önüne geçmektir.

Tüm bu adımlar sonucunda ortaya çıkan rotalar incelenmiş ve aşağıdaki Tablo 4.1'deki gibi özetlenmiştir.

Tablo 4.1 Rotalamaların Nokta Sayısı ve Rota Uzaklıkları

Varsayılan		Tasarruf Algoritması		K-Ortalamalar ile birlikte Tasarruf Algoritması		Bulanık C-Ortalamalar ile birlikte Tasarruf Algoritması	
Nokta Sayısı	Rota Uzaklığı	Nokta Sayısı	Rota Uzaklığı	Nokta Sayısı	Rota Uzaklığı	Nokta Sayısı	Rota Uzaklığı
486	39579 m	461	79699 m	583	41108 m	418	33901 m
470	57095 m	441	52096 m	481	37127 m	310	22100 m
242	40383 m	451	42470 m	114	33840 m	321	61837 m
857	78002 m	471	65356 m	390	38163 m	209	23201 m
186	32851 m	407	79437 m	214	35032 m	447	55326 m
281	84556 m	478	42149 m	224	36148 m	524	43491 m
714	88525 m	432	47892 m	340	36951 m	376	36141 m
357	27327 m	427	47221 m	444	46293 m	256	36089 m
372	61350 m	434	42020 m	360	43077 m	444	40397 m
288	27310 m	463	67795 m	609	49696 m	761	54894 m
421	56935 m	418	48735 m	657	46900 m	565	40292 m
193	23757 m	425	35699 m	598	40763 m	515	39264 m
235	26749 m	399	37863 m	550	33455 m	261	15218 m
352	34638 m	407	40352 m	368	30478 m	257	22529 m
494	39755 m	386	44490 m	536	33479 m	662	44991 m
613	43977 m	477	46849 m	456	37768 m	228	14259 m
245	32938 m	381	36261 m	364	28200 m	614	47785 m
676	89580 m	124	17972 m	194	39479 m	314	36387 m
7482	***885307 m***	***7482***	***874356 m***	***7482***	***687957 m***	***7482***	***668102 m***

Sonuçlar incelendiğinde, sadece tasarruf algoritmasının sonuçları anlamlı bir optimizasyon sunmuyor olsa da veri seti üzerinde 3 varyasyon da ilk sonuçlardan daha kısa rotalar çıkartarak daha kısa rotalar ortaya koymuştur. Rotalarda tasarruf miktarları Tablo 4.2'deki gibidir.

Tablo 4.2 Rotalama Algoritmaların Performansları

	Rota Uzunluğu	İyileşme Oranı
Varsayılan	***885307***	**-**
Tasarruf Algoritması	***874356***	***1,252%***
K-Ortalamalar ile birlikte Tasarruf Algoritması	***687957***	***28,686%***
Bulanık C-Ortalamalar ile birlikte Tasarruf Algoritması	***668102***	***32,511%***

Elde edilen sonuçlara göre, tasarruf algoritması ön işleme adımı olmadan, varsayılan rotaya göre, rota uzunluğu olarak %1,252'lik bir iyileşme göstermiştir. Verilerde ön işleme adımı olarak K-ortalamalar kümeleme algoritmasının kullanıldığı tasarruf algoritması varyasyonunda ise ilk rota uzunluğuna göre %28,686'lık bir iyileşme gözlenmiştir. Bulanık C-ortalamalar algoritmasının, tasarruf algoritmasının ön adımı olarak kullanıldığı varyasyonda ise, ilk rota uzunluğuna göre %32,511'lik bir iyileşme hesaplanmıştır. Elde edilen sonuçlara bakıldığında, tasarruf algoritmasının ön adımı olarak kümeleme algoritmalarının kullanılmasının, özellikle büyük veri setleri ile çalışıldığında, algoritmanın performansını olumlu etkilediği gözlemlenmiştir.

5. SONUÇLAR

Tasarruf algoritması, araç rotalama problemi için sezgisel çözüm sunan bir algoritmadır. Algoritmanın akışı gereği, depoya en uzak ve birbirine en yakın ikilileri bir araya getirmeyi amaçlayan bu algoritma, çok büyük veri setlerinde bu ikilileri seçerken, ikililerin birbirine yakınlığı dikkate almayabilmektedir. Bu nedenle, birbirine uzak ikililer, sadece tasarruf miktarları fazla olduğu için bir araya gelebilmekte ve toplam rota maliyetinin artmasına neden olabilmektedir.

Bu çalışmada, tasarruf algoritmasının, daha iyi sonuçlar vermesi amacıyla birbirine geometrik olarak yakın noktaların bir araya getirilmesi amaçlanmıştır. Bu sayede tasarruf algoritmasının performansının artırılmasının gözlemlenmesi hedeflenmiştir.

