T.C.
ERCİYES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

AV ARAMA ALGORİTMASI İLE SIKI ZAMAN PENCERELİ ARAÇ ROTALAMA PROBLEMLERİNİN ÇÖZÜMÜ

Hazırlayan
Nihan GÜLSOY

Danışman
Yrd. Doç. Dr. Sinem KULLUK

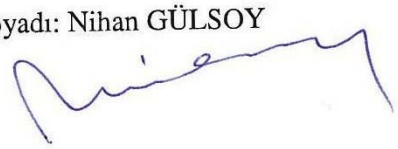
Yüksek Lisans Tezi

OCAK 2013
KAYSERİ

BİLİMSEL ETİĞE UYGUNLUK

Bu çalışmadaki tüm bilgilerin, akademik ve etik kurallara uygun bir şekilde elde edildiğini beyan ederim. Aynı zamanda bu kural ve davranışların gerektirdiği gibi, bu çalışmanın özünde olmayan tüm materyal ve sonuçları tam olarak aktardığımı ve referans gösterdiğimi belirtirim.

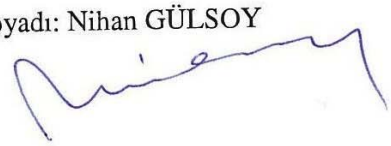
Adı-Soyadı: Nihan GÜLSOY

İmza : 

BİLİMSEL ETİĞE UYGUNLUK

Bu çalışmadaki tüm bilgilerin, akademik ve etik kurallara uygun bir şekilde elde edildiğini beyan ederim. Aynı zamanda bu kural ve davranışların gerektirdiği gibi, bu çalışmanın özünde olmayan tüm materyal ve sonuçları tam olarak aktardığımı ve referans gösterdiğimi belirtirim.

Adı-Soyadı: Nihan GÜLSOY

İmza : 

Yrd. Doç. Dr. Sinem KULLUK danışmanlığında **Nihan GÜLSOY** tarafından hazırlanan **"Av Arama Algoritması İle Sıkı Zaman Pencereli Araç Rotalama Problemlerinin Çözümü"** adlı bu çalışma, jürimiz tarafından Erciyes Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Anabilim Dalında **Yüksek Lisans** tezi olarak kabul edilmiştir.

.

16 / 01 / 2013

(Tez savunma sınav tarihi yazılacaktır.)

JÜRİ:

Danışman	:Yrd. Doç. Dr. Sinem KULLUK
Üye	: Doç. Dr. Lale ÖZBAKIR
Üye	: Yrd. Doç. Dr. Hülya BEHRET

ONAY:
Bu tezin kabulü Enstitü Yönetim Kurulunun 22/01/2013 tarih ve 2013/04-03 sayılı kararı ile onaylanmıştır.

22 / 01 / 2013

Prof. Dr. Necmettin MARAŞLI

Enstitü Müdürü

ÖNSÖZ / TEŞEKKÜR

Çalışma boyunca farklı bakış açıları ve bilimsel katkılarıyla beni aydınlatan, yakın ilgi ve yardımlarını esirgemeyen sayın hocam Yrd. Doç. Dr. Sinem KULLUK'a teşekkürü bir borç bilirim.

Bu zorlu süreçte daima yanımda olan arkadaşım Berrin UÇARKUŞ'a ve çalışmalarım süresince sabır göstererek beni daima destekleyen aileme en içten teşekkürlerimi sunarım.

Nihan GÜLSOY

Kayseri, Ocak 2013

AV ARAMA ALGORİTMASI İLE SIKI ZAMAN PENCERELİ ARAÇ ROTALAMA PROBLEMLERİNİN ÇÖZÜMÜ

Nihan GÜLSOY

Erciyes Üniversitesi, Fen Bilimleri Enstitüsü

Yüksek Lisans Tezi, Ocak 2013

Danışman: Yrd. Doç. Dr. Sinem KULLUK

KISA ÖZET

Av arama optimizasyon algoritması, hayvanların grup olarak avlanmasından esinlenilerek, sürekli optimizasyon problemleri için geliştirilmiş yeni bir algoritmadır. Bu çalışmada lojistik sisteminin temelini oluşturan araç rotalama problemlerinden sıkı zaman pencereli araç rotalama problemi av arama algoritması ile çözümlenmiştir. Bu amaçla, av arama algoritması rassal anahtarlama metodu ve ayırma algoritması ile SZPARP problemlerin çözümü için uygun hale getirilmiştir. Çalışma sonucunda geliştirilen SZPARP için av arama algoritması literatürde sıkça kullanılan Solomon'un problemleri için çözümlenmiştir. Sonuçlar incelendiğinde, küçük boyutlu problemlerde, diğer algoritmalarla rekabet edilebilecek düzeyde çözüm elde edildiği görülmüştür.

Anahtar Kelimeler: Av arama algoritması; sıkı zaman pencereli araç rotalama problemi; rassal anahtarlama metodu; ayırma algoritması.

THE SOLUTION VEHICLE ROUTING PROBLEM WITH HARD TIME WINDOWS WITH HUNTING SEARCH ALGORITHM

Nihan GÜLSOY

Erciyes University, Graduate School of Natural and Applied Sciences

M. Sc. Thesis, January 2013

Supervisor: Assist. Prof. Sinem KULLUK

ABSTRACT

Hunting search optimization algorithm is a novel algorithm developed for continuous optimization problems, inspired by group hunting behavior of animals. In this study, vehicle routing problem with hard time windows (VRPHTW) which underlies the logistic system is analyzed by hunting search algorithm. In order to do this, hunting search algorithm is modified using random-key procedure and splitting algorithm to make it compatible with vehicle routing problem with hard time windows. The developed hunting search algorithm for VRPHTW is analyzed on frequently used Solomon problems in literature. Obtained results showed that, proposed algorithm is competitive with other algorithms on small sized problems.

Keywords: Hunting search algorithm; vehicle routing problem with hard time windows; Rrndom-Key procedure; splitting algorithm.

İÇİNDEKİLER

AV ARAMA ALGORİTMASI İLE SIKI ZAMAN PENCERELİ ARAÇ ROTALAMA PROBLEMLERİNİN ÇÖZÜMÜ

BİLİMSEL ETİĞE UYGUNLUK i
YÖNERGEYE UYGUNLUK ii
KABUL VE ONAY SAYFASI iii
ÖNSÖZ / TEŞEKKÜR iv
KISA ÖZET v
ABSTRACT vi
İÇİNDEKİLER vii
KISALTMALAR x
TABLOLAR LİSTESİ xi

GİRİŞ 1

1. BÖLÜM

ARAÇ ROTALAMA PROBLEMİ

1.1. Araç Rotalama Problemlerinin Kısa Tarihçesi 4
1.2. Araç Rotalama Probleminin Bileşenleri 6
1.3. Standart Araç Rotalama Problemi Formülasyonu 7
1.4. Araç Rotalama Problemlerinin Sınıflandırması 9
1.4.1. Kapasite Kısıtlı ARP 9
1.4.2. Mesafe Kısıtlı ARP 10

1.4.3. Önce Dağıt Sonra Topla ARP 11
1.4.4. Eş Zamanlı Topla-Dağıt ARP 11
1.4.5. Bölünmüş Dağıtımlı ARP 12
1.4.6. Çok Depolu ARP 12
1.4.7. Periyodik ARP 12
1.4.8. Zaman Pencereli ARP 13
1.5. Araç Rotalama Problemlerine Çözüm Yöntemleri 13
1.5.1. Kesin Yöntemler 14
1.5.2. Sezgisel Yöntemler 14
1.5.3. Meta-Sezgisel Yöntemler 16

2. BÖLÜM

AV ARAMA ALGORTİMASI

2.1. Av Arama Algoritmasının İşleyişi 24

3. BÖLÜM

ZAMAN PENCERELİ ARAÇ ROTALAMA PROBLEMİ

3.1. Sıkı Zaman Pencereli Araç Rotalama Problemi Matematiksel Formülasyonu 33
3.2. ZPARP Literatürü 34
3.2.1. Deterministik – Tam Çözüm Algoritmaları 34
3.2.2. Yerel Arama Sezgiselleri 35
3.2.3. Meta-Sezgiseller 35

4. BÖLÜM

AV ARAMA ALGORİTMASI İLE SZPARP'NİN ÇÖZÜMÜ

4.1. Problem Tanımı 39
4.2. Dizi Gösterimi 40
4.2.1. Prins'in Ayırma Algoritması 41
4.2.2. SZPARP İçin Uyarlama 43

4.3. SZPARP için Önerilen AA Algoritmasının Temel İşleyişi..... 44
4.4. Deneysel Çalışmalar..... 47
4.4.1. Sayısal Sonuçlar..... 48

5. BÖLÜM

TARTIŞMA-SONUÇ VE ÖNERİLER

5.1. Tartışma-Sonuç ve Öneriler..... 51

KAYNAKÇA 53
ÖZGEÇMİŞ..... 61

KISALTMALAR

AA	Av Arama
AG	Avcı Grubu
AGB	Avlanma Grubu Boyutu
AGDO	Avlanma Grubu Değerlendirme Oranı
ARP	Araç Rotalama Problemi
DG	Diferansiyel Gelişim
EIS	Epokta İterasyon Sayısı
ES	Epok Sayısı
GA	Genetik Algoritma
GSP	Gezgin Satıcı Problemi
HA	Harmoni Arama
KKO	Karınca Koloni Optimizasyonu
LMH	Lider Arkasından Maksimum Hareket
OCGA	Optimize Çaprazlama Operatörü
PSO	Parçacık Sürü Optimizasyonu
Ra	Uzaklık Yarıçapı
SZPARP	Sıkı Zaman Pencereli Araç Rotalama Problemi
TA	Tabu Arama
TB	Tavlama Benzetimi
YAK	Yapay Arı Kolonisi
ZPARP	Zaman Pencereli Araç Rotalama Problemi

TABLOLAR LİSTESİ

Tablo 1.1. Araç rotalamanın kısa tarihçesi.....5
Tablo 2.1. Av arama algoritması adımları.....26
Tablo 4.1. Müşteri sırası oluşturma.....40
Tablo 4.2. Prins'in etiketleme algoritması.....42
Tablo 4.3. SZPARP için etiketleme algoritması.....42
Tablo 4.4. SZPARP için önerilen AA algoritması.....45
Tablo 4.5. Önerilen AA algoritması parametre değerleri.....47
Tablo 4.6. Önerilen AA algoritmasıyla elde edilen sayısal sonuçlar.....48
Tablo 4.7. Önerilen AA algoritmasıyla PSO algoritması sonuçlarının karşılaştırılması.49

GİRİŞ

Günümüzde hem yerel hem de küresel pazarlarda yoğun bir rekabet söz konusudur. İşletmeler, yoğun rekabetin olduğu bu müşteri odaklı pazarlarda var olabilmek ve rekabet ile mücadele edebilmek için mamullerinin kalitesini yükseltirken, maliyetlerini düşürmek durumundadırlar. İşletmelerin toplam maliyetleri içinde önemli bir büyüklüğe sahip olan maliyet kalemlerinden biri lojistik maliyetleridir [1]. Lojistik yönetiminin temel amacı, hammaddenin satın alınmasından, bitmiş ürünün satışına kadar olan dağıtım, toplama ve depolama süreçlerinin mümkün olan en düşük maliyetle planlanmasıdır. Tüm bu dağıtım ve toplama faaliyetlerinin planlanmasının en zor ayağını ise araç rotalama problemleri (ARP) oluşturmaktadır.

ARP, bir veya birden çok depodan alınan malzemelerin, özellikleri belirli araçlar ile belirtilen kısıtlar altında en düşük maliyet ile müşterilere dağıtılması olarak tanımlanabilir. Araç rotalama problemleri, sadece üretim sistemleri içerisinde karşılaşılan bir problem değil, aynı zamanda personel veya öğrenci servislerinin rotalarının planlanması, toplu taşıma sistemlerinin planlanması, şehir içi ve dışı kargo sistemlerinin yönetimi, katı atık toplama sistemlerinin yönetimi gibi günümüzde birçok problemin temelini oluşturmaktadır [2].

NP-zor problem yapısına sahip ARP'nin birçok farklı tipi bulunmaktadır. Tez çalışmasında, araç rotalama problem çeşitlerinden sıkı zaman pencereli araç rotalama problemi (SZPARP) için yeni geliştirilmiş meta-sezgisel bir algoritma olan Av Arama (AA) algoritması ile çözüm aranmıştır.

Araç rotalama problemine, her bir müşterinin servisinin başlayacağı en erken ve en geç zaman kısıtlarının eklenmesiyle problem zaman pencereli araç rotalama problemine (ZPARP) dönüşür. Gerçek uygulamadaki problemlere bakıldığında ZPARP'nin

akar-yakıt dağıtımı, posta dağıtımı, tam zamanlı üretim için satıcı dağıtımı gibi problemlere daha uyumlu olduğu görülmektedir [3]. SZPARP'de ise servisin en erken zamandan önce, ya da en geç zamandan sonra başlaması söz konusu değildir. Yani en erken başlama zamanından önce gelen araç servis zamanını beklerken, en geç başlama zamanından sonra gelen araç ise servise başlayamamaktadır.

Av arama algoritması, hayvanların grup olarak avlanmasından esinlenilerek geliştirilmiş meta-sezgisel bir algoritmadır. Henüz sadece sürekli problemlere uygulanan bu algoritma, kombinatoryal bir problem olan SZPARP için çalışılmıştır. Tez çalışmasındaki temel amaç, birçok meta-sezgisel ile iyi çözümler elde edilmeye çalışılan SZPARP için yeni geliştirilmiş AA meta-sezgiseli ile de başarılı sonuçlara ulaşabilmektir.

Bu doğrultuda tez çalışmamızın ilk bölümünde ARP, bileşenleri, çeşitleri ve çözüm yöntemleri sunulmuştur. İkinci bölümde AA meta-sezgiseli detaylı olarak incelenmiştir. Üçüncü bölümde çalışma için seçilen SZPARP, detaylı olarak ele alınmış ve literatürdeki zaman pencereli araç rotalama problemleri irdelenmiştir. Dördüncü bölümde, sıkı zaman pencereli araç rotalama problemine AA algoritması ile çözüm aranmıştır. Son bölümde ise çalışmanın sonuçları sunulup tartışılarak, öneriler ve konuyla ilgili gelecekte yapılabilecek çalışmalardan bahsedilmiştir.

1. BÖLÜM
ARAÇ ROTALAMA PROBLEMİ

Araç Rotalama Problemleri, kapasitesi belirli araçlar ile belirlenen müşterilere, en uygun şekilde hizmet vermek için her araca en uygun rotayı atamayı kapsar [2]. ARP için bugüne kadar birçok tanım yapılmıştır. Aşağıda bu tanımlardan iki tanesi yer almaktadır:

"*Bir araç filosu ile müşterilerin önceden bilinen taleplerinin, merkez depoda başlayıp bitecek şekilde minimum maliyetli rotalamayla karşılanmasıdır*" [4].

"*Bir veya daha fazla depodan müşterilere servis yapmak üzere araç filosunun optimum rotalanması işlemidir*" [5].

ARP, literatürde sıkça yer alan Gezgin Satıcı Problemi (GSP) ile benzerlik göstermektedir. GSP, ARP'nin en temel ve üzerinde en çok çalışılan türüdür. Aralarındaki temel fark, GSP'de gezgin satıcının (aracın) kapasitesi yoktur, dolayısıyla bir tek satıcı (araç) tüm müşterilere hizmet verebilmektedir.

Kombinatoryal optimizasyonun en önemli ve çok çalışılan problemlerinden biri olan ARP'de, belirli bir müşteri kümesine hizmet edecek olan bir araç filosunun izleyeceği rotaların en iyisinin belirlenmesine çalışılır. Klasik ARP problemlerinin çözümü, her rotanın depodan başlayıp, depo ile bittiği ve her bir müşteriye bir kez uğranması kısıtının sağlandığı rotalar kümesidir. Bunun yanı sıra, problemin türüne göre bazı yan kısıtların da sağlanması gerekebilmektedir. En yaygın olan yan kısıtlar; kapasite kısıtı, bir rotada olabilecek en fazla talep noktası kısıtı, bir rotada aracın toplam süre kısıtı, talep noktalarına hizmetin başlanabileceği zaman penceresi kısıtı, bir talep noktasının başka bir talep noktasından önce ziyaret edilmesinin gerektiği öncelik kısıtıdır.

Toplam taşıma maliyetlerinin en küçüklenmesi araç rotalama probleminin amacıdır. Maliyeti yaratan etkenler gidilen mesafe, araç ve diğer elemanların kullanım süreleri, taşıma maliyetleri vb. olabilmektedir. Dolayısıyla toplam gidilen mesafe, dağıtım süreleri ya da araç sayısı azaltılarak amaç fonksiyonu en küçüklenebilmektedir. Yani ARP'de, bir depo, *m* adet araç ve *n* adet müşteri bulunmaktadır ve depoda yer alan araçlar, toplam maliyet en az olacak şekilde söz konusu müşterilerin ayrık miktardaki (kesikli) taleplerini karşılamak durumundadırlar.

ARP'de birden fazla ve genelde birbiriyle çelişen amaçlar dikkate alınabilmektedir. En bilinen amaçlar şöyle sıralanabilir:

- Araçların toplam kat ettiği mesafeye (ya da toplam seyahat süresine) ve kullanılan araçların sabit maliyetlerine (ve ilgili sürücüye) bağlı olan toplam taşıma maliyetlerini en küçüklemek,
- Tüm müşterilere hizmet etmek için gerekli olan toplam araç sayısını (ya da sürücü sayısını) en küçüklemek,
- Rotaları seyahat süresi ve araç yükü açısından dengelemek,
- Müşterilere parçalı dağıtım yapılmasından kaynaklanan cezaları en küçüklemek.

Yukarıda sayılan amaçların çeşitli kombinasyonları da problemin amaç fonksiyonunu oluşturabilmektedir [6].

1.1. Araç Rotalama Problemlerinin Kısa Tarihçesi

Araç rotalama ile ilgili ilk çalışma Danztig, Fulkerson ve Johson [7] tarafından yapılmıştır. Bu çalışmada, 15 müşterili bir ARP ele alınmış ve çözümlenmiştir. Clarke ve Wright [8], birden çok aracın bulunduğu araç rotalama problemlerini ilk defa literatüre kazandırmışlardır. Bu çalışma tam anlamıyla ilk araç rotalama çalışması olarak değerlendirilmektedir.

"Araç Rotalama" ismi ile yapılan ilk çalışmalar ise, Golden vd. [9] tarafından yapılmıştır. ARP'ye benzer olan ilk çalışmalar ise; uçak rotalama [10], ulaşım ağı tasarımı [11], hizmet araçlarının rotalanması [12] ve katı atık toplama sistemlerinin

tasarımı [13] olarak sıralanabilir. Solomon [14], literatürde "Solomon Problemleri" olarak anılan örnek problem setlerini hazırlamış ve birçoğunu optimal olarak çözmüştür. Bu problemler, literatürde geliştirilen ARP algoritmalarının test edilmesi amacıyla yaygın olarak kullanılmaktadır.

1980'lerde, işlemci hızlarının çok düşük olması, araştırmacıları daha karmaşık olan ARP'lerden uzak tutmuştur. Fakat 1990'lar ile başlayan ve günümüze kadar uzanan süreçte, ARP üzerinde birçok farklı çalışmalar yapılmıştır. Araştırmacılar, optimal veya optimale yakın çözümleri bulmak amacıyla, kesin çözüm yöntemleri, sezgisel veya meta-sezgisel yöntemler önermişlerdir [2]. ARP'lerin çözüm yöntemlerinin tarihsel gelişimi özet olarak Tablo 1.1'de verilmiştir.

Tablo 1.1. Araç rotalamanın kısa tarihçesi [15]

1950'ler:	ARP, tamsayılı programlama olarak formüle edilmiş ve 10-20 müşterili küçük problemler çözülmüştür.
1960'lar:	Rota kurma sezgiselleri sunulmuş ve 30-100 müşterili problemler çözülmüştür.
1970'ler:	İki fazlı sezgiseller ve interaktif (insan-makine) sezgiseller geliştirilmiş, yaklaşık 50 müşterili problemler eniyileme metotları ile çözülebilir hale gelmiştir.
1980'ler:	Matematiksel programlama esaslı prosedürler literatüre sunulmuştur. Etkileşimli (interaktif: insan-makine) sezgiseller geliştirilmiştir. Eniyileme yöntemleri kullanılarak yaklaşık 50 müşteriye sahip bazı problemler çözülmüştür.
1990'lar:	ARP'ye meta-sezgiseller uygulanmıştır. 50–100 müşteriye sahip bazı problemler için en iyi çözümler bulunmuştur.
2000'ler:	Meta-sezgiseller üzerine yapılan araştırmalar yoğunluk kazanmıştır. Birçok farklı meta-sezgisel ARP'lere uygulanmıştır. Melezleştirme teknikleri geliştirilmiş ve daha karmaşık ARP'ler üzerinde uygulamalar yapılmıştır.

1.2. Araç Rotalama Probleminin Bileşenleri

Araç rotalama problemlerine ait birtakım bileşenler söz konusudur. Bunlar talep yapısı, araç filosunun yapısı, başlangıç noktasının mahiyeti (tek veya çok, aynı veya farklı), dağıtım ve toplama noktalarının durumu gibi özelliklerdir [16].

Talep yapısı: ARP' de talebin önceden bilinip bilinmemesi, dağıtım ya da toplama sırasında değişip değişmemesi problemin modellenmesi ve çözümü açısından büyük önem arz etmektedir. Talep önceden biliniyorsa ve sabitse araçların rotalanması işlemi turlar öncesinde gerçekleştirilir ve bu çözüme göre hareket edilir. Dinamik yapıda talep söz konusu ise tur sırasında değişimler söz konusudur. Bir ya da daha fazla müşteriye ait talebin değişmesi ile rotalamanın güncellenmesi gerekebilir. Bu hem mesafenin en küçüklenmesi açısından hem de kapasite kısıtı varsa araçların durumu açısından bir gerekliliktir.

Araç filosunun yapısı: Genel olarak ARP problemlerinde filodaki araçların homojen olduğu varsayılmaktadır. Araç kapasiteleri bilinmektedir ve aynıdır. Filonun heterojen olması durumunda ise filodaki araçların kapasiteleri farklıdır. Heterojen yapıdaki araç filosuna sahip problemlerin çözümünde rotalama sırasında araçların ayrı ayrı kapasiteleri ele alınarak çözüm buna göre üretilmelidir. Ayrıca birçok rotalama probleminin çözüm sürecinde etkili olmayan hız, yakıt tüketim miktarları gibi bir takım özellikler de araç filosunun yapısı hakkında söylenebilecek unsurlardır.

Başlangıç noktasının mahiyeti: Başlangıç noktası genellikle ARP problemlerinde tektir ve birinci şehirdir. Filodaki araçlar merkez depo olarak da adlandırılan birinci şehirden çıkarlar, turlarını tamamladıktan sonra tekrar depoya dönerler. Dolayısıyla bu tip problemlerde birinci şehrin talebi sıfır olarak belirtilir. Bu problemler tek depolu olarak adlandırılır. Birden çok depo söz konusu olduğunda problem çok depolu ARP haline dönüşür. Depolara ait araçlar turlarından sonra tekrar kendi depolarına dönüyorsa, her bir depoya ait filonun turlarını kapsayan çözüm ayrı bir kapasite kısıtlı ARP problemi olarak ele alınır.

Dağıtım ve toplama noktalarının durumu: Genel olarak ARP problemlerinde toplama noktası denilince, merkez depo veya depolar anlaşılmaktadır. Dağıtım noktaları ise müşteriler ya da bayilerdir. Müşterilere ait dağıtım noktaları sabitse araçların rotaları

turlardan önce belirlenir. Aksi durumda dağıtım noktaları için bir seçim söz konusudur. Servislerin personelinin veya öğrencilerin taşımasını içeren problemlerde dağıtım ve toplama noktaları aynı olabilmektedir. Okul veya şirket sabah dağıtım noktası iken akşam toplama noktası olmaktadır. Ev ve duraklar ise sabah toplama noktası iken akşam dağıtım noktası olmaktadır.

1.3. Standart Araç Rotalama Problemi Formülasyonu

Standart Araç Rotalama Probleminin formülasyonu 1.1-1.6 eşitliklerinde verilmektedir [17].

$$\min Z = \sum_{i,j} c_{ij} \sum_{k} x_{ijk} \tag{1.1}$$

Öyle ki

$$\sum_{k} y_{ik} = \begin{matrix} 1, & i = 2, \dots, n \\ m, & i = 1 \end{matrix} \tag{1.2}$$

$$\sum_{j} x_{ijk} = y_{ik} \quad i = 1, \dots, n \quad k = 1, \dots, m \tag{1.3}$$

$$\sum_{i,j \in S} x_{ijk} \leq |S| - 1 \; tüm \; S \in \{2, \dots, n\} \quad k = 1, \dots, m \tag{1.4}$$

$$y_{ik} \in \{0,1\} \quad i = 1, \dots, n \quad k = 1, \dots, m \tag{1.5}$$

$$x_{ik} \in \{0,1\} \quad i, j = 1, \dots, n \quad k = 1, \dots, m \tag{1.6}$$

Burada;

$$x_{ik} = \begin{cases} 1, k \; aracı \; i \; düğümünden \; hemen \; sonra \; j \; düğümüne \; giderse \\ 0, diğer \; durumda \end{cases}$$

$$y_{ik} = \begin{cases} 1, k \; aracı \; i \; düğümünü \; ziyaret \; ederse \\ 0, diğer \; durumda \end{cases}$$

$m = araç \; sayısı,$

$n = düğüm \; sayısı,$

$S = düğümler \; setinin \; alt \; seti,$

$|S| = S\ alt\ setindeki$ düğüm $sayısı,$

ARP için geliştirilmiş çok sayıda matematiksel model bulunmaktadır. Literatürde yer alan Christofides vd.'nin [18] yaptığı çalışmadaki tamsayılı model 1.7-1.13 eşitlikleri ile ifade edilmektedir. Bu modelde *M* araç sayısını ve *N* müşteri sayısını göstermektedir.

$$x_{ijk} = \begin{cases} 1, eğer\ k\ nolu\ araç\ i\ müşterisinden\ j\ müşterine\ gidiyorsa \\ 0, diğer\ durumda \end{cases}$$

$c_{ij} = i$ müşterisi $ile\ j$ müşterisi $arasındaki\ mesafe\ (maliyet)$

$Q = bir\ araca$ yüklenebilecek $toplam$ ürün $miktarı$

$q_i = i$ müşterisinin $talep\ miktarı$

$y_i = alt\ turları\ engellemek\ için\ kullanılan\ rastgele$ değişken

$$Min\ Z = \sum_{i=0}^{N} \sum_{j=0}^{N} (c_{ij} \sum_{k=1}^{M} x_{ijk} \tag{1.7}$$

Öyle ki;

$$\sum_{i=0}^{N} \sum_{k=1}^{M} x_{ijk} = 1,\ \ j = 1, \ldots, N \tag{1.8}$$

$$\sum_{i=0}^{N} x_{ipk} - \sum_{j=0}^{N} x_{pjk} = 0,\ \ k = 1, \ldots, M \quad p = 0, \ldots, N \tag{1.9}$$

$$\sum_{i=1}^{N} (q_i \sum_{j=0}^{N} x_{ijk}) \leq Q,\ \ k = 1, \ldots, M \tag{1.10}$$

$$\sum_{j=1}^{N} x_{0jk} = 1,\ \ k = 1, \ldots, M \tag{1.11}$$

$$y_i - y_j + N \sum_{k=1}^{M} x_{ijk} \leq N - 1,\ \ i \neq j = 1, \ldots N \tag{1.12}$$

$$x_{ijk} \in \{0,1\}, \forall, i, j, k \quad y_i\ işareti\ serbest \tag{1.13}$$

Eşitlik (1.7)'deki amaç fonksiyonu toplam yol maliyetini (mesafesini) en küçüklemektedir. (1.8) kısıtına göre, her müşteri yalnızca bir kez ziyaret edilecektir. İkinci kısıt ise, eğer bir araç bir müşteri ziyaret ediyorsa aynı zamanda o müşteriden hareket etmek zorunda olduğunu belirtir. Her aracın taşıyabileceği toplam ürün miktarının bir sınırı vardır ve bununla ilgili kapasite kısıtı üçüncü kısıt ile ifade edilmiştir. Dördüncü kısıta göre, her araç yalnızca bir kez kullanılmalıdır, yani bir

rotaya atanmalıdır. Problemin çözümünde alt turların engellenmesi için son kısıt eklenmiştir [18].

1.4. Araç Rotalama Problemlerinin Sınıflandırması

ARP, belirli bir problemi göstermek yerine, Klasik ARP'nin yapısındaki geniş çeşitlilikten kaynaklanan birbirinden farklı problemleri içermektedir. Pratikte farklı durumlar için geliştirilmiş çok sayıda ARP ile karşılaşılmaktadır. Buna göre problemde yer alan işletme birimleri, araçlar, sürücüler, yollar ve müşterilerin özelliklerine göre çeşitli ARP türleri tanımlanmaktadır. Bu nedenle farklı kriterler dikkate alınarak ARP için çeşitli sınıflandırmalar yapılmıştır. Örneğin; çevre durumuna göre deterministik ve stokastik, rotaların durumuna göre açık uçlu ve kapalı uçlu, yolların durumuna göre simetrik ve asimetrik ARP'den bahsedilebilir.

Sınıflandırmada kullanılan kriterlerin en önemlisi kısıtlardır ve problem çoğu zaman ARP'de dikkate alınan kısıtın ismi ile beraber anılmaktadır. Ancak literatürde kısıtlarına göre sınıflandırma için farklı yaklaşımlar mevcuttur. Çünkü artan rekabet şartları ve değişen çevre koşulları sebebiyle rotalama problemlerine giderek daha fazla kısıt eklenmektedir. Probleme kısıt eklendikçe, tanımlanması ve çözümü zorlaşmaktadır. Pratikte kurulan lojistik sistemlerinde ARP'nin birebir modellenip çözümlenmesi zor olduğundan, bazı önemli kısıtlar seçilip diğerleri göz ardı edilerek problemler tanımlanmaya çalışılır. Aşağıda dikkate alınan önemli kısıtlara göre, ARP türlerinden kısaca bahsedilmiştir [19].

1.4.1. Kapasite Kısıtlı ARP

Dağıtım ağı en iyilemenin en önemli problemlerden biri olan ve ARP'nin en yaygın türü olan Kapasite Kısıtlı ARP'de her aracın belirli bir kapasitesi vardır ve müşterilerin talepleri önceden bilinmektedir. En basit kapasiteli araç rotalama probleminde her aracın kapasitesi eşittir ve araçlar bir depodan harekete başlarlar ve en son yine bu depoya dönerler. Müşterilerin talepleri tek seferde teslim edilmektedir, parçalama söz konusu değildir.

Literatürde üzerinde çok fazla çalışma olan bu problem türünde, her müşterinin belirli talep miktarı vardır ve her müşteri yalnızca bir araç tarafından ve sadece bir kez ziyaret

edilebilir. Depo ve müşterilerin birbirlerine olan uzaklıkları simetriktir ve araçların seyahat süresi bu mesafeyle doğru orantılı alınabilmektedir. Bu problemde de sıklıkla amaç, araçların kat ettiği toplam mesafeyi en küçüklemektir.

Müşterilere sadece bir kez uğranması gereken Kapasite Kısıtlı ARP'de amaç, talepleri bilinen müşterilerin toplam hizmet maliyetlerini en küçüklemektir. NP-zor sınıfına giren ARP için polinom zamanda en iyi çözümü bulabilen etkili bir algoritma yoktur. Matematiksel yöntemlerin kullanılması en iyi çözümü verecektir, fakat bu kesin yöntemler çözüm uzayının tamamını arayacağından hesaplama karmaşıklığına sahip olan ARP için, problemin büyüklüğü arttıkça çözüm için gereken süre de üssel olarak artacaktır [6].

Literatürde kapasite kısıtlı ARP'nin ele alındığı birçok çalışma bulunmaktadır. Chen vd. [20], kesikli parçacık sürü optimizasyonu ve tavlama benzetimi melez algoritmasını kapasite kısıtlı ARP'ye uygulamışlardır. Ralphs vd.'nin [21] yaptığı çalışmada, ARP amaç programlama ile çözümlendiği gibi diğer kombinatoryal problemlerin ARP'ye uygulanması için bir taslak sunulmuştur. Konu ile ilgili daha ayrıntılı bilgi için Ralphs vd.'nin yapmış oldukları çalışma incelenebilir.

1.4.2. Mesafe Kısıtlı ARP

Mesafe kısıtlı ARP, rotalara atanan her aracın gidebileceği maksimum mesafe kısıtının bulunduğu problemlerdir. Bu durum gerçek bir dağıtım probleminde taşınan ürünün cinsinden, araç veya sürücü kısıtlarından dolayı söz konusu olabilir. Eğer taşınan ürünün uzun süre taşıma nedeniyle bozulabilmesi söz konusuysa, ya da araç kullanıcısının sürekli olarak belirli bir süreden daha fazla yolculuk yapamaması söz konusu ise bu kısıt eklenmelidir [6].

Literatürde mesafe kısıtlı ARP'nin de farklı versiyonları bulunmaktadır. Örneğin, mesafe kısıtı yerine seyir süresi sınırlaması da konulabilir. Bu durumda her müşteri için servis süresi dikkate alınmaktadır. Ayrıca yükleme kapasitesi ile mesafe kısıtının birlikte kullanılması da karşılaşılan bir durumdur [19].

1.4.3. Önce Dağıt Sonra Topla ARP

ARP'nin bu türünde müşteriler, ürün teslim edilecek ve tam tersi ürün teslim alınacak müşteriler olmak üzere iki çeşittir. Bir rotaya her iki müşteri tipinden de atanabilmektedir, fakat önce ürün dağıtımı yapılacak olan müşterilere uğranacak, daha sonra toplama işlemi yapılacaktır. Dağıtımın ve toplamanın rastgele yapılması durumunda aracın yükleme yapılan tarafının, yeniden düzenlenmesi ekonomik ve olurlu değildir. Teslim edilecek ve toplanılacak miktarlar daha önceden bilinmektedir ve tüm araçlar aynı özelliklere sahiptir, yani aynı miktarda taşıma kapasitesine sahiptir. Burada dikkat edilmesi gereken diğer bir nokta da araçların hem dağıtım yapacağı miktarın, yani depodan hareket etmeden önce araca yüklenecek ürün miktarının, hem de teslim alınacak ürünlerin toplam miktarının, söz konusu araç kapasitesini geçememesidir [6].

Literatürde önce dağıt sonra topla ARP ile ilgili birçok çalışma bulunmaktadır. Bu çalışmalardan Brandao'nun [22] çalışmasında, problem tabu arama meta-sezgiseli ile çözülmüştür. Gendreau vd. [23], probleme komşuluk arama ile çözüm aramışlardır.

1.4.4. Eş Zamanlı Topla-Dağıt ARP

Eş Zamanlı Topla-Dağıt ARP'de ise, isminde de belirtildiği gibi, eş zamanlı olarak bir müşteriye hem ürün teslimatı yapılabilir hem de müşteriden ürün teslim alınabilir. Bu problem türünde müşteriler iki ayrı gruba ayrılmazlar. Yani önce dağıtım yapılacak ve sonra toplama yapılacak gibi bir kısıt yoktur. Araçlar rotaları boyunca dağıtım ve toplamayı bir arada yapabildiklerinden dolayı araç kapasitesinin her an korunması gibi zor bir durum söz konusudur. Her müşteriye teslimat yapılacak ürün miktarı, ya da toplama yapılacak miktar önceden bilinmektedir.

Bianchessi ve Righin'in [24] çalışmalarındaki tanıma göre, Eş Zamanlı Topla-Dağıt ARP, yapılacak işlemler arasında bir öncelik ilişkisi olmadan aynı anda ürün dağıtımının ve atık toplamanın uygun şekilde birleştirilmesidir. ARP'nin bu türü, ürün dağıtımının yapıldığı ve kullanılmış ürünlerin geri dönüşüm tesislerinde tekrar işlenmesi için geri toplanması ile ilgilenen, ters lojistik uygulamaları için uygundur [6].

1.4.5. Bölünmüş Dağıtımlı ARP

Bu problem tipinde, bir müşterinin talepleri birden fazla araç tarafından temin edilebilmektedir. Yani bir müşteriye farklı araçlar tarafından birden fazla uğranabilmektedir. Bu dağıtım şeklini, ortalama müşteri talebi çok büyük olduğunda kullanmak anlamlıdır. Bölünmüş dağıtımlı ARP araçların gittiği toplam mesafeyi en küçüklerken, klasik ARP probleminden sadece sağladığı şu kısıt ile farklılaşmaktadır: Bir müşterinin talebi bir veya daha fazla araç tarafından temin edilir. Bir diğer farklılığı da yukarıda belirtildiği gibi ortalama müşteri talebi klasik ARP'dekinden çok fazladır [6].

1.4.6. Çok Depolu ARP

Daha önce bahsedilen ARP problemlerinde tek deponun olduğu kabul edilmiştir, bu problem türünde ise isminden de anlaşıldığı üzere araçların harekete başlayabileceği birden fazla depo bulunmaktadır. Depoların ve müşterilerin konumları önceden bilinmektedir ve her depo tüm müşterilerin toplam taleplerini karşılayabilecek kapasiteye sahiptir. Bu problemde her araç hareket ettiği depoya geri dönmek durumundadır. Birden fazla deposu olan bir dağıtım şirketinin araç rotalaması yapılmakta ise, çok depolu olma durumunu yapılan modele ilave etmek gerekecektir. Bu ARP türü de NP-zor bir problemdir ve en iyi çözümün elde edilebileceği verimli bir yöntem bulunmamaktadır [6].

1.4.7. Periyodik ARP

Periyodik ARP'de belirli bir dönemin planı en başta yapılmaktadır ve müşteriler bu süreçte birden fazla hizmet görmektedir. Müşterilere yapılacak servis sayısı müşterilerin talep miktarlarına, stok alanlarına göre değişmektedir. Eğer bir müşterinin talep miktarı çok fazla ise, az miktarda talebi olan müşteriye göre; ya da stoklama alanı küçük ise, büyük olan müşteriye göre daha fazla ziyaret edilecektir. Bu problem sınıfı bakkaliye, alkolsüz içki endüstrisi, atık toplama gibi alanlarda ortaya çıkmaktadır [6].

1.4.8. Zaman Pencereli ARP

Zaman Pencereli ARP'de, kapasite kısıtlı ARP'de olduğu gibi belirli sayıda ve aynı kapasiteye ve özelliklere sahip özdeş araçlar vardır, talep miktarları ve konumları bilinen müşteriler ve yine konumu bilinen bir depo bulunmaktadır. Bu problem türünü diğerlerinden ayıran ve problemi daha da güçleştiren kısıt, her müşteriye servise başlanabilecek belirli bir (a_i, b_i) zaman aralığının olmasıdır. Bu zaman aralığı içerisinde söz konusu müşteriye hizmete başlamak gerekmektedir. Her müşteri için dağıtım ya da ürün toplamak için belirli bir servis süresi (t_i) vardır ve araç müşteriye uğradıktan sonra bu (t_i) hizmet süresi kadar kalacaktır, daha sonra bir sonraki müşteriye ya da depoya hareket edecektir. Gerçek hayatta müşteriler ürün teslimatını belirli zaman aralığında kabul edebilirler, bu durumlarda bu problem türü için geliştirilen yöntemleri kullanmak uygundur.

Zaman Pencereli ARP iki alt sınıfa ayrılmaktadır. Bunlar;

Esnek Zaman Pencereli ARP: Esnek zaman pencereli ARP'de, müşterilere ilgili zaman pencerelerinin dışında hizmet verilebilmektedir, fakat bu durumda bir ceza maliyeti söz konusudur. Deponun zaman penceresi sağlanmak zorundadır [6].

Sıkı Zaman Pencereli ARP: Eğer bir araç müşterinin söz konusu zaman aralığının başlangıcından önce gitmişse beklemek durumundadır, eğer son tarihinden sonra gitmişse de hizmet verememektedir. Sıkı zaman pencereli araç rotalama problemi ilk olarak Christofides vd. tarafından 1981 yılında çalışılmıştır. Sıkı zaman pencereli ARP'nin çalışıldığı literatürde çok sayıda çalışma bulunmamakla birlikte, son dönemde Çetin ve Gencer [25], kesin zaman pencereli - eş zamanlı dağıtım toplamalı araç rotalama probleminin matematiksel modeli üzerinde çalışmışlardır.

1.5. Araç Rotalama Problemlerine Çözüm Yöntemleri

NP-zor bir problem olan ARP'nin çözümü için literatürde çok sayıda yöntem bulunmaktadır. Bu yöntemler, en iyi çözümü veren kesin yöntemler ve sezgisel algoritmaları içeren yaklaşık yöntemler olmak üzere iki ana gruba ayrılabilir.

Geliştirilen klasik sezgisel algoritmalar genel olarak yapısal, iyileştirmeli ve çok aşamalı olmak üzere üç türdür. Son yıllarda hızlı bir yükselişe geçen çözüm yöntemi ise, yine en iyiyi garanti etmeyip yaklaşık çözümler veren meta-sezgisellerdir [19].

1.5.1. Kesin Yöntemler

Kesin yöntemler, problemin en iyi çözümünü bulan yöntemlerdir. Ancak kesin yöntemler, problem boyutu büyüdükçe ya da probleme başka kısıtlar eklendikçe çözüm süresinin artmasıyla birlikte makul sürelerde iyi çözümler bulamamaktadır.

Tamsayılı model olarak formüle edilen ARP'nin çözümü için dal-sınır (branch and bound), kesme düzlemi (cutting plane) ve dal-kesme (branch and cut) yöntemleri kullanılmaktadır. Ayrıca dinamik programlama, lagranj ayrıştırma (lagrangean relaxation), ağaç arama ve sütun yaratma (column generation) da kesin çözüm yöntemleri arasında yer almaktadır [19].

1.5.2. Sezgisel Yöntemler

ARP'nin çözümünde iyi sonuçlara kısa sürelerde ulaşmak için genellikle sezgisel yöntemler kullanılmaktadır. Sezgisel algoritmalar, problemin çözüm uzayı çok büyük olduğunda çözümün aranmasını sınırlamak üzere herhangi bir kural, strateji, sadeleştirme gibi alternatif hareketlerden etkili olanını kullanmak amacıyla tanımlanan kriterler veya bilgisayar yöntemleri olarak tanımlanabilir.

Sezgisel yöntemler arama uzayında oldukça kısıtlı bir alanda tarama yaparak, kısa sürede iyiye yakın çözümler üretmeyi amaçlarlar. Dolayısıyla bu tür algoritmalar yakınsama özelliğine sahip olsa da kesin çözümü garanti etmezler ve sadece en iyi çözümün yakınındaki bir çözümü garanti edebilirler.

Sezgisel metotlar genel olarak miyop metotlar olarak adlandırılır. Bunun sebebi, iterasyonların her zaman daha iyi çözüme doğru ilerlemesi ve kötü çözümün hiçbir zaman yeni çözüm olarak kabul edilmemesidir. Sezgisel yöntemlerde küresel en iyi çözüme gitmek yerine, çoğu zaman yerel en iyi çözümlerde kalınır. Fakat, sezgisel yöntemler kısa işlem sürelerinde yakın-optimal sonuçlar verebilirler.

ARP için geliştirilen klasik sezgisel algoritmalar genel olarak üç ana sınıfa ayrılmaktadır. Bunlar: [19]

- Tur kurucu sezgiseller
- Tur iyileştirici sezgiseller
- İki aşamalı metotlar

Tur Kurucu Sezgiseller: Bu yöntemlerde, genellikle her müşteri depoya atanarak basit turlar oluşturulur. Algoritmanın her aşamasında turlar araç kapasitesi dikkate alınarak birleştirilir ve uygun çözüm elde edilir. Tur kurucu metotlar arasında en çok tercih edileni Clarke ve Wright'ın [8], Dantzig ve Ramser'in çalışmasından esinlenerek geliştirdikleri tasarruf algoritmasıdır. Tasarruf algoritması bir başlangıç noktasının depo olarak ele alınmasıyla başlar. Başlangıçta tüm durakların depodan ziyaret edildiği varsayılır. Daha sonra, iki alt arkın birleştirilmesiyle elde edilen tasarruflar hesaplanır. En büyük tasarruf değerinden başlayarak arklar birleştirilir [2]. Desrochers ve Verhoog'in 1989 [26], Altinkemer ve Gavish'in 1991 [27] yılında yaptıkları çalışmalarda tasarruf algoritmasının birçok farklı çeşidi mevcuttur.

Tur İyileştirici Sezgiseller: Bu yöntemlerin tur kurucu sezgisellerden en büyük farkı, uygun bir çözümle aramaya başlamalarıdır. Her iterasyonda dal kombinasyonlarında sınırlı değişiklikler yapılarak, mevcut çözümden daha iyi amaç fonksiyonu değerine sahip başka bir çözüm elde edilmeye çalışılır, bu çözüm bulunduğunda ise mevcut çözüm iyileştirilmiş olur. Bu şekildeki bir algoritma, en iyileme literatüründe yerel arama algoritması olarak tanımlanmaktadır. Diğer bir ifade ile tur iyileştirici sezgiseller aslında birer yerel arama algoritmalarıdır. Yerel aramayı uygulayabilmek için öncelikle komşuluk yapısının tanımlanması gerekmektedir. Arama uzayında bir nokta olan s çözümünün komşuluk yapısı $N(s)$ ile gösterilir. $N(s)$ mevcut s çözümünden, tanımlanan bir hareketle ulaşılabilecek ve bir takım ölçütlere göre s noktasının yakınında yer alan tüm çözümleri içeren kümedir. Yerel aramanın her iterasyonunda mevcut çözüm, kendisinden daha iyi olan bir komşu çözüm ile yer değiştirilmektedir [19]. Thompson ve Psaraftis [28] ve Kinderwater ve Savelsbergh'in [29] çalışmaları bu tipteki yöntemlere örnektir.

İki Aşamalı Sezgiseller: İki-aşamalı yöntemler genel olarak, müşterilerin kapasite kısıtına uygun gruplara ayrılması ve her grup içinde rotalama işleminin yapılması olmak

üzere iki adımdan oluşmaktadır. Bu adımlardan hangisinin önce gerçekleştirildiğine göre, “önce grupla sonra rotala” veya “önce rotala sonra grupla” olmak üzere iki yöntem mevcuttur. Genellikle ikinci adımda yapılan işlem, ilk adımda yapılan işleme geri beslemeler ile sürdürülmektedir [19]. Bu sınıftaki yöntemlerin en önemlileri, Fisher ve Jaikumar Algoritması [30], Petal Algoritması [31] ve Süpürme (Sweep) Algoritması [32] olarak sıralanabilir.

1.5.3. Meta-Sezgisel Yöntemler

Meta-sezgisel yöntemler sezgisel yöntemlere göre çok daha özelleşmiş ve daha zeki bir şekilde küresel en iyiyi arayan metotlardır. Ancak, işlem sürelerinin tam olarak kestirilmesi güçtür. Bu yöntemler, küresel en iyi çözüme giderken sezgisel yöntemlere göre daha çok süre harcayabilirler.

ARP’lerin çözümü için araştırmacılar birçok meta-sezgisel üzerine araştırmalar yapmışlardır. Tabu arama (TA), tavlama benzetimi (TB), genetik algoritmalar (GA), karınca koloni optimizasyonu (KKO), diferansiyel gelişim (DG), harmoni arama (HA), yapay arı kolonisi (YAK) ve parçacık sürü optimizasyonu (PSO) algoritmaları ARP’lerin çözümünde kullanılan bazı meta-sezgisellerdir.

TA ve TB, komşuluk stratejisi kullanarak ilerleyen yöntemlerdir. Başka bir deyişle, bunlar popülasyon tabanlı olmayıp, yeni çözümlerin bir önceki çözümden üretildiği mekanizmalara sahiptir. GA, DG ve PSO popülasyon tabanlı meta-sezgisellerdir. GA ve DG çaprazlama ve mutasyon gibi bir veya iki bireyin kullanıldığı operatörlerle çözüm arayan yöntemlerdir. GA’nın getirdiği en önemli yenilik, yeni bireylerin seçiminde popülasyondaki tüm bireylerin bilgilerine başvurulmasıdır. KKO ve PSO’da ise, yeni çözümler bir önceki nesildeki tüm bireylerin bilgilerine başvurularak oluşturulur. Böylece, daha çok bilgi kullanılarak oluşturulan yeni nesiller, daha geniş bir alanda arama yaparken, algoritmanın küresel optimum noktasına gidişini de hızlandırırlar [2].

Tabu Arama: TA tekniğine ait ilk çalışmalar Glover’a aittir. Tabu arama, insan hafızasının çalışmasından esinlenilerek önerilmiş bir yerel arama yöntemidir. TA yerel en iyinin ötesindeki çözüm uzayını keşfetmek için yerel bir sezgisel arama prosedürüne kılavuzluk eden bir meta-sezgiseldir. TA bu kılavuzluğu, hafızasında aramanın hikayesini tutarak, yani bazı adaylara "yasak" koyarak, aramayı sınırlandırması ve yerel

en iyiden kurtarmasıyla yerine getirmektedir. TA'da arama boyunca en iyi komşu tasarımların elde edilmesini sağlayan hareketler yapay bir hafızaya kaydedilir. Bu hafıza kısa dönemli hafıza olarak adlandırılır. Bu yapay hafıza kullanılarak aramada önceki tasarımların tekrar elde edilmesini sağlayacak hareketler yasaklanır. Arama esnasında yapılmasına izin verilmeyen bu hareketlere tabu denir. Tabu hareketler tabu listesi adı verilen ve yapay hafıza görevi gören tek boyutlu bir listede muhafaza edilir. Tabu listesine kaydedilen her hareket, bir tasarım değişkenini temsil eder. TA'da her tasarım değişkeni için tanımlanan komşu tasarımlar içerisinde en iyi olan tasarım, tabu listesi de kontrol edilerek mevcut tasarımın yerini alır. Arama işlemine mevcut tasarımın yerini alan en iyi komşu tasarım ile devam edilir. Bunlara ek olarak, bulunan komşu çözüm tabu listesinde olmasına rağmen en iyi çözümden daha iyi ise, belirli bir kritere göre kabul edilir. Bu kritere aspirasyon kriteri denir. Bu kriterle kabul edilen bir hareketin tabu olma durumu kaldırılmaz ve hareket tabu listesinden çıkarılmaz. Aynı işlem diğer tasarım değişkenleri için de tekrarlanarak bir iterasyon tamamlanır. Kabul edilen durdurma kriteri sağlanıncaya kadar her iterasyon için bu işlemler tekrarlanır [33].

Braysy'nin 2002'de tabu arama sezgiselini zaman pencereli araç rotalama problemine uyguladığı çalışmasında amaç, tüm tabu arama algoritmalarının karşılaştırılması olmuştur. Bu doğrultuda, çalışmada öncelikle ZPARP problemi ve tabu arama kısaca anlatılmış, sonrasında uygulamanın sonuçları sunularak analizler yapılmıştır. Solomon problemleri için elde edilen sonuçların analizinde, literatürdeki tabu algoritmalarının karşılaştırması yapılarak Gehring ve Homberger [34], Cordeau ve ark. [35], Taillard ve ark. [36] ve [37] algoritmalarının en iyi performans gösteren algoritmalar olduğu görülmüştür.

Tavlama Benzetimi: TB algoritmasının temelini oluşturan fikirler, ilk olarak Metropolis vd. tarafından 1953 [38] yılında önerilmiştir. TB algoritması, malzemelerin sıcak banyolarda soğutulmasından esinlenilerek geliştirilmiştir. Eğer, katı malzeme erime noktasına kadar ısıtılır ve katı hale geçinceye kadar tekrar soğutulursa, katı malzemenin soğutulmuş halinin yapısal özellikleri soğutma oranına bağlı olur. Örneğin, kristaller, çok yavaş bir soğutma ile büyüyebilirler, fakat hızlı bir soğutma, kristal yapısında kusurların oluşmasına neden olabilir. Tavlama sürecinin, malzemeyle ilgili bir partiküler sistemi olarak benzetimi yapılabilir. Temel olarak, Metropolis algoritması, soğutma sürecinden geçirilen bir sistemin kararlı bir duruma gelinceye kadar,

enerjisindeki değişikliklerin benzetimini yapar. Otuz yıl sonra, Kirkpartrick vd. [39], bu tür bir benzetimin, optimal sonuca yakınsama amacıyla, optimizasyon problemlerine uygulanabileceğini söylemişlerdir. TB yaklaşımı, uygun çözümlerin bir alt kümesi içerisinde, mevcut çözümden komşu çözümlere tekrarlı bir arama tekniği olan, yerel (komşu) arama tekniğinin bir varyasyonu olarak algılanabilir. Bir minimizasyon problemi için yerel arama tekniği, aramanın daima gelişme yönünde olduğu bir iniş stratejisini çalıştırır. Bununla birlikte, böyle bir strateji, küresel bir çözümden çok, yerel bir çözüme yakınsar. Algoritmanın birkaç değişik başlangıç çözümüyle uygulanması veya komşulukların karmaşıklığını amaçların kapsamını genişleterek/arttırarak da tatminkâr bir çözüm bulma yoluna gidilebilir [40]. TB'de TA'daki gibi, yeni çözümler, komşuluk oluşturma stratejisi kullanılarak elde edilir. Fakat bu algoritmada yasak listesi yerine, stokastik bir yapı kullanılır. Bir en küçükleme problemi için, TB basitçe şu şekilde çalışmaktadır: Mevcut çözüm *X* iken komşuluk stratejisi ile üretilen komşu çözüm x' olarak alınırsa ve eğer $delta, \Delta = f(x) - f(x')$ ise yeni çözüm $x', (x = x')$ olarak kabul edilir. Burada *f(x)*, amaç fonksiyonunu ifade eder. Eğer delta sıfırdan büyük ise $(\Delta > 0), x', e^{-\Delta/T} \leq 0.5$ olduğu durumlarda kabul edilir, diğer durumlarda ise yeni çözüm olarak kabul edilmez. *T*, sıcaklık parametresi olup, her iterasyonda belirli bir oranda düşürülür [2].

Czech ve Czarnas'ın [41] zaman pencereli araç rotalama problemi çözümünde paralel tavlama benzetimi algoritması ile yaptığı çalışmalarında, paralellik kullanılarak iyi sonuçlar elde edilmeye çalışılmıştır. Sonuçlar paralel tavlama benzetiminin, iki kriterli optimizasyon problemlerinde başarılı sonuçlar elde edilebildiğini göstermiştir. Solomon problem setlerinin bazılarında çok iyi sonuçlar elde edildiği de görülmüştür.

Karınca Koloni Optimizasyonu Algoritması: Karınca koloni meta-sezgiseli, doğal karıncaların yuvaları ile besin kaynakları arasında takip ettikleri yolların izlenmesi sonucu ortaya çıkan bilimsel gerçekler üzerine doğmuştur. Gerçek karıncalar ile ilgili deneyler ilk olarak 1989 yılında laboratuar ortamında yetiştirilmiş karınca kolonileri üzerinde yapılmıştır. Bu çalışmalarda elde edilen sonuçlar şöyledir: pek çok karınca türü neredeyse körtür. Karıncalar yuvalarından yiyecek kaynağına veya tersi yönde hareket ederlerken geçtikleri yollara feremon adı verilen bir tür kimyasal madde bırakmaktadırlar. Karıncalar bir yol seçmeleri gerektiği zaman bu seçimi alternatif yollar üzerine bırakılmış feremon yoğunluğuna göre belirlemektedirler ve karıncaların

bu hareketleri merkezi bir kontrol ile sağlanmamaktadır şeklinde özetlenmiştir [42]. Karıncalar, yuvalarından bir gıda kaynağına giden en kısa yolu, herhangi görsel ipucu kullanmadan bulma kabiliyetine sahiptirler. Ayrıca, çevrelerindeki değişikliklere çok iyi uyum sağlama özellikleri vardır. Örnek vermek gerekirse, bir gıdaya giden yolda herhangi bir problem meydana gelmesi (bir engelin ortaya çıkması gibi) ve yolun kullanılamaz olması durumunda, yeniden en kısa yolu bulabilirler. Karıncalar ilerlerken, belirli bir miktar feremon depo ederler ve olasılığa dayanan bir yöntemle feremonun daha çok olduğu yolu az olduğu yola tercih ederler.

Yıldırım ve Çatay'ın [43] zaman pencereli ve zamandan bağımsız araç rotalama problemi için yaptıkları karınca koloni algoritması ile olan çalışmalarında, problem iki durum için ele alınmıştır. Çalışmada yerel aramanın karınca koloni algoritmasının çözüm kalitesini önemli ölçüde artırdığı gözlemlenmiştir. İlerleyen çalışmalarda zaman penceresi ile görünürlük fonksiyonu oluşturularak algoritmanın hesaplama süresi ve kalitesini artırıcı analizler yapılması planlanmıştır.

Genetik Algoritmalar: Genetik algoritmaların temel ilkeleri ilk kez Michigan Üniversitesi'nde John Holland tarafından ortaya atılmıştır. Holland 1975 yılında yaptığı çalışmaları "Adaptation Natural and Artificial Systems" adlı kitabında bir araya getirmiştir. Genetik algoritmalar, doğada geçerli olan en iyinin yaşaması kuralına dayanarak sürekli iyileşen çözümler üretir. Bunun için "iyi"nin ne olduğunu belirleyen bir uygunluk fonksiyonu ve yeni çözümler üretmek için yeniden üretim, mutasyon, çaprazlama gibi operatörler kullanılır [9]. Genetik algoritmaların bir diğer önemli özelliği de popülasyon temelli çalışmasıdır. Bu sayede, çok sayıda çözümün içinden iyileri seçilip kötüleri elenebilir. Genetik algoritmalar genel anlamda, dizilerden oluşan bir popülasyona üreme, çaprazlama ve mutasyon operatörlerinin uygulanmasını içerir. Bu operatörlerin uygulanmasından sonra yeni bir popülasyon (yavru popülasyon) oluşur. Yeni popülasyon eski (ebeveyn) popülasyon ile yer değiştirir. Her dizinin bir uyum değeri mevcuttur. Diziler uyum değerlerine göre seçilirler. Ortalama uyum değerlerinin üzerine uyuma sahip dizilerin gelecek nesillerde temsil edilme olasılığı daha yüksektir. Evrim süreci, popülasyonun ortalama uyumunu giderek artırır ve ilerleyen nesillerde daha iyi uyum değerleri elde edilmesini sağlar [1].

Ombuki vd. 2006 [44] yılında yaptıkları çalışmada, çok amaçlı genetik algoritmayı zaman pencereli araç rotalama problemine uygulanmışlardır. Çalışmada çok amaçlı problem ve genetik algoritma pareto tekniği ile kullanılarak çözümlenmeye çalışılmıştır. Amaçlar olarak, araç sayısı ve toplam maliyet ele alınmıştır. Elde edilen sonuçların literatürdeki diğer çalışmalarla rekabet edebilecek düzeyde olduğu görülmüştür.

Parçacık Sürü Optimizasyonu Algoritması: PSO, 1995 yılında J.Kennedy ve R.C.Eberhart [45] tarafından; kuş sürülerinin davranışlarından esinlenilerek geliştirilmiş popülasyon tabanlı stokastik bir optimizasyon tekniğidir. Doğrusal olmayan problemlerin çözümü için tasarlanmıştır. Çok parametreli ve çok değişkenli optimizasyon problemlerine çözüm bulmak için kullanılmaktadır. PSO, genetik algoritmalar gibi evrimsel hesaplama teknikleriyle birçok benzerlik gösterir. Sistem rastgele çözümler içeren bir popülasyonla başlatılır ve nesilleri güncelleyerek en uygun çözümü araştırır. Kuşların uzayda, yerini bilmedikleri yiyeceği aramaları, bir probleme çözüm aramaya benzetilir. Kuşlar yiyecek ararken yiyeceğe en yakın olan kuşu takip ederler. Parçacık olarak adlandırılan her tekil çözüm, arama uzayındaki bir kuştur. Parçacık hareket ettiğinde, kendi koordinatlarını bir fonksiyona gönderir ve böylece parçacığın uygunluk değeri ölçülmüş olur (yani yiyeceğe ne kadar uzaklıkta olduğu ölçülmüş olur). Bir parçacık; koordinatlarını, hızını (çözüm uzayındaki her boyutta ne kadar hızla ilerlediği), şimdiye kadar elde ettiği en iyi uygunluk değerini ve bu değeri elde ettiği koordinatları hatırlamalıdır. Çözüm uzayının her boyutundaki hızının ve yönünün her seferinde nasıl değişeceği, komşularının en iyi koordinatları ve kendi kişisel en iyi koordinatlarının bir birleşimi olacaktır [46]. Parçacık sürü optimizasyonu çizelgeleme problemi [47] ve gezgin satıcı problemleri gibi [48-49] bir çok kesikli optimizasyon problemine uygulanmıştır.

Chen'in 2006 [20] yılındaki çalışmasında ise, melez parçacık sürüsü algoritması kapasite kısıtlı araç rotalama problemine uygulamıştır. Çalışmada parçacık sürüsü algoritması yerel ve global aramada etkili sonuçlar veren tavlama benzetimi ile birleştirilmiştir. Elde edilen sonuçlar, büyük boyutlu kapasite kısıtlı araç rotalama problemlerinde algoritmanın oldukça etkili olduğunu göstermiştir [20]. Zhu'nun [50] zaman pencereli araç rotalama problemi için parçacık sürüsü algoritması ile yaptığı çalışmasında ise, parçacık sürüsü algoritması ile elde edilen sonuçlar genetik algoritma ile kıyaslanmıştır. Buna göre literatürdeki birçok problemin çözümünde parçacık sürüsü

algoritmasının genetik algoritmaya göre daha etkili sonuçlar verdiği, ayrıca en iyi çözüme daha hızlı ulaşıldığı görülmüştür.

Diferansiyel Gelişim Algoritması: DG, Price ve Storn tarafından 1995 [51] yılında geliştirilmiş, özellikle sürekli verilerin söz konusu olduğu problemlerde etkin sonuçlar verebilen, isleyiş ve operatörleri itibariyle genetik algoritmaya dayanan popülasyon temelli sezgisel bir optimizasyon tekniğidir. Aynı anda birçok noktada araştırma yapmaktadır. İterasyonlar boyunca, operatörler yardımıyla problemin çözümü için daha iyi sonuçlar araştırılmaktadır. Klasik ikili GA' dan farklı olarak değişkenler gerçek değerleriyle temsil edilmektedir. GA'da da gerçek değerlerle kodlama kullanılmaktadır. Ancak Price ve Storn genetik operatörlerdeki birtakım değişikliklerle, gerçek değerlerle kodlamanın kullanıldığı problemlerin çözüm performansını arttırmaya çalışmışlardır. Genetik algoritmadaki çaprazlama, mutasyon ve seçim operatörleri DG'de de kullanılmaktadır. Farklı olarak her bir operatör tüm popülasyona sırayla uygulanmamaktadır. Kromozomlar tek tek ele alınmakta, rastgele seçilen diğer üç kromozomda kullanılarak yeni bir birey elde edilmektedir. Bu işlemler sırasında mutasyon ve çaprazlama operatörleri kullanılmış olmaktadır. Mevcut kromozomla elde edilen yeni kromozomun uygunlukları karşılaştırılarak uygunluğu daha iyi olan, yeni birey olarak bir sonraki popülasyona aktarılmaktadır. Böylelikle seçim operatörü de kullanılmış olmaktadır [52]. Sürekli problemler için geliştirilen diferansiyel gelişim algoritmasının, literatürde kesikli problemlere uygulandığı birçok çalışma mevcuttur. Literatürdeki bu çalışmalara sırt çantası problemi [53], kombinatoryal ihalelerde teklif kombinasyonlarının belirlenmesi [54], gezgin satıcı problemi ve N vezir problemleri [55] örnek olarak verilebilir.

Mingyong, Erbao [56] diferansiyel gelişim algoritmasını eş zamanlı topla-dağıt araç rotalama problemi için çözümleyerek genetik algoritma ile karşılaