T.C.
NECMETTİN ERBAKAN ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ZAMAN PENCERELİ ARAÇ ROTALAMA PROBLEMİNİN ÇÖZÜMÜ İÇİN HİBRİT BİR YÖNTEMİN GELİŞTİRİLMESİ

Zekai ÖZKÖK

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı

Aralık-2021
KONYA

ÖZET

YÜKSEK LİSANS TEZİ

ZAMAN PENCERELİ ARAÇ ROTALAMA PROBLEMİNİN ÇÖZÜMÜ İÇİN HİBRİT BİR YÖNTEMİN GELİŞTİRİLMESİ

Zekai ÖZKÖK

Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Assoc. Prof. Dr. İlhan İLHAN

2021, 92 sayfa

Jüri

Doç. Dr. İlhan İLHAN
Doç. Dr. Mehmet HACIBEYOĞLU
Dr. Öğr. Üyesi Mehmet Akif ŞAHMAN

Küreselleşmeyle birlikte rekabetin hızla arttığı bir dünyada firmaların sürdürülebilir başarı sağlamasında müşteri memnuniyeti belirleyici faktörlerden biridir. Müşterilerin beklentilerine cevap verebilmek için güçlü bir tedarik alt yapısı oluşturmak hayati derecede önemlidir. Tedarik zincirindeki en önemli süreçlerden biri de rota planlamasıdır. Günümüzde araç rotalama problemleri üzerine planlama yapmak özellikle lojistik ve hizmet sektöründe faaliyet gösteren şirketler için önemli hale gelmiştir. Artan talepleri karşılamak ve maliyetleri azaltmak için geliştirilen iş süreçlerinde rotalama ve optimizasyon çözümleri şirketler için oldukça önemlidir. Kötü yönetilen rota planları, şirketlerin müşterilerine hizmet vermesi açısından sorun teşkil etmekte, diğer lojistik ve hizmet süreçlerinde aksamalara neden olmaktadır. Hızla gelişen ve değişen küresel rekabet ortamı işletmeleri daha iyi süreç yönetimi için planlama yapmaya zorlamaktadır.

Bu planlamanın doğru yapılması için hem müşteri memnuniyetine önem verilmeli hem de işletme maliyetleri düşürülmelidir. Araç rotalama maliyetleri, lojistik sistem içindeki taşıma ve dağıtım maliyetlerinin önemli bir parçasıdır. Araç rotalama probleminin amacı, bir araç filosu için en küçük maliyetli rota kümesini tasarlamaktır. Araç rotalama problemlerinden biri olan zaman pencereli araç rotalama problemi (ZPARP) birçok araştırmacının çalışma alanını oluşturmaktadır. ZPARP her bir müşteriye ait bir zaman aralığı kısıtı olan araç rotalama problemidir. Bu problemde dağıtım aracı, her bir müşteriye belirli bir zaman aralığında hizmet vermek zorundadır.

Bu çalışmada, ZPARP'nin çözümü için yapay arı kolonisi (Artificial Bee Colony-ABC) algoritması ile Order Crossover (OX1) operatörü melezleştirilmiş ve ABC-OX1 olarak adlandırılan hibrit bir yöntem önerilmiştir. ABC-OX1 yönteminde başlangıç çözümleri rastgele olarak üretilmiş ve bu çözümler tur geliştirici 3-opt algoritması ile iyileştirilmiştir. İşçi arı fazında insertion, swap ve divideandswap yerel arama operatörleri eşit olasılıkla kullanılmıştır. Gözcü arı fazında OX1 operatörü ile besin çeşitliliği sağlanmış ve kaliteli çözümler üzerinde aramaya devam edilmiştir. Kâşif arı fazında ise yeni çözümler rastgele oluşturulmuş ve bu çözümler tur geliştirici 2-opt algoritması ile iyileştirilmiştir. ABC-OX1 yönteminde popülasyondaki çözümlerin uygunluk değerlendirmesi için Bellman algoritması kullanılmıştır. ABC-OX1, Solomon'un 56 adet farklı veri kümesinden oluşan örnekler üzerinde test edilmiştir. Sonuçlar ABC-OX1'in temel ABC'ye göre daha yüksek bir performansa sahip olduğunu göstermiştir.

Anahtar kelimeler: Çeşitlilik, Sıralı çaprazlama, Yapay arı kolonisi algoritması, Yerel arama, Zaman pencereli araç rotalama problemi

ABSTRACT

MS THESIS

DEVELOPMENT OF A HYBRID METHOD FOR THE SOLUTION OF THE VEHICLE ROUTING PROBLEM WITH TIME WINDOWS

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE OF NECMETTİN ERBAKAN UNIVERSITY

THE DEGREE OF MASTER OF SCIENCE / COMPUTER ENGINEERING

Advisor: Assoc. Prof. Dr. İlhan İLHAN

2021, 92 Pages

Jury

Advisor Assoc. Prof. Dr. İlhan İLHAN
Assoc. Prof. Dr. Mehmet HACIBEYOĞLU
Asst. Prof. Dr. Mehmet Akif ŞAHMAN

In a world where competition is increasing rapidly with globalization, customer satisfaction is one of the determining factors in the sustainable success of companies. It is vitally important to create a strong supply infrastructure in order to meet the expectations of the customers. One of the most important processes in the supply chain is route planning. Today, planning on vehicle routing problems has become important especially for companies operating in the logistics and service sector. Routing and optimization solutions in business processes developed to meet increasing demands and reduce costs are very important for companies. Poorly managed route plans pose a problem for companies to serve their customers and cause disruptions in other logistics and service processes. The rapidly developing and changing global competitive environment forces businesses to plan for better process management. In order for this

planning to be done correctly, both customer satisfaction should be given importance and operating costs should be reduced. Vehicle routing costs are an important part of transportation and distribution costs within the logistics system. The purpose of the vehicle routing problem is to design the least costly set of routes for a fleet of vehicles. Vehicle routing problem with time window (WRPTW), which is one of the vehicle routing problems, constitutes the study area of many researchers. WRPTW is a vehicle routing problem with a time interval constraint for each customer. In this problem, the delivery vehicle has to serve each customer within a certain time interval.

In this study, artificial bee colony (ABC) algorithm and order crossover (OX1) operator were hybridized for the solution of WRPTW and a hybrid method called ABC-OX1 was proposed. In the ABC-OX1 method, the initial solutions were randomly generated and these solutions were improved with the tour developer 3-opt algorithm. Local search operators insertion, swap and divideandswap were used with equal probability in the worker bee phase. In the onlooker bee phase, food diversity was provided with the OX1 operator and the search for quality solutions continued. In the scout phase, new solutions were randomly generated and these solutions were improved with the tour developer 2-opt algorithm. In the ABC-OX1 method, the Bellman algorithm was used for the suitability evaluation of the solutions in the population. ABC-OX1 was tested on samples from Solomon's 56 different datasets. The results showed that ABC-OX1 had a higher performance than basic ABC.

Keywords: Diversity, Order crossover, Artificial bee colony algorithm, Local search, Vehicle routing problem with time window

ÖNSÖZ

Tüm tez çalışması boyunca değerli görüş, öneri ve zamanını benden esirgemeyen, her zaman en güzel yorumlarıyla tezime yardımcı olan ve beni yönlendiren, iş ahlâkına hayranlık ve saygı duyduğum danışman hocam sayın Doç. Dr. İlhan İLHAN'a teşekkür ederim. Tüm çalışmalarım boyunca seferberlik ilan eden her zaman desteğini üzerimde hissettiğim sabırla beni, maddi ve manevi destekleyen sayın Mehmet ESEROĞLU hocama teşekkürü bir borç bilirim. Akademik hayatım boyunca beni her türlü özveri ile destekleyen kıymetli annem, babam ve eşime sonsuz şükran borçluyum. İnsanoğlu amaçsız ve bir hedefi olamadan yaşayamaz. Mutlaka kısa veya uzun vadede bir hedef üzerine yaşantısını şekillendirir. Bu gaye üzerine bir hayali daha gerçekleştirmem için imkân ve sıhhat veren rabbime hamd ederim.

Zekai ÖZKÖK

KONYA-2021

İÇİNDEKİLER

ŞEKİLLER

Sayfa No

ÇİZELGELER

Sayfa No

SİMGELER VE KISALTMALAR

Simgeler

A	:	Yol ağları (Bağlantı yolları)
a	:	Rota kümesi
a_i	:	En erken varış zamanı
b_i	:	En geç varış zamanı
C	:	Müşteri kümesi
c_i	:	i. müşteri düğümü
c_k	:	K aracının kapasitesi
c_o	:	Merkez depo düğümü
D	:	Çeşitlilik yüzdesi (Diversity percentage)
d_{ij}	:	i ve j noktaları arasında öklid uzaklığı
e_j	:	j müşterisinin zaman aralığının başlangıç zamanı
c_{ij}	:	i müşterisi ve j müşterisi arasındaki maliyet
I	:	İterasyon sayısı
k	:	Araç sayısı
L	:	Rota kısıtlaması
l_j	:	j müşterisinin zaman aralığının bitiş zamanı
l_o	:	Deponun zaman aralığının bitiş zamanı
n	:	Müşteri sayısı
N	:	Düğüm kümesi
P	:	Popülasyon
si	:	Servis süresi
T	:	Maksimum deneme limiti
t_i	:	Müşteri varış zamanı
t_{ij}	:	i. müşterisi ve j. müşterisi arasındaki seyahat süresi
V	:	Özdeş araçları kümesi
y_i	:	Alt turları engellemek için rasgele değişken
x_{ijk}	:	İkili karar değişkeni
w_i	:	Müşteri bekleme süresi
q_i	:	i. düğümünün (müşteri) talep miktarı
Q	:	Özdeş araçların maksimum kapasitesi
x,y	:	Kordinat düzlemi

λ : Rotadaki takas

λ_j : Lagrange çarpanı

∞ : Sonsuzluk

% Hata : Hata sapma yüzdesi (Bilinen en iyi değerden)

Kısaltmalar

ARP	: Araç Rotalama Problemi
ABA	: Ateş Böceği Algoritması
ABC	: Artificial Bee Colony (Yapay arı kolonisi)
BKS	: Best Known Solution (Bilinen en iyi çözüm)
BT	: Benzetilmiş Tavlama
CUM	: Kümülatif
ÇDARP	: Çok Depolu Araç Rotalama Problemi
DGA	: Diferansiyel Gelişim Algoritması
DARP	: Dinamik Araç Rotalama Problemi
EZPARP	: Esnek Zaman Pencereli Araç Rotalama Problemi
GA	: Genetik Algoritma
GSP	: Gezgin Satıcı Problemi
KKA	: Karınca Koloni Algoritması
KKARP	: Kapasite Kısıtlı Araç Rotalama Problemi
OX1	: Order Crosover Operator (Sıralı çaprazlama operatörü)
ÖDST-ARP	: Önce Dağıt Sonra Topla Araç Rotalama Problemi
PARP	: Periodik Araç Rotalama Problemi
PROB	: Probability (İstatislik)
PSO	: Parçacık Sürü Optimizasyonu
SARP	: Stokastik Araç Rotalama Problemi
SZ	: Müşteri için sunulan hizmet süresi
SZPARP	: Sıkı Zaman Pencereli Araç Rotalama Problemi
TAA	: Tabu Arama Algoritması
YAK	: Yapay Arı Kolonisi
$ZPAS_j$	: $Müşteri_j$ için zaman penceresi alt sınırı
ZPARP	: Zaman Pencereli Araç Rotalama Problemi
$ZPUS_j$	: $Müşteri_j$ için zaman penceresi üst sınırı
Std. Sap.	: Standart Sapma

1. GİRİŞ

Günümüzdeki teknolojik gelişmeler hem ülkemizde hem de küresel pazarlarda büyük etkiler göstermektedir. Teknolojinin yarattığı imkanlar firmaların online pazarda büyümelerini sağlamakla birlikte küreselleşmelerine de fırsat sunmaktadır. Şirketlerin genişleyen pazar alanı üretim süreçlerinin artmasına da neden olmaktadır. Bu gelişmeler artan rekabet ortamı ile işletmeleri daha hızlı ve etkin kararlar almaya zorlamaktadır. Gelişerek büyüyen bu pazarda işletmelerin müşterilerin taleplerine hızlı, zamanında ve eksiksiz bir şekilde cevap verebilmesi gerekir. Bu nedenle ürün veya hizmetlerin transfer süreçleri hızlı bir şekilde yapılmalıdır. Bu zorunluluk lojistik sektörünün önemini artırmıştır. Lojistik süreçlerin hem etkin hem de ekonomik olması firmalara rekabet için önemli avantajlar sağlamaktadır. Bu nedenle ürün veya hizmetlerin transfer sürecinin de minimum maliyetle gerçekleştirilmesi önemlidir.

Lojistik sektörünün her geçen gün gelişmesi araç rotalama problemlerine (ARP) olan ilgiyi artırmıştır. Günümüzde ulaşım ve lojistik sektöründe faaliyet gösteren firmaların rota oluşturmaları oldukça karmaşık ve maliyetli bir hale gelmiştir. Özellikle ulaşım ve lojistik sektöründeki artan rekabet ve gelişen teknolojiyle birlikte işletmelerin de bu yeniliklere uyum sağlaması, sürekli kendilerini yenilemeleri gerekmektedir. Müşteri taleplerinin zamanında ve eksiksiz, minimum maliyetle yapılması, zor ve karmaşık bir optimizasyon problemi olan ARP kullanılarak çözülmektedir [1].

ARP, NP-zor problem sınıfında yer almaktadır. Kullanılan kısıtlar ve parametrelere göre problem daha da zor ve karmaşık bir hale gelebilmektedir. Bu nedenle, farklı optimizasyon algoritmaları kullanılarak kısa sürede, en uygun çözümler elde edilmeye çalışılır. Problemin boyutu büyüdükçe çözüm süresi de uzar. ARP'nin çözümünde kesin çözüm yöntemlerinin yanında farklı sezgisel yöntemler de kullanılır. Bu sezgisel yöntemler sayesinde büyük ve karmaşık gibi görünen problemler için kısa sürede optimuma yakın çözümler üretilir. ARP, bir veya birden çok depodan hareket eden özdeş araçların farklı konumlardaki müşterilere belirli kısıtlar altında hizmet vermek üzere optimum rotaların planlanması problemidir. ARP, literatürde ilk defa 1959 yılında Dantzig ve Ramser tarafından yayınlanan akademik bir makalede yerini almıştır. Yayınlanan bu çalışmada, benzin istasyonlarına akaryakıt dağıtımı yapan araçlar için gezgin satıcı probleminden (GSP) bahsedilmiştir [2]. Yine 1964 yılında Clarke ve Wright, tasarruf algoritmasını kullanmışlar ve çözümü daha da geliştirmişlerdir. ARP

alanında yapılan çalışmalara sürekli yeni kısıtlar ve algoritmalar eklenerek farklı türde modeller geliştirilmiştir [3].

ARP modelleri hayatta karşılaştığımız dağıtım problemlerinin türlerine göre çeşitlenmektedir. Birden fazla deponun olması, müşterilerin öncelikleri, farklı özelliklere sahip araçlar, müşteri taleplerinin eş zamanlı topla-dağıt gibi özelliklere sahip olması karşılaşılan bu problemlerden bazılarıdır [4]. ARP problem tiplerine göre kapasite kısıtlı, periyodik, topla-dağıt, çoklu depo, geri toplamalı ve zaman pencereli gibi gruplara ayrılmaktadır.

Bu çalışmada, müşterilerin sadece belirli bir zaman dilimi içerisinde servis alma zorunluluğu bulunan zaman pencereli araç rotalama problemi (ZPARP) ele alınacaktır. Kombinatorik bir problem olan ZPARP, her bir düğüme zaman aralığı kısıtının eklenmesi ile geliştirilmiş bir ARP türüdür. ZPARP için son yıllarda literatürde yapılan çalışmalar artmaktadır. Genel olarak ZPARP, merkezi depoda bulunan özdeş kapasiteye sahip araçların, belirlenen zaman ve kapasite kısıtları altında minimum araç ile optimum rota planlarının oluşturulması olarak ifade edilir. Her bir müşteriye belirli bir zaman aralığı içerisinde hizmet verme zorunluluğu vardır. Müşterilerin zaman penceresi kısıtı aşılmadan hizmet verilmesi sıkı zaman pencereli araç rotalama (SZPARP) türüne örnektir. Tüm bu amaçlar gerçekleştirilirken maliyetin minimum olması istenmektedir. Bu yönüyle aynı zamanda çok amaçlı rotalama türüne de bir örnektir [5].

Bu çalışmanın amacı, lojistik ve hizmet sektörlerindeki gecikmelerden kaynaklanan maliyet, zaman ve müşteri memnuniyetsizliklerini önlemek üzere rota planlarını optimize etmektir. Bu amaca yönelik olarak yapılan çalışma altı ana bölümden oluşmaktadır. Birinci bölümde tezin konusu, amacı ve ARP ile ilgili genel bilgiler yer almaktadır. İkinci bölümde ZPARP için literatürdeki çalışmalar ve yaygın kullanılan çözüm yöntemlerinden bahsedilmiştir. Üçüncü bölümde ARP'nin türleri ve çalışmamızın konusu olan ZPARP'ye yer verilmiştir. Dördüncü bölümde önerilen hibrit yöntem hakkında bilgiler sunulmaktadır. Beşinci bölümde deneysel çalışmaların sonuçlarına yer verilmiştir. Son bölümde ise elde edilen hibrit yöntemin sonuçları yorumlanmış, geleceğe yönelik yapılacak çalışmalar için önerileriler de bulunulmuştur.

1.1. Araç Rotalama Problemi

1.1.1. Araç Rotalama Problemi Nedir?

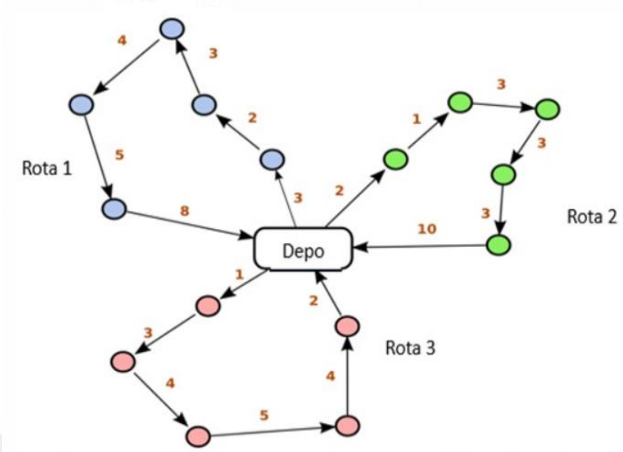
ARP üzerinde yapılan çalışmalar son yıllarda oldukça artış göstermiştir. Her geçen gün popülaritesi artan bir çalışma alanıdır. İlk defa 1956 yılında Dantzig ve Ramser [2] tarafından matematiksel olarak formüle edilmiştir. Yayınladıkları makalede benzin istasyonlarına akaryakıt dağıtımı yapan araçların sorununu çözmek için algoritmik bir yaklaşım geliştirmişlerdir.

Genel olarak, ürün ve hizmetlerin merkezi bir yerde belirlenen konumlara planlı bir şekilde dağıtımıyla ilgilenen problemlere ARP denir. Diğer bir ifadeyle farklı depolardan servis vermek üzere dağınık konumlardaki müşterilere hizmet sunulması için optimum rotaların belirlenmesi problemidir [6].

ARP'nin çözümünde kullanılan tüm araçlar merkezi bir veya birden çok depodan hareket etmeli ve yine depoya dönmelidir. Tüm bu hizmetlerin sunulması sırasında müşterilerin talepleri yerine getirilirken belirtilen kısıtların da karşılanması gerekir. Rotalar belirlenirken iki amaç söz konusudur: Araç sayısı minimize edilmeli ya da araçların toplam kat ettikleri mesafe minimize edilmelidir. Rotalama problemlerinde kullanılan parametreler problemin türüne göre değişiklik gösterse de genel olarak parametreler: müşteri sayısı, konumları, talepleri, depo, servis süresi, araç sayısı vb. şeklinde sıralanabilir.

ARP'nin her bir müşteri arasındaki bağlantı yoluna kenar denilmektedir. Depodan hareket eden bir aracın tekrar depoya dönene kadar güzergâh boyunca eklediği her bir düğüme ait bağlantı yollarının oluşturduğu ağa rota denilmektedir. ARP'nin birden çok rota kümesi vardır. Her bir rota kümesi aynı zamanda ilgili problemin rota sayısı olarak kabul edilir. Eğer tek bir rota kümesinden oluşur ise bu probleme gezgin satıcı problemi denir. Depodan hareketine başlayan bir aracın rota boyunca her bir düğüme uğrayarak tekrar depoya dönene kadar kat ettiği toplam yol uzunluğu o rotanın toplam mesafesi olarak ifade edilmektedir. Her bir rota için ölçülen toplam mesafe aynı zamanda rota süresi olarak da ilişkilendirilebilir. Bütün bu rota kümelerinin oluşturduğu alana graf denilmektedir. Graf probleme ait tüm müşterilerin konumlarını gösteren noktaların oluşturduğu kümenin tamamı olarak ifade edilmektedir. Graf alanında her bir (x,y) koordinat düzlemine isabet eden müşteri konumuna düğüm denilmektedir. ARP, graf alanda dağıtılmış vaziyette bulunan müşterilerin istenilen kısıtlara göre minimum araç

sayısı ile maksimum müşteriye hizmet verebilmesi için oluşturulmuş rota planlama problemidir. ARP'ye düğümler arasındaki kenar yollarının uzunluğu ilgili rotanın toplam uzunluğu olarak hesaplanırken tüm rota kümelerinin oluşturduğu uzunluk ise probleme ait toplam kat edilen mesafe olarak ifade edilmektedir.



Şekil 1.1. Araç rotalama problemi graf alanı [7]

ARP alanında çalışma yapan araştırmacıların sayısı gün geçtikçe artmaktadır. ARP alanında farklı türlerde formüle edilmiş birçok problem bulunmaktadır. Fakat genel itibari ile bir ARP için müşterilerin karakteristik özellikleri aşağıdaki gibi ifade edilebilir:

- Müşteri taleplerinin önceliği
- Müşteri konumlarının graf alanında dağılma özellikleri
- Toplanacak veya dağıtılacak farklı türde ve miktarda olan müşteri talepleri
- Zaman penceresi kısıtı ile müşterilerin belirli bir zaman aralığında hizmet alma zorunluluğu (müşterinin servise hazır olma zamanı, müşteriye son servis zamanı)

Müşterilerin talepleri ve özelliklerine göre bazen hizmet sürelerinde aksamalar veya hizmet verememe gibi durumlarla karşılaşılabilir. Böyle bir durumda talep miktarları azaltılabilir veya tamamen hizmet verilememe gibi durumlar yaşanabilir. Bu durumlarda hizmet önceliklendirme, ceza maliyetleri veya çeşitli yaptırımlar olabilir [8]. Araç rotalama problemlerinde kullanılan araçlara ait karakteristik özellikler;

- Araçların toplam kat ettikleri yol hesaplanır.
- Araçlar kapasitesinden fazla ürün veya yük taşıyamaz.
- Depodan hareket eden araç, tekrar depoya dönmek zorundadır.
- Araç özelliğine veya ürün tiplerine göre (katı, sıvı...) araca yükleme yapılır. [9]

Depodan aktif edilen bir aracın tüm rota boyunca her bir müşteriye hizmet verdikten sonra tekrar depoya dönmesi gerekir. Lojistik uygulamalarında bazen belirli bölgelerdeki müşteriler en yakın oldukları depolara göre bölünebilirler. Bu durumda her araç kendi bölgesindeki müşterilere hizmet verdikten sonra tekrar başladıkları depoya dönmek zorundadır. Buna benzer gerçek hayat problemleri birden çok depoya ayrıştırılarak bağımsız birer probleme dönüşebilirler ve ait oldukları problem türüne göre bir veya birden fazla amaca göre dizayn edilirler. Bu amaçlardan birkaç tanesini sıralayacak olursak;

- Seyahat sürelerini minimize etmek
- Tüm müşterilere hizmet verirken aynı zamanda araç sayısını minimize etmek
- Araçların toplam aldıkları mesafeyi, süreyi, sabitleri (sürücü giderleri) vb. maliyetleri minimize etmek
- Müşterilerin zaman kısıtından kaynaklanan cezalarını minimize etmek. Esnek zaman pencereli araç rotalama problemi (EZPARP) müşterilerin zaman aralığı dışında hizmet alması durumunda oluşan ceza maliyetleri

Yukarıda bahsedilen amaçlardan biri veya birkaçı bazen ARP'nin amaç fonksiyonu veya fonksiyonlarından biri olabilir. ARP'nin çözümünde oluşturulan çözüm kümesi ayrık şekildedir. Rassal olarak oluşturulur ve her çözüm sonucunda elde edilen araç sayısı aynı zamanda rota sayısı olarak da görülür. Müşterilerin hangi rotaya dahil olacakları rassal olarak oluşturulan çözüm kümesi ve problem kısıtlarına göre değişebilmektedir. GSP araç kapasitesinin olmadığı veya dikkate alınmadığı en temel araç rotalama problemidir. ARP zor ve karmaşık yapısı ile gerçek hayattaki birçok sorunu çözmektedir. Bu nedenle sürekli farklı türde yeni yaklaşımlar geliştirilmekte, çözüm üretilen problem sayısı da farklılaşarak artmaktadır.

1.1.2. Araç Rotalama Probleminin Amacı

Günümüzde büyük ve karmaşık araç rotalama problemlerinin çözümünde kullanılan optimizasyon yöntemleri işletmelere büyük kolaylık sağlamaktadır. Özellikle işletmelerin lojistik faaliyetlerinde maliyet, zaman, hız ve müşteri memnuniyeti açısından büyük avantajlar sunmaktadır. Yapay zekâ optimizasyon algoritmaları, ulaştırma özellikle de lojistik sektöründe, müşterilerin ihtiyaç ve beklentilerine istenilen kısıtlar altında en optimal rotaları oluşturarak çözümler üretebilmektedir. Literatürde kesin

çözüm yöntemleri ve sezgisel yöntemler kullanılmaktadır. Fakat problemin boyutu büyüdükçe kesin çözüm yöntemleri ile istenilen sonuca ulaşamayabilir. Bu nedenle, kısa sürede makul ve kabul edilebilir çözümler üreten sezgisel yöntemler kullanılır.

Bu amaca yönelik olarak literatürde genetik algoritması, diferansiyel gelişim algoritması, parçacık sürü optimizasyonu, benzetilmiş tavlama yöntemi, yapay arı kolonisi algoritması, karınca kolonisi algoritması, ateş böceği ve tabu arama algoritmaları vb. kullanılarak araç rotalama problemleri çözülmektedir.

ARP belirlenen müşteri grubuna hizmet verecek araçlar için en uygun rotaların belirlenmesi problemidir. Müşterilerin, talepleri ve depoya göre konumu bilinen, belirli sayıda özdeş kapasiteye sahip araçlar ile zaman ve kapasite kısıtları altında, toplam yolu ve araç sayısını minimize etmek amaçlanmaktadır. Depodan hareket eden aracın her noktayı bir kez ziyaret ederek tekrar depoya dönmesi ile rota tamamlanmaktadır. Aslında ARP'nin, problemin öncelik durumuna göre bazen araç sayısı bazen de toplam kat edilen mesafe minimize edilir [1]. İşletmeler için oluşturulan rotalama planları hem müşteri memnuniyeti hem de rekabet avantajı sağlar. Aynı zamanda lojistik faaliyetler etkinlik kazanarak gider maliyetlerini de azaltır.

1.1.3. Araç Rotalama Probleminin Önemi

Günümüzde araç rotalama problemleri üzerine planlama yapmak özellikle de lojistik ve hizmet sektöründe faaliyet gösteren şirketler için önemli hale gelmiştir. Artan talepleri karşılamak ve maliyetleri azaltmak için geliştirilen iş süreçlerinde rotalama ve optimizasyon çözümleri şirketler için oldukça önemlidir. Lojistik sektörünün sürekli büyümesi ve gelişmesiyle birlikte ihtiyaçlar da artırmıştır. Bu alanda faaliyet gösteren firmaların rota planı oluşturmaları fazlaca karmaşık ve maliyetli bir hal almıştır. İşletmeler rekabet edebilmek adına lojistik faaliyetlerindeki giderleri azaltmak zorundadırlar. Özellikle lojistik sektöründeki rekabetin artması, firmaların küreselleşme çabalarıyla birlikte bu yeniliklere uyum sağlaması ve sürekli kendilerini yenilemeleri gerekmektedir. Bu nedenle müşteri taleplerinin zamanında ve eksiksiz, minimum maliyetle karşılanması önem kazanmaktadır. Etkin bir rotalama planı hazırlanması önemli bir maliyet avantajı sağlamakla birlikte aynı zamanda müşterilerin memnuniyetini de artıracaktır. Bu nedenle lojistik planların oluşturulması zor ve karmaşık bir optimizasyon problemi olan ARP kullanılarak çözülmektedir.

ARP çözüm yöntemlerinden biri olan kesin çözüm yöntemi ile çözülemeyecek kadar zor, karmaşık ve uzun süreler gerektiren problemler olabilmektedir. Bu nedenle müşterilerin ihtiyaçlarına hızlı çözümler sağlayabilmek için sezgisel yöntemler tercih edilir [10]. Bu çalışmada, dağıtım sistemlerindeki ARP zaman penceresi kısıtları altında incelenecektir. Optimizasyon algoritmaları için başlangıç parametreleri ve kısıtlar ayarlanabilecektir.

1.1.4. Araç Rotalama Probleminin Prensipleri

Literatürde çok sayıda ARP ve bu problemlerin çözümüne yönelik her biri için geliştirilmiş farklı yapılarda algoritmalar bulunmaktadır.

ARP üzerine literatürde çok geniş kapsamlı çalışmalar yer almaktadır. Farklı türlerde geliştirilmiş algoritmalar ile gerçek hayattaki problemlere çözüm bulunmaktadır. Sezgisel yöntemler kısa sürede optimuma yakın sonuç üretse de henüz gerçek hayattaki bir problemin kesin sonucuna ulaşamamaktadır. Bu nedenle ARP üzerine ilgi artarak devam etmektedir. Her yıl kaynaklarda yayınlanmış birçok yeni çalışma görülmektedir. Özellikle yeni geliştirilen hibrit yöntemler oldukça ilgi görmektedir.

Literatürde yayınlanan kaynaklarda, ARP'nin çözümünde aşağıda bahsedilen temel prensiplerin dikkate alınması tavsiye edilmiştir.

- Rotalama yapılırken birbirine en yakın düğümler seçilmelidir. Bu şeklide toplamda en kısa mesafe elde edilir.
- Benzer bölgedeki dağıtımlar aynı tarihte yapılarak bir kez gidilmesi sağlanmalıdır.
- Rotalama işlemine başlarken depoya en uzak düğüm seçilmelidir. Bu şekilde toplam maliyet azaltılır.
- Rota oluşturulurken aynı yönde ve birbirine yakın düğümler seçilmelidir. Böylece uzak noktalar için maliyet faydası sağlanır.
- Rotalama için en yüksek kapasiteye sahip olan araç seçilmelidir. Bu sayede maliyet azalacaktır.
- Mümkünse topla-dağıt aynı araç ile yapılmalıdır. Bu şekilde hem toplam maliyet hem de rotalama zamanından tasarruf sağlanacaktır.
- Graf alanında rota güzergahı haricindeki düğümlere hizmet için küçük araçlar tercih edilmelidir.

- Eğer gerekli ve mümkünse müşterilere yapılan serviste topla-dağıt zamanları yeniden değerlendirilerek zaman tasarrufu elde edilmelidir [11].

1.1.5. Araç Rotalama Probleminin Temel Bileşenleri

ARP'nin temel bileşenlerini oluşturan maddeleri şu şeklide sıralayabiliriz:

a) Lojistiği yapılacak malzeme/ürün tipi
b) Toplama ve dağıtım yapılacak noktaların konumları
c) Filo araçlarının durumları
d) Müşterilerin talep yapıları

a) *Malzeme/Ürün Tipi:* Müşterilere sunulan hizmetler sırasında müşterilerin taleplerine göre farklı tipte malzeme veya ürünler taşınabilir. Bunlara tehlikeli madde (akaryakıt) taşımacılığı, yiyecek (gıda) madde taşımacılığı, kargo dağıtımı gibi örnekler verilebilir. Diğer taraftan hizmet sektöründe yürütülen faaliyetlerde ise toplu ulaşım, işçi servisleri, öğrenci servisleri gibi daha karmaşık bir yapı vardır. Özel amaçla gerçekleştirilen askeri sevkiyat (tank, mühimmat vs.) faaliyetlerinde ise ürün ve talepler değişik yapıdadır. Kısaca taşınacak ürün veya malzemeler katı veya sıvı olabileceği gibi taşınan malzemelerin fiziksel özellikleri simetrik koli şeklinde ya da asimetrik araç-gereç türünde de olabilir. Bahsedilen malzeme veya ürünler lojistik taşımacılığın önemli bir bileşenini oluştururlar.

b) *Toplama/Dağıtım Noktaları*: Araç rotalama probleminde genelde depolar toplama merkezi olarak kabul edilirken müşterilerin konumları ise dağıtım noktası olarak ifade edilmektedir. Üretilen malların fabrikadaki depodan toptancılara veya pera:kendecilere kadar dağıtım süreci örnek olarak verilebilir. Depolar rota planlarının başlayıp tekrar sonladığı noktalardır. Problemin türüne göre bazen birden çok depo da olabilir. Global firmalar, aynı anda birçok farklı merkezde benzer ürünleri üretip dağıtımını yapabilirler. Bu tür araç rotalama problemleri çok depolu veya birbirinden bağımsız ilişkili rotalama problemine dönüşebilir.

c) *Filo Araçları*: Araç rotalama problemlerinde yapı itibariyle genelde araç kapasiteleri özdeş kabul edilir. Farklı yapıdaki problem türüne göre bazen araç kapasiteleri farklılık gösterebilir. Gerçek hayatta da standart iş süreçleri haricinde

dinamik üretim ve değişken talepli lojistik hizmetlerinde rotalama işlemi yapılırken ekstra karar değişkeni belirlenerek hangi aracın ilgili rotaya hizmet vereceği belirlenir. Spesifik olarak araç filolarında araçlara ait özellikler de farklı olabilir. Hız, yakıt tüketimi ve taşınacak malzeme (katı/sıvı veya büyük hacimli vb.) duruma göre uygun araç yapısı olabilir. Fakat bu özellikler rotalama işlemi için ilave bir katkı sağlamamaktadır.

d) *Talep Yapıları*: Araç rotalama problemlerinde bazen problem türüne göre değişiklik gösterse de müşterilerin talepleri genelde sabittir. Fakat dinamik bir talebe de dönüşebilir [12]. Talep değişikliği araç rota üzerinde bilinen güzergâh üzerinden devam ederken önceden belirlenmiş talepler dışındaki bazı düğümler de o anda belirlenmektedir.

1.1.6. Araç Rotalama Probleminin Uygulama Alanları

ARP'nin ilgi alanı çok geniştir. Ulaşım hizmetlerinden, lojistik sektöre birçok alanda dinamik faaliyetler yürütülmektedir. Farklı sektörlerde ve iş kollarında yürütülen faaliyetlere yönelik yapılan bilimsel çalışmaların haylice fazla olduğu görülmektedir. İş veya hizmet kollarının detaylarına göre uygulama alanları aşağıdaki gibi sıralanabilir:

- İşçi servisleri için rotalama problemleri
- Engelli hizmetleri için rotalama problemi
- Topla-Dağıt tipinde olan eş zamanlı problemler
- Atık maddelerin toplanması (tıbbi, evsel, hafriyat vb.)
- Hava yolu trafiği için uçak rotalarının belirlenmesi problemi
- Güvenlik ve asayiş hizmetleri (polis, asker vb. devriye hizmetleri)
- Online market, gıda satışı yapılan ürünler (yerel servis hizmetleri)
- Online satış yapılan ürünlerin teslimatı (kargo taşımacılığı bölgesel)
- Dinamik olarak değişen öğrenci/okul servis güzergahlarının belirlenmesi
- Hizmet veya ürünlerin bir veya daha fazla depodan farklı talep noktalarına dağıtımı (ulaşım ve lojistik sektöründe yapılan faaliyetler)

Tıbbi atık veya evsel atıkların merkezi depodan arıtma merkezlerine taşınması sırasında oluşacak maliyetler araç sayısı, toplam yol ve harcanan zamanın optimize edilmesi için araç rotalama planları yapılmaktadır. Lojistik sektöründe üretim yapan

işletmelerin ürünlerinin depolardan son kullanıcılara kadar ulaştırılması sırasında, oluşan dağıtım problemlerini çözmek için rotalama uygulaması kullanılmaktadır. Ulaşım hizmetleri sunan özel veya kamu (toplu taşıma) araçlarının, müşterilerin taleplerinin karşılanması ve optimum maliyetlerle hizmet sunulması amacıyla rotalama yapılmaktadır. Okul servis güzergahlarının belirlenmesi sırasında araç sayısı ve toplam yolu minimize etmek adına çeşitli çalışmalar yapılmaktadır [13]. Hava yolu trafiği için yapılan çalışmalarda uçuş noktalarının konumları, kapasiteleri ve yoğunluklarına göre uçuş planları yapılmaktadır.

2. LİTERATÜR TARAMASI VE YAYGIN KULLANILAN YÖNTEMLER

ZPARP hakkında birçok bilimsel çalışma yayımlanmıştır. Genel olarak araştırmacılar, ZPARP'yi mevcut araç rotalama probleminden çok daha zor ve karmaşık bir problem olarak ifade etmektedirler. Literatürde ZPARP'nin kambinatorik NP-zor problem sınıfında yer aldığı belirtilmektedir. ZPARP için literatürde kullanılan çözüm yöntemleri üç gruba ayrılmaktadır. İlk olarak kesin (tam) çözüm yöntemleri, probleme ait tüm olası çözümleri değerlendirerek optimum sonucu verirler. Kesin çözüm yöntemleri, ZPARP'nin çözümünde, problemin doğal yapısından kaynaklanan ve problemin boyutuna bağlı olarak da çözüm süresi üstel olarak artan yapıdadır.

Tam çözüm yöntemleri büyük boyutlu problemin çözümünde yetersiz kalmaktadırlar. Bu nedenle büyük boyutlu problemlerde daha makul sürelerde yaklaşık çözümler üreten diğer bir çözüm yöntemi olan sezgisel yöntemler tercih edilmektedir. Klasik başlangıç sezgiselleri makul sürelerde arama uzayında olası yaklaşık sonuçları üretirler. Elde etikleri sonuçlar genelde yerel en iyi minimum değer olarak görülür. Kısıtlı bir alanda arama gerçekleştirirler. Fakat hiçbir zaman optimum sonucu garanti etmezler.

Üçüncü ve son yöntem ise meta-sezgisel çözüm teknikleri yapay zekâ optimizasyon algoritmaları ile yapılan çalışmaları içermektedir. Bu yöntemler çok hızlı bir şekilde arama uzayında çözüm üretebilirler. Sezgisel yöntemlere göre daha kısa sürede sonuca ulaşırlar. Meta-sezgisel yöntemler ile çözüm kalitesi artırılır. Fakat yine de en iyi çözümü garanti etmezler. Performansları itibari ile optimuma yakın sonuç üretirler. Lokal minimuma takılmadan global çözümlere ulaşırlar. Bu alanda geliştirilmiş birçok algoritma vardır. Gerçekte meta-sezgisel yöntemler kesin sonucu vermese de kısa sürede makul ve nihai sonuçlar üretirler. Bu nedenle kullanım alanları ve problem türleri her geçen gün genişlemektedir. Meta-sezgisel yöntemlerde elde edilen başarılar araştırmacıların bu yöne doğru ilgisinin artmasını sağlamıştır. ZPARP üzerine yapılan çalışmalarda farklı yaklaşımlar sergileyen yeni türde modeller literatüre kazandırılmaktadır. Zaman gibi çok kıymetli bir değer üzerine yapılan çalışmalarda sürekli yeni algoritmalar geliştirilmektedir.

Solomon, 1984 yılında yayınladığı bir makalede zaman penceresinden bahsederek bir benzin istasyonu için dağıtım yapan ARP için çözüm yaklaşımından bahsetmiştir. Bu tarihten itibaren zaman kısıtı altında araç rotalama problemleri için birçok çalışma yapılmıştır.

Literatür incelendiğinde, aşağıda isimleri belirtilen araştırmacıların çalışmalar yaptıkları görülmektedir. Bu çalışmalarda kullanılan tasarruf, en yakın komşu, dal ve sınır, lokal arama, tabu arama, tavlama benzetimi vb. gibi birçok farklı algoritma teknikleri ile toplam seyahat zamanı, toplam yol ve toplam araç sayısı minimize edilmiştir. Çizelge 2.1'de ZPARP üzerine günümüze kadar yapılan çalışmalardan bazılarına yer verilmiştir.

Çizelge 2.1. ZPARP kullanılan çözüm yöntemleri [5]

Yazar	Yılı	Kullanılan Yöntem	Amaç Fonksiyonu
Solomon	1987	En Kısa Yol Algoritması	Toplam mesafeyi minimize etme
Soumis ve ark.	1998	Tasarruf, En Yakın Komşu	Araç ve toplam süreyi minimize etme
Kallehauge ve ark.	2001	Kesme Düzlemi Algoritması	ZPARP ile ilişkili dif. opt. yöntemi
Bard ve ark.	2002	Dal ve Sınır Metodu	Toplam araç ve yolu minimize etme
Özaydın	2003	Hibrit Sezgisel Yaklaşım	Toplam mesafeyi minimize etme
Letchford ve ark.	2008	Küme Kaplama Yöntemi	Toplam mesafeyi minimize etme
Amini ve ark.	2010	Parçacık Sürü Optimizasyonu	Araç sayısını minimize etme
Ghoseiri ve ark.	2010	Genetik Algoritma	Toplam araç ve süreyi minimize etme
Ursani ve ark.	2011	Genetik Algoritma	Toplam süreyi ve yolu minimize etme
Yu ve ark.	2011	Karınca Koloni Algoritması	Toplam süreyi minimize etme
Balseiro ve ark.	2011	Karınca Koloni Algoritması	Toplam rota ve yolu minimize etme
Ding ve ark.	2012	Karınca Koloni Algoritması	Rota maliyetini minimize etme
Wang ve ark.	2012	Genetik Algoritma	Toplam araç ve yolu minimize etme
Vidal ve ark.	2013	Genetik Algoritma	Toplam seyahat maliyetini minimize
Chiang ve ark.	2013	Yenilikçi algoritma	Toplam maliyeti minimize etme
Yu ve ark.	2013	Tabu Arama Algoritması	Toplam mesafeyi minimize etme
Taş ve ark.	2014	Tabu Arama Algoritması	Toplam süreyi minimize etme
Belhaiza ve ark.	2014	Tabu Arama Algoritması	Toplam süreyi minimize etme
Jiang ve ark.	2014	Tabu Arama Algoritması	Toplam araç ve süreyi minimize etme
Fernandez ve ark.	2014	Olasılık Dağılımı	Toplam araç ve süreyi minimize etme
Tan ve ark.	2015	Bakteriyel Besin Arama	Toplam mesafeyi minimize etme
Armas ve ark.	2015	Değişken Komşuluk Arama	Toplam araç sayısını minimize etme
Brito ve ark.	2015	Değişken Komşuluk Arama	Toplam süreyi minimize etme
Zhou ve ark.	2017	Tabu Arama Mekanizmalı	Toplam süreyi minimize etme
Göçken ve ark.	2019	Genetik Algoritma	Bekleme sürelerini minimize etme.
Alberto ve ark.	2020	Genetik Algoritma	Toplam süresimi minimize etme
Zhang ve ark.	2021	Hibrit Çok Amaçlı Evrimsel	Toplam araç ve süreyi minimize etme
Triani Aulya fitri	2021	Yapay Arı Kolonisi	Yakıt maliyeti ve cezaları opt. etme
A. Ham ve ark.	2021	Karışık Doğrusal Programlama	Toplam araç ve yolu minimize etme
Meenal ve ark.	2021	Evrimsel Algoritma	Toplam araç ve yolu minimize etme

2.1. Kesin Çözüm Yöntemleri

Kesin çözüm yöntemleri tam sayılı programlama modeli olarak ifade edilir. Probleme ait çözümleri matematiksel olarak hesaplayabilen ve en uygun çözümleri bulan yöntemlerdir. ZPARP için sonlu çözüme ulaştıran algoritma modellerine kesin

(deterministik) çözüm yöntemleri denilmektedir. Genelde sonlu yapıda olan probleme ait tüm kombinasyonların sırayla denendiği ve o probleme ait kesin sonucun sonlu sürede elde edildiği yöntemlere denilmektedir. Kesin çözüm yöntemleri her zaman aynı sonucu veren tutarlı yöntemlerdir. Bu yöntem tüm kombinasyonel olasılıkları deneyerek en iyi çözümü bulmayı garanti ederler [14]. Kesin çözüm yöntemlerinin ZPARP'yi çözmek için kullandığı süre, problemin boyutu ve kısıtlarına bağlı olarak üstel bir şekilde artar. Ancak problemin boyutu büyüdükçe daha da karmaşık bir hal almasından dolayı çözüm süresi uzayacak ve bu bilgiyi işleyecek bilgisayarın kapasitesi yetersiz kalacaktır. Bu sebeple kesin çözüm yöntemleri daha çok başlangıçta, müşteri sayısının az olduğu küçük kapasiteli problemleri çözmede etkin olarak kullanılmıştır. Kesin çözüm yöntemleri ile her türlü ARP'nin çözülebilmesi mümkün olmadığı için genellikle müşteri sayısının az olduğu küçük ve orta ölçekli problemlerin çözümünde yararlı olmaktadır.

ZPARP zor ve karmaşık bir problem sınıfında olduğu için çözümünde de doğrusal programlama tercih edilir. Fakat problemin kısıtları ve karmaşıklığı fazla olması nedeniyle son yıllarda ZPARP'nin çözümü için kullanılan kesin çözüm teknikleri azalmaktadır. Çünkü kesin çözüm yöntemleri dar bir alanda hizmet vermektedir [3].

Literatürde ARP için kesin çözüm yöntemlerinden dal-kesme (Branch and cut), dal-sınır (Branch and bound), dinamik programlama ve küme-bölme algoritmalarının tam sayılı problemlerde sıklıkla kullanıldığı görülmektedir.

ZPARP için kullanılan yaygın kesin çözüm yöntemlerini aşağıdaki gibi sıralayabiliriz:

- Lagrange yaklaşımı (Lagrange relaxion)
- Sütun yaratma yaklaşımı (Column generation)
- Dinamik programlama yaklaşımı

2.1.1. Lagrange Tabanlı Kesin Çözüm Yöntemi

ZPARP'nin bir alt türü olan SZPARP için Lagrange modeli kullanılır. Çözüm kısıtları çok zorlaştırılmış sıkı problem türlerinde çözümü gevşeterek (rahatlatarak) nihai çözüme ulaşılır. Bu rahatlama için Lagrange tabanlı çarpan kullanılarak gevşemeden kaynaklı kısıtlarda oluşan ceza maliyetleri hesaplanarak problemdeki eşitsizlikler giderilir.

ZPARP'nin zaman aralığı kısıtlarının çözümü zor olduğundan Lagrange tabanlı kesin çözüm yöntemleri ile problem alt dallara ayrılarak kolaylıkla çözülür [5]. Lagrange tabanlı çarpan için aşağıdaki matematiksel model kullanılır.

$$\min \Sigma_{k\epsilon m} \Sigma_{i\epsilon N} \Sigma_{j\epsilon N} (c_i - \lambda_j) x_{ij}^k + \Sigma_{j\epsilon m} \lambda_j \quad (2.1)$$

Denklem 2.1'de amaç fonksiyonumuz ceza maliyeti λ_j Lagrange çarpanı ile gösterilmektedir. Bu şekilde j nolu müşteriye hizmet verme imkânı sağlanır [15]. Lagrange rahatlaması c_i den c_j ye giden tek bir bağlantı yolu olma durumunu, garanti edilmiş kısıtlar haricindeki tüm kısıtların rahatlatılmasında kullanılır [16].

Madsen [17], 1990 yılında ZPARP'ye alt sınır bulmak için Lagrange tabanlı gevşeme yöntemini kullanmıştır. Yaklaşık 31 müşteriye kadar olan problemler için optimal sonuçlar elde etmiştir.

Fisher ve ark. [18], 1997 yılında Lagrange gevşetmesi tabanlı iki yaklaşım önermişlerdir. Bunlar; K-ağaç yaklaşımı ve değişken parçalama yaklaşımıdır. K-ağaç yönteminde, özel tasarlanmış matematiksel model ile kısıtlara Lagrange gevşetmesi uygulanmaktadır. Diğer yöntem değişken parçalama yaklaşımında ise bir atama problemi zaman pencereli ve kapasite kısıtlı en kısa yol problemleri elde edilmektedir.

Kohl ve Madsen [19], 1997 yılında ZPARP'nin her müşteriye bir kez ziyaret edilmesi kısıtını gevşeten bir yaklaşım önermiştir. Ana problemin çözümünde optimal Lagrange çarpanlarının bulunması, alt problem çözümünde ise ZPARP için en kısa mesafe problemi elde edilir.

Kallehauge ve ark. [20], 2006 yılında ZPARP'nin atama kısıtlarının gevşetilmesiyle oluşturulan Lagrange dual problemini çözmek için doğrusal programlama çatısı altında bir kesen düzlem algoritması geliştirmişlerdir. Kesen düzlem algoritmasını bir dal-sınır algoritması içerisinde kullanarak geliştirdikleri Lagrange dal-kesim maliyet algoritması ile çözüm oluşturma zamanlarında önemli iyileştirmeler sağlamıştır.

2.1.2. Sütun Yaratma Tabanlı Kesin Çözüm Yöntemi

Sütun oluşturma tabanlı kesin çözüm yaklaşımlarında doğrusal programlama modeli, çözülmesi gereken çok fazla değişken olması durumunda, problemi daha basit alt problemler şekline getirerek, etkin çözüm metodunu kullanır. Bu metot, kesme düzlemi yöntemine benzemektedir. Özel kısıtlar oluşturularak, çözümün graf alanı belirlenir. Bu yaklaşımda probleme özgü çok fazla değişken olması ve bu değişkenlerin zor olması nedeniyle, alt küme problemlerinin değerlendirilmesine odaklanılır. Amaç fonksiyonu iyileşme potansiyeli yüksek olanları üretmeye odaklanır.

ZPARP'nin çözümünü kolaylaştırmak için problem basit alt çözümlere ayrıştırılır. Böylece hızlı ve etkili sonuçlar elde edilir. Bu şekilde karmaşık gibi görünen problemlerin çözümünde karşılaşılan, zaman penceresine bağlı oluşan ceza maliyetleri minimuma indirilir. Bu yöntem ZPARP için en optimum sonuçların elde edilmesinde etkilidir [21].

Bu modele yönelik ilk çalışma Desrosiers [22] tarafından 1984 yılında yapılmıştır. Bir okula ait 6 adet servis aracının rotalama işleminde sütun yaratma (Column generation) yöntemi kullanılarak servis taşıma problemi çözülmüştür.

Baker [23], 1983 yılında yaptığı çalışmada zaman pencere kısıtının darlığı nedeniyle dal-sınır ağacı algoritmasında köşe noktalarının etkilendiğini ifade etmiştir.

Deterministik alanında yapılan çalışmaların ilkini Desrosiers ve ark. [24] 1986 yılında yayınlamıştır. Yaptıkları çalışmada gezgin satıcı probleminin zaman kısıtını çözmüşlerdir. Bu çözümlerinde basit, dal-sınır algoritmasını kullanmışlardır. Küçük veri kümelerine ait basit problemlerde deney yaparak optimum sonuca ulaşmışlardır.

Kolen ve ark. [25], 1987 yılında ZPARP çözümü için dal-sınır algoritma modelini önermişlerdir. Yaklaşık 14 müşteriye kadar sınırlı problemler için dört araçla optimum çözümü elde etmişlerdir. 1990 yılından sonra ise sezgisel ve meta-sezgisel yöntemler kullanılmaya başlanmıştır.

Kohl ve ark. [17], 1999 yılında 2-yol kesimi (2-path cut) sütun yaratma için yeni bir dal-sınır şeması geliştirmişlerdir. Bu çalışmada bazı 25-50 müşterilik örnek problemler ve birkaç tane de 100 müşterilik örnek problem için optimal sonuç elde edilebilmiştir.

Min Hokey [26], çalışmasında ise rassal tam sayılı algoritma modeli kullanarak gecikmeye neden olan olumsuzlukları minimuma indirmeye çalışmıştır.

Desrochers ve ark. [27] veri kümelerini kapsamak için bir formülasyon yani sütun oluşturma kriteri kullanan optimizasyon algoritması geliştirmişler. Geliştirdikleri bu yeni algoritma sayesinde mevcut problemlere göre daha büyük veri kümelerinin çözümünde optimal sonucu elde etmişlerdir.

2.1.3. Dinamik Programlama Tabanlı Kesin Çözüm Yöntemi

Dinamik programlama yaklaşımı, çok değişkene sahip karmaşık yapıdaki problemleri basitleştirerek, alt problemlere yinelemeli bir şeklide uygulanmasıyla kullanılan bir yöntemdir. Problemde var olan tüm kararlar dağınık ve değerlendirme imkânı olmadığı için tekrarlı bir şeklide alt kararlara dağıtılır. Bu şekilde alt problemlere bölünerek ve daha sonra da adım adım yenilenerek çözüm elde ediliyorsa alt yapısal olarak ifade

edilebilir [28]. Yani büyük bir problem, ilişkisel olarak alt problemlere ayrıştırılarak dinamik programlama şeklinde çözülebiliyorsa, ana problem ile alt problem arasında direk bir ilişki vardır.

Dinamik programlama yaklaşımındaki amaç oluşan ceza maliyetlerini minimum yapmaktır. En yaygın olarak tercih edilen alan ise müşteriler için kesin hizmete başlama süresine karar vermektir [16].

Bir elektrik dağıtım firmasının elektrik hizmetlerinin sunulması sırasında ani kesintilere karşı hızlı karar alma ve uygulamaya konulması veya hizmet sektöründeki bir yetkili servis müşterilerine hizmet sunarken dinamik gelişen durumlara karşı karar alma problemlerini örnek olarak verebiliriz. Bahsedilen benzer problemlerin çözümünde kullanılan dinamik programlama yöntemleri için literatürdeki çalışmalardan bazıları aşağıda örnek olarak sunulmuştur.

Psaraftis 1988 yılında ilk olarak bir kargo şirketine ait araçların rotalama problemini çözen yöntem önermiş daha sonra 1995 yılında bu çalışmalarını derlediği dinamik araç rotalama problemini (DARP) çözen bir algoritma modeli geliştirmiştir [29].

Kilby ve ark. [30], 1998 yılında geliştirdikleri DARP için test problemleri üretmişlerdir. Dinamik hale getirilen test problemlerine, çalışma günü uzunluğu, taleplerin ortaya çıkış zamanları, her bir talebin servis süresi gibi zamana bağlı parametreler eklenmiştir.

Montemanni ve ark. [31] 2005 yılında karınca koloni algoritması (KKA) ile çözüm önerisi sunmuştur. Bu çalışmada, planlanmış eşit zaman aralıkları ve bu aralıklarda oluşan problemlere çözüm üretmek için algoritma modeli önermişlerdir. Planlama periyodunun kaç eşit alt zaman aralığına bölünmesi gerektiği ile ilgili test çalışmaları yapılmıştır.

Hanshar ve Ombuki-Berman [32], 2007'de Montemanni önerdiği yaklaşıma benzer bir yöntemi genetik algoritma ile kullanarak literatüre sunmuştur.

Khouadjia ve ark. [29] 2012 yılında parçacık sürü optimizasyonu ile yine Montemanni önerdiği yaklaşımı kullanarak yeni bir algoritma modeli geliştirmişlerdir.

2.2. Sezgisel Çözüm Yöntemleri

ARP'nin çözümü kesin çözüm yöntemleriyle çözülemeyecek kadar büyük ve karmaşık yapıda olabilir. Bu türde bir problemin çözümünün mümkün olmadığı durumlarda sezgisel yöntemler tercih edilir. Araç rotalama problemlerinde müşteri

sayısına bağlı olarak alternatif rotaların sayısı da kombinasyonel olarak artmaktadır. Rota sayısındaki artış hesaplama sürelerine etki etmektedir. Sezgisel yöntemler kısa sürede optimale yakın sonuçlar elde etmek için kullanılır. Bu nedenle ARP'nin çözümünde sezgisel yöntemlerin tercih edilmesi kaçınılmaz olmaktadır. Sezgisel yöntemler optimum sonucu garanti etmezler, yalnızca nihai çözümü sunarlar. Yani kesin sonuca alternatif olarak yakın bir sonuç üretirler. Bu özelliği nedeniyle karar problemi olarak adlandırılır. Genelde elde edilen sonuçlar değerlendirildiğinde çözüm kalitesinden kısmen vazgeçerek hesaplama süresi en aza indirgenir. Ancak arama uzayında dar bir alanda kısa sürede yaklaşık ve makul çözümler elde ederler [16]. Sezgisel yöntemlerle elde edilen sonuçlar lokal alt sınır çözümleri olarak değerlendirilir.

Problem çözme sırasında, sezgisel algoritma şu şekilde hareket eder: Rastgele bir ilk çözümle başlanır. Daha sonra iyi bir çözüm bulununcaya kadar arama devam eder. Tekrarları azaltan işlem süreci ile mevcut çözümü geliştirmek için uzayda sürekli aramaya devam eder. Her adımda bir önceki çözüm geliştirilerek en iyi sonuç elde edilmeye çalışılır. Problemin kısıtlarına bağlı olarak arama uzayının tamamında arama yapmak bazen mümkün olmayabilir. Sezgisel yöntemler dar ve sınırlı bir alanda arama yaparlar. Sürekli iyi çözümler üzerinden devam ettikleri için yerel minimuma takılmaları sıklıkla görülmektedir. Bu nedenle sezgisel yöntemler yerel en iyi çözümleri verirler.

Sezgisel yöntemler genelde tek bir amaca yönelik hızlı arama yaparak kısa sürede sonuca ulaşırlar [7]. Örneğin; toplam en kısa yolun bulunması, dizi elemanlarını sıralama veya küme elemanlarını gruplama gibi.

Sezgisel yöntemlerin çözüm adımlarını aşağıdaki gibi sıralayabiliriz:

- Eldeki verilerden herhangi birinin başlangıç çözümü için seçilmesi
- Başlangıç çözümüne mümkün olan testleri uygulayarak mevcut çözümün değiştirilmesi
- Elde edilen yeni çözüm için karar verilmesi
- Gereksiz çözümlerin, çözüm kümesinden çıkarılması
- Sonuca ulaşılmış ise sürecin tamamlanması ya da yeni çözümlerin ele alınarak tekrarlanması

Sezgisel yöntemlerde kullanılan çözüm yöntemleri sürekli ve ayrık optimizasyon şeklinde olabilir. Bu tamamen tercih edilen problemin türüne göre değişebilir. Ayrık optimizasyon tekniğinde tamsayı değerleri olur iken, sürekli optimizasyonda tam sayı

olmayan nümerik değerler de olabilir. ARP'nin sezgisel çözüm algoritmalarını genel olarak üç sınıfa ayırabiliriz:

- Rota kurucu sezgiseller
- Rota geliştirici sezgiseller
- İki aşamalı metotlar

Cordone ve Calvo'nun 2001 yılında ZPARP için önerdikleri model, üç basit prosedürün birleşiminden oluşmaktadır. İlk olarak tur geliştirici k-opt değişimleri ile çözüm iyileştirilir. Özel bir prosedür, araç sayısını azaltır ve ikinci bir amaç fonksiyonu aramayı yerel optimumdan uzaklaştırır. Parametre ayarlaması gerekmez ve rastgele seçim yapılmaz [33].

2.2.1. Rota Kurucu Sezgiseller

Rota oluşturma yöntemleri, depo ile her bir müşteri arasında oluşturulan bağlantı yollarıdır. Algoritma, rota oluşturma işleminin her aşamasında araçların kapasite sınırını dikkate alarak rotalara bir dal bağlantısı ekleyerek iki farklı rota kümesini birleştirir. Bu şekilde tasarruf hesabı yapılarak en uygun çözümler elde edilmiş olur. Rotaların birleştirilmesinde en büyük tasarruf sağlanan önceliklidir. Tur kurucu yöntemler içerisinde en çok tercih edilen algoritma Clarke ve Wrigh'ın, Dantzig ve Ramser'in çalışması üzerine geliştirdikleri tasarruf algoritmasıdır [34].

Bilimsel kaynaklarda tasarruf algoritmanın temel metodu değiştirilmeden geliştirilmiş veya çeşitlendirilmiş birçok algoritma modeli bulunmaktadır. Örneğin; Altınkemer ve Gavish'in [35] 1991 yılında geliştirdikleri paralel tasarruf algoritmasını örnek olarak gösterilebilir.

ZPARP için rota kurucu sezgiseli Clarke ve Wright [34] 1964 yılında literatüre kazandırdığı tasarruf algoritmasıdır. Solomon'un en yakın komşu algoritmasında her adımda en yakın komşu noktasını rotaya eklenerek iyi sonuç elde edilmeye çalışılır. Tasarruf algoritması, en iyi bilinen rota kurucu sezgisellerinden biridir. Her müşteriye bire bir teslimat şeklinde rotalar kurulur. Sonra rotalar en uyumlu olacak şekilde birleştirilerek toplam kazanç elde edilir.

Van Landeghem çalışmasında, ZPARP için geliştirdiği yeni sezgisel metodunu anlatmıştır. Bu metot, Clarke ve Wright'ın tasarruf algoritması üzerine eklenen yeni mekânsal ve zamansal kavramların etkileşimini tasarlayan yaklaşımları içermektedir.

Yapılan deneysel çalışmalarda bu sezgisel yaklaşımın, doğal rotalama sezgisellerine göre daha etkin sonuçlar ürettiği görülmüştür [36].

2.2.2. Rota Geliştirici Sezgiseller

Rota iyileştirme yöntemleri problemdeki komşuluk yapısına göre daha iyi olan komşu çözümleri bulmak için mevcut çözüm üzerinden sürekli arama yapan sezgisel algoritmalardır. Başlangıç çözümünü tekrarlı arama yöntemi ile iterasyonlar boyunca yerel arama yaparak geliştirmeye çalışır. Rota kurucudan farklı olarak makul çözümlerle aramaya başlar. Arama sırasında daha iyi bir çözüm bulunursa mevcut çözüm üzerinde değişiklik yapılarak iyileştirme sağlanır. ZPARP için geliştirilen yöntem Lin'in λ-Opt tekniğidir. Bu yöntemde λ rotadaki takas edilecek noktaların (köşelerin) sayısıdır. Graf alanındaki bir rotanın köşeleri çıkarılarak hesaplamaya başlanır. Sonra uygun bir noktada çıkarılan noktaların eklenmesi ile yeni çözüm geliştirilmeye çalışılır [5]. Rota üzerindeki noktaların yerlerinin değiştirilmesi ile daha iyi sonuç veren yeni rotalar bulunur. 2-opt ve 3-opt ise rotadaki çıkarılacak köşelerin sayısını ifade eder.

Rota geliştirme sezgiselleri, rotalar üzerindeki düğümlerin yerlerini değiştirerek veya farklı rotalardaki noktaları karşılıklı yer değiştirerek çözümü geliştirirler. Bu metot ZPARP'ye uygulanarak gerçekçi problemlerin çözümünde kullanılır.

Brasys ve Gendreau'nın [37], 2005 yılında geliştirdikleri algoritma ilk kabul değeri veya en iyi kabul değerinin bulunmasını hedeflemektedir. Kısaca mevcut çözüme karşı üretilen komşu bir çözüm varsa ve bu çözüm daha iyi ise kabul edilerek aramaya devam edilir. Bulunan komşu çözüm mevcut çözümden kötüyse kabul edilmez. Bu şekilde arama uzayındaki tüm seçenekler durdurma kriteri sağlanana kadar test edilir. Bu süreçler sırasında birinci aşamada rota kurma sonra zincir atma yöntemi ile rota sayıları azaltılmak istenir. Son aşamada ise sıralı değişim ile rota yolu kısaltılmaya çalışılmaktadır. Elde edilen en iyi değer sonuç olarak kabul edilir. Bu algoritma hızlı ve rekabetçidir.

Literatürde yer alan rota geliştirici sezgisellere bakıldığında, genellikle 2-opt algoritması tercih edilir. Bu tez çalışmasında da 2-opt ve 3-opt sezgiselleri ZPARP'nin çözümünde kullanılmıştır. 2-opt algoritması Croes [38] tarafından 1958 yılında geliştirilmiştir.

Lin [39] 1965 yılında 2-opt algoritmasına göre performansı daha yüksek ve kaliteli sonuçlar üreten fakat 2-opt algoritmasına göre daha yavaş olan 3-opt algoritmasını literatüre kazandırmıştır.

Verhoeven ve ark. [40], 1995 yılında 2-opt algoritmasına göre daha hızlı çözüme ulaştıran benzer bir 2-opt algoritmasını geliştirmişlerdir.

Potvin ve Rousseau [41], 1995 yılında ZPARP için 2-opt, 3-opt ve Or-opt sezgisellerini karşılaştırmıştır. Ayrıca rotalar arası takas modeli olan 2-opt* tekniğini önermişlerdir. Bu yöntem farklı rotalar arasındaki düğümleri karşılıklı olarak takas eder. Takas edilen düğümlerin sıralamaları değiştirilmeden, rotalar arasında yer değiştiren bir sezgisel yöntem önermişlerdir.

Taillard ve ark. [42], 1997 yılında 2-opt* ve Or-opt değiştirmelerini genelleştiren çapraz-değiştirme (Cross-exchange) sezgiselini önermiştir.

Nilsson [43], 2003 yılında 2-opt, 3-opt ve k-opt algoritmalarını karşılaştırmalı olarak incelemiştir.

2.2.3. İki Aşamalı Metotlar

İki aşamalı yöntemin ilk aşamasında müşteriler kapasiteleri aşmayacak şekilde araçlara atanır. İkinci aşamada her bir araç için GSP'de kullanılan sezgisel algoritmalar da olduğu gibi aynı şekilde rota oluşturulur. İki aşamalı metotlara örnek olarak önce grupla-sonra rotala tipindeki algoritmaları verebiliriz. [44]

Gillet ve Miller'in [45], geliştirdikleri süpürme algoritmasını, Fisher ve Jaikumar'ın [46], 1981 yılında önerdikleri algoritmayı, Bent ve ve Hentenryck. [47], geliştirdikleri algoritmaları iki aşamalı metotlara örnek olarak verebiliriz.

Fagerholt, 2001 yılında ZPARP'nin bir türü olan esnek zaman pencereli araç rotalama problemi üzerine çalışma yapmıştır. Topla-dağıt problem modeli üzerinde müşterilere esnek zaman penceresinde hizmet vermenin zorluğundan bahsetmiş, bu sebeple oluşan ceza maliyetlerini hesaplamanın zorluğuna dikkat çekmiştir. Müşterilere hizmet verirken oluşan uygunsuz maliyetler için en uygun kümeleme yapılarak toplam maliyetin en aza indirgenmesi üzerine hesaplama modeli geliştirmiştir [48].

2.3. Meta-Sezgisel Çözüm Yöntemleri

Bilgisayar teknolojilerinin gelişmesiyle birlikte 90'lı yıllardan sonra yapay zekâ yöntemleri üzerine çalışmalar artmıştır. Özellikle işlemci kapasitelerinin artırılması, büyük ve karmaşık problemlerin çözümü üzerine yeni yaklaşımların literatüre kazandırılmasını sağlamıştır. Kombinatoryal optimizasyon problemlerinin çözümünde derin arama gerçekleştiren meta-sezgisel algoritmalar kullanılmıştır.

Meta-sezgisel yöntemler kesin çözüm yöntemleri ile makul sürelerde çözülemeyen zor ve karmaşık ARP'nin çözümü için genellikle doğadaki olaylardan esinlenerek geliştirilmiş algoritma modelleridir. Genelde sezgisel yöntemler ile kısıtlı ve dar alanda arama yaparak yerel sonuçlar elde edilirken, meta-sezgisel yöntemlerle graf alanın tüm noktalarında arama yapılarak en iyi çözüme ulaşılır. Sezgisel yöntemlere göre daha global sonuçlar elde etmek için tercih edilir. ARP'nin arama uzayında oldukça fazla sayıda yerel minimum nokta bulunmaktadır. Meta-sezgisel teknikler yerel minimuma takılmadan uzaydaki en optimum sonucu bulurlar. Yerel minimuma takılmadan en iyi çözümleri elde etmek zor olduğu için ileri zekâ tekniği gerekmektedir. Bu nedenle meta-sezgisel yöntemler arama uzayında çok hızlı, kararlı ve zekice arama yapmaları sebebiyle kullanılırlar [49].

Meta-sezgisel yöntemler, sıklıkla rota kurucu ve iyileştirici sezgisel yöntemlerle birlikte kullanılırlar. Bu tez çalışmasında tasarlanan yapay arı kolonisi algoritmasına 2-opt ve 3-opt algoritmaları eklenerek birlikte kullanılmıştır. Sezgisel arama tekniği, komşuluk yapısına bağlı olarak kısır ve kör bir yapıda arama yapar. Geliştirilemeyen çözüm ile bazen algoritma yerel minimumda sonlandırılır. Fakat arama uzayındaki daha kötü çözümlerin maliyetlerine katlanarak, global çözümler elde etmek için meta-sezgisel yöntemler tercih edilir. Meta-sezgisel yöntemler uygun olmayan çözümleri dikkate alarak yeni çözümler keşfedebilme ve bu çözümleri geliştirebilme yeteneğine sahiptir. Böylece bütün arama uzayında en iyi çözümü elde etmeye çalışır. Bu yöntemler genelde global sonuçları bulan optimizasyon yöntemleridir. Meta-sezgisel yöntemler her ne kadar kesin çözümü garanti etmese dahi çok kısa sürelerde optimuma yakın kalitede çözüm üretirler. Özellikle büyük boyutlu ve bütünleşik yapıdaki gerçek yaşam problemlerinin çözümünde en pratik yol olarak kabul edilir. Çözüm uzayını birden çok noktada hızlıca arar ve optimale yakın çözüm sağlar. Bu nedenle en çok tercih edilen yöntemlerdir. Bu yüzden araştırmacılar sürekli olarak hibrit yöntemler geliştirmektedirler. Literatürde en çok kullanılan meta-sezgisel algoritmalara örnek olarak tabu arama, tavlama benzetimi, karınca kolonisi, yapay arı kolonisi ve genetik algoritmalar verilebilir [3].

Hibrit modellerin bilinirliği ve kullanımı son yıllarda oldukça artmıştır. Farklı problem türlerine ufak değişikliklerle uygulanabilirler. Meta-sezgisel yöntemlerin dezavantajı ise algoritmada kesin bir durdurma kriterinin bulunmamasıdır. Durdurma problemi, belirli bir iterasyon boyunca gelişme olmaz ise aramayı bitir şeklinde kısıtlar eklenerek aşılmaktadır [50].

2.3.1. Popülasyon Tabanlı Meta-Sezgisel Yöntemler

Popülasyon kavramı daha çok biyoloji biliminde kullanılan ve belirli bir zamanda o bölgede yer alan aynı türlerden oluşan bireylerin bir araya gelmesi ile oluşan topluluğa denilmektedir. Genetik biliminde ise birden fazla genin oluşturduğu kromozom topluluğudur. Kombinatoryal problemlerde kullanılan popülasyon kavramı ise graf alanındaki düğümlerin oluşturduğu çözüm kümesidir. ZPARP çözümünde popülasyon tabanlı meta-sezgisel yöntemler kullanılır. Popülasyon tabanlı çözüm yaklaşımlarında aynı anda birden fazla çözüm kümesi kullanılarak yeni çözümler elde edilir. Bu yöntemleri kullanan algoritmalara genetik algoritma, yapay arı kolonisi, karınca kolonisi ve parçacık sürü optimizasyonu örnek verilebilir.

2.3.2. Yerel Arama Tabanlı Meta-Sezgisel Yöntemler

Yerel arama yaklaşımları tek bir noktadan hareket eder ve çözüm sayısına göre arama uzayında en iyi çözümü bulmaya çalışır. Yerel arama algoritması uygun bir çözüm ile aramaya başlar ve bir sonraki adımda komşu çözüme gider. Eğer geldiği çözüm bir önceki çözümden daha iyi ise kabul edilir. Değilse iteratif olarak diğer çözümler araştırılır. Arama uzayında gidilen her noktada elde edilen en iyi çözüm güncellenerek belirli bir alanda arama yapılır. Yerel arama yöntemleri sistemli bir şekilde yinelemeli arama modeli olarak tasarlanmıştır. Graf alanındaki komşuluk yapısı problemin çözüm süresinde ve kaliteli sonuçlar bulmasında etkilidir. Komşuluk yapısına göre arama yaptıkları için çoğu zaman yerel optimum sonuçlar elde ederler [51].
Yerel arama tabanlı meta-sezgisel algoritmaların sahip olduğu arama özellikleri;

- İlk olarak uygun bir çözüm ile aramaya başlanır.
- Adım adım komşu çözümler üretilir.
- Olası uygun çözümler değerlendirilir.
- Elde edilen en iyi çözüm hafızaya alınır.

Yerel arama tabanlı meta-sezgisel yöntemlerden popüler olanları tabu arama ve tavlama benzetimi algoritmalarıdır.

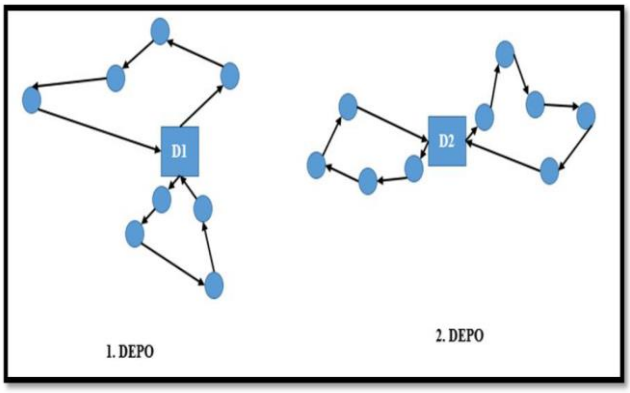
3. ARP TÜRLERİ VE ZPARP'NİN MATEMATİKSEL MODELİ

ARP tedarikçilerden son noktadaki müşterilere doğru ürün veya hizmetlerin dağıtılması sırasında ortaya çıkan problemdir. Gerçek hayattaki dinamik süreçler rotalama planlarını da etkilemektedir. Dağıtım türlerindeki çeşitlilikten kaynaklanan farklı çözüm yöntemlerine sahip ARP türleri de vardır. ARP çözümünde değişen kısıtlar (kapasite, zaman vb.) nedeniyle yeni ARP türleri ortaya çıkmıştır. Literatürde farklı problem türlerini çözmeye yönelik geliştirilmiş birçok ARP türü bulunmaktadır. ARP'nin çözümünde müşteri önceliklerine göre çözüm yöntemleri de çeşitlenmektedir. Şekil 3.1'de ARP türleri gösterilmektedir.

Şekil 3.1. ARP türleri

3.1. Çok Depolu Araç Rotalama Problemi

Araç rotalama problemi genelde tek bir depo ile müşterilere hizmet vermektedir. Çok depolu araç rotalama problemi (ÇDARP) adından da anlaşılacağa üzere birden çok depoya sahiptir. Şekil 3.2'de ÇDARP gösterilmiştir. Bu problem, farklı depolardan hareket eden araç filolarının müşterilere hizmet vermesi ve tekrar başladıkları depolara dönmesi ile sonuçlanır.



Şekil 3.2 Çok depolu araç rotalama problemi [5]

Bu tür problemlerin gerçek hayattaki karşılığı tedarikçiler, fabrika vb. üretim yapan tesislerdir. Pazar alanlarının genişliği nedeniyle birden çok depo ile müşterilerine hizmet sunarlar. Müşterilerin aynı tip ürün taleplerini farklı bölgelerdeki depolardan karşılamaya çalışırlar. ÇDARP'nin amacı, toplam seyahat mesafesini (maliyetini) minimize etmektedir. Klasik araç rotalama problemlerine göre çok daha kompleks bir yapıdadır ve çözümü oldukça zor olan bir problem türüdür. Kaynaklarda halen etkin bir çözüm yöntemi önerilmemiştir. Bu yapıdaki problemler genellikle doğrusal programlama algoritmaları ile oluşturulan "Transportasyon metodu" ile çözülmektedir [11].

3.2. Kapasite Kısıtlı Araç Rotalama Problemi

Kapasite kısıtlı araç rotalama problemi (KKARP), ARP'nin en yaygın bilinen ve kullanılan modelidir. Araçların kapasite kısıtları olmadığında, problem gezgin satıcı problemine dönüşür. Günümüzde KKARP yaygın olarak çalışılan konulardan biridir. NP-zor problemler sınıfına ait olup, yüksek zaman karmaşıklığına sahiptir. KKARP, bir dizi müşteriye hizmet vermek için minimum maliyetle maksimum rotaları belirlemeyi amaçlar. Maliyet, toplam seyahat mesafesine göre belirlenir. Müşteriler farklı coğrafi konumlara dağılmıştır ve farklı talepleri vardır [52]. Her birinin tek bir araç tarafından yalnızca bir kez ziyaret edilmesi gerekmektedir. Araçlar genellikle aynı kapasite kısıtlamalarına sahiptir. Müşteri talepleri araç kapasitesinden büyük olmaz. KKARP genelde araçların kat ettikleri mesafeyi minimize etmektedir. Araçların sabit çalışma maliyetlerinden dolayı, bazen amaç fonksiyonları, öncelikle araç sayısını minimize edecek şekilde tasarlanmaktadır.

3.3. Periyodik Araç Rotalama Problemi

Bilinen araç rotalama problemlerinde rota planlamaları günlük olarak kabul edilir. Periyodik araç rotalama probleminde (PARP) ise rota bir defada oluşturulur. Fakat belirlenen planlama periyodu değiştirilmeden farklı günlerde tekrarlı olarak uygulanır. Yani müşteriler baştan belirlenen planlama periyoduna bağlı olarak farklı günlerde birden fazla hizmet almaktadırlar. Müşteri ziyaret etme sıklığı, belirlenen frekans değerine göre gerçekleştirilir. Müşterilerin sayısı, talep miktarları, stok alanı vb. gibi durumlara bağlı olarak toplam ziyaret süresi minimize edilmeye çalışılır. PARP genellikle pera:kende sektöründe, mal ve hizmetlerin süreklilik gösterdiği market, kırtasiye, kasap, tamir ve bakım işleri ile atık toplama gibi alanlarda tercih edilmektedir.

3.4. Stokastik Araç Rotalama Problemi

Stokastik araç rotalama problemi (SARP) rotadaki elemanların veya parametrelerin dinamik olarak değiştiği problem türüdür. Klasik ARP'nin parametreleri sabit ve önceden bilinirken, Stokastik araç rotalama probleminde dinamik süreçler olduğu için önceden bilmek mümkün değildir. Gerçek hayat problemlerinde dinamik koşullardan dolayı müşteri sayısı, talepleri ve zamanı belirsizdir. Dinamik rotalama modeli, beklenmeyen aniden oluşan durumlar için karar alma süreçlerini kolaylaştırmaktadır. Yeni oluşan şartlara göre karar almak bazen tüm süreçleri etkilemektedir. Bu yüzden oluşan bu durumdan minimum etkilenmek için SARP algoritmaları kullanılmaktadır.

SARP süreçleri firmaların karar alma sürelerini ve maliyetlerini minimize etmek için kullanılır [53]. Amaç fonksiyonunu optimize etmek için problem iki aşamada çözülmektedir. İlk aşamada rassal gerçekleşme değeri bilinmeden problem çözülür. Bazı verilerin rassal olduğu durumlarda, rassal değerlerin tüm gerçekleşme durumlarında kısıtların karşılanmasını beklemek artık imkansızdır. Bu nedenle ikinci adımda ilk aşamada çözülen değere düzeltici işlemler yapılır. Karar verici ya belirli bir olasılıkla bazı kısıtların karşılanması koşulunu koyabilir ya da herhangi bir kısıt bozulduğunda probleme düzeltici işlemler ekleyebilir. Örneğin; araç kapasitesi tam dolmasa dahi bir sonraki müşteride araç kapasitesinin aşacağı bilinirse araç depoya dolmadan geri dönebilir. Yeni müşteri talebi olduğu zaman kapasite aşımı da olabilecektir. Bu durumdan kaçınmak için probleme yeni kısıtlar eklenerek olasılıklı çözüm yöntemleri denenmektedir. Bunlar;

- Araç dolu ise depoya dön yükü boşalt ve sonra planlanan şekilde talepleri toplamaya devam et.
- Araç rotalama yaparken kapasitesi dolar ise depoya dönsün yükü boşaltıp sonra planlanan kısım yeniden optimize edilsin.
- Araç dolu olmasa dahi depoya dönmek için önleyici bir geri dönüş planla. Bu kararı, araçtaki birikmiş yük miktarı ile aracın depoya olan uzaklığına göre belirle.

Belirsizlik, problemin çeşitli parametrelerinde olabilir. Rassal müşteriler, talebin rassal olması, servis zamanı ve seyahat zamanının rassal olması gibi sıralayabiliriz [11]. Problem stokastik olunca parametre özelliklerine göre de türleri değişmektedir. Bunlar; stokastik müşteriler, stokastik talepler ve stokastik zamanlar olarak ifade edilebilir.

3.5. Önce Dağıt Sonra Topla Araç Rotalama Problemi

Önce dağıt sonra topla araç rotalama probleminde (ÖDST-ARP) tek bir merkezi depodan hareket eden araçların önce ürünleri müşterilere dağıtmaları, sonra tedarikçilerden talepleri toplamaları beklenir. Bu modelde müşteriler ve tedarikçiler olmak üzere iki tür ilişki vardır. Rotalama işlemi başlamadan önce tüm talepler bellidir. Dağıtılacak ve toplanacak tüm taleplerin araç kapasitesini aşmaması gerekir. Rotalama planına göre önce talepler dağıtılır, sonra toplama işlemine geçilir. Eğer dağıtım esnasında toplama işlemi de yapılırsa aracın dağıtım yerinden hariç, diğer tarafına toplanan ürünler yerleştirilir. Fakat bu çok tercih edilen bir yöntem değildir. Dağıt ve topla şeklinde yapılan modelde amaç toplam maliyeti en aza indirmektir. Örnek olarak sebze-meyve veya et-süt entegre üretim tesislerini verebiliriz. Bu sektörlerde dağıtım tarafında olan müşterileri market, toplama tarafında olanları ise tedarikçiler veya toptancılar olarak düşünebiliriz.

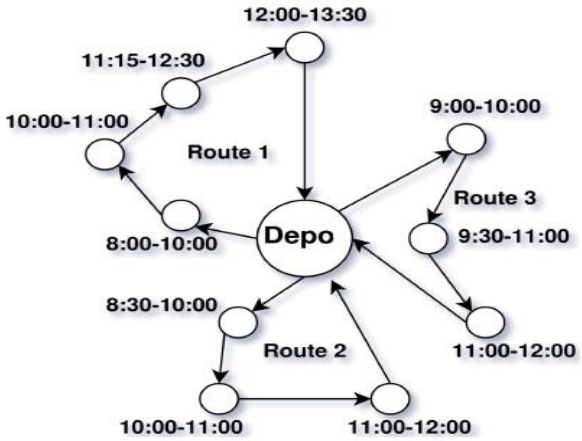
Literatürde ÖDST-ARP ile ilgili oldukça çalışma bulunmaktadır. Bu çalışmalardan Brandao'nun çalışmasında, problem tabu arama meta-sezgiseli ile çözülmüştür [54].

3.6. Zaman Pencereli Araç Rotalama Problemi

Bu tez çalışmasında, müşterilerin sadece belirli bir zaman dilimi içerisinde servis alma zorunluluğu bulunan ZPARP ele alınacaktır. Kombinatorik bir problem olan ZPARP Şekil 3.3'te görüldüğü gibi her bir düğüme zaman aralığı kısıtının eklenmesi ile geliştirilmiş bir araç rotalama türüdür [55]. Her i müşterisi için $[a_i, b_i]$ zaman penceresi tanımlanır. Bu zaman penceresinin dışında müşterilere hizmet verilemez. Graf alanındaki her bir müşteri için s_i servis süresi vardır. Araç, müşterinin servise hazır olma zamanından önce gitmiş ise beklemek zorundadır. Son servis süresi dışında gelirse de hizmet verememektedir. Her aracın rotası depoda başlar ve tekrar depoda sonlanır. Rota araç kapasitesi, müşteri zaman penceresi ve depo son tarih kısıtlarına uyularak oluşturulur. ZPARP'de merkezi bir depoda bulunan özdeş kapasiteye sahip araçların sayısı, bu araçların kat ettikleri toplam mesafe ve hizmet süreleri minimum yapılmaya çalışılır. Tüm bu amaçlar gerçekleştirilirken maliyetin minimum olması istenmektedir. Bu yönüyle aynı zamanda çok amaçlı rotalama türüne de bir örnektir.

Yapısı nedeniyle okul servisi rotalama, posta, gazete dağıtımı, akaryakıt dağıtımı, tam zamanlı üretim için satıcı dağıtımı, güvenlik devriyesi kontrolleri, kentsel atık toplama ve zincir mağaza dağıtım lojistiği gibi gerçek hayat problemlerine uyumludurlar [56]. Bu

nedenlerden dolayı diğer araç rotalama problemlerine göre daha çok araştırma yapılmıştır.



Şekil 3.3. Zaman pencereli araç rotalama problemi [55]

ZPARP bilgi teknolojisindeki gelişmeler sonucunda firmaların verimlilik ve zaman odaklı tedarik zincirine yönelmeleri ile tedarik zinciri için vazgeçilmez bir araç haline gelmiştir. Literatürde ZPARP'nin, KKARP'ye göre daha karmaşık olduğu ifade edilmektedir. Bu yüzden ZPARP için daha çok sezgisel ve meta-sezgisel yöntemler ile çözümler aranmaktadır.

ZPARP kendi içinde iki alt sınıfa ayrılmaktadır:

Sıkı Zaman Pencereli Araç Rotalama Problemi

Her bir müşteriye belirli bir zaman aralığı içerisinde hizmet sağlanması zorunluluğu vardır. Müşterilerin zaman penceresi kısıtları aşılmadan hizmet verilmesi SZPARP türüne örnektir [9]. Zaman pencere kısıtları sağlanmazsa müşteri hizmet alamamaktadır. Bu tez çalışmasında SZPARP ele alınmıştır.

Esnek Zaman Pencereli Araç Rotalama Problemi

EZPARP, belirli bir ceza maliyetine katlanılması durumunda müşterilere ilgili zaman pencerelerinin dışında hizmet verilebilmesine olanak sağlar. Sıkı zaman penceresi gibi müşterinin hizmet alamama gibi bir durumu söz konusu değildir. Ceza maliyeti hesaplanarak grafdaki tüm müşterilere hizmet sağlanır. Diğer yandan depo son tarihine uyulmalıdır.

ZPARP'nin çözümünde kullanılan genel kurallar;

- Rotalar simetrik ve asimetrik olabilir.
- Her müşteriye sadece bir araçla hizmet verebilir.
- Rotadaki her bir düğüm bir defa ziyaret edilmelidir.

- Tek bir depo vardır ve başlangıç noktası kabul edilir.
- Her rota depoda başlayıp depoda sonlanmalıdır.
- Müşterilere belirtilen zaman aralığında hizmet verilir.
- Müşteri talepleri araç kapasitesinden küçük olmalıdır.
- Bir rotadaki toplam talep araç kapasitesini aşmamalıdır [39].

3.6.1. ZPARP'nin Matematiksel İfadesi

ZPARP'nin matematiksel ifadesi oluştururken aşağıda belirtilen modeller kullanılır. Problem alt türüne göre değişiklik gösterebilir. Sıkı veya esnek problem tipine bağlı olarak formülasyona parametreler eklenebilir veya çıkartılabilir. Bu tamamen kullanılacak veri setinin özelliklerine ve probleme göre değişiklik gösterebilir. Genel itibari ile aşağıda gösterilen amaç fonksiyonları ve parametreler kullanılır. Bunlar; kümeler, karar değişkenleri ve parametreler olarak sıralanır.

ZPARP (N, A) yol ağları ile ifade edilir. Düğüm kümesi N, müşteri kümesi C olmak üzere 0 ile n+1 düğümden oluşur. Toplam müşteri sayısı 1, 2, …, n ile gösterilecektir. A rota kümesindeki düğümler arasında kurulan bağlantı yolları olarak tanımlanmaktadır. Her bir bağlantı yolu n+1 düğümünden çıkamayacağı gibi yine 0 düğümünde de sonlanamaz. Rotaların tamamı 0 düğümünde başlar ve n+1 düğümünde sonlanır. C_{ij}, i-j müşterisi arasındaki maliyeti belirtir. Her bir rota için t_{ij} seyahat süresine i. müşterideki hizmet süreside (s_i) dahildir. Her bir müşteriye tanımlanan hizmet süresi s_i olarak ifade edilir. Her bir $(i, j) \in A$ arasındaki bağlantı yolları, rota yolu olarak tanımlanmıştır. V, özdeş araçların kümesi olarak tanımlanmıştır. Her araç eşit tanımlı q kapasitesine ve q_i ($i \in C$) talebine sahiptir. Her bir müşterinin servise hazır olma zaman aralığında $[a_i, b_i]$, ($i \in C$) olmalıdır. Her müşteri için a_i servise en erken başlama zamanını, b_i ise servise en geç başlama zamanını belirtir. Ayrıca her bir araç depodan $[a_0, b_0]$ zaman aralığında hareket etmeli ve tekrar $[a_n+1, b_n+1]$ zaman aralığında depoya dönmelidir. Araç, müşterinin servise hazır olma zamanından önce gelirse bekler. Fakat müşterinin son servis tarihinden sonra gelirse ilgili müşteriye hizmet verilemez. Bu yüzden mutlaka son servis süresinden önce müşteriye aracın ulaşması beklenir. Müşteriye erken ulaşma t_i durumunda beklenecek süre w_i için ilave bir maliyet söz konusu değildir. Düğümler arasındaki mesafe d_{ij} olarak tanımlıdır. Düğümdeki j müşterisinin e_j-l_j zaman aralığıdır. Depo son kapanma zamanı l_o olarak belirtilmiştir. Başlangıç zamanı sıfır kabul edilir.
ZPARP'nin matematiksel formülasyonu aşağıda gösterilmiştir.

Kümeler:

C $\{1,2,\ldots...,n\}$ Müşteri kümesi

N $\{0,1,2,\ldots,n\}$ Düğüm kümesi (0. düğüm depo)

V $\{1,2,\ldots...,v\}$ Araç kümesi

Karar değişkenleri:

$$Xijk = \begin{cases} 1, & \text{k nolu araç i müterisinden j müşterisine gidiyorsa} \\ 0, & \text{aksi takdirde} \end{cases}$$

t_i : i. müşteri varış zamanı (i. düğüm zamanı)

w_i: i. bekleme süresi (i. müşteriye erken ulaşma durumunda bekleme süresi ya da i. düğümün servise hazır olma zamanı için geçen süre)

Parametreler:

d_{ij} i ve j müşterileri arasındaki uzaklık

a_i i. müşteride servise en erken başlama anı

b_i i. müşteride servise en geç başlama anı

c_{ij} i müşterisi ve j müşterisi arasındaki maliyet

t_{ij} i müşterisi ve j müşterisi arasındaki seyahat süresi

q_i i düğümünün (müşteri) talep miktarı

e_j j müşterisinin zaman aralığının başlangıç zamanı

s_i i müşterisindeki hizmet süresi ($t_i + w_i$)

l_j j müşterisindeki son hizmet alma zamanı

l_0 Depo zaman aralığı son tarihi (bir aracın maksimum seyahat süresi)

Q Özdeş araçların maksimum kapasitesi

k Toplam araç sayısı

n Toplam müşteri sayısı

c_i i. müşteri düğümü $i \in \{1,2,...,N\}$

c_0 Merkez depo düğümü

Amaç fonksiyonları:

Minimum

$$\sum_{i=0}^{n}\sum_{j=0}^{n}\sum_{k=1}^{v} d_{ij} * X_{ij} \tag{3.1}$$

$$\sum_{k=1}^{v}\sum_{j=1}^{n} X_{0jk} \tag{3.2}$$

Şu kısıtlara göre:

$$\sum_{k=1}^{v}\ \sum_{j=1}^{n}\ X_{0jk}\ \leq v \tag{3.3}$$

$$\sum_{j=1}^{n} X_{0jk}\ = \sum_{j=1}^{n}\ X_{0jk}\ \leq 1,\ \forall k\ \in V \tag{3.4}$$

$$\sum_{k=1}^{v}\ \sum_{i=0,i\neq j}^{n}\ X_{ijk}\ = 1,\ \forall j\ \in\ C \tag{3.5}$$

$$\sum_{k=1}^{v}\ \sum_{j=0,j\neq i}^{n}\ X_{ijk}\ = 1,\ \forall i\ \in\ C \tag{3.6}$$

$$\sum_{k=1}^{n}\ \left(\ m_i\ * \sum_{j=0,i\neq j}^{n} X_{ijk}\ \right) \leq Q,\ \forall k\ \in V \tag{3.7}$$

$$\sum_{i=0}^{n}\ \sum_{j=0,j\neq i}^{n} X_{ijk}\ \ \left(\ d_{ij}\ + s_i\ + w_i\ \right) \leq\ l_o\ ,\ \forall k\ \in V \tag{3.8}$$

$$t_0\ = w_0\ = s_0\ = 0 \tag{3.9}$$

$$t_i\ + w_i\ + s_i\ + d_{ij} =\ t_{j,}\quad \forall (i,j) \in N, i\ \neq j, if\ X_{ijk}\ = 1 \tag{3.10}$$

$$w_i = max\ \{e_i - t_i, 0\}, \forall i \in C \tag{3.11}$$

$$e_j \leq (t_j + w_j) \leq l_j, \forall j \in C \tag{3.12}$$

$$x_{ijk} \in \{0,1\}, \forall k \in V, \forall\ (i,j) \in N \tag{3.13}$$

$$t_i\ \geq\ 0,\ \forall i \in C \tag{3.14}$$

$$w_i \geq 0,\ \forall i \in C \tag{3.15}$$

Denklem (3.1) ZPARP için genel amaç fonksiyonu ifade eder. Toplam gidilen yolu minimum yapmaya çalışır. Denklem (3.2) ise araç sayısını minimum yapılmaya çalışılır. Denklem (3.3) depodan hareket edecek araç sayısının en fazla v olmasını sağlayarak tüm araçların kullanılma zorunluluğunu ortadan kaldırır. Denklem (3.4) her bir rotanın depodan başlayıp tekrar depoda sonlanacağını belirtir. Denklem (3.5) ve (3.6) ile her bir müşteriye tek bir araç ile hizmet verilmesi sağlanır. Denklem (3.7) bir rotadaki toplam müşteri talep miktarının araç kapasitesini aşmamasını sağlar. Denklem (3.8)-(3.12)

zaman penceresi kısıtlarını ifade etmektedir. Denklem (3.8) maksimum seyahat süresi kısıtıdır. Denklem (3.9), i. düğümünün 0 olması durumunda depo için kullanılacak karar değişkenlerini ve Denklem (3.10) ise 1 parametresi k nolu aracın i. müşterinden j. müşteriye gitmesini gösterir. Denklem (3.11) aracın müşteriye erken gelmesi durumunda bekleme süresinin hesaplanmasını ifade eder. Denklem (3.12) müşteri için hizmet zaman aralığını ifade eder. Bu aralıkta hizmetin verilmesi sağlanır. Denklem (3.13)-(3.15) karar değişkenlerinin alabileceği değer kümelerini belirtir.

3.6.2. SZPARP İçin Örnek Model

Bu tez çalışmasının konusunu oluşturan SZPARP, 4 müşteriden oluşan basit bir örnek problem üzerinde anlatılacaktır. Rotaların belirlenme süreci probleme girilen parametre değerleri, kısıtları ve amaç fonksiyonuna göre oluşturulur. Problem 4 müşteri ve 4 araçtan oluşmaktadır. Araç kapasiteleri özdeş ve 90 adet olup müşterilere hizmet verilirken geçen servis süreleri her bir müşteri için eşit 80 zaman birimidir. Probleme ait veriler Çizelge 3.1'de verilmiştir. Deponun müşteri numarası sıfır (0) olarak tanımlanmıştır. Gerçek problemlerin çözümünde başlangıç çözümü veya çözümleri rassal atama ile oluşturulur. Bu örnek problemin çözümünün kolay anlaşılmasını sağlamak için sırasıyla 1-2-3-4 başlangıç çözümü ele alınmıştır. Müşterilerin ve deponun graf alanındaki konumları Çizelge 3.2'de gösterilmiştir. Problemin çözümü sırasında uzaklık ve zaman birimi eş değer olarak kabul edilecektir [57].

Çizelge 3.1. SZPARP için örnek veri kümesi

Müşteri Numarası	X Koordinatı	Y Koordinatı	Talep Miktarı	İlk Tarih	Son Tarih	Servis Süresi
0	14	21	0	0	700	0
1	25	32	50	320	435	80
2	17	12	30	275	390	80
3	23	29	30	450	565	80
4	38	42	40	395	505	80

Çizelge 3.2'de gösterilen uzaklık matrisi, her müşterinin depoya ve birbirlerine olan uzaklıkları Denklem (3.16)'da öklid formülü ile hesaplanmıştır.

$$d_{i-j} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \qquad (3.16)$$

Denklem (3.16) x-y koordinat düzlemindeki iki noktanın birbirlerine olan uzaklıklarını hesaplamak için kullanılır. Öklid formülüne göre depo ile 1 nolu müşteri arasındaki uzaklık aşağıda gösterildiği gibi hesaplanır.

d (Depo-0 & Müşteri-1) = $\sqrt{(14-25)^2 + (21-32)^2} \cong 15.55$ olarak hesaplanır.

Öklid hesaplama modeline göre probleme ait simetrik mesafe haritası Çizelge 3.2'de ki gibi oluşturulur.

Çizelge 3.2. SZPARP örneği için uzaklık haritası

	DEPO	**Müşteri-1**	**Müşteri-2**	**Müşteri-3**	**Müşteri-4**
DEPO	0	15.55	9.48	12.04	31.89
Müşteri-1	15.55	0	21.54	3.60	16.40
Müşteri-2	9.48	21.54	0	18.02	36.61
Müşteri-3	12.04	3.60	18.02	0	19.84
Müşteri-4	31.89	16.40	36.61	19.84	0

Adım 1: Algoritma rotalama turuna başlarken ilk önce depodan bir aracı aktif eder ve rassal oluşturulan rota güzergahına göre ilk müşteriye gider. Problemimizde bize verilen çözüm sıralaması 1-2-3-4 şeklindedir. Bu sıralama düzenine göre mevcut kısıtlar altında müşterilerin rotalara ataması yapılacaktır. Aracın ilk andaki konumu depo, kapasitesi sıfır ve şimdiki zamanı sıfırdır. İlk müşterinin herhangi bir araca atanıp ya da atanmadığı kontrol edilir. Eğer ilk müşteri rotaya atanmamış ise sırasıyla önce aracın kapasitesi müşteri talebini karşılıyor mu kontrol edilir. Sonra müşterinin servis zaman aralığı (zaman penceresi) içerisinde aracın müşteriye ulaşma durumu kontrol edilir. Çünkü müşteri hazır olma zamanından önce araç müşteriye gelmiş ise beklemek zorundadır. Bu durumda depodan 1 nolu müşteriye giden aracın kapasitesi 90 adet ve 1 nolu müşterinin talebi 50 adet olduğu için araç kapasite şartı sağlanmış olur. Aracın mevcut kapasitesi artık 50 adettir. Depo ile 1 nolu müşteri arasındaki uzaklık Çizelge 3.2'de görüldüğü üzere 15.55 birimdir. Müşteri-1 servise hazır olma zamanı 320 birim olduğu için 1 nolu aracımız beklemek zorundadır. Son durumda aracın şimdiki zamanı 320 birim olarak güncellenecektir. 1 nolu müşterinin servis hizmet süresi 80 birimdir. Bunun için şimdiki zaman artık 320 birim ve 80 birimde hizmet süresi eklendiğinde 1. müşteriden ayrıldığı anda aracın şimdiki zamanı 400 birim olacaktır. Son olarak araç sıradaki müşteriye hareket etmeden önceden depoya dönebiliyor mu kontrol edilir. Bunun için depo ile müşteri arasındaki uzaklık yukarıda belirtildiği gibi simetrik olduğundan 15.55 birimdir. Son durumda aracın depoya dönebilmesi için gereken toplam zaman 415.55 birimdir ve bu değer depo son tarihi olan 700 birimden küçüktür. Dolayısıyla 1 numaralı aracın

rotasındaki ilk sıraya 1 nolu müşteri atanacaktır. Şimdi artık aracın mevcut konumu müşteri-1, kapasitesi 50 adet ve şimdiki zamanı 400 birimi olmuştur. Bir sonraki adımda araç, 1 numaralı müşteriden başlangıç çözümünde belirtilen sıraya göre 2 numaralı müşteriye doğru hareket edecektir.

Adım 2: Bu durumda konumu müşteri-1 olan araç başlangıçta verilen çözüm sırasına göre müşteri-2'ye hareket edecektir. Algoritmada belirlenen kısıtlar sırasıyla kapasite ve zaman kısıtları kontrol edilir. Mevcut aracın kapasitesi 50 adet 2 nolu müşterinin talebi 30 adet olduğu için araç kapasite şartı sağlanmış olur. Müşteri-1 ile müşteri-2 arasındaki uzaklık 21.54 birimdir. Aracın müşteri-1'den müşteri-2'ye ulaştığında şimdiki zamanı 400 + 21.54 = 421.54 birim olduğu için müşteri zaman penceresi aşılmış olur. Çünkü 2 nolu müşterinin son servis tarihi 390 birimdir. Bu durumda zaman penceresi kısıtı sağlanamadığı için müşteri-2'nin rotaya ataması yapılamaz. Bu durumda sıradaki müşteriyi rotaya eklemek için devem edilecektir.

Adım 3: Başlangıç çözüm sırasına göre devam edilir. Sıradaki 3 nolu müşteri için daha önce herhangi bir rotaya atanmış mı kontrol edilir. Eğer rotaya ataması yapılmamış ise 3 nolu müşteri için talep ve zaman kısıtları aynı şekilde kontrol edilir. Hesaplama modeli uygulandığında 3 numaralı müşteri 1 numaralı rotaya atanacaktır. 1 numaralı araç sıradaki 4 nolu müşteri için hareket edecektir. 1 nolu aracın şimdiki zamanı 530 birimdir. Çizelge 3.2'den kontrol edildiğinde müşteri-3 ile müşteri-4 arasındaki uzaklık 19.84 birim son durumda 1 nolu araç 4 numaralı müşteriye ulaştığında 530+19.84 =549.84 birimdir. Hesaplanan bu zaman birimi 4 nolu müşterinin zaman penceresi dışında kalmaktadır. Sırada rotaya atanacak herhangi yeni bir müşteri kalmadığı için 1 nolu araç rotasına, müşteri-1 ve müşteri-3 ekleyerek depoda rotasını sonlandırır. Depodan aktif edilen ilk aracın kapasite veya zaman kısıtları diğer rotaya atanamayan müşteriler için sağlanamadığı durumda depodan ikinci bir araç aktif edilerek diğer müşterilere hizmet verilir. Böylece tüm müşteriler bir rotaya atanana kadar problemin çözümüne devam edilir. Bu örnekte rota güzergahı yani çözüm sırası bize önceden 1-2-3-4 şeklinde sıralı olarak dikte edilmiştir. Son durumda rotaya atanamayan müşteri-2 ve müşteri-4 için 2 nolu araç turuna başlar. Yukarıda bahsedilen adım 1-2 ve 3'teki kısıtlar sırayla uygulanarak herhangi bir rotaya atanmamış müşteri kalmayana kadar adımlar sırasıyla uygulanır. Son durumda bu ayrık çözüm dizisine göre elde edilen sonuç aşağıda belirtilmiştir.

Problemin çözüm sırasına göre 2 rota oluşturulmuştur ve rota 1 için {1-3}numaralı müşteriler, rota 2'ye ise {2-4} numaralı müşteriler atanmıştır. Rota sayısı aynı zamanda problemde kullanılan araç sayısında ifade etmektedir. Amaç fonksiyonumuz bu problem için toplam mesafeyi minimize edecek şekilde tasarlanmıştır. Buna göre;

Toplam Mesafe => 109.17 birim olarak hesaplanmıştır. Sırasıyla müşterilerin atandığı rotalar: 1. müşteri 1 nolu rotaya, 2. müşteri 2 nolu rotaya, 3. müşteri 1 nolu rotaya ve 4. müşteri 2 nolu rotaya atanmıştır.

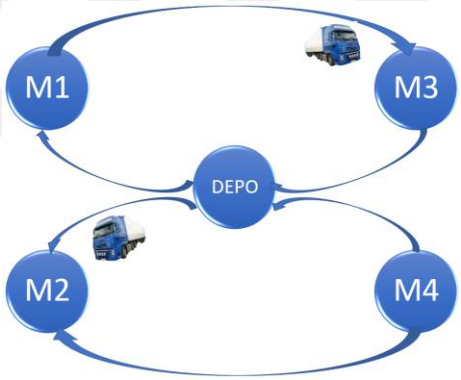
Çözüm sırası: 1-2-3-4'e göre => 4 müşteri için 2 Rota oluşturuldu.

(Rota 1 & Araç 1) : Depo → M1 → M3 → Depo

(Rota 2 & Araç 2) : Depo → M2 → M4 → Depo

Problemde kullanılan 2 aracın toplamda kat ettikleri mesafe aşağıdaki hesaplama modeli ile gösterilmiştir. Rota güzergahı Şekil 3.4'te gösterilmiştir.

Toplam mesafe = d(Depo, M1) + d(M1, M3) + d(M3, Depo) + d(Depo, M2) + d(M2, M4) + d(M4, Depo)= 15.55 + 3.60 + 12.04 + 9.48 + 36.61 + 31.89 = 109.17 birimdir.



Şekil 3.4. SZPARP örneği için rota oluşturma modeli

Bu sonucun en iyi sonuç olmadığını görmek için 4 müşterinin tüm olasılık durumları 4! yani 16 çözüm seçeneğinin tamamı incelenmelidir. Aşağıda rastsal oluşturulmuş 8 örnek çözümden oluşan durumlardan bazıları gösterilmiştir. Çözüm sıralaması değiştiğinde toplam maliyette azalma olduğu görülmektedir. En iyi çözüm maliyetinin 1-2-4-3, 1-4-2-3 ve 2-3-1-4 sıralamasında olduğu tespit edilmiştir. Kullanılan araç sayısı 2 olmuştur. Problemin çözülebilmesini mümkün kılmak için bazı hususlara dikkat edilmesi gerekir. Örneğin müşteri sayısına bağlı olarak uygun araç sayısı belirlenmelidir. Müşteri talebi araç kapasitesinden küçük olmalıdır. Müşteri zaman penceresi ve depo son tarihleri problemin yapısına uygun olarak seçilmelidir. Bu gibi benzer kısıtlar altında problemin çözümünü mümkün kılmak için uygun veri setleri olmalıdır. Aksi durumlarda problem çözülemez.

Popülasyon büyüklüğü: 8 Araç sayısı: 4 Araç kapasitesi: 90 parametre değerlerine göre Çizelge 3.3'teki sonuçlar elde edilmiştir.

Çizelge 3.3. Örnek model için deneysel çalışmalar

Çözüm Sırası	**Rota Sayısı**	**Müşterilerin Atandığı Rotalar**	**Toplam Mesafe Km birim**
3-2-1-4	3	2 2 1 3	134.44
4-2-1-3	2	2 2 1 1	110.36
1-2-3-4	2	1 2 1 2	109.20
1-2-4-3	**2**	**1 2 2 1**	**103.40**
1-4-2-3	**2**	**1 2 2 1**	**103.40**
3-4-1-2	4	3 4 1 2	137.95
4-1-2-3	3	2 3 1 1	113.86
3-1-2-4	3	2 3 1 2	106.90

Eğer çözüm sırası dikte edilmeseydi ve rasgele olsaydı çözüm uzayında 4 müşteri için oluşacak durum sayısıda 4! şeklinde olacaktı. Müşteri sayısının 100 olduğu bir problemde haliyle 100! kadar bir çözüm oluşacaktır. Bu durum neredeyse problemin çözümünü imkânsız kılacak ya da problemin çözümü için çok uzun süreler gerekecektir. Bu sebeple karmaşık ve büyük boyutlu problemlerde daha kısa sürede optimuma yakın sonuçlar üreten meta-sezgisel yöntemler kullanılır.

4. ÖNERİLEN ABC-OX1 YÖNTEMİ

Bu çalışmada, yapay arı kolonisi algoritması ile sıralı çaprazlama (Order Crossover-OX1) operatörü melezleştirilmiş ve ABC-OX1 olarak isimlendirilen yeni hibrit bir yöntem önerilmiştir. Bu algoritmanın başlangıcında giriş parametreleri tanımlanır, veri seti okutularak parametre modeli oluşturulur. Öklid formülü ile müşterilerin uzaklık matrisi hesaplanır. Geliştirilen ABC-OX1 yönteminde başlangıç çözümleri permütasyon kodlamalı (tam sayı değerli) olarak rastgele ayrık bir dizi şeklinde oluşturulur. Girişte tanımlanmış parametre değerlerine göre besin matrisinin büyüklüğü belirlenir. Örneğin; problemin giriş parametresine 100 değeri tanımlandığında 100 satırdan ve müşteri sayısı kadar sütundan oluşan rastsal sıralanmış başlangıç popülasyonu (besin matrisi) üretilir. Bu çözümler daha sonra tur geliştirici 3-opt algoritması ile iyileştirilir. İşçi arı fazında insertion, swap ve divideandswap yerel arama operatörleri eşit olasılıkla kullanılmıştır. Gözcü arı fazında OX1 operatörü ile besin çeşitliliği sağlanmış ve kaliteli çözümler üzerinde aramaya devam edilmiştir. Kâşif arı fazında ise yeni çözümler rastgele olarak oluşturulmuştur. Daha sonra bu çözümler tur geliştirici 2-opt algoritması ile iyileştirilmiştir. ABC-OX1 yönteminde popülasyondaki çözümlerin uygunluk değerlendirmesi için Bellman [58] algoritması kullanılmıştır. ABC-OX1 algoritmasındaki öncelikli amaç toplam mesafeyi minimum yapmaktır. ABC-OX1'in kaba kodu Çizelge 4.1'de verilmiştir.

Çizelge 4.1'de görülebildiği gibi ilk prosedür I, P, T, D ve ZPARP parametrelerini başlatmak için kullanılır. ZPARP'nin çözümü için Solomon'un test örneklerinde oluşan ".xlxs" uzantılı dosyalar kullanılmıştır. Bu dosya müşterilerin koordinat düzlemindeki konumları (x-y), müşteri taleplerinin miktarı, müşterilerin servise hazır olma zamanları (ready time), müşterilerin son servis tarihleri (due date) ve hizmet almaları için gereken servis sürelerini içerir. Bu bilgiler kaba koddaki ilk 3 satır kullanılarak oluşturulur. Algoritmadaki diğer prosedür ve satırlar aşağıdaki alt bölümlerde tanımlanmıştır.

Çizelge 4.1. ABC-OX1 algoritmasının kaba kodu [58]

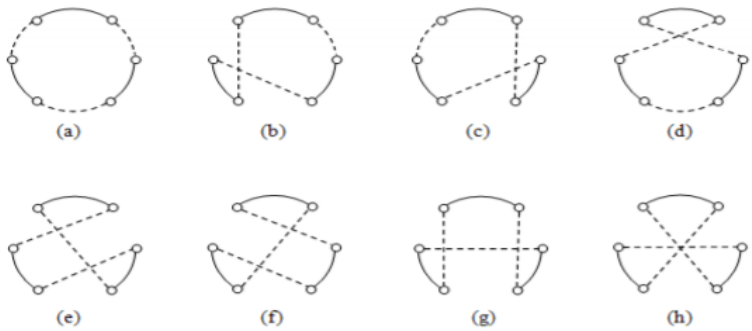
Input: I: İterasyon sayısı, P: Popülasyon büyüklüğü, T: Maksimum deneme limiti,
D: Çeşitlilik yüzdesi, ZPARP: Solomon örnekleri.
Output: p_{best}: En iyi çözüm, $f(p_{best})$: En iyi çözümün uygunluk değeri,
$v(p_{best})$: En iyi çözümün araç sayısı

```
1  begin
       // Parametrerelerin başlatılması I, P, T, D ve ZPARP
2      ParametreleriBaşlat()
       // Test örneklerinin ZPARP için yüklenmesi.
3      M = ZPARPÖrnekleriniYükle(ZPARP)
       // Başlangıç pülasyonunun rastgele üretilmesi
4      p = İlkPopülasyonuOluştur(M, P)
       // 3-opt operatörünün başlangıç popülasyonundaki çözümlere uygulanması
5      p = 3-OptOperatorUygula(M,p)
       // Çözümler için deneme sayaçlarının sıfırlanması
6      t = DenemeSayaçlarınıSıfırla(p)
       // İterasyonların başlangıcı
7      for k = 1 to I
       // İşçi Arı Fazı
8          foreach pi in p
               // Yerel arama operatörünün uygulanması
9              pnew = YerelAramaOperatörleriniUygula(pi)
               // Popülasyondaki pnew klon sayısının bulunması
10             clone = KlonSayısınıBul(p, pnew)
11             if f(pnew) < f(pi) and clone <= P*(D/100) then
12                 pi = pnew
13                 ti = 0
14             else
15                 ti = ti + 1
16             end if
17         end foreach
           // Uygunluk değerleri kullanılarak çözümler için kümülatif olasılıkların hesaplanması
18         cum = KümülatifOlasılıklarıHesapla(f)
           // Gözcü Arı Fazı
19         j = 1
20         while j <= P
               // Kümülatif olasılıklara dayalı bir çözümün rastgele seçimi
21             i = ÇözümSeç(cum)
               // Bir çözümün rastgele seçimi
22             r = Randi(P)
               // Sıralı çaprazlama operatörünün uygulanması
23             (o1, o2) = OX1OperatörünüUygula(pi, pr)
24             if f(o2) < f(o1) then
25                 o1 = o2
26             end if
               // Popülasyondaki pnew klon sayısının bulunması
27             clone = KlonSayısınıBul(p, o1)
28             if f(o1) < f(pi) and clone <= P*(D/100) then
29                 pi = o1
30                 ti = 0
31             else
32                 ti = ti + 1
33             end if
34             j = j + 1
35         end while
           // İzci (Kaşif) Arı Fazı
36         for i = 1 to P
37             if ti > T then
                   // Çözümün rastgele üretilmesi
38                 pi = ÇözümÜret(M)
                   // 2-opt operatörünün çözüme uygulanması
39                 pi = 2-Opt OperatörünüUygula(M, pi)
40             end if
41         end for
       // İterasyonların sonu
42     end for
43     return(pbest, f(pbest), v(pbest))
44 end
```

4.1. Başlangıç Popülasyonunun Oluşturulması ve Uygunluk Değerlendirmesi

Geliştirilen ABC-OX1 yönteminde başlangıç çözümleri permütasyon kodlamalı (tam sayı değerli) olarak rastgele ayrık bir dizi şeklinde oluşturulur. Bir çözümün (yiyecek kaynağı) uzunluğu müşterilerin sayısına eşittir. Popülasyondaki çözümlerin sayısı işçi arıların sayısına eşittir. Popülasyondaki her bir çözüm rastgele olarak üretilir ve müşteriler kümesinin bir permütasyonu olarak temsil edilir. Daha sonra her bir çözüm, tur geliştirici 3-opt sezgisel algoritması kullanılarak geliştirilir.

3-opt algoritması ilk olarak Lin [39] tarafından önerilmiştir. Bu algoritmada bir çözümün üç kenarı kaldırılır. Kaldırılan bu kenarların yerine üç yeni kenar eklenir ve uygunluk değeri açısından bu kenarlar yeniden değerlendirilir. Bu işlem üç yeni kenarın sekiz farklı kümesi için tekrarlanır. Bu oluşan sekiz yeni durum için olası çözümler şu şekilde gerçekleşir: Öncelikle bir turun üç kenarı üç seçenekli bir hareketle kaldırılır ve en fazla üç kenar değiştirilerek yeni bir tur elde edilir. Bu bağlamda, üç kenarın kaldırılması, daha sonra Şekil 4.1'de gösterildiği gibi sekiz farklı şekilde tam bir turda yeniden birleştirilebilen üç yeni yol ile sonuçlanacaktır. Yine de sekiz yoldan sadece dördü (e, f, g, h) aslında üç yeni kenar sunarken, diğer dört yol (a, b, c, d) 2-opt hareketle elde edilebilir [59]. Bu kümeler içerisinde en iyi olan tespit edilir ve yeni çözüm olarak kabul edilir. 3-opt algoritması yardımı ile kaliteli çözümlerden oluşan başlangıç popülasyonu üretilmiş olur. 2-opt algoritmasına göre daha yavaş çalışır. Fakat daha kaliteli sonuçlar elde etmede 3-opt algoritması daha başarılıdır. Bu tez çalışmasında ABC-OX1 algoritmamıza 3-opt sezgiseli eklenerek kaliteli başlangıç çözümlerinin oluşturulması sağlanmıştır.

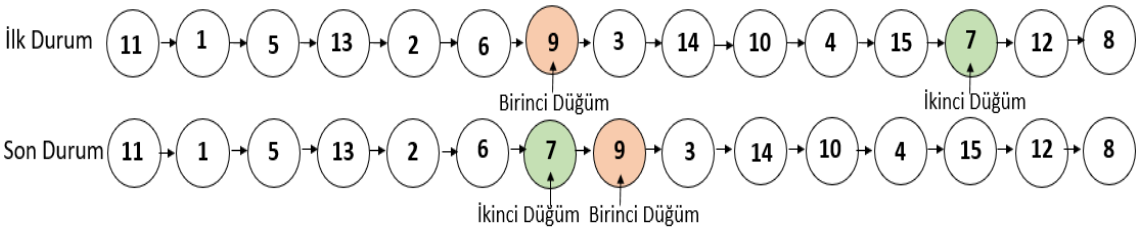


Şekil 4.1. 3-opt kombinasyonel durumları [59]

ABC-OX1'de bir çözüm tamsayı dizisi şeklinde temsil edilir. Bu nedenle dizi rota bilgisini içermez. Çözümü rotalara ayırmak ve uygunluk değerini hesaplamak ZPARP için geliştirilen Bellman algoritması kullanılır.

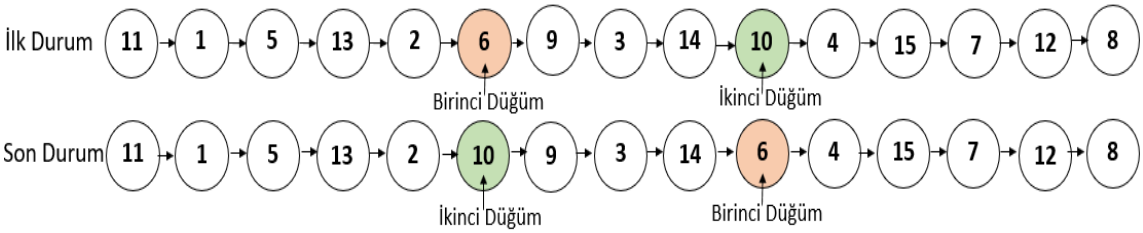
4.2. İşçi Arı Fazı

Popülasyondaki çözümler yerel arama operatörleri kullanılarak geliştirilir. Bu çalışmada yerel arama operatörleri olarak insertion, swap ve divideandswap operatörleri kullanılmıştır. Herhangi bir çözüm için bu operatörlerden hangisinin kullanılacağına ise eşit olasılıkla karar verilmiştir. Insertion operatörü çözüm içerisinde rastgele iki düğüm seçer ve ikinci düğümü birinci düğümün arkasına ekler. Bu şekilde ikinci düğümden sonraki diğer düğümler otomatik bir şekilde birer adım sağa kaymış olur. Etkili yerel arama operatörlerinden biridir. Şekil 4.2'de insertion operatörünün örnek bir uygulamasını gösterir.



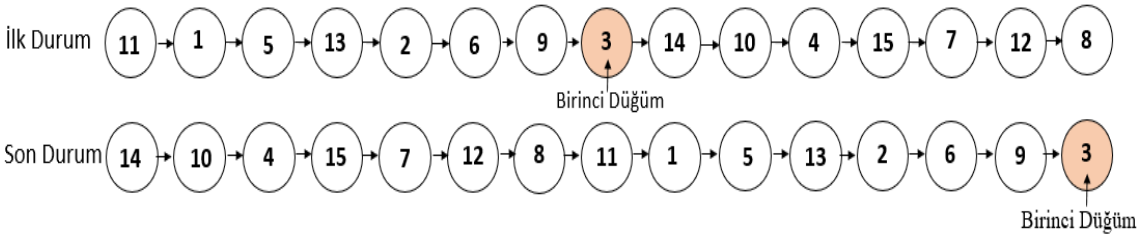
Şekil 4.2. Insertion operatörü

Swap operatörü çözüm içerisinde rastgele iki düğüm seçer ve iki düğümü karşılıklı olarak takas eder. Şekil 4.3'te swap operatörünün örnek bir gösterimi verilmiştir. Bu operatör her ne kadar dizideki iki düğümün yerini değiştirmiş olsa da gerçekte çözümün geliştirilmesinde ve toplam maliyetin azaltılmasında etkisi çok büyüktür.



Şekil 4.3. Swap operatörü

Divideandswap operatörü rastgele bir düğüm seçerek çözümü ikiye böler. Daha sonra bu iki bölümü takas eder ve yeni çözümü üretir. Oluşan yeni çözüm için maliyet hesabı yapılarak önceki çözümle karşılaştırılır. Şekil 4.4'te divideandswap operatörünün örnek bir uygulaması gösterilmiştir.

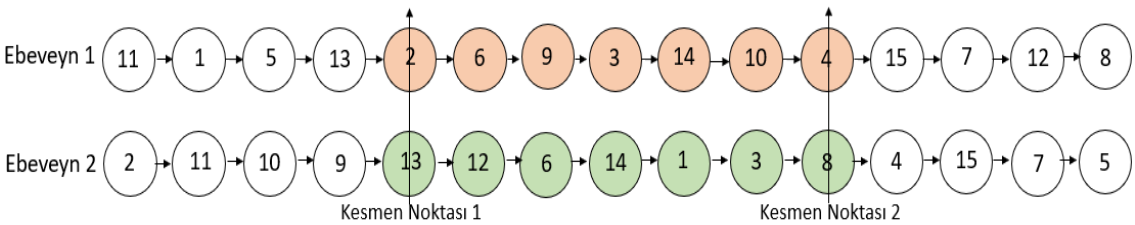


Şekil 4.4. Divideandswap operatörü

Popülasyonda rastgele bir çözüm satırı seçilir. Seçilen çözüm satırındaki sıralamayı değiştirmek için yerel arama operatörleri eşit olasılık hesabı ile kullanılır. Yeni oluşan çözüm sıralaması için Bellman algoritması ile maliyet hesabı yapılır. Eğer yeni çözüm eski çözümden daha iyi ise kabul edilir. Değilse geliştirilememe sayacı (t_i) bir artırılır. Clone değişkeni ile popülasyondaki çeşitlilik (aynı bireylerin sayısı) sürekli olarak kontrol edilir. Bulunan her yeni çözüm besin matrisinde kontrol edilir. Popülasyondaki çeşitliliğin azalmasına bağlı olarak clone değeri de artacaktır. Popülasyondaki çeşitliliği kontrol altında tutmak için diversity (benzerlikleri kabul etmeme oranı) kullanılır. Benzer çözümlerin sayısının popülasyonda belirli bir oranın üzerine çıkması diversity ile sınırlandırılmıştır. İteratif aramalar ile mevcut çözümler üretilen her yeni çözüm ile güncellenecektir. Bu şekilde sürekli arama yapılarak mevcut çözüme göre daha iyi çözümler araştırılır. Bulunan her yeni ve daha iyi çözüm popülasyondaki eski çözümün yerini alır. Güncelleme sırasında clone değerinin benzerlikleri kabul etmeme oranında küçük ya da eşit olması gerekir. Aksi halde ilgili çözümün geliştirilememe sayacı (t_i) bir artırılır.

4.3. Gözcü Arı Fazı

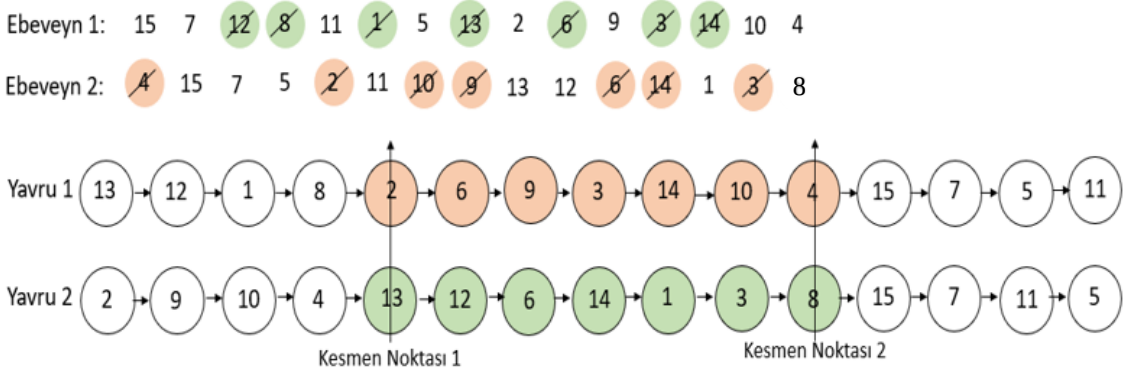
Gözcü arılar belirli bir olasılıkla çözümlere yönlendirilirler. Yerel arama operatörleri, popülasyondaki çözümleri iyileştirir. Çaprazlama operatörleri, ebeveynler tarafından sınırlandırılan bir alt uzay içinde yavru çözümler yaratır. Şekil 4.5'de örnek ebeveyn için sınırlandırılan alanlar gösterilmiştir. İlk ebeveyn için (2-6-9-3-14-10-4) ve ikinici ebeveyn için (13-12-6-14-1-3-8) noktaları yeni üretilecek yavru çözümlere aktarılacak diğer kalan noktalar sıralı çaprazlama tekniğine göre yeniden oluşturulacaktır. Böylece algoritmanın daha kaliteli çözümleri elde etme olasılığı artırılırken yeni çözümlerle global sonuçlara kısa sürede ulaşması hedeflenir.



Şekil 4.5. Ebeveyn çözümler ve kesim noktaları [58]

Önerilen ABC-OX1 yönteminde, bu fazda, çözümler OX1 operatörü ile geliştirilmiştir. Herhangi bir i. çözümün olasılığı $\boldsymbol{prob_i = e^{-1*f(p_i)/mean(f)}}$ formülü ile hesaplanır. Burada f(p_i), i. çözümün uygunluk değerini, mean (f) ise çözümlerin uygunluk

değerlerinin ortalamasını gösterir. Kümülatif olasılıklar ise $\boldsymbol{cum_i = \sum_{i=1}^{P} prob_i}$ formülü ile hesaplanır. Burada (P) popülasyon büyüklüğüdür. OX1 operatöründe kullanılacak ilk ebeveyn çözüm kümülatif ihtimallere göre, ikinci ebeveyn çözüm ise popülasyon içerisinden rastgele olarak seçilir. OX1 operatörü rastgele iki kesme noktası seçer. Şekil 4.5'teki gibi ebeveynler için seçilen noktalar aynen yerlerini korur. İlk yavru çözümü oluşturmak için şekil 4.6'da gösterildiği gibi 1. ve 2. ebeveynde sınırlandırılmış alan haricinde kalan noktalar için karşılaştırma yapılır. Bunun için ilk önce 1. ebeveyndeki noktalar ile 2. ebeveyndeki sınırlandırılmış noktalar karşılaştırılır ve benzer olanlar çıkartılır. Aynı şekilde 2. ebeveyn içinde benzerlikler 1. ebeveynin sınırlandırılmış noktalarına göre elemine edilir. Şekil 4.6'da gösterildiği üzere yeşil renkle elemine edilmiş noktalar 2. ebeveynin sınırlandırılmış alanı için kullanılmaktadır. Bu benzerlikten dolayı seçimden çıkarılmıştır. Aynı durum 2. ebeveyn için pembe renkle işaretlenmiş olarak gösterilmektedir. Benzerliklerde çıkarıldıktan sonra 1. ebeveyn için diğer ebeveyne aktarılan yeni noktalarda dikkate alınarak ilk ebeveynin kullanılmayan noktalarına çapraz sıralı olarak soldan sağa yerleştirilerek yeni bir yavru çözüm elde edilir. Benzer teknik kullanılarak 2. yavru çözümde elde edilir. [11]. Yavru çözümlerin son hali Şekil 4.6'da gösterilmiştir.



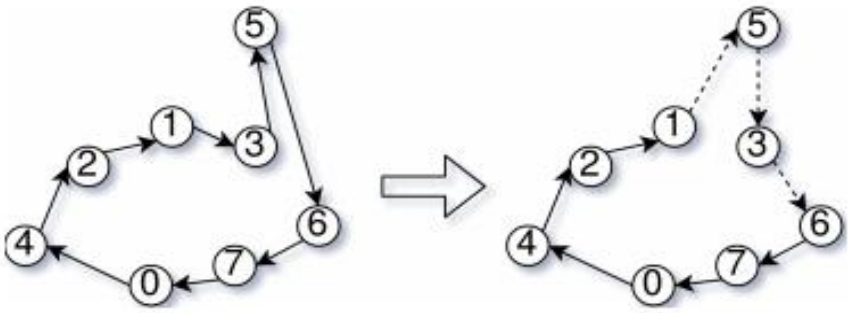
Şekil 4.6. OX1 sıralı çaprazlama operatörü [52]

Ana çözümler, önceden tanımlanmış çaprazlama olasılığına göre seçilmiştir. Eğer yavru çözümler ebeveynlerden daha iyi uygunluk değerlerine sahip ise yeni popülasyonda ebeveyn çözümlerin yerine bu yavru çözümler kullanılır. Şekil 4.6'de gösterildiği gibi OX1 operatörü yavru çözümleri oluşturmak için ikinci kesme noktasından itibaren düğümleri sıralar. Daha sonra sıralı düğümlerden kesme noktaları arasında kalan düğümleri kaldırır ve yavru çözümleri oluşturur. Bu çözümler de aynı işçi arı fazında olduğu gibi, popülasyondaki bireylerin aynı olma sayılarını ifade eden clone değeri ile iteratif olarak kontrol edilir [52]. Elde edilen her yeni yavru çözüm ebeveyn çözüme göre daha iyi ise popülasyondaki eski çözümün yerini alır. Aksi durumda iteratif arama ile

devam edilir. Clone değişkeni besin matrisindeki benzer çözümleri itiratif arama süresince kontrol eder, her yeni çözüm besin matrisindeki çözüm ile aynı olması durumda clone değeri 1 artırılır. Aramalar boyunca besin matrisi içerisindeki benzer çözümlerin oranı diversity ile kontrol edilir. Her iteratif arama sırasında clone değeri diversity (kabul etmeme oranı) değerinden küçük olmalıdır. Aksi halde ilgili çözümün geliştirilememe sayacı (trials) işçi arı fazında olduğu gibi bir artırılır.

4.4. Kâşif Arı Fazı

ABC-OX1 algoritmasında herhangi bir çözümün deneme sayacı algoritmaya giriş olarak verilen denemelerin maksimum limitine ulaştığında ilgili çözüm terk edilir. Bu çözümün yerine rastgele bir çözüm üretilir ve mevcut çözüm olarak kabul edilir. Daha sonra bu çözüm, tur geliştirici 2-opt sezgisel algoritması kullanılarak geliştirilir. 2-opt algoritması, ilk olarak Croes [38] tarafından gezgin satıcı problemini çözmek için önerilmiştir. Sezgisel optimizasyon tekniklerinde 2-opt yaygın olarak kullanılan bir yerel arama yöntemidir. Bu algoritmada bir çözümün iki kenarı kaldırılır. Kaldırılan bu kenarların yerine maliyeti düşürecek şekilde iki yeni kenar eklenir. Müşteri kümesinin sayısına bağlı olarak farklı noktaların tüm kombinasyonları denenerek rota iyileştirme sezgiseli ile daha iyi bir sonuç elde edilmeye çalışılır. Yeni oluşan çözüm önceki çözümden daha iyi ise kabul edilir. Aksi takdirde önceki çözümle devam edilir. Bu işlem, ikili değişikliklerin çözümü iyileştiremediği noktaya kadar devam eder. Şekil 4.7'de 2-opt uygulaması gösterilmektedir. Bunun için aşağıdaki örnekte gösterildiği gibi orijinal rotamız 0-4-2-1-3-5-6-7-0 şeklinde olsun. Sonra 2-opt algoritması gerçekleştirildiğinde, alt rota 3 ile 5'in sırası yer değiştirilir. Böylece oluşan yeni rota 0-4-2-1-5-3-6-7-0'dır. Bu yöntem rota içi iyileştirme olarak ifade edilmektedir.



Şekil 4.7. 2-Opt rota içi iyileştirme modeli [55]

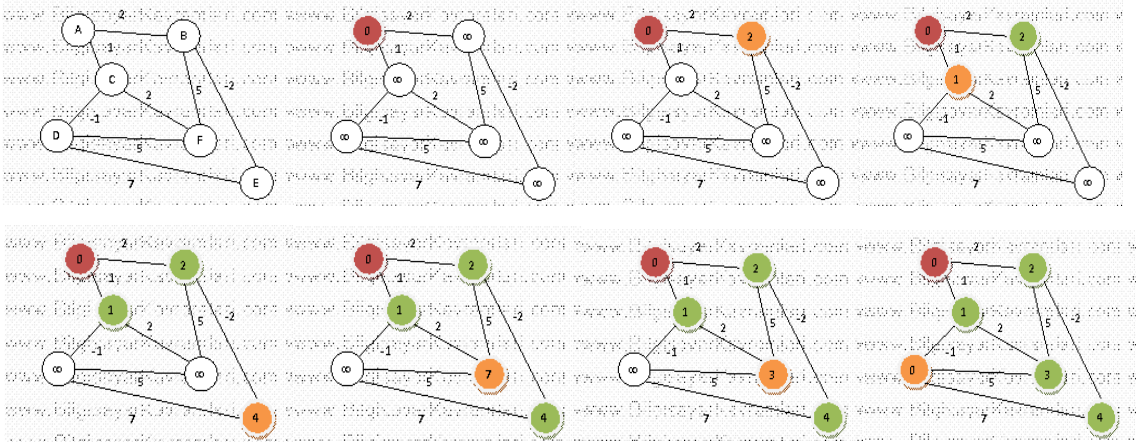
4.5. Popülasyon Çeşitliliği

ABC-OX1 algoritmasında işçi ve gözcü arı fazlarında popülasyon çeşitliliğini kontrol etmek ve yerel minimuma takılmamak için yeni bir yaklaşım kullanılmıştır. Bu yaklaşımda, popülasyon içindeki benzer çözümlerin sayısının belirli bir yüzdelik oranının üzerine çıkması engellenmiştir. Bunun için algoritmaya giriş olarak verilen çeşitlilik yüzdesi (Diversity percentage-D) parametresi kullanılmıştır. İşçi ve gözcü arı fazlarında üretilen her yeni çözüm için bu oran kontrol edilmiştir. Eğer üretilen yeni çözümden popülasyon içinde mevcutsa ve bu çözümlerin popülasyon içindeki yüzdelik oranı algoritmaya giriş olarak verilen çeşitlilik yüzdesi oranından büyük ise yeni çözüm reddedilir. Başka bir ifadeyle, popülasyonda aynı bireyden kaç tane olduğu ve yüzdesi bulunur. Yüzde değer çeşitlilik yüzdesi oranından büyük ise yeni birey popülasyona dahil edilmez. Böylece ABC-OX1 algoritmasının lokal minimumlara takılması önlenir.

4.6. Bellman Algoritması

Bellman algoritmasının amacı graf alan üzerindeki bir noktadan başlayarak belirlenen hedef noktasına giderken en kısa yolu bulmaktır. Algoritma ağırlıklı yol maliyeti üzerinden çalışır ve bu yönü ile literatürdeki Dijkstra algoritmasına benzer. Fakat graflardaki eski değerleri de hesaplama yeteneğinden dolayı daha başarılıdır.

Algoritma Dijkstra algoritmasındaki gibi en küçük değere sahip kenarı tercih etmez. Bütün graf alanındaki kenarları test eder. Bu sayede aç gözlü yaklaşım handikabına düşmez ve her düğüme sadece bir kere geçerek en kısa yolu bulmuş olur [60]. Şekil 4.8'de Bellman algoritmasının çalışması örnek bir graf üzerinde gösterilmiştir.



Şekil 4.8. Bellman algoritması [60]

Çizelge 4.2'de görüldüğü üzere algoritmamız tüm olası kenarları hesaplamak üzere bir yol haritası belirler. Başlangıç noktamız için depo değeri sıfır olarak belirlenir.

Depodan doğrudan erişimi bulunmayan tüm düğüm noktalarına sonsuz değer atanırken erişimi bulunan düğümlere ise maliyet hesabı yapılır. Her gidilen kenar düğümü için maliyet hesabı yapılır. Daha düşük bir maliyet elde edilmiş ise ilgili düğümün maliyeti güncellenir. Bir düğüme birden çok bağlantı olması durumunda en uygun maliyetli olan yol seçilir. Bu şekilde graf alandaki tüm düğümlerden bir kez geçilerek en uygun maliyetli rota elde edilir.

Bellman algoritmasının ilk adımında araç kapasitesi ve ilgili düğümün son servis tarihine bakılır. Eğer kapasite ve zaman kısıtları sağlanıyorsa maliyet hesabı yapılır. Bu işlem sırasında problem alt çözüm kümelerine ayrıştırılarak çözülür. Kısıtlama ihlalleri dikkate alınarak uygun kısıtlar altında rotalar birleştirilerek toplam maliyet belirlenmiş olur.

Çizelge 4.2'deki Bellman algoritmasının detaylarını bir örnek üzerinden açıklayalım. Algoritma tarafından, ilk olarak depo ile doğrudan bağlantısı olmayan tüm düğüm noktalarına sonsuz değer atanır. Sonra çözüm sırasına göre ilk müşterinin maliyet hesabı için depodan bir araç aktif edilir. Konumu depo olan aracın kapasitesi ve mevcut maliyeti sıfır (0) olarak belirlenir.

j, müşteri; n, müşteri sayısı; cost, maliyet; i, adım (iterasyon sayısı); load, yük miktarı; cap, araç kapasitesi ve deadline, müşteri son servis tarihi olarak tanımlanır.

Adım 1: Çözüm kümesindeki tüm müşteriler herhangi bir rotaya atanana kadar yani ($j<=n+1$) şartı için bu işlem tekrarlanır. Rotaya atanan müşterilerin toplam talep miktarları araç kapasitesinden küçükse ($load<=cap$) ve ilgili düğüme aracın ulaştığı andaki zamanı müşteri$_j$'nin son servis kabul tarihinden küçük veya eşit olduğu durumda ($cost_j<=deadline$), müşteri$_j$ talebi mevcut araç kapasitesi için güncellenir ($load=load+dem_j$). Aksi durumda ($i=2: n+1$) çözüm sırasına göre diğer müşteriler için sayaçlar sıfırlanarak *Adım 1* yeniden kontrol edilir.

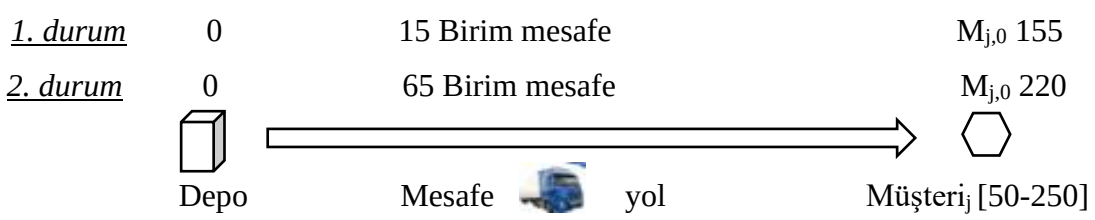
Adım 2: ($i=j$) çözüm sırası ile müşteri$_j$ konumu eşitlenir. ($j<=n+1$) geçerli olduğu sürece çözüm kümesindeki tüm müşterilerin rotaya atanmaları Çizelge 4.2'de gösterildiği gibi gerçekleşecektir. İlk durum ($i=j$) olduğunda yani rotaya atanacak ilk müşteri için her defasında *Adım 2 ve Adım 3* geçerli olacaktır. *Adım 4*'te gösterilen ($i\neq j$) eşitsizliği olduğunda ise aynı rota için çözüm sırasına göre diğer müşterilerin rotaya dahil edilmesini ifade eder. Çizelge 4.2'de rotaya atanma durumları detaylı olarak gösterilmiştir.

Çizelge 4.2. Müşteri$_j$ için rotaya atanma durumu

i	j	Rotaya Atanma Durumu	i=j	i≠j
2	2	√	Rotaya eklendi	
2	3	√		Rotaya eklendi
2	4	x		Rotaya eklenemedi (i+1 ve i=j)
3	3	x	Rotaya eklenemedi (i+1)	
4	4	√	Rotaya eklendi	
4	5	x		Rotaya eklenemedi (i+1 ve i=j)
5	5	√	Rotaya eklendi	
5	6	√		Rotaya eklendi
5	7	√		Rotaya eklendi
5	8	x		Rotaya eklenemedi (i+1 ve i=j)
6	6	√	Rotaya eklendi	

Adım 3: Depodan ilk müşteriye giden araç müşteri$_j$ için zaman penceresi alt sınırı (ZPAS) yani müşteri$_j$ için en erken servise hazır olma tarihinden önce ulaşmış ise beklemek zorundadır. Bu tarihten önce müşteriye ulaşan araç ZPAS'yi bekledikten sonra hizmet verebilir.

Örneğin; ilk müşteriye ulaştığı andaki aracın zamanı 15 birim olsun ve müşterinin en erken servise hazır olma zamanı 50 birim olduğunu düşünürsek araç müşteriye erken ulaştığı için 50 tarihini beklemek zorundadır. Maliyet hesabı yapılırken aşağıdaki Şekil 4.9'da gösterilen ilk durum için depo ile müşteri$_j$ arasındaki 15 birimlik mesafe hesaplama maliyetine dahil edilmez. Çünkü aracımız müşterinin en erken hazır olma zamanı 50 birim kadar beklediği için artık mevcut zamanı 50 birim, servis süresi 90 birim ve depoya dönüş için harcanan maliyet simetrik mesafe kabul edildiğinden dolayı 15 birim olarak hesaplanır. Son durumda ilk müşteri rotaya eklenmiş ve aracımız depo son tarihinden önce depoya dönüş yapmıştır. İlk müşteri için maliyetimiz 50+90+15=155 birim olacaktır. İkinci durumda ise aracımız müşterinin tanımlı zaman penceresi 50 ile 250 zaman birimi arasında müşteriye ulaşması durumunda örneğin aracımızın müşteriye ulaştığı andaki zamanı 65 birim olsun. Depoya dönüş uzaklığını (simetrik mesafe) ve hizmet servis süresi 90 birimi eklediğimizde son durumda maliyetimiz 65+90+65=220 birim olacaktır. Her iki durumun sonunda da algoritma *Adım 6'*dan devam edecektir.

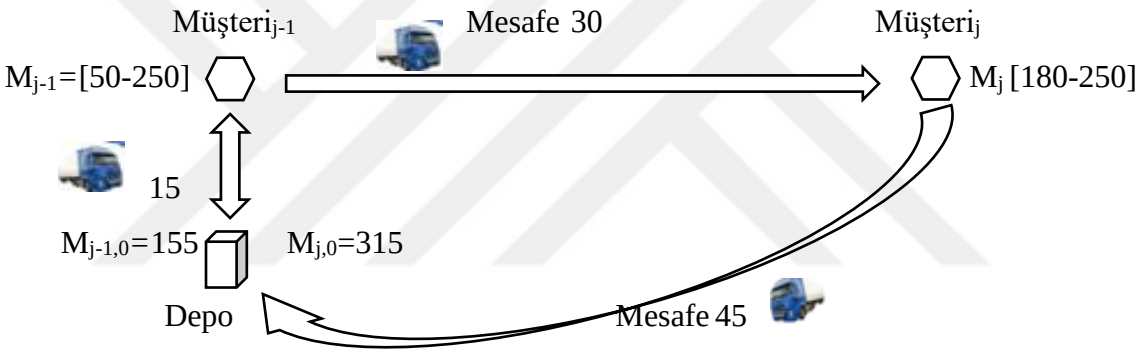


Şekil 4.9. Müşteri$_j$ için rotaya ekleme modeli

Adım 4: (*j<=n+1*) geçerli olduğu sürece ve (*i≠j*) durumunda çözüm sırasına göre bir sonraki müşteri veya müşterileri rotaya eklemek üzere yukarıdaki Çizelge 4.2'de bahsedilen hesaplama modeli gerçekleşir. Bir sonraki müşteriyi rotaya eklemek için *Adım 5'*ten devam edilir.

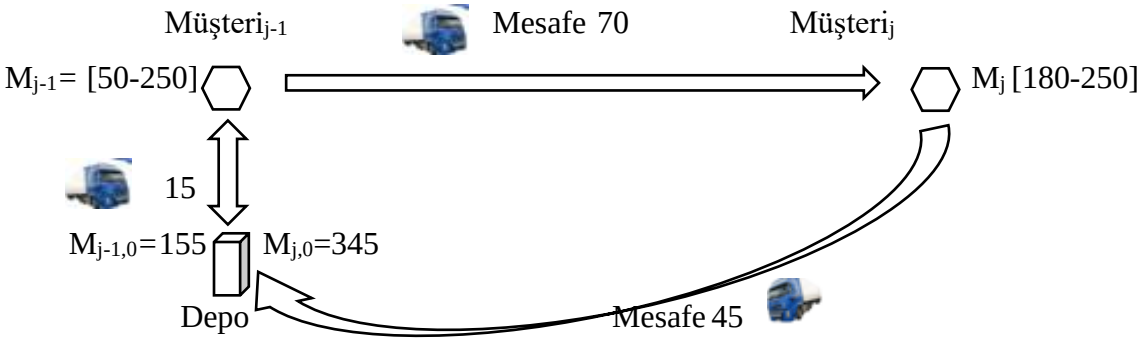
Adım 5: Müşteri$_{j-1}$'den hareket eden aracın bir sonraki müşteri$_j$'yi rotaya eklemesi için müşteri$_{j-1}$'nin depoya dönüş maliyeti çıkartılır. Artık aracın müşteri$_j$ hareket etmeden önceki mevcut zamanı Şekil 4.10'da gösterildiği üzere 155-15=140 birimdir.

İlk durumda aracın bir sonraki müşteri ile arasındaki uzaklık mesafesi eklendiğinde (müşteri$_j$'ye ulaştığı andaki zamanı) 140+30=170 birim olacaktır. Müşteri$_j$'ye ulaştığı andaki zamanı ZPAS'den küçük olduğu için müşteri$_j$'nin en erken servise hazır olma zamanı 180 birimi beklemek zorundadır. Depoya dönüş maliyetini de hesaplarsak servis süresi 90 birim ve depoya dönüş için harcanan mesafe simetrik 45 birim kabul edildiği için 180+90+45=315 birim olacaktır.



Şekil 4.10. Müşteri$_j$ < ZPAS$_j$ durumu

Şekil 4.11'de gösterildiği gibi ikinci durumda eğer araç müşteri$_j$ için tanımlanan zaman penceresi 180 ile 250 zaman birimi arasında müşteriye ulaşması durumunda örneğin aracımızın müşteri$_j$'ye ulaştığı andaki zamanı 210 birim olsun. Hizmet servis süresi ve depoya dönüş uzaklığı 45 birim simetrik mesafe kabul edildiği için son durumda maliyetimiz 210+90+45=345 birim olacaktır. Algoritma *Adım 6'*dan devam edecektir.

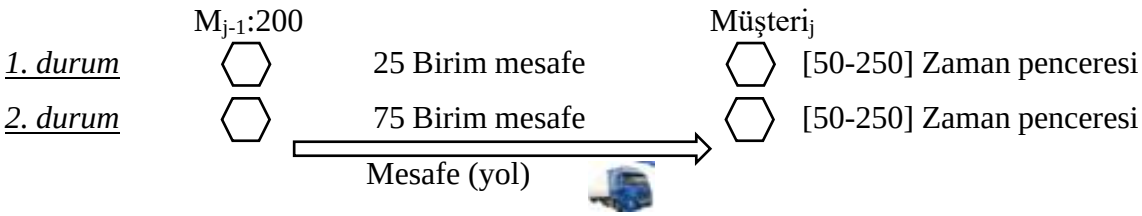


Şekil 4.11. ZPAS$_j$ < müşteri$_j$ < ZPUS$_j$ müşteri zaman penceresi

Adım 6: Kısıtlama ihlallerinin doğrulanması kontrol edilir. Araç kapasitesi, depo son tarihi ve müşteriye ait zaman penceresi üst sınırının (ZPUS) ihlal edilme durumu doğrulanacaktır. Adım 6'ya kadar olan hesaplama modelinde aracın mevcut müşteriye ait tanımlı ZPAS'den önce gelme durumu veya zaman penceresi içerisinde gelmesi halinde oluşan durumlar değerlendirilmiştir. Şimdi ise müşteriye ait ZPUS'nin ihlal edilme durumu değerlendirilecektir.

Şekil 4.12'deki ilk durumumda gösterilen örnekte müşteri$_{j-1}$'nin mevcut maliyeti 200 birim müşteri$_j$'ye ulaşma maliyeti 25 birim eklendiğinde 200+25=225 birim ile müşteri$_j$'ye ulaşmıştır. Çizelge 4.3'te ifade edildiği gibi araç kapasitesi ve müşteri$_j$'ye ulaştığı andaki maliyeti ZPUS (250) değerinden küçük olduğu için müşteriye ait düğüm maliyeti güncellenecektir. Algoritmanın ilk başında depodan doğrudan erişimi bulunmayan noktalara sonsuz değer atamıştık. Artık ilgili düğüme daha uygun bir maliyet ile ulaşıldığı için düğüme ait maliyet değeri güncellenir. Rotaya eklenen her düğüm P vektöründe alt çözüm olarak kaydedilir. P vektöründeki çözüm sırasına ait bilgiler daha sonra toplam maliyet ve rota sayısının hesaplanmasında değerlendirilir.

Şekil 4.12'de gösterilen 2. durumda ise müşteri$_{j-1}$'nin mevcut maliyeti 200 birim müşteri$_j$'ye ulaşma maliyeti 75 birim eklendiğinde 200+75=275 birim ile müşteri$_j$'ye ulaşmıştır. Müşteri$_j$ için son servise kabul zamanı ZPUS aşılmış olduğu için rota ekleme işlemi yapılamaz. Müşteri$_j$ için tanımlı zaman penceresi üst sınırı 250 birimdir. Araç müşteri$_j$'ye ulaştığı andaki zamanı 275 birim olduğu için müşteri$_j$'ye hizmet veremeyecektir. Kısıtlama ihlali doğrulanmadığı için (*i=2: n+1*) sayaçlar sıfırlanır ve mevcut müşteriye depodan aktif edilen farklı bir araç ile hizmet verilir.



Şekil 4.12. Müşteri$_j$ < ZPUS$_j$ durumu

Yukarıda bahsedilen örnek için Bellman algoritmasına ait modelin kaba kodu Çizelge 4.3'te gösterilmiştir. Çizelgede ifade edilen kısaltmalar aşağıda açıklanmıştır.

SZ: Bir müşteri için sunulan hizmet süresini ifade etmektedir.

dem$_j$: Müşteri talebi j (j müşterisine ait talep miktarını göstermektedir)

L: Rota kısıtlaması (Toplam maliyet, depo son tarihinden küçük olmalı)

K: Araç kapasitesi (Araç kapasitesi, aracın taşıyabileceği maksimum yük miktarı)

Çizelge 4.3. Bellman algoritması kaba kodu [58]

```
1 dist0 = 0                                          // Başlatma
2 for i = 1 to n do
3    disti = +∞                                      // Her bir düğüm için çok büyük bir başlangınç değeri ayarlama
4 end for
5  for i = 1 to n do                                 //  Düğüm
6      load = 0; cost = 0;  j = i                    // Kapasite, maliyet ve ilk konumun ayarlanması
7      repeat                                        // Kapsitesi, düğüm ve zaman penceresi uygun olduğu sürece tekrarla
8         load = load + demj                         // Artan rota talebi (Araç kapasitesini güncelle)
9         if i = j then                              // Maliyet hesabı
10           if  edge0,j < ZPASj then                // Aracın alt zaman sınırından önce gelme durumunun kontrol edilmesi
11              cost = ZPASj + SZ + edge0,j          // Zaman aralığı sınırı
12           else                                    // Aracın düğüme zaman penceresinde gelmesi
13              cost = edgej,0 + SZ + edge0,j        // 0,j + servis süresi + j,0  rotaya ekleme maliyeti
14           end if
15        else
16           if  cost – edgej-1,0 + edgej-1,j < ZPASj then     // Bir sonraki düğümü ekleme<ZPASj
17              cost = ZPASj +  edgej,0 +  SZ
18           else                                    // ZPASj < costj < ZPUSj zaman penceresine uygun ise
19              cost = cost - edgej-1,0 + edgej-1,j + edgej,0 + SZ
20           end if
21        end if
                                  // Kısıtlama ihlali doğrulanıyor kapasite maliyet ve müşteri hizmet alma üst sınırı için
22        if (load ≤ K) and (cost ≤ L) and (cost - edgej,0 – SZ ≤ ZPUSj) then
23           if disti-1 + cost  < distj then          // Elde edilen maliyet düğüm maliyetinden küçükse
24              distj = disti-1 + cost                // Düğüm maliyeti güncelle
25              Pj = i -1                             // Sonlandırma noktasının ayarlanması rota haritası
26           end if
27              j = j + 1                             // Artan işaretçi (bir sonraki konum veya müşteri)
28        end if
29     until (j > n) or (load > K) or (cost > L) or (cost – edgej0 – SZ > ZPUSj)
30 end for
```

Probleme ait çözüm tamsayı dizisi şeklinde temsil edilir ve rota bilgisi içermez. Çözümü rotalara ayırmak ve uygunluk değerini hesaplamak için aşağıdaki adımlar takip edilir. P vektörüne ait çözümlerden ARP çözümünü elde etmek için Çizelge 4.5'teki algoritma uygulanır**.**

Adım 1: P çözüm kümesi için i=P_j sonlandırma noktasıyla rota atamaları gerçekleştirilir. Bunun için P_j'deki sonlandırma noktasına göre müşteriler sırayla t+1 yolculuğuna eklenir.

İlk durumda; j=101 için Çizelge 4.4a'da gösterildiği gibi P_j dizisine ait son dizi elemanının (indis değeri: 101) 94 olduğu görülmektedir. Bu duruma göre ilk sonlandırma noktası i=P_{101} için 94'ten 101'e kadar olan dizi elemanlarını içermektedir. $P_{94\text{-}101}$ dikkate alınırsa (indis değeri:94-101) A = [95 96 97 98 99 100 101] için t+1 yolculuğu olarak belirlenir.

Adım 2: Bir önceki adımda elde edilen alt çözüm kümesi yani A vektöründeki dizi (indis değeri) sırasına göre Çizelge 4.4b'deki çözüm sırasından çıkararak ilk rotamızı elde etmiş oluruz. Buna göre; Çizelge 4.4b'de 95. İndis değerine karşılık gelen çözümün düğüm değeri 55 olarak bulunur. Benzer eşleştirme çözüm sırasına göre yapıldığında ROTA-1= [55 54 57 59 61 60 81] elde edilmiş olur.

Adım 3: Rota sonlandırmasının ilk değeri i=94 olarak belirlenmişti. Diğer rota elemanlarını da belirlemek üzere (*j=i)* sonlandırma noktasına göre *Adım 1*'den döngü devam edecektir. P_{94}'ten rota alt çözümlerini elde etmek için i=P_{94} sonlandırma noktasıyla rota atamaları gerçekleştirilir. P vektöründeki 94. çözüm sırası (indis değeri:94) 91 olduğu görülmektedir. Buna göre bir önceki sonlandırma noktasına göre hesaplama yapılırsa C= [92 93 94] için t+1 yolculuğu olarak belirlenir. Benzer eşleştirme çözüm sırasına göre yapıldığında ROTA-2= [58 56 69] elde edilmiş olur. Bu şekilde P vektöründeki tüm çözümler elde edilene kadar algoritma tekrar edecektir. Sıfır (0) depoyu temsil etmektedir.

Çizelge 4.4. Vektör P_j ve ARP çözümünün elde edilmesi

0	1	1	1	1	1	1	7	7	9
9	9	9	9	14	14	14	14	18	18
20	20	20	20	20	20	20	20	20	21
30	30	32	32	32	35	35	37	38	39
39	41	41	41	41	41	41	41	48	48
48	48	48	48	54	54	56	57	57	57
57	57	57	63	63	63	63	67	67	67
67	67	67	73	73	75	75	75	75	79
79	81	81	81	81	81	81	87	87	87
87	91	91	91	94	94	94	94	94	94
94									

a) P_j

1	82	79	77	72	71	74	10	80	84
83	85	86	89	91	88	87	90	99	92
97	96	95	93	94	98	101	100	3	2
6	76	4	8	5	9	78	7	11	14
12	18	19	20	16	17	15	13	25	26
28	31	53	50	44	48	42	43	41	45
47	46	49	33	32	52	51	34	36	38
39	40	37	30	35	29	27	24	23	21
22	68	66	64	63	67	70	75	73	62
65	58	56	69	55	54	57	59	61	60
81									

b) Çözüm sırası

Çizelge 4.5. Vektör P'den ARP çözümünü çıkarmak için algoritma [58]

```
1  for i = 1 to n do
2      trip(i) = Φ                    // Başlatma
3  end for
4  t = 0                              // Alt çözümlerin sayısı
5  j = n
6  repeat                             // Pj+1'den j'ye müşterileri t+1 yolculuğuna ekleyin
7     t = t + 1                       // Seyahat artışı
8     i = Pj                          // Sonlandırma noktası rota atamaları
9    for
10       k = i +1 to j  do           //  sırya al(trip(t),k)    ( i+1 den j'ye müşterileri içeren rota)
11   end for
12       j = i                       // Ayarlama Pj den rotaları çıkarmak için
13 until  i = 0
```

Vektör P'den elde edilen rotalar ile toplam maliyet ve rota sayısı hesaplanır.

5. ARAŞTIRMA SONUÇLARI VE TARTIŞMA

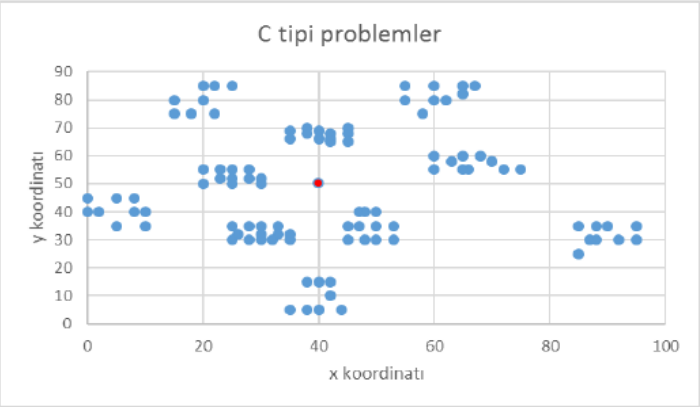
ABC-OX1 algoritması Matlab'da kodlanmıştır. Deneysel çalışmalar Intel Core i5 1.60 Ghz CPU, 16 Gb RAM ve 64 bitlik Windows 10 işletim sistemli bir notebook bilgisayarda uygulanmıştır. Önerilen yöntemin performans testleri Solomon'un veri kümeleri üzerinde yapılmıştır.

5.1. Kullanılan Veri Kümeleri

Bu tez çalışmasında SZPARP'nin çözümü için yeni bir yöntem önerilmiştir. Önerilen yöntemin testi için Solomon'un veri kümeleri kullanılmıştır. Solomon'un ZPARP için önerdiği veri kümeleri http://w.cba.neu.edu/~msolomon/problems.htm adresinden alınmıştır. Bu veri kümeleri tip-1 ve tip-2 şeklinde iki gruba ayrılmaktadır. Her grup da kendi içerisinde C, R ve RC olarak üç farklı veri kümesinden oluşmaktadır. İlk gruptaki örneklerde araç kapasiteleri küçük ve zaman pencereleri dar iken diğer gruptaki örneklerde araç kapasiteleri büyük ve zaman pencereleri daha geniştir. C tipi örneklerde müşterilerin hizmet alma süreleri genişken R ve RC tipi örneklerde ise hizmet alma süreleri dardır. Tip-1 R veri kümesinde 12 adet, C veri kümesinde 9 adet ve RC veri kümesinde 8 adet örnek bulunurken tip-2 veri kümelerinde sırasıyla R 11, C 8 ve RC 8 adet olmak üzere toplamda 56 adet örnek veri kümesi bulunmaktadır [25].

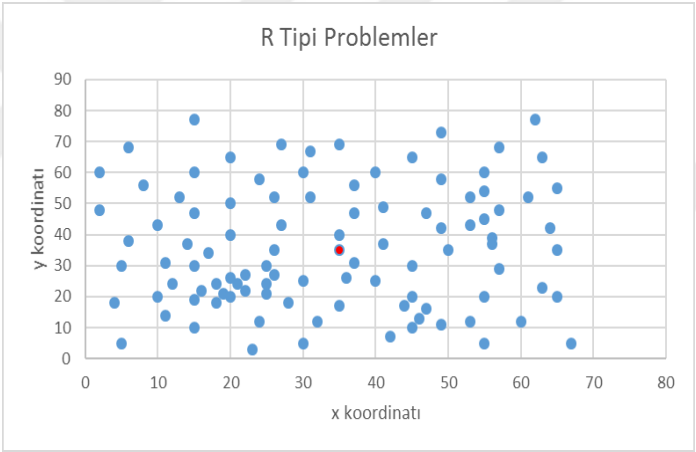
Örnek veri kümelerinde müşteri numarası, (x,y) düzlemindeki koordinatları, talep miktarı, servise en erken hazır olma zamanı, en son servis zamanı ve hizmet alma süreleri gibi bilgiler bulunmaktadır.

Şekil 5.1'de mavi ile belirtilen noktalar C tipi müşterilerin x ve y koordinat düzlemindeki konumlarını göstermektedir. Kırmızı nokta işareti depoyu göstermektedir. C tipi örneklerde müşteriler belirli bölgelerde yoğunlaşmış ve toplu halde bulunmaktadır. Müşterilerin hizmet alma süreleri 90 birimdir. Araç kapasiteleri C101'den C109'a kadar olan veri kümeleri için 200 birim diğer tip-2 C201'den C208'e kadar olan örnekler için 700 birimdir.



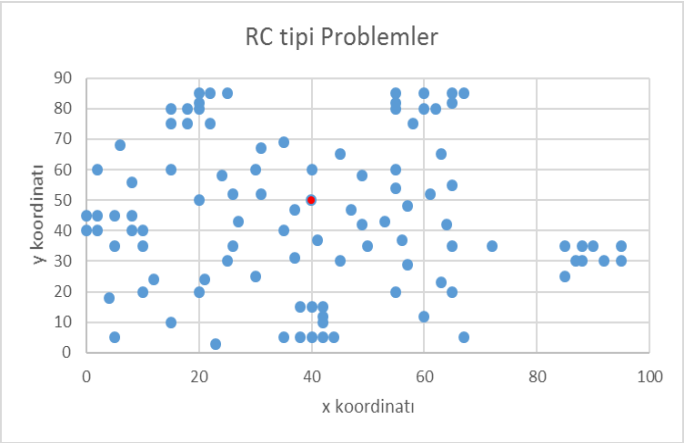
Şekil 5.1. C tipi problem müşteri konumları [5]

Şekil 5.2'de R tipi problem için rastgele dağılmış müşteri konumları gösterilmiştir. Bu tür problemleri zaman pencereli olarak çözmek zor ve karmaşık bir optimizasyon problemi olarak görülmektedir. R tipi problemlerde müşterilerin hizmet alma süreleri 10 birimdir. Araç kapasiteleri R101'den R112'ye kadar olan veri kümeleri için 200 birim diğer tip-2 örnek veri kümeleri R201'den R211'e kadar olan örnekler için 1000 birimdir.



Şekil 5.2. R tipi problem müşteri konumları [5]

Şekil 5.3'te RC tipi problemde bir grup düğüm kümelenmiş halde iken diğer düğümler rastgele dağıtılmıştır. RC tipi problemlerin de çözümü oldukça güçtür. Bu problemlerde müşterilerin hizmet alma süreleri 10 birimdir. Araç kapasiteleri RC101'den RC112'ye kadar olan veri kümeleri için 200 birim diğer tip-2 örnekleri RC201'den RC211'e kadar olan için 1000 birimdir.



Şekil 5.3. RC tipi problem müşteri konumları [5]

Solomon'un veri kümeleri toplamda 56 adet örnekten oluşmaktadır. Her bir örnek 100 müşteri ve bir depodan oluşmaktadır. Veri kümeleri maksimum 25 araç ile çözülür. Her bir müşteriye ait servise hazır olma zamanı ve son hizmet alma zamanları bulunmaktadır. Yani her müşteri için tanımlı bir zaman penceresi bulunmaktadır. Müşteri talepleri farklılık gösterse de bir rotadaki müşterilerin talepleri toplamı araç kapasitesinden büyük olamaz. Her araç rota turuna depodan başlar ve rota turunu tamamladıktan sonra depo son tarihinden önce tekrar depoya dönmek zorundadır. Örnek veri setlerinde müşteri numarası, (x,y) düzlemindeki koordinatları, talep miktarı, servise en erken hazır olma zamanı, en son servis alma zamanı ve servis hizmet süreleri gibi bilgiler bulunmaktadır. Depo ve müşterilerin yerleri, koordinat düzleminde (x,y) şeklinde belirlenmiş ve sabittir. Müşterilerin birbirlerine olan uzaklıkları Öklid bağlantısı ile hesaplanır. Müşterilerin birbirlerine olan uzaklıkları ve toplamda elde edilen rota mesafesi için hesaplanan değer birim olarak ölçülmektedir. Ölçülen bu değer aynı zamanda zaman değeri (süre) olarak da değerlendirilmektedir.

5.2. ABC-OX1 İçin Yerel Arama Operatörü Seçimi

ABC-OX1 algoritmasında kullanılacak yerel arama operatörlerine birtakım ön deneyler uygulanarak karar verilmiştir. Güvenilirliği artırmak için her deney 10 kez tekrarlanmıştır. Sonuçların en iyi ve ortalama değerleri kaydedilmiştir. Bu operatörlerin performans analizleri 12 farklı örnek (C101, C102, C201, C102, R101, R102, R201, R202, RC101, RC102, RC201 ve RC202) üzerinden gerçekleştirilmiştir.

Yerel arama operatörlerinin performans analizleri, iterasyon sayısı 1000, popülasyon büyüklüğü 200, limit değeri 75 ve çeşitlilik yüzdesi 15 alınarak gerçekleştirilmiştir. Deneylerde insertion, reversion, swap ve divideandswap operatörlerinin tekli, ikili ve

üçlü kombinasyonları kullanılmıştır. İlk turda 10 çalıştırma için her operatör tek tek sınanmış, elde edilen en iyi değerler ile ortalama değerler çizelgeye kaydedilmiştir. İkinci turda ise insertion-swap, insertion-reversion, insertion-divideandswap, reversion-swap, reversion-divideandswap ve swap-divideandswap ikili grupları eşit olasılık hesabına göre çalıştırılmış elde edilen en iyi değerler ile ortalama değerler çizelgeye kaydedilmiştir. Üçüncü turda insertion-swap-reversion, insertion-reversion-divideandswap, insertion-swap-divideandswap ve reversion-swap-divideandswap üçlü grupları eşit olasılık hesabına göre çalıştırılmış ve sonuçlar aynı şekilde ilgili çizelgelere kaydedilmiştir. Son olarak dört operatör insertion, reversion, swap ve divideandswap birlikte eşit olasılık hesabına göre çalıştırılarak elde edilen en iyi değerler ile ortalama değerler çizelgeye kaydedilmiştir.

Çalışma sonuçları Çizelge 5.1 ve 5.2'de verilmiştir. Çizelge 5.1 elde edilen en iyi değerleri, Çizelge 5.2 ise ortalama değerleri göstermektedir. Her iki çizelgede en iyi değerlere sahip olanlar koyu renkle belirtilmiştir. Deney sonuçları değerlendirildiğinde insertion-swap-divideandswap operatör grubunun diğer operatör gruplarına göre 6 kez en iyi, 2 kez de en iyi ortalama değeri elde ettiği gözlenmiştir. Çizelge 5.2'de elde edilen ortalamaların tekrar ortalaması alındığında insertion-swap-divideandswap operatör grubunun yine en iyi değeri aldığı görülmüştür.

Böylece tüm çalışma süresince belirlenen örnekler ve hesaplama modeline göre en iyi değerleri insertion-swap-divideandswap operatör grubunun ürettiği gözlenmiştir. Sonuç olarak, ABC-OX1 algoritması için en uygun yerel arama operatörlerinin insertion-swap-divideandswap operatör grubu olduğuna karar verilmiştir.

Çizelge 5.1. Elde edilen en iyi değer açısından yerel arama operatörlerinin karşılaştırılması

OPERATÖR GRUPLARI	C101	C102	C201	C202	R101	R102	R201	R202	RC101	RC102	RC201	RC202	En iyi değer Ort.
Divideandswap	**828.94**	882.64	**591.56**	**591.56**	1772.55	1584.86	1297.56	1156.81	1809.58	1655.00	1474.27	1250.42	1241.31
Insertion	**828.94**	833.66	**591.56**	**591.56**	1715.12	1536.88	1252.66	1132.92	1742.18	1586.99	1422.59	1191.38	1202.20
Reversion	**828.94**	874.61	**591.56**	609.21	1763.10	1602.95	1273.22	1171.17	1755.46	1647.63	1435.38	1211.49	1230.39
Swap	**828.94**	**828.94**	**591.56**	638.59	1723.71	1592.66	1247.70	1143.99	1742.85	1581.87	1466.31	1197.36	1215.37
Insertion-Reversion	**828.94**	862.01	**591.56**	**591.56**	1731.17	1535.31	1244.69	1137.95	1776.45	1609.92	1427.92	1211.01	1212.37
Insertion-Swap	**828.94**	**828.94**	**591.56**	**591.56**	**1692.25**	1549.98	1259.11	1124.42	1740.86	**1545.62**	1448.84	1205.13	1200.60
Insertion- Divideandswap	**828.94**	**828.94**	**591.56**	**591.56**	1725.57	1535.29	1230.58	1140.93	1735.94	1588.06	1378.32	1217.77	1199.45
Reversion- Divideandswap	**828.94**	884.93	**591.56**	**591.56**	1726.35	1610.73	1254.63	1183.76	1826.70	1611.68	1445.39	1236.25	1232.70
Reversion-Swap	**828.94**	887.35	**591.56**	618.58	1744.99	1578.45	1243.44	1141.76	1820.08	1618.49	1394.39	1224.98	1224.41
Swap- Divideandswap	**828.94**	873.36	**591.56**	627.46	1771.46	1590.89	1261.12	1157.35	1792.55	1590.03	1460.52	1205.33	1229.21
Insertion-Reversion- Divideandswap	**828.94**	832.61	**591.56**	**591.56**	1717.38	1563.39	1234.25	1157.33	1740.18	1615.12	1408.53	1202.90	1206.98
Insertion-Reversion-Swap	**828.94**	**828.94**	**591.56**	628.91	1709.41	**1530.63**	1249.94	1139.62	**1695.86**	1562.76	1418.66	1192.25	1198.12
Insertion-Swap- Divideandswap	**828.94**	**828.94**	**591.56**	**591.56**	1708.11	1562.42	**1216.01**	1126.03	1747.23	1591.08	**1356.46**	1213.15	**1196.79**
Reversion-Swap- Divideandswap	**828.94**	868.31	**591.56**	**591.56**	1754.34	1585.18	1252.49	1153.41	1742.21	1624.55	1411.43	1162.19	1213.85
Insertion-Swap-Reversion- Divideandswap	**828.94**	**828.94**	**591.56**	**591.56**	1729.58	1545.76	1261.48	**1108.72**	1757.53	1600.52	1385.79	**1151.44**	1198.48

Çizelge 5.2. Ortalama değerler açısından yerel arama operatörlerinin karşılaştırılması

OPERATÖR GRUPLARI	C101	C102	C201	C202	R101	R102	R201	R202	RC101	RC102	RC201	RC202	Ort. Ortalaması
Divideandswap	872.16	959.32	614.36	627.94	1806.11	1632.33	1340.11	1220.25	1864.16	1717.35	1517.22	1295.46	1288.90
Insertion	851.11	879.09	600.71	629.82	1743.28	**1586.36**	1281.21	1179.62	1793.38	1627.17	1469.67	1243.04	1240.37
Reversion	870.37	933.91	625.96	658.06	1811.09	1670.48	1318.19	1214.46	1843.02	1687.73	1509.24	1264.16	1283.89
Swap	864.02	929.97	605.28	668.08	1778.29	1638.60	1296.68	1205.35	1808.88	1658.61	1515.48	1253.69	1268.58
Insertion-Reversion	843.81	927.37	607.25	643.76	1770.29	1598.75	1271.51	1174.47	1815.26	1650.96	1481.24	1249.90	1252.88
Insertion-Swap	849.52	895.40	598.27	628.31	**1736.79**	1598.54	1292.56	1171.54	1793.37	**1608.21**	1494.87	1227.21	1241.22
Insertion Divideandswap	850.41	**885.92**	598.12	627.40	1752.43	1590.65	1268.78	**1165.49**	1818.51	1629.40	**1446.57**	1254.53	1240.68
Reversion-Divideandswap	864.50	945.01	611.76	634.14	1792.36	1640.50	1311.97	1201.97	1874.14	1687.35	1509.21	1265.41	1278.19
Reversion-Swap	859.68	916.52	616.86	649.96	1800.70	1638.15	1298.23	1197.61	1850.54	1655.21	1473.25	1274.97	1269.31
Swap-Divideandswap	868.81	932.59	605.81	648.93	1806.64	1623.25	1287.09	1187.52	1836.03	1643.08	1507.71	1247.45	1266.24
Insertion-Reversion-Divideandswap	835.06	889.54	604.82	628.87	1764.73	1626.73	1274.28	1181.70	1805.67	1666.78	1478.20	1243.79	1250.01
Insertion-Reversion-Swap	853.01	897.03	603.05	653.62	1759.29	1591.38	1289.12	1168.64	**1776.90**	1646.98	1466.88	1243.17	1245.76
Insertion-Swap-Divideandswap	846.95	895.37	**591.56**	624.23	1747.46	1599.47	**1258.69**	1174.37	1794.79	1627.64	1454.86	1238.58	**1237.83**
Reversion-Swap-Divideandswap	**834.84**	909.50	595.15	634.50	1806.64	1628.01	1299.49	1188.27	1823.09	1680.17	1476.39	1261.80	1261.49
Insertion-Swap-Reversion- Divideandswap	854.99	904.23	603.37	**614.20**	1775.86	1598.86	1293.97	1168.67	1807.38	1652.50	1473.59	**1226.31**	1247.83

5.3. ABC-OX1 İçin Parametre Seçimi

Parametre değerleri algoritmaların performanslarını doğrudan etkilerler. Bu yüzden doğru parametreleri belirlemek kaliteli sonuçlar elde etmek için önemlidir. Yerel arama operatörlerinde olduğu gibi ABC-OX1 algoritmasında da parametre değerlerini tespit etmek için de birtakım ön deneyler yapılmıştır. Güvenilirliği artırmak için her deney 10 kez tekrarlanmıştır. Sonuçların en iyi ve ortalama değerleri kaydedilmiştir. Bu parametrelerin tespitinde kullanılan, operatör tespitinde de kullandığımız veri kümelerinden 12 örnek (C101, C102, C201, C102, R101, R102, R201, R202, RC101, RC102, RC201 ve RC202) üzerinde uygulanmıştır.

Yerel arama operatörü olarak daha önce en iyi operatör grubu olarak belirlenen insertion-swap-divideandswap operatörleri eşit olasılıkla kullanılmıştır. İterasyon sayısı sabit ve 1000 olarak alınmıştır. İterasyon sayısı, popülasyon büyüklüğü, deneme limiti ve çeşitlilik yüzdesi değerleri için 10 farklı parametre grubu oluşturulmuştur. Veri kümelerinin çeşitliliğinin fazla olması ve çözümlerinin çok uzun gerektirmesinden dolayı parametre tespitinde aşağıda ifade edilen parametre grupları üzerinden deneysel çalışmalar sürdürülmüştür. Bu gruplar için elde edilen sonuçlar Çizelge 5.3 ve 5.4'te verilmiştir. Sonuçlar incelendiğinde, ABC-OX1 için en iyi parametre değerlerinin iterasyon sayısı, popülasyon büyüklüğü, limit değeri ve çeşitlilik yüzdesi için sırasıyla 1000, 500, 125 ve 15 olduğu tespit edilmiştir.

Çizelge 5.3. Elde edilen en iyi değer açısından ABC-OX1'in parametre performanslarının karşılaştırılması

Örnekler	1000-200-50-15	1000-200-100-20	1000-300-75-15	1000-300-150-20	1000-400-100-15	1000-400-150-20	1000-400-200-25	1000-500-125-15	1000-500-175-20	1000-500-250-25
C101	**828.94**	**828.94**	**828.94**	**828.94**	**828.94**	**828.94**	**828.94**	**828.94**	**828.94**	**828.94**
C102	**828.94**	**834.64**	**828.94**	**828.94**	**828.94**	**828.94**	**828.94**	**828.94**	**828.94**	**828.94**
C201	**591.56**	**591.56**	**591.56**	**591.56**	**591.56**	**591.56**	**591.56**	**591.56**	**591.56**	**591.56**
C202	**591.56**	**591.56**	**591.56**	**591.56**	**591.56**	**591.56**	**591.56**	**591.56**	**591.56**	**591.56**
R101	1726.35	1703.73	1697.90	1706.88	1702.07	1691.76	1696.66	1702.01	**1674.74**	1679.62
R102	1534.19	1544.77	1531.43	1552.57	1530.64	1515.70	1538.75	**1503.81**	1531.77	1525.69
R201	1248.72	1249.00	1217.15	1248.02	1220.56	**1203.88**	1219.35	1212.89	1227.49	1212.68
R202	1143.45	1131.66	1104.21	1121.59	1129.74	1115.90	1131.62	1104.34	1118.81	**1101.27**
RC101	1750.48	1774.11	1719.33	1718.07	1719.17	1737.55	1729.53	**1704.00**	1731.04	1724.43
RC102	**1534.10**	1573.49	1564.88	1560.25	1582.76	1567.04	1543.31	1556.94	1565.93	1550.28
RC201	1423.36	1422.79	1387.32	1436.89	1418.99	1402.32	1374.62	**1340.19**	1383.43	1367.49
RC202	1186.97	1177.08	1175.93	1183.82	1160.13	1176.66	1176.03	**1142.67**	1161.05	1142.70

Çizelge 5.4. Ortalama değer açısından ABC-OX1'in parametre performanslarının karşılaştırılması

Örnekler	1000-200-50-15	1000-200-100-20	1000-300-75-15	1000-300-150-20	1000-400-100-15	1000-400-150-20	1000-400-200-25	1000-500-125-15	1000-500-175-20	1000-500-250-25
C101	855.32	844.45	830.79	842.40	**828.94**	**828.94**	**828.94**	**828.94**	843.82	**828.94**
C102	889.87	911.97	868.40	866.62	870.82	**838.64**	858.40	848.05	850.81	869.14
C201	619.56	**591.56**	**591.56**	**591.56**	**591.56**	**591.56**	596.40	**591.56**	**591.56**	596.36
C202	619.77	638.20	602.08	619.21	616.09	605.67	606.92	**595.51**	603.94	596.28
R101	1763.65	1754.53	1719.89	1749.67	1721.40	1722.07	1723.30	1726.71	1718.01	**1716.17**
R102	1592.48	1592.67	1579.51	1584.03	1567.47	**1547.58**	1558.91	1549.49	1562.90	1561.41
R201	1298.78	1281.41	1255.88	1266.56	1245.06	1249.92	1251.02	1239.88	1245.19	**1233.76**
R202	1174.87	1170.16	1169.77	1147.81	1154.25	1154.57	1165.23	**1135.36**	1149.47	1137.50
RC101	1810.78	1811.72	1768.11	1796.11	1757.31	1776.65	1771.78	**1753.44**	1780.51	1757.63
RC102	1620.70	1638.08	1617.53	1615.02	1621.74	1613.68	1589.91	1589.78	1599.59	**1588.35**
RC201	1461.52	1479.51	1449.67	1470.07	1451.10	1442.21	1429.85	**1402.05**	1421.00	1433.41
RC202	1245.35	1229.06	1209.22	1225.60	1211.57	1214.93	1207.75	**1193.80**	1197.52	1201.98

5.4. Deneysel Sonuçlar ve Tartışma

ABC-OX1 algoritması için yerel arama operatörlerinin ve parametre değerlerinin nasıl tespit edildiği yukarıdaki bölümlerde anlatılmıştır. Bu bölümde ise temel ABC algoritması ile yeni geliştirilen ABC-OX1 hibrit algoritmasının performansları karşılaştırılmıştır. Bunun için Solomon'un 56 adet örnek kullanılmıştır. ABC ve ABC-OX1 algoritmaları Matlab'da kodlanmıştır. Deneysel çalışmalar Intel Core i5 1.60 GHz CPU, 16 GB RAM ve 64 bitlik Windows 10 işletim sistemli bir notebook bilgisayarda uygulanmıştır.

ABC ve ABC-OX1 algoritmalarını adil bir şekilde karşılaştırmak için yerel arama operatörleri ve parametre değerleri aynı alınmıştır. Yerel arama operatörleri insertion, swap ve divideandswap eşit olasılık hesabı ile kullanılmıştır. Parametre değerleri ise itreasyon sayısı 1000, popülasyon büyüklüğü 500, limit değeri 125 ve çeşitlilik yüzdesi 15 olarak alınmıştır. ABC-OX1 algoritması, popülasyon çeşitliliğini sağlama stratejisi ve gözcü arı fazında kullanılan OX1 operatörü ile ABC algoritmasından farklıdır. Diğer bütün prosedürler her iki algoritmada da aynıdır.

Çizelge 5.5 için Solomon'un tip-1 C serisi 9 örnek, Çizelge 5.6 için tip-2 C serisi 8 örnek kullanılmıştır. C serisi veri kümeleri daha çok müşterilerin toplu halde bulunduğu, müşterilerin hizmet alma sürelerinin geniş olduğu ve çözümü kolay olan problem türüne örnektir. ABC ve ABC-OX1 algoritmaları her bir örnek üzerinden 10 kez çalıştırılmıştır. ABC-OX1'de kullanılan OX1 operatörü ve popülasyondaki çeşitliliği kontrol etme yaklaşımı ABC ile ABC-OX1 arasındaki temel farklardır. Her iki yöntem için elde edilen sonuçlar Çizelge 5.5 ve Çizelge 5.6'da gösterilmiştir.

Çizelgelerde BKS (Best Known Solution) literatürdeki bilinen en iyi çözümü ifade etmektedir. ABC ve ABC-OX1 algoritmaları 10 kez aynı şartlar altında tekrarlı olarak çalıştırılmıştır. Elde edilen sonuçlar içerisindeki en iyi çözüm değerleri Çizelge 5.5 ve Çizelge 5.6'da en iyi değer sütununda gösterilmiştir. 10 çalıştırmanın ortalama değerleri ise ortalama değer sütununda verilmiştir. Std. Sap., (standart sapma) değerini ifade etmektedir. %Hata, bilinen en iyi değere göre hata yüzdesini göstermektedir. %Hata 10 kez çalıştırılan algoritmada elde edilen en iyi çözüm ile literatürde bilinen en iyi çözümün karşılaştırılmasıdır. Bu karşılaştırma modeli yüzdelik sapma olarak hesaplanmaktadır. Hata yüzdesi aşağıdaki gibi formüle edilir.

$$\%\text{Hata} = (\text{En iyi değer} - \text{BKS}) / \text{BKS} \quad * 100$$

Çizelgelerde gösterilen mesafe ve araç sütunu toplam tur uzunluğunu ve araç sayısını ifade etmektedir. Çizelge 5.5'te her iki algoritma içinde %Hata sütunu incelendiğinde ABC algoritması 4 problem için bilinen en iyi çözümü elde ederken, ABC-OX1 hibrit yöntemi ise 8 problem için bilinen en iyi çözümü yakalamıştır. ABC-OX1 yönteminin toplu veri kümelerini çözmede gösterdiği performans oldukça başarılıdır. Sadece C104 örnek için bilinen en iyi değerden %0.8 oranında sapma olduğu gözlemlenmiştir. Genel olarak ABC-OX1 algoritması tip-1 C serisi problemlerde literatürde neredeyse bilinen en iyi mesafe sonuçlarını elde etmiştir. Araç sayısı öncelikli amaç fonksiyonumuzu oluşturmasa da yine de 9 örnekte bilinen en iyi araç sayısı yakalanmıştır. ABC algoritması ise C102 ve C103 için en iyi araç sayısını elde edememiştir. %Hata sütunu açısından ABC-OX1 yöntemi 9 problem örneğinden 5'i için ABC yöntemine göre daha iyi sonuca ulaşmıştır. ABC-OX1 hibrit yönteminin ABC'ye göre üstünlüğü ortalama ve en iyi değer sütununda görülmektedir. Çizelge 5.5 ve 5.6'da ifade edilen ortalama değer sütunundaki tüm değerler ABC-OX1 algoritmasının ABC'ye göre daha başarılı sonuçlar elde ettiğini göstermiştir. Karşılaştırma testleri sırasında ABC-OX1 algoritması C101, C105 ve C106 örneklerinin çözümünde 10 kez bilinen en iyi değeri elde etmiştir.

Çizelge 5.6'da gösterildiği üzere %Hata sütununda ABC algoritması 2 örnek için bilinen en iyi çözümü elde ederken, ABC-OX1 algoritması ise 8 örnekten 6 tanesi için bilinen en iyi çözümü elde etmiştir. C202 ve C203 örneklerinde ise ABX-OX1 algoritmasının hata sapma yüzdeleri çok küçük kalmıştır. Tip-2 C serisi problemlerde ABC-OX1 için bilinen en iyi araç sayısı elde edilmiştir. ABC algoritması ise sadece C202 için en iyi araç sayısını elde edememiştir. ABC-OX1 hibrit modelin ABC'ye göre üstünlüğü ortalama ve en iyi değer sütununda da görülebilmektedir. Çizelge 5.6'da ABC-OX1'in ortalama değer sütunundaki tüm değerleri ABC yönteminin aynı sütunundaki değerlerine göre daha başarılıdır. ABC-OX1 algoritması C201, C205 ve C207 örnekleri için 10 kez çalıştırılmış her defasında en iyi değeri elde etmiştir. Bu nedenle ortalama değerleri bilinen en iyi değerlerle aynıdır. Diğer kalan veri kümelerinde de küçük sapma oranları ile yakın değerler elde edilmiştir. Çizelge 5.5 ve 5.6'da ortalama değer sütunu tüm çalıştırmaların ortalamasını ifade etmektedir. Ortalama değer sütunu bilinen en iyi değer sütununa ne kadar çok yakın ise algoritmanın da o kadar kararlı çalıştığını söyleyebiliriz.

Çizelge 5.5. Solomon tip-1 C modeli için ABC ve ABC-OX1 sonuçları

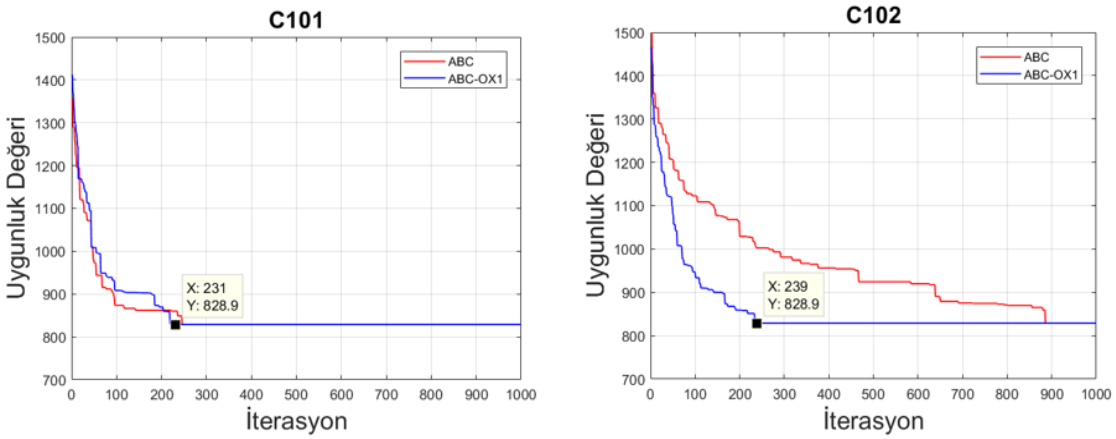
Solomon Örnekleri	BKS		ABC						ABC-OX1					
	Bilinen En İyi Değer		En İyi Değer		Ortalama Değer		Std. Sap.	% Hata	En İyi Değer		Ortalama Değer		Std. Sap.	% Hata
	Araç	Mesafe	Araç	Mesafe	Araç	Mesafe			Araç	Mesafe	Araç	Mesafe		
C101	10	828.94	10	**828.94**	10.6	849.61	31.90	**0%**	10	**828.94**	10.0	**828.94**	**0**	**0%**
C102	10	828.94	11	891.80	12.2	944.93	44.24	7.58%	10	**828.94**	10.7	869.45	42.6	**0%**
C103	10	828.94	11	904.95	12.1	995.40	66.53	9.29%	10	**828.06**	10.6	878.49	43.6	**0%**
C104	10	828.94	10	862.06	10.6	907.77	35.34	4.52%	10	831.46	10.1	861.29	19.7	0.8%
C105	10	828.94	10	**828.94**	10.9	872.01	37.25	**0%**	10	**828.94**	10.0	**828.94**	**0**	**0%**
C106	10	828.94	10	**828.94**	11.0	871.52	47.62	**0%**	10	**828.94**	10.0	**828.94**	**0**	**0%**
C107	10	828.94	10	**828.94**	11.0	895.97	47.27	**0%**	10	**828.94**	10.1	835.27	20.0	**0%**
C108	10	828.94	10	829.29	10.9	880.60	38.79	0.04%	10	**828.94**	10.0	831.05	6.7	**0%**
C109	10	828.94	10	833.59	11.0	910.22	29.69	0.56%	10	**828.94**	10.1	835.36	13.7	**0%**

Çizelge 5.6. Solomon tip-2 C modeli için ABC ve ABC-OX1 sonuçları

Solomon Örnekleri	BKS		ABC						ABC-OX1					
	Bilinen En İyi Değer		En İyi Değer		Ortalama Değer		Std. Sap.	% Hata	En İyi Değer		Ortalama Değer		Std. Sap.	% Hata
	Araç	Mesafe	Araç	Mesafe	Araç	Mesafe			Araç	Mesafe	Araç	Mesafe		
C201	3	591.56	3	**591.56**	3.7	618.90	34.37	**0%**	3	**591.56**	3.0	**591.56**	**0**	**0%**
C202	3	591.56	4	637.38	4.5	675.48	28.14	7.75%	3	**591.56**	3.1	595.87	13.6	**0%**
C203	3	591.17	3	620.48	4.1	665.46	33.25	4.96%	3	600.54	3.4	615.19	17.91	1.5%
C204	3	590.60	3	603.02	3.5	632.20	15.40	2.10%	3	596.89	3.1	606.34	14.73	1.0%
C205	3	588.88	3	**588.88**	3.6	618.98	29.63	**0%**	3	**588.88**	3.0	**588.88**	**0**	**0%**
C206	3	588.49	3	591.72	3.7	621.76	28.14	0.55%	3	**588.49**	3.1	591.27	8.52	**0%**
C207	3	588.29	3	589.94	3.7	624.83	28.26	0.28%	3	**588.29**	3.0	**588.29**	**0**	**0%**
C208	3	588.32	3	588.88	3.3	609.88	18.40	0.09%	3	**588.32**	3.0	588.67	1.08	**0%**

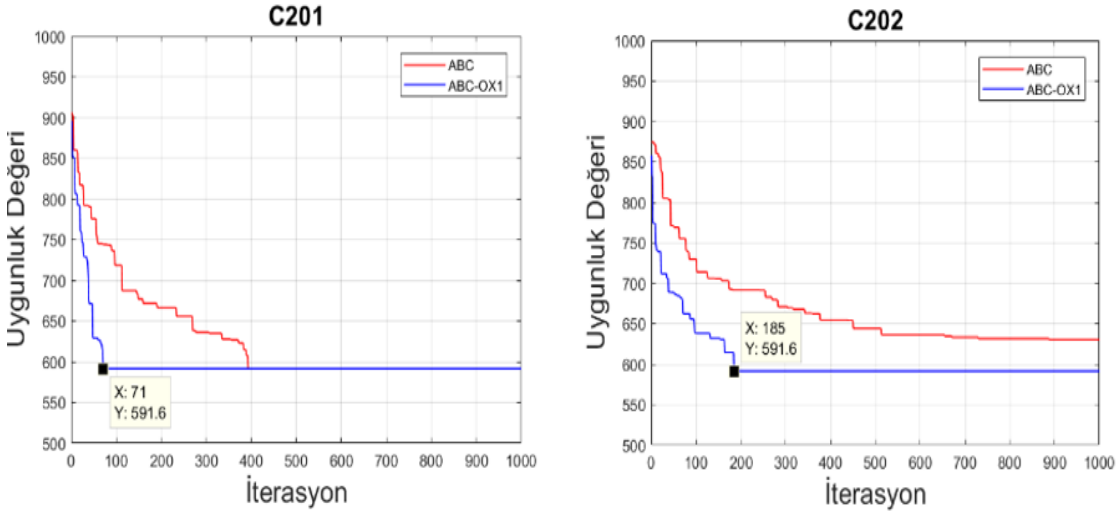
Şekil 5.4'teki C101 ve C102 örneklerine ait yakınsama grafiklerinde kırmızı çizgi ABC algoritmasını, mavi çizgi ise ABC-OX1 algoritmasını temsil etmektedir. Deneysel çalışmalarda kullanılan diğer yakınsama grafiklerinde de çizgiler aynı renkler ile gösterilmiştir. Şekil 5.4'teki C101 grafiği incelendiğinde her iki algoritmanın da yakınsama performanslarının birbirlerine yakın olduğu görülmektedir. C101 örneğinde iteratif aramaya başlanırken uygunluk değeri ABC algoritması için 1356.5 birim iken ABC-OX1 1413.1 birim olarak görülmüştür. C101 örneği 1000 iterasyon sayısı için çalıştırıldığında ABC 246. ve ABC-OX1 ise 231. iterasyonda en iyi değeri elde etmiştir. Aynı şeklide C102 örneğinde uygunluk değerleri ABC için 1518.1 birim iken ABC-OX1 1466.2 birim ile iteratif aramaya başlamışlardır. ABC 886. ve ABC-OX1 239. iterasyonda en iyi değere ulaşmıştır. Böylece her iki problemde de algoritmalar bilinen en iyi değeri

elde ederken ABC-OX1 algoritmasının ABC'ye göre daha erken yakınsadığı Şekil 5.4'te görülmüştür.



Şekil 5.4. ABC ve ABC-OX1 için C101 ve C102 yakınsama eğrisi

C201 ve C202 örneklerine ait yakınsama grafikleri Şekil 5.5'te verilmiştir. Bu grafikler incelendiğinde her iki örnekte de ABC-OX1 algoritmasının daha erken yakınsadığı görülmüştür. C201 örneğinde ABC algoritması aramaya başlarken 906 birim uygunluk değerine sahiptir. ABC-OX1 ise 895.5 birim uygunluk değerine sahiptir. C201 örneği için ABC-OX1 algoritması 71. ABC ise 393. iterasyonda en iyi değere ulaşmıştır. C202 örneğinde ABC algoritması aramaya başlarken 874.9 birim uygunluk değerine sahiptir. ABC-OX1 ise 858.3 birim uygunluk değerine sahiptir. ABC-OX1 185. iterasyonda en iyi değere ulaşırken ABC algoritması 888. iterasyonda 630.3 birimde kalarak bilinen en iyi değere ulaşamamıştır. Böylece C201 örneğinde her iki algoritmada bilinen en iyi değeri elde etmiştir. Fakat C202 örneğinde ABC-OX1 yöntemimiz bilinen en iyi değeri 71. iterasyonda elde ederken, ABC algoritması bilinen en iyi değere ulaşamamıştır. ABC-OX1 hibrit algoritmasının kararlı çalışması bu örnekte de görülmüştür. Şekil 5.4 ve 5.5'ten de görüldüğü üzere her iki algoritma da iteratif aramaya yaklaşık aynı uzaklık değeri ile başlamaktadır. Bu durum algoritmaların başlangıç popülasyonu oluşturma ve rota iyileştirme süreçlerinin aynı olmasından kaynaklanmaktadır.



Şekil 5.5. ABC ve ABC-OX1 için C201 ve C202 yakınsama eğrisi

Çizelge 5.7'de gösterildiği üzere tip-1 R serisi örneklerin çözümünde 25 araçla 100 müşteriye hizmet verilmesi hedeflenmektedir. R serisi veri kümesi toplamda 12 örnekten oluşmaktadır. Bu problemlerde zaman aralığı dar ve araç kapasiteleri küçüktür. ABC-OX1 algoritmasının yüzdelik hata sapma değeri %3 ile %14 arasında gerçekleşmiştir. Bu oran dağınık örnekler için makul kabul edilebilmektedir. Çünkü karmaşık problemlerin çözümü zor ve güçtür. ABC'de ise yüzdelik hata sapma değeri %8 ile %20 arasındadır. Bilinen en iyi çözüm değeri elde edilememiş olsa da ABC-OX1 hibrit yöntemi yaklaşık değerleri elde etmiştir. R108 örneği için bilinen en iyi araç sayısı yakalanmıştır.

Çizelge 5.8'de ise tip-2 R serisi veri kümesine ait çözümler görülmektedir. Bu veri kümesi 11 örnekten oluşmaktadır. Zaman penceresi geniş ve araç kapasitesi büyüktür. Müşteriler dağınık vaziyette konumlandırılmıştır. ABC-OX1 algoritmasının tip-2 R serisi örnekler için yüzdelik hata sapma değeri %4 ile %10 arasındadır. Karmaşık ve zor problemlerin çözümünde birçok olasılık değerlendirildiği için yaklaşık çözümler bulmak normal kabul edilmektedir. ABC'nin yüzdelik hata sapma değeri %6 ile %15 arasındadır. Çizelge 5.7 ve 5.8 birlikte incelendiğinde R serisi örneklerin tamamında ABC-OX1'in ABC'ye göre daha iyi sonuçlar ürettiği görülmektedir.

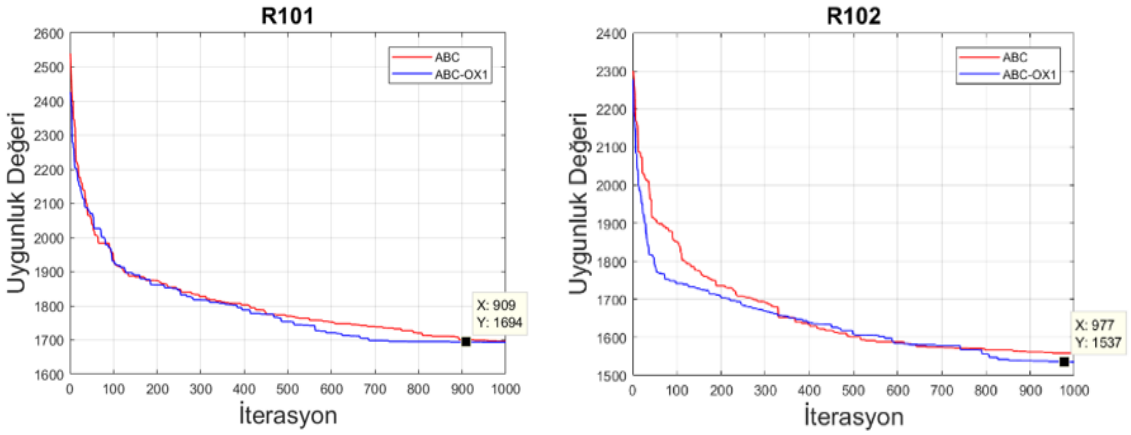
Çizelge 5.7. Solomon tip-1 R modeli için ABC ve ABC-OX1 sonuçları

Solomon Örnekleri	BKS		ABC						ABC-OX1					
	Bilinen En İyi Değer		En İyi Değer		Ortalama Değer		Std. Sap.	% Hata	En İyi Değer		Ortalama Değer		Std. Sap.	% Hata
	Araç	Mesafe	Araç	Mesafe	Araç	Mesafe			Araç	Mesafe	Araç	Mesafe		
R101	20	1642.87	21	1725.57	23.4	1750.29	16.86	%16.31	21	1701.52	22.9	1734.42	28.7	%3.57
R102	18	1472.62	19	1551.73	20.2	1607.58	30.83	%14.44	19	1524.47	19.6	1554.43	19.7	%3.52
R103	14	1213.62	16	1352.33	17.2	1374.50	14.60	%19.32	16	1293.71	16.6	1324.58	17.8	%6.60
R104	10	982.01	12	1081.12	13.0	1123.34	19.22	%11.65	12	1040.81	12.9	1086.57	23.7	%5.99
R105	15	1360.78	18	1457.48	19.2	1524.33	32.94	%15.44	17	1430.07	19.1	1499.20	30.2	%5.09
R106	13	1241.52	16	1393.42	17.3	1416.00	24.52	%15.95	15	1339.20	16.2	1371.70	27.0	%7.87
R107	11	1076.13	13	1183.20	14.5	1233.20	25.88	%12.48	13	1171.28	14.1	1198.24	18.9	%8.84
R108	10	948.573	11	1033.33	11.9	1061.39	21.44	%8.94	**10**	1006.88	11.4	1037.02	19.0	%6.15
R109	13	1151.84	15	1281.61	16.1	1305.77	17.69	%15.42	14	1211.17	14.7	1244.20	21.6	%5.15
R110	11	1080.36	14	1194.13	14.6	1239.04	27.58	%10.53	13	1153.55	13.6	1175.67	20.0	%6.77
R111	12	1053.5	14	1176.20	14.5	1223.03	25.63	%19.07	13	1146.45	13.8	1181.03	24.1	%8.82
R112	10	953.63	11	1090.08	11.8	1129.22	20.60	%14.31	11	1082.72	11.5	1098.26	10.9	%13.54

Çizelge 5.8. Solomon tip-2 R modeli için ABC ve ABC-OX1 sonuçları

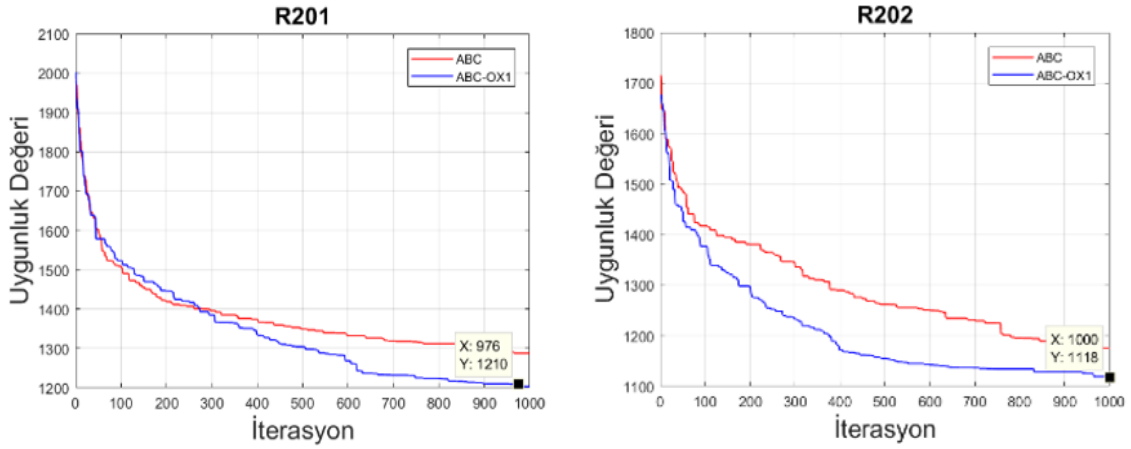
Solomon Örnekleri	BKS		ABC						ABC-OX1					
	Bilinen En İyi Değer		En İyi Değer		Ortalama Değer		Std. Sap.	% Hata	En İyi Değer		Ortalama Değer		Std. Sap.	% Hata
	Araç	Mesafe	Araç	Mesafe	Araç	Mesafe			Araç	Mesafe	Araç	Mesafe		
R201	9	1147.8	12	1262.45	13.9	1322.52	40.38	%9.92	11	1200.27	12.1	1245.24	19.9	%4.57
R202	5	1039.32	10	1162.87	10.9	1200.92	27.53	%10.78	9	1112.62	10.2	1149.74	28.7	%7.05
R203	5	874.87	7	1004.81	8.6	1039.55	22.92	%11.64	7	940.58	7.7	971.55	24.7	%7.51
R204	3	735.80	4	827.63	5.4	847.78	15.68	%7.16	5	776.57	5.4	808.40	20.4	%5.54
R205	5	954.16	9	1090.45	9.9	1121.25	27.44	%13.62	8	1024.34	9.0	1059.75	22	%7.36
R206	4	884.25	7	981.93	8.7	1050.82	25.15	%9.24	7	970.44	7.4	985.65	14.3	%9.75
R207	4	797.99	6	935.39	6.6	961.23	14.06	%14.80	5	860.42	5.8	891.19	23.8	%7.82
R208	3	705.62	**3**	762.94	4.0	787.94	12.96	%6.65	4	736.70	4.2	757.07	11.3	%4.40
R209	5	860.11	7	975.93	8.5	1003.34	20.54	%10.96	6	901.73	7.4	945.04	24.7	%4.84
R210	5	910.98	9	1016.67	9.3	1051.89	17.13	%8.98	7	953.54	7.8	986.88	20.4	%4.67
R211	4	755.82	5	828.36	6.3	853.59	13.41	%8.84	6	807.49	6.2	829.87	19.8	%6.84

Şekil 5.6'daki yakınsama grafiği incelendiğinde her iki algoritmanın da benzer şekilde yakınsadıkları görülmektedir. R101 örneğinde aramaya başlanırken ABC algoritması için uygunluk değeri 2539.7 iken ABC-OX1 için 2426.7 birimdir. R101 veri kümesi için her iki algoritma da bilinen en iyi değere ulaşamamıştır. Benzer şekilde R102 örneğinde aramaya başlanırken ABC algoritması için uygunluk değeri 2300.3 iken ABC-OX1 için 2277.6 birimdir. ABC-OX1 yönteminin R102 örneği için 800. iterasyondan sonra belirgin bir şeklide ABC'ye göre daha hızlı yakınsadığı görülmektedir.



Şekil 5.6. ABC ve ABC-OX1 için R101 ve R102 yakınsama eğrisi

Şekil 5.7'deki yakınsama grafikleri incelendiğinde ABC-OX1 algoritmasının ABC algoritmasına göre çok daha erken yakınsadığı görülmektedir. R201 örneğinde aramaya başlanırken ABC algoritması için uygunluk değeri 1988.8 iken ABC-OX1 için 2002.9 birimdir. R201 ve R202 örnekleri için her iki algoritma da bilinen en iyi değere ulaşamamıştır. Benzer şekilde R202 örneğinde aramaya başlanırken ABC için uygunluk değeri 1716.4 iken ABC-OX1 için 1676.1 birimdir. R201 örneğinde ABC-OX1 yöntemi 300. iterasyondan sonra ABC'ye göre daha hızlı yakınsamıştır. R202 örneğinde ise ABC-OX1 ilk aramaya başladığı andan itibaren ABC algoritmasına göre çok daha başarılı bir yakınsama göstermiştir.



Şekil 5.7. ABC ve ABC-OX1 için R201 ve R202 yakınsama eğrisi

Çizelge 5.9'da tip-1 RC örnekleri için elde edilen sonuçlar görülmektedir. Bu veri kümesi toplamda 8 örnekten oluşmaktadır. Bu problemlerde zaman aralığı sıkı, araç kapasiteleri küçük ve müşterilerin hizmet alma süreleri kısadır. ABC-OX1 algoritmasının yüzdelik hata sapma değeri %4 ile %10 arasında gerçekleşmiştir. Bu sonuçlar dağınık veri kümeleri için daha önce de ifade edildiği gibi makul kabul edilmektedir. ABC algoritmasının yüzdelik hata sapma değeri ise %9 ile %24 arasında gerçekleşmiştir. ABC-

OX1 ve ABC algoritmaları RC örneklerinde bilinen en iyi değeri elde edememişlerdir. Fakat ABC-OX1 hibrit yöntemi ABC algoritmasına göre yaklaşık değerler elde etmiştir. RC105 veri kümesi, bilinen en iyi değere göre %4 hata sapma oranı ile en çok yaklaşılan problem olmuştur.

Çizelge 5.10'da ise tip-2 RC örnekleri için elde edilen sonuçlar görülmektedir. Bu veri kümesi toplamda 8 örnekten oluşmaktadır. Zaman penceresi geniş, araç kapasitesi büyük ve müşterilerin hizmet alma süreleri kısadır. Müşteriler hem kümelenmiş hem de rassal olarak dağınık vaziyette konumlanmıştır. %Hata sütunu incelendiğinde ABX-OX1 algoritmasının ABC yöntemine göre daha başarılı sonuçlar elde ettiği görülmektedir. Tip-2 RC örnekleri için yüzdelik hata sapma değeri ABC-OX1 için %4 ile %11 arasındadır. ABC algoritması için ise yüzdelik hata sapma değeri %6 ile %16 arasındadır. ABC-OX1 hibrit yöntemi RC206 ve RC207 örnekleri için en iyi araç sayısını elde etmiştir. Çizelge 5.9 ve 5.10 birlikte incelendiğinde RC serisi örneklerin tamamında ABC-OX1'in ABC'ye göre daha iyi sonuçlar ürettiği görülmektedir.

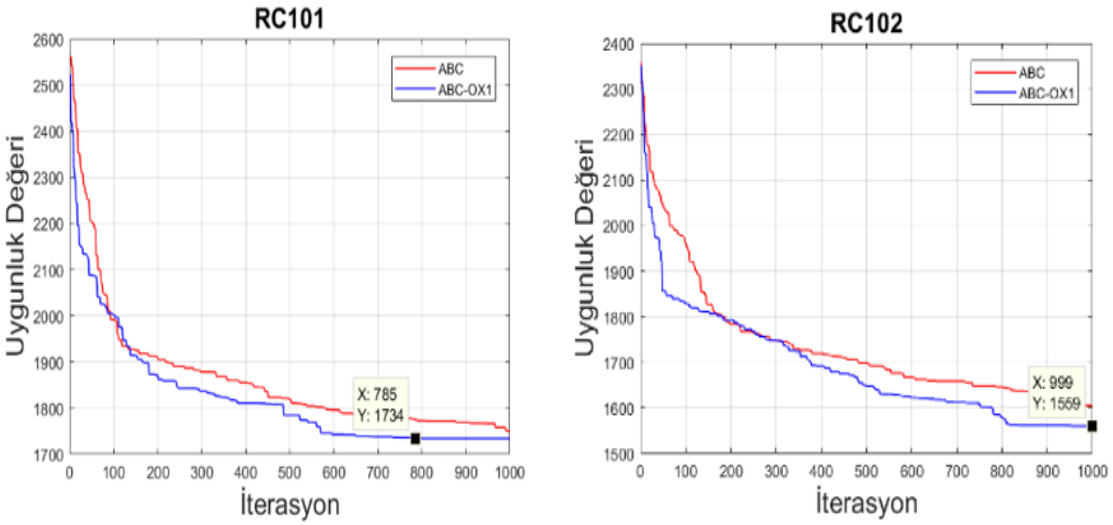
Çizelge 5.9. Solomon tip-1 RC modeli için ABC ve ABC-OX1 sonuçları

Solomon Örnekleri	BKS		ABC						ABC-OX1					
	Bilinen En İyi Değer		En İyi Değer		Ortalama Değer		Std. Sap.	% Hata	En İyi Değer		Ortalama Değer		Std. Sap.	% Hata
	Araç	Mesafe	Araç	Mesafe	Araç	Mesafe			Araç	Mesafe	Araç	Mesafe		
RC101	15	1623.58	18	1750.52	20.0	1821.41	32.14	%18.18	18	1724.72	19.6	1773.45	27.0	%6.23
RC102	14	1466.84	16	1591.62	17.7	1655.43	49.70	%14.07	16	1550.18	17.3	1605.20	37.1	%5.68
RC103	11	1261.67	13	1386.68	14.5	1484.51	39.88	%13.52	13	1377.44	13.8	1416.90	24.7	%9.18
RC104	10	1135.48	11	1243.76	12.1	1260.62	16.26	%9.54	11	1191.21	11.7	1233.00	21.4	%4.91
RC105	16	1518.60	17	1673.47	18.5	1715.30	25.74	%23.58	17	1584.37	18.0	1660.90	46.8	%4.33
RC106	13	1377.35	15	1509.88	15.8	1539.78	23.40	%23.09	15	1474.11	15.4	1508.46	26.5	%7.02
RC107	12	1212.83	13	1330.80	13.7	1368.65	17.85	%15.62	13	1300.24	13.4	1330.83	23.2	%7.21
RC108	11	1117.52	12	1207.52	12.7	1248.46	26.34	%12.14	12	1175.80	12.3	1206.85	25.2	%5.21

Çizelge 5.10. Solomon tip-2 RC modeli için ABC ve ABC-OX1 sonuçları

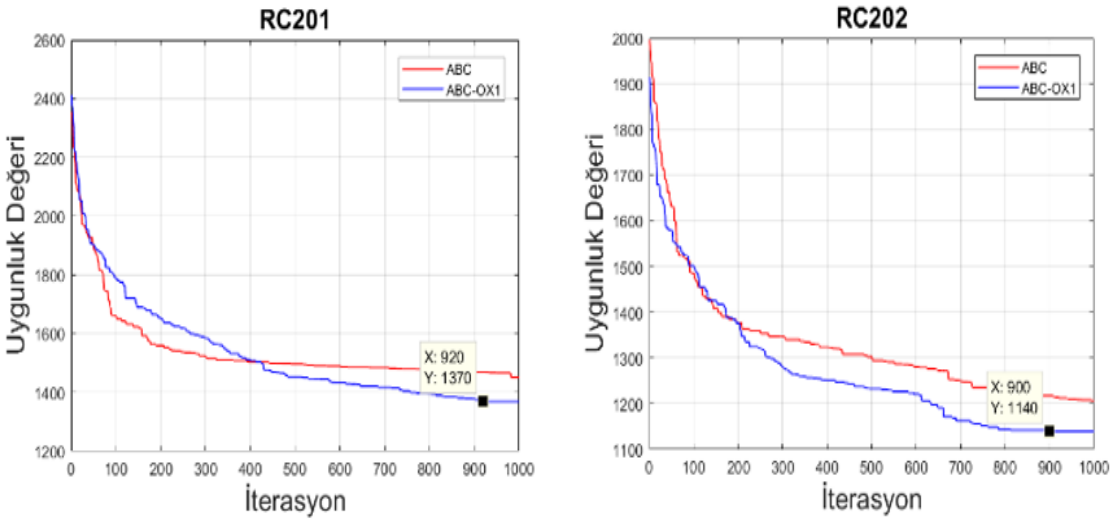
Solomon Örnekleri	BKS		ABC						ABC-OX1					
	Bilinen En İyi Değer		En İyi Değer		Ortalama Değer		Std. Sap.	% Hata	En İyi Değer		Ortalama Değer		Std. Sap.	% Hata
	Araç	Mesafe	Araç	Mesafe	Araç	Mesafe			Araç	Mesafe	Araç	Mesafe		
RC201	9	1266.11	13	1439.8	13.7	1509.6	51.0	%13.7	11	1401.63	12.3	1429.9	22.50	%10.7
RC202	8	1096.75	10	1241.93	11.2	1303.72	37.26	%13.24	9	1157.56	9.6	1207.52	23.40	%5.54
RC203	5	926.89	7	1038.07	8.0	1111.59	35.82	%12.00	6	982.02	7.0	1024.93	29.73	%5.95
RC204	4	786.38	5	875.74	5.6	908.96	20.12	%11.36	5	845.11	5.1	864.33	12.56	%7.47
RC205	7	1157.55	10	1304.82	11.1	1349.13	32.77	%12.72	8	1208.68	9.8	1244.34	26.71	%4.42
RC206	7	1056.21	9	1216.22	10.2	1284.73	48.52	%15.15	**7**	1122.93	8.3	1157.05	35.45	%6.32
RC207	7	966.08	8	1103.61	9.2	1166.62	32.63	%14.24	**7**	1041.47	8.2	1079.25	20.06	%7.80
RC208	4	779.84	5	828.18	5.9	870.87	24.63	%6.20	5	821.58	5.8	839.45	10.46	%5.35

Şekil 5.8 yakınsama grafikleri incelendiğinde ABC-OX1 hibrit yönteminin ABC algoritmasına göre daha erken yakınsadığı görülmektedir. RC101 örneğinde aramaya başlanırken ABC algoritması için uygunluk değeri 2561 birim iken ABC-OX1 algoritması için 2523 birimdir. RC101 örneği için her iki algoritma da bilinen en iyi değere ulaşamamıştır. Benzer şekilde RC102 örneğinde aramaya başlanırken ABC algoritması için uygunluk değeri 2359 iken ABC-OX1 için 2350 birimdir. ABC-OX1 yönteminin RC101 örneği için 130. iterasyondan sonra ve RC102 örneği içinse 380. iterasyondan sonra belirgin bir şeklide ABC algoritmasına göre daha hızlı yakınsadığı görülmektedir.



Şekil 5.8. ABC ve ABC-OX1 için RC101 ve RC102 yakınsama eğrisi

Şekil 5.9'daki yakınsama grafiklerinden gözlemlendiği üzere ABC-OX1 hibrit yöntemi ABC algoritmasına göre daha erken yakınsamaktadır. RC201 örneği için aramaya başlanırken ABC algoritması için uygunluk değeri 2353 birim iken ABC-OX1 algoritması içinse 2412 birimdir. RC201 örneği için her iki algoritmada bilinen en iyi değere ulaşamamıştır. Benzer şekilde RC202 örneğinde aramaya başlanırken ABC algoritması için uygunluk değeri 1917 iken ABC-OX1 için 1995 birimdir. ABC-OX1 yönteminin RC201 veri kümesi için 420. iterasyondan sonra RC202 veri kümesi içinse 200. iterasyondan sonra belirgin bir şeklide ABC algoritmasına göre daha hızlı yakınsadığı görülmüştür.



Şekil 5.9. ABC ve ABC-OX1 için RC201 ve RC202 yakınsama eğrisi

6. SONUÇLAR VE ÖNERİLER

6.1. Sonuçlar

Bu çalışmada ZPARP'nin çözümü için ABC algoritması ile order crossover (OX1) operatörü melezleştirilmiş ve ABC-OX1 olarak adlandırılan hibrit bir yöntem önerilmiştir. ABC-OX1 yönteminde başlangıç çözümleri rastgele olarak üretilmiş ve bu çözümler tur geliştirici 3-opt algoritması ile iyileştirilmiştir. İşçi arı fazında insertion, swap ve divideandswap yerel arama operatörleri eşit olasılıkla kullanılmıştır. Gözcü arı fazında OX1 operatörü ile besin çeşitliliği sağlanmış ve kaliteli çözümler üzerinde aramaya devam edilmiştir. Kâşif arı fazında ise yeni çözümler rastgele olarak oluşturulmuştur. Daha sonra bu çözümler tur geliştirici 2-opt algoritması ile iyileştirilmiştir. ABC-OX1 yönteminde popülasyondaki çözümlerin uygunluk değerlendirmesi için Bellman algoritması kullanılmıştır. ABC-OX1, Solomon'un C, R ve RC veri kümelerini oluşturan 56 örnek üzerinde test edilmiştir. ABC-OX1, 56 örnekten 14'ü için BKS'ye ulaşmıştır. ABC ile ABC-OX1 algoritmaları aynı şartlarda birtakım deneyler yapılarak karşılaştırılmıştır. Hata paylarını minimuma indirmek için algoritmalar her bir örnek üzerinde 10'ar kez koşturulmuştur. Elde edilen sonuçlara göre yeni geliştirilen hibrit yöntemin ABC algoritmasına göre daha hızlı yakınsadığı görülmüştür. Elde edilen en iyi ve ortalama değerler incelendiğinde tüm örnekler üzerinde ABC-OX1 hibrit yöntemi ABC'ye göre daha iyi sonuçlar üretmiştir.

6.2. Öneriler

Önerilen ABC-OX1 hibrit yönteminin başarısını arttırmak için üzerinde birtakım iyileştirmeler yapılabilir. Bunlardan ilki ve en önemlisi başlangıç çözümlerinin daha da iyileştirilmesidir. Bunun için Solomon'un rota kurucu tasarruf, ekleme ve en yakın komşu algoritmaları kullanılabilir. İkinci iyileştirme ise tur geliştirme aşamasında yapılabilir. Bunun için bu aşamada 2-opt*, Or-opt ve ruin-recreate gibi algoritmalar kullanılabilir. Üçüncü iyileştirme ise uygunluk değerlendirmesi aşamasında yapılabilir. Bunun için Bellman algoritması yerine farklı bir algoritma kullanılabilir.

KAYNAKLAR

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

[2] Dantzig, G.B. and Ramser, J.H., 1959, The truck dispatching problem, *Management Science,* 6, 80-91.

[3] Şahin, Y. ve Eroğlu, A., 2014, Kapasite kısıtlı araç rotalama problemi için meta-sezgisel yöntemler: Bilimsel yazın taraması. *Süleyman Demirel Üniversitesi, İktisadi ve İdari Bilimler Fakültesi Dergisi*, Isparta, 19 (4), 337-355.

[4] Gökçen, T. ve ark., 2018, Zaman pencereli araç rotalama probleminin çözümü için çok amaçlı genetik algoritma yaklaşımı. *Gazi Üniversitesi, Fen Bilimleri Dergisi*, Ankara, 6 (4), 774-786.

[5] Kuram, Ç., 2016, Zaman pencereli araç rotalama probleminin Popülasyon tabanlı sezgisel yöntemler ile optimize edilmesi, Yüksek lisans tezi, *İstanbul Üniversitesi, Fen Bilimleri Enstitüsü,* İstanbul, 16-23.

[6] Güngör G. ve Ergülen A.**,** 2006, Bulanık araç rotalama problemlerine bir model önerisi ve bir uygulama, Yönetim ve Ekonomi, *Celal Bayar Üniversitesi, İktisadi ve İdari Bilimler Fakültesi Dergisi,* Manisa, 13 (1), 1-8.

[7] Tezer, T., 2009, Toplama ve dağıtım zaman pencereli araç rotalama problemi için kesin çözüm yaklaşımı ve örnek uygulamalar. Yüksek Lisans Tezi, *Balıkesir Üniversitesi, Fen Bilimleri Enstitüsü*, Balıkesir 17-27.

[8] Meng, F., Ding, Y. Li., W. and Guo, R., 2019, Customer-Oriented Vehicle Routing Problem with Environment Consideration: Two-Phase Optimization Approach and Heuristic Solution, *Mathematical Problems in Engineering*, [online], https://doi.org/10.1155/2019/1073609 [Ziyaret Tarihi: 5 Şubat 2021].

[9] Toth, P. and Vigo, D., 2002, The Vehicle Routing Problem. *Philadelphia: Society for Industrial and Applied Mathematics.* [online], https://epubs.siam.org/doi/pdf/10.1137/1.9780898718515.fm [Ziyaret Tarihi: 9 Şubat 2021].

[10] Koç, Ç. ve Karaoğlan, İ., 2014, Zaman bağımlı araç rotalama problemi için bir matematiksel model. *Gazi Üniversitesi, Mühendislik Mimarlık Fakültesi Dergisi,* Ankara, 29 (3), 549-558.

[11] Şeker, Ş., 2007, Araç rotalama problemleri ve zaman pencereli stokastik araç rotalama problemine genetik algoritma yaklaşımı, Yüksek Lisans Tezi, *Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü*, İstanbul 26-39.

[12] Bowerman, R. et al, 1995, A multi-objective optimization approach to urban school bus routing: Formulation and solution method, *Transportation Research Part A: Policy and Practice*, 29 (2), 107-123.

[13] Keskintürk, T. ve ark., 2015, Araç rotalama problemleri ile çözüm yöntemlerinin sınıflandırılması ve bir uygulama, *İstanbul Üniversitesi, Fen İşletme Bilimi Dergisi,* İstanbul, 3 (2), 77-107. [online], https://dergipark.org.tr/tr/download/ article-file/213329 [Ziyaret Tarihi: 9 Ağustos 2020].

[14] Laporte, G., 1992, The vehicle routing problem: An overview of exact and approximate algorithms, *European Journal of Operational Research*, 59 (3), 345–358.

[15] Karimi, H. and Seifi, A., 2012, Acceleration of Lagrangian Method for the Vehicle Routing Problem with Time Windows. *International Journal of Industrial Engineering, Production Research,* 309-315.

[16] İnkçi, M., 2019, Zaman pencereli araç rotalama problemine uygulanan meta-sezgisel çözüm önerilerinin karşılaştırılması, Yüksek lisans tezi, *Çukurova Üniversitesi, Sosyal Bilimleri Enstitüsü,* Adana, 5-7.

[17] Kılıç, S., 2008, Bulanık karar ortamında karınca kolonisi optimizasyonu yöntemiyle araç rotalama, Doktora Tezi, *İstanbul Teknik Üniversitesi, Fen Bilimleri Enstitüsü*, İstanbul 6-35.

[18] Fisher, M.L. et al, 1997, Vehicle Routing with Time Windows: Two Optimization Algorithms, *Operations Research*, 45 (3), 488-492.

[19] Kohl, N., and Madsen, B.G., 1997, An Optimization Algorithm for the Vehicle Routing Problem with Time Windows Based on Lagrangian Relaxation, *Operations Research*, 45 (3), 395-406

[20] Kallehaugea, B. et al, 2006, Madsena lagrangian duality applied to the vehicle routing problem with time windows, *Operations Research*, 33, 1464–1487.

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

[22] Desrosiers, J., Soumis F., Desrochers, M. and Sauvé, M., 1986, Methods for routing with time windows, *European Journal of Operational Research*, 23, 236-245.

[23] Baker, E. K., 1983, An exact algorithm for the time constrained traveling salesman problem, *OperationsResearch*, 31 (5), 938-945.

[24] Desrosiers, J., Soumis,F. and Desrochers, M., 1984, Routing with time Windows by column generation. *Networks,* 14 (4), 545-565.

[25] Kolen, A.W.J., Rinnooy Kan, A.H.G. and Trienekens, H.W.J.M., 1986, Vehicle routing with time windows, *Operations Research*, 35 (2), 266-273 [online], https://www.jstor.org/stable/170698?seq=1 [Ziyaret Tarihi: 9 Şubat 2021].

[26] Min, H., 1991, A multiobjective vehicle routing problem with soft time windows: the case of a public library distribution system, *Socio-Economic Planning Sciences*, 25 (3), 179-188.

[27] Desrochers, M., Desrosiers, J. and Solomon, M., 1992, A new optimization algorithm for the vehicle routing problem with time windows, *Operations Research*, 40 (2), 342-354.

[28] Çetin, S. ve Gencer, C., 2010, Kesin zaman pencereli – Eş zamanlı dağıtım toplamalı araç rotalama problemi: Matematiksel Model, *Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi*, 25 (3), 579-585.

[29] Demirtaş, Y.E., ve Özdemir, E., 2017, Dinamik araç rotalama problemleri için yeni bir çözüm önerisi, Süleyman Demirel Üniversitesi, *İktisadi ve İdari Bilimler Dergisi*, Isparta, 22, (3) 807-823.

[30] Kilby, P. Prosser, P., and Shaw, P., 1998, Dynamic VRPs: A Study of Scenarios, University of Strathclyde Technical Report, Australia, [online], https://www.researchgate.net/publication/2944001_Dynamic_VRPs_A_Study_of_Scenarios [Ziyaret Tarihi: 15 Haziran 2021].

[31] Montemanni, R., Gambardella, L.M., Rizzoli, A.E. and Donati, A.V., 2005, Ant colony system for a dynamic vehicle routing problem, *Journal of combinatorial optimization,* Switzerland

[32] Hanshar, F., T. and Ombuki-Berman B., 1998, Using Genetic Algorithms for Multi-depot Vehicle Routing, Canada, [online], https://link.springer.com/ article/10.1007/BF02098286 [Ziyaret Tarihi: 5 Temmuz 2021].

[33] Cordone, R. and Calvo, R.W., 2001, A heuristic for the vehicle routing problem with time windows, *Journal of Heuristics*, 7, 107-129. [online], https://www.academia.edu/9556143/ [Ziyaret Tarihi: 24 Mart 2021].

[34] Clarke, G. and Wright, J., W., 1964, Scheduling of vehicles from a depot to a number of delivery points, *Operations Research*, 12, 568–581.

[35] Önder, E., 2011, Araç rotalama problemlerinin parçacık sürü ve genetik algoritma ile optimizasyonu, Doktora tezi, *İstanbul Üniversitesi, Sosyal Bilimler Enstitüsü*, İstanbul, 46-66.

[36] Van Landeghem H.R.G., 1998, Abi-criteria heuristic for the vehicle routing problem with time windows, *European Journal of Operational Research*, North-Holland, 36, 217-226

[37] Braysy, O. and Gendreau, M., 2005, Vehicle routing problem with time windows, PartI: Route construction and local search algorithms. *TransportationScience*, 39 (1), 104-118.

[38] Croes, G.A., 1958, A method for solving traveling-salesman problems, *Operations Res*, 6 (6), 791–812. [online], https://www.jstor.org/stable/167074 [Ziyaret Tarihi: 8 mart 2021].

[39] Lin, S., 1965, Computer solutions of the traveling salesman problem, *Bell System Technical Journal,* 44 (10), 2245-2269.

[40] Keskintürk, T. ve ark., 2016, 2-opt algoritması ve başlangıç çözümünün algoritma sonuçları üzerindeki etkisi, *İstanbul Üniversitesi, İşletme fakültesi, Endüstri mühendisliği dergisi,* İstanbul, 27 (3), 2-12.

[41] Potvin, J. and Rousseau, J.M., 1995, An exchange heuristic for routeing problems with time windows, *Journal of Operational Research*, 46, 1433-1466.

[42] Taillard, E.D. and Badeau, P., 1997, A tabu search heuristic fort he vehicle routing problem with soft time windows. *Operations Research*, [online], https://pubsonline.informs.org/doi/abs/10.1287/trsc.31.2.170 [Ziyaret Tarihi: 19 Temmuz 2021].

[43] Nilsson, C., 2003, Heuristics for the traveling salesman problem, *Tech. Report*, Linköping University, Sweden.

[44] Eryavuz, M. ve Gencer, C., 2001, Araç rotalama problemine ait bir uygulama. *Süleyman Demirel Üniversitesi, İktisadi ve İdari Bilimler Fakültesi Dergisi*, Isparta, 6 (1), 139-155.

[45] Gillet, B. and Miller, L., 1974, A heuristic algorithm fort he vehicle dispactching problem, *Operations Research*, 22, 340-349.

[46] Fisher, M. L. And Jaikumar, R., 1981, A generalised assigment heuristic for vehicle routing, *networks*, 11, 109-124.

[47] Bent, R. and Hentenrryck, P. V., 2004, A Two-Stage Hybrid Local Search for the Vehicle Routing Problem with Time Windows, *Transportation Science*, 38 (4), 515-530.

[48] Fagerholt, K., 2001, Ship Scheduling with Soft Time Windows: An Optimisation Based Approach, *European Journal of Operational Research*, 131, 559-571.

[49] Aksakal, B., 2014, Bir firmanın zaman pencereli belirli talepli araç rotalama probleminin genetik algoritma kullanılarak çözülmesi, *Yüksek lisans tezi, İstanbul Üniversitesi Fen Bilimleri Enstitüsü*, İstanbul, 29-30.

[50] Breedam, A.V., 2002, A parametric analysis of heuristics for the vehicle routing problem with side-constraints, *European journal of operational research*, 137 (2), 348-370.

[51] Vansteenwegen, P., Mateo, M., 2014, An iterated local search algorithm for the single-vehicle cyclic inventory routing problem. *European Journal of Operational Research*, 237, 802–813.

[52] İlhan, İ., 2021, An improved simulated annealing algorithm with crossover operator for capacitated vehicle routing problem, *Science Direct*, 64, 1-8.

[53] Ağayeva, Ç. ve Takan, M. A., 2020, Stokastik talepli kapasite kısıtlı araç rotalama problemine yönelik karşılaştırmalı bir yaklaşım, *Bilecik Şeyh Edabalı Üniversitesi, Fen Bilimleri Dergisi*, Bilecik, 2, 971-979.

[54] Brandao, J., 2006, A new tabu search algorithm for the vehicle routing problem with Backhauls, *European Journal of Operational Research*, 173, 540–555.

[55] Shen, Y., Mingde, L., Jian, Y. and Yuhui, S., 2020, A Hybrid Swarm Intelligence Algorithm for Vehicle Routing Problem With Time Windows, *Digital Object Identifier* 10.1109/ACCESS.2020.2984660

[56] Gülsoy, N., 2013, Av arama algoritması ile sıkı zaman pencereli araç rotalama probleminin çözümü, Yüksek lisans tezi, *Erciyes Üniversitesi Fen Bilimleri Enstitüsü,* Kayseri 25-36.

[57] Dursun, P., 2009, Zaman pencereli araç rotalama problemi'nin genetik algoritma ile modellenmesi, *Yüksek Lisans Tezi, İstanbul Üniversitesi, Fen Bilimleri Enstitüsü,* İstanbul 50-62.

[58] Ursani Z., Essama, D., Cornforthb, D and Stocker R., 2011, Localized genetic algorithm for vehicle routing problem with time Windows, *Science Direct*, 11 (8), 5375-5390.

[59] Tan, W.F., L.S. Lee, Z.A. Majid and H.V. Seow, 2012, Ant Colony optimization for capacitated vehicle routing problem, *Journal of Computer Science* 8 (6), 846-852.

[60] Şeker, Ş.E., 2010, [online], https://bilgisayarkavramlari.com/2010/05/26/bellman-ford-algoritmasi/ [Ziyaret Tarihi: 8 Haziran 2021].

Linkler:

Solomon's benchmark problems

https://neo.lcc.uma.es/vrp/vrp-instances/capacitated-vrp-with-time-windows-instances/

http://web.cba.neu.edu/~msolomon/problems.htm

Diğer link erişimleri

https://epubs.siam.org/doi/pdf/ 10.1137/1.9780898718515.fm

https://dergipark.org.tr/tr/download/ article-file/213329

https://www.jstor.org/stable/170698?seq=1

https://www.academia.edu/9556143/

https://epubs.siam.org/doi/pdf/10.1137/1.9780898718515.fm

https://ieeexplore.ieee.org/stamp/stamp.jsp?tp

http://matejgazda.com/tsp-algorithms-2-opt-3-opt-in-python/

https://bilgisayarkavramlari.com/2010/05/26/bellman-ford-algoritmasi/

EKLER

C101 Problem verisi Araç sayısı: 25 adet Araç kapasitesi: 200 birim

Müşteri Numarası	X Koordinatı	Y Koordinatı	Talep Miktarı	Hazır O. Zamanı	Son Tarih	Servis Süresi
0	40	50	0	0	1236	0
1	45	68	10	912	967	90
2	45	70	30	825	870	90
3	42	66	10	65	146	90
4	42	68	10	727	782	90
5	42	65	10	15	67	90
6	40	69	20	621	702	90
7	40	66	20	170	225	90
8	38	68	20	255	324	90
9	38	70	10	534	605	90
10	35	66	10	357	410	90
11	35	69	10	448	505	90
12	25	85	20	652	721	90
13	22	75	30	30	92	90
14	22	85	10	567	620	90
15	20	80	40	384	429	90
16	20	85	40	475	528	90
17	18	75	20	99	148	90
18	15	75	20	179	254	90
19	15	80	10	278	345	90
20	30	50	10	10	73	90
21	30	52	20	914	965	90
22	28	52	20	812	883	90
23	28	55	10	732	777	90
24	25	50	10	65	144	90
25	25	52	40	169	224	90
26	25	55	10	622	701	90
27	23	52	10	261	316	90
28	23	55	20	546	593	90
29	20	50	10	358	405	90
30	20	55	10	449	504	90
31	10	35	20	200	237	90
32	10	40	30	31	100	90
33	8	40	40	87	158	90
34	8	45	20	751	816	90
35	5	35	10	283	344	90
36	5	45	10	665	716	90
37	2	40	20	383	434	90
38	0	40	30	479	522	90
39	0	45	20	567	624	90
40	35	30	10	264	321	90
41	35	32	10	166	235	90
42	33	32	20	68	149	90
43	33	35	10	16	80	90
44	32	30	10	359	412	90
45	30	30	10	541	600	90
46	30	32	30	448	509	90
47	30	35	10	1054	1127	90

48	28	30	10	632	693	90
49	28	35	10	1001	1066	90
50	26	32	10	815	880	90
51	25	30	10	725	786	90
52	25	35	10	912	969	90
53	44	5	20	286	347	90
54	42	10	40	186	257	90
55	42	15	10	95	158	90
56	40	5	30	385	436	90
57	40	15	40	35	87	90
58	38	5	30	471	534	90
59	38	15	10	651	740	90
60	35	5	20	562	629	90
61	50	30	10	531	610	90
62	50	35	20	262	317	90
63	50	40	50	171	218	90
64	48	30	10	632	693	90
65	48	40	10	76	129	90
66	47	35	10	826	875	90
67	47	40	10	12	77	90
68	45	30	10	734	777	90
69	45	35	10	916	969	90
70	95	30	30	387	456	90
71	95	35	20	293	360	90
72	53	30	10	450	505	90
73	92	30	10	478	551	90
74	53	35	50	353	412	90
75	45	65	20	997	1068	90
76	90	35	10	203	260	90
77	88	30	10	574	643	90
78	88	35	20	109	170	90
79	87	30	10	668	731	90
80	85	25	10	769	820	90
81	85	35	30	47	124	90
82	75	55	20	369	420	90
83	72	55	10	265	338	90
84	70	58	20	458	523	90
85	68	60	30	555	612	90
86	66	55	10	173	238	90
87	65	55	20	85	144	90
88	65	60	30	645	708	90
89	63	58	10	737	802	90
90	60	55	10	20	84	90
91	60	60	10	836	889	90
92	67	85	20	368	441	90
93	65	85	40	475	518	90
94	65	82	10	285	336	90
95	62	80	30	196	239	90
96	60	80	10	95	156	90
97	60	85	30	561	622	90
98	58	75	20	30	84	90
99	55	80	10	743	820	90
100	55	85	20	647	726	90