Çalışmada veri seti olarak Maltepe Belediyesinden alınan KML dosyaları kullanılmıştır. Bu dosyalar, Maltepe belediyesinin sınırları içerisinde bulanan mahallelerdeki çöp kutularının lokasyon bilgilerini içermektedir. Bu veri setinin işlenmesi ve rotalamaya hazır hale getirilmesi bu tezin amaçlarından biridir. Veri içerisindeki gerçek uzaklıklar hesaplanmış varsayılan rota uzaklığı ortaya çıkarılmıştır.

Önce tasarruf algoritması, benzer rotaları oluşturacak şekilde çalıştırılmış, sonuçlar kaydedilmiştir. Sonrasında tasarruf algoritması, kümeleme algoritmalarının oluşturduğu kümeler ile çalıştırılmıştır. Sonuçlar kaydedilmiştir. Bu tezin ilk adımı olan verinin işlenerek, kümeleme ve tasarruf algoritmaları için kullanılabilir bir hale getirilmesi tamamlanmış olup, tezin çalışmasının bu kısmı başarılı bir şekilde gerçekleştirilmiştir.

Tezin ikinci amacı olan, mevcut rotanın iyileştirilmesi kısmı, tasarruf algoritmasının 3 farklı varyasyonu ile denenmiştir. Elde edilen sonuçlar incelediğinde, yalnızca tasarruf algoritmasının çalıştırıldığı varyasyonda anlamlı bir fark ortaya koyulamadığı, ancak ön işleme adımı olarak kümeleme algoritmalarının kullanıldığı varyasyonlarda ise, %28.686'lık ve %32.511'lik iyileşmeler olduğu gözlenmiştir. Bu durum tasarruf algoritmasının kullanılması öncesinde, ön işleme adımı olarak kümeleme algoritmalarının kullanılmasının, tasarruf algoritmasının performansını artırdığını ortaya koymaktadır. Bu durum tezin ikinci amacı olan, rotalama performansının artışının ve rota maliyetinin azalmasının başarılı şekilde ortaya koyulduğunu göstermektedir.

Bu tez çalışmasında, birçok varsayım ile çalışılmıştır. Bu varsayımlar, ilgili varsayımlar kısmında listelenmiştir. Gerçek hayata daha uygulanabilir çözümlerin, gerçekçi bir şekilde yansıtılabilmesi için, varsayımların olabildiğince azaltılması gerekmektedir. Bu kapsamda, geliştirilecek olan başka çalışmalarda bu varsayımların azaltılması yönünde ilerlenebilir. Ayrıca, çalışmadan görüldüğü üzere, sezgisel algoritmaların öncesinde, ön işleme adımı olarak kümeleme algoritmaların çalıştırılması, sezgisel algoritmaların, çözüm uzayında daha optimum sonuçların olduğu alanları taramalarına neden olabilmektedir. Bu nedenle K-Ortalamalar ve Bulanık C-Ortalamalar kümeleme algoritmaları, başka sezgisel algoritmalar ile beraber kullanılıp, rotalama maliyeti üzerine olan etkileri izlenebilir.

KAYNAKLAR

[1] Dantzig, G. B., & Ramser, J. H. (1959) The Truck Dispatching Problem. *Management Science*, 6(1), pp. 80-91.

[2] Clarke, G., & Wright, J. W. (1964) Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Operations Research*, 12(4), pp. 568-581.

[3] Toth, P., & Vigo, D. (Eds.). (2002). *The vehicle routing problem*. Society for Industrial and Applied Mathematics.

[4] Moghaddam, B. F., Ruiz, R., & Sadjadi, S. J. (2012). Vehicle routing problem with uncertain demands: An advanced particle swarm algorithm. *Computers & Industrial Engineering*, *62*(1), 306-317.

[5] Osman, Ibrahim Hassan. "Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem." *Annals of operations research* 41.4 (1993): 421-451.

[6] Tanyaş, M. (2009). Lojistik ve Tedarik Zinciri Yönetimi Sunumu. *İstanbul: Yeditepe Üniversitesi Lojistik Kulübü*.

[7] Bowerman, R., Hall, B., & Calamai, P. (1995). A multi-objectıve optımızatıon approach to urban school bus routıng: formulatıon and solutıon method. *Transpn. Res-A, 29A(2),* 107-123.

[8] Kara, İ., & Bektaş, T. (2006). Integer linear programming formulations of multiple salesman and its variations. *European Journal of Operational Research,* 174, 1449–1458

[9] Maden, W., Eglese, R., & Black, D. (2010). Vehicle routing and scheduling with time varying data: a case study. *The Journal of the Operational Research Society, 61(3),* 1524-1532.

[10] Desaulniers, G. (2010). Branch-and-price-and-cut for the split-delivery vehicle routing problem with time windows. *Operations Research, 58(1),* 179-192.

[11] Archetti, C., Bouchard, M., & Desaulniers, G. (2011). Enhanced branch and price and cut for vehicle routing with split deliveries and time windows. *Transportation Science, 45(3),* 285-298.

[12] Çetin, S., & Gencer, C. (2011). Heterojen araç filolu zaman pencereli eş zamanlı dağıtım-toplamalı araç rotalama problemleri: matematiksel model. *International Journal of Research and Development, 3(1),* 19-27.

[13] Atmaca, E. (2012). Bir kargo şirketinde araç rotalama problemi ve uygulaması. *TÜBAV Bilim Dergisi, 5(2),* 12-27.

[14] Koç, Ç. (2012). Çok kullanımlı ve zaman pencereli araç rotalama problemi için bir matematiksel model. *Gazi Üniv. Müh. Mim. Fak. Der, 27(3),* 569-576.

[15] Zachariadis, E., Tarantilis, C., & Kiranoudis, C. (2012). The pallet-packing vehicle routing. *Transportation Science, 46(3),* 341-558.

[16] Lee, C., Lee, K., & Park, S. (2012). Robust vehicle routing problem with deadlines and travel time/demand uncertainty. *The Journal of the Operational Research Society, 63(9),* 1294-1306.

[17] Vansteenwegen, , P., Souffriau, W., & Sörensen, K. (2012). The travelling salesperson problem with hotel selection. *Journal of the Operational Research Society, 63,* 207-217.

[18] Wang, Z., Liang, W., & Hu, X. (2014). A metaheuristic based on a pool of routes for the vehicle routing problem with multiple trips and time windows. *Journal of the Operational Research Society, 65,* 37-48.

[19] Hezer, S., & Kara, Y. (2013). Eşzamanlı dağıtım ve toplamalı araç rotalama problemlerinin çözümü için bakteriyel besin arama Optimizasyonu Tabanlı Bir Algoritma. *Gazi Üniv. Müh. Mim. Fak. Der.*, 28(2), 373-382.

[20] Martinez, L., & Amaya, C. (2013). A vehicle routing problem with multi-trips and time windows for circular items. *Journal of the Operational Research Society 64*, 1630-1643.

[21] Eryavuz, M., & Gencer, C. (2001) Araç Rotalama Problemine Ait Bir Uygulama. *Süleyman Demirel Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, *6*(1), pp. 139-155.

[22] Baker, B. M., & Ayechew, M. A. (2003) A Genetic Algorithm for the Vehicle Routing Problem. *Computers & Operations Research*, 30(5), pp. 787-800.

[23] Nuortio, T., Kytöjoki, J., Niska, H., & Bräysy, O. (2006) Improved Route Planning and Scheduling of Waste Collection and Transport. *Expert Systems With Applications*, 30(2), pp. 223-232.

[24] Ustundag, Alp., & Cevikcan, E. (2008) Vehicle Route Optimization for RFID Integrated Waste Collection System. *International Journal of Information Technology & Decision Making*, 7(4), pp. 611-625.

[25] Freitas, T. R., Coelho, A., & Rossetti, R. J. (2010) Correcting Routing Information Through GPS Data Processing. In *13th International IEEE Conference On Intelligent Transportation Systems*, pp. 706-711.

[26] Erdoğan, S., & Miller-Hooks, E. (2012) A Green Vehicle Routing Problem. *Transportation*

Research Part E: Logistics and Transportation Review, 48(1), pp. 100-114.

[27] Güvez, H., Dege, M., & Eren, T. (2012) Kırıkkale'de Araç Rotalama Problemi ile Tıbbi Atıkların Toplanması. *International Journal of Engineering Research and Development*, 4(1), pp. 41-45.

[28] Bozyer, Z., Alkan, A., & Fığlalı, A. (2014) Kapasite Kısıtlı Araç Rotalama Probleminin Çözümü İçin Önce Grupla Sonra Rotala Merkezli Sezgisel Algoritma Önerisi. *Bilişim Teknolojileri Dergisi*, 7(2), pp. 29-37.

[29] Yazgan, H. R., Ercan, S., & Arslan, C. (2014) Talep ve Kapasite Kısıtlı Optimizasyon Problemi İçin Yeni Bir Melez Algoritma. *Endüstri Mühendisliği*, 25(1), pp. 16-28.

[30] Das, S., & Bhattacharyya, B. K. (2015) Optimization of Municipal Solid Waste Collection and Transportation Routes. *Waste Management*, 43, pp. 9-18.

[31] Aziz, R., Kedia, M., Dan, S., Basu, S., Sarkar, S., Mitra, S., & Mitra, P. (2016) Identifying and Characterizing Truck Stops from GPS Data. In *Industrial Conference on Data Mining*, pp. 168-182.

[32] Nowakowski, P., Szwarc, K., & Boryczka, U. (2018) Vehicle Route Planning in E-Waste Mobile Collection on Demand Supported By Artificial Intelligence Algorithms. *Transportation Research Part D: Transport And Environment*, 63, pp. 1-22.

[33] Žunić, E., Hindija, H., Beširević, A., Hodžić, K., & Delalić, S. (2018) Improving Performance of Vehicle Routing Algorithms Using GPS Data. In *2018 14th Symposium on Neural Networks and Applications (NEUREL)*, pp. 1-4.

[34] Çetin, O., & Özçakar, N. (2019) Akaryakıt Dağıtımında Araç Rotalama Problemi İçin Bir Başlangıç Çözümü. *Trakya Üniversitesi Sosyal Bilimler Dergisi*, 21(2), pp. 461-474.

[35] Özoğlu, B., Çakmak, E., & Tuğçe, K. O. Ç. (2019) Clarke & Wright's Savings Algorithm and Genetic Algorithms Based Hybrid Approach for Flying Sidekick Traveling Salesman Problem. *Avrupa Bilim ve Teknoloji Dergisi*, pp. 185-192.

[36] Hannan, M. A., Begum, R. A., Al-Shetwi, A. Q., Ker, P. J., Al Mamun, M. A., Hussain, A., Basri, H., & Mahlia, T. M. I. (2020) Waste Collection Route Optimisation Model for Linking Cost Saving and Emission Reduction to Achieve Sustainable Development Goals. *Sustainable Cities and Society*, 62, 102393.

[37] Žunić, E., Delalić, S., & Đonko, D. (2020) Adaptive Multi-Phase Approach for Solving the Realistic Vehicle Routing Problems in Logistics with Innovative Comparison Method for Evaluation Based on Real GPS Data. *Transportation Letters*, pp. 1-14.

[38] Özalp, M., & Alp, S. (2020) Uygun Dağıtım Rotası Belirlenmesi Probleminde Hibrit Sezgisel Bir Yöntem Önerisi: Bir Kargo Firması Örneği. *Akıllı Ulaşım Sistemleri ve Uygulamaları Dergisi*, 3(1), pp. 59-70.

[39] Akdaş, Hatice Şahin, et al. "Vehicle Route Optimization for Solid Waste Management: A Case Study of Maltepe, Istanbul." *2021 13th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*. IEEE, 2021.

[40] Gu, Ruixue, et al. "A hierarchical solution evaluation method and a hybrid algorithm for the vehicle routing problem with drones and multiple visits." *Transportation Research Part C: Emerging Technologies* 141 (2022): 103733.

[41] Zhang, Zheng, Bin Ji, and Samson S. Yu. "An Adaptive Tabu Search Algorithm for Solving the Two-dimensional Loading Constrained Vehicle Routing Problem with Stochastic Customers." *Sustainability* 15.2 (2023): 1741.

[42] Wang, Yong, et al. "A clustering-based extended genetic algorithm for the multidepot vehicle routing problem with time windows and three-dimensional loading constraints." *Applied Soft Computing* 133 (2023): 109922.

EKLER

ÖZGEÇMİŞ

EĞİTİM

Bölüm	Kurum	Derece	Mezuniyet Tarihi
Bilgisayar Mühendisliği	Marmara Üniversitesi Fen Bilimleri Enstitüsü	Yüksek Lisans	Şubat 2023
Endüstri Mühendisliği	İstanbul Üniversitesi Mühendislik Fakültesi	Lisans	Haziran 2018
Sayısal	Samiha Ayverdi Anadolu Lisesi	Lise	Haziran 2014

MESLEKİ DENEYİM

Birim	Kurum	Pozisyon	Başlangıç Tarihi	Ayrılma Tarihi
Yazılım Geliştirme	TAAC Havacılık Teknolojileri A.Ş.	Yazılım Mühendisi	Mayıs 2021	-
Yazılım Geliştirme	Altınay Savunma Teknolojileri A.Ş.	Yazılım Geliştirme Mühendisi	Mayıs 2020	Mayıs 2021
Yazılım Geliştirme	Op Bilişim Danışmanlık Arge ve Yazılım Tic. Ltd.Şti	Yazılım Geliştirici	Ocak 2019	Mayıs 2020
Yazılım Geliştirme	MARS Bilgi Teknolojileri A.Ş.	Yazılım Geliştirici	Eylül 2018	Aralık 2018

BİLİMSEL ESERLER

Buyukozdemir Kerem, Yildiz, Kazim, Bas, Anil, & Uslu Banu Calis. (2021, December). A Review of Heuristic Approaches to Vehicle Routing Problems. In *2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)* (pp. 1-5). IEEE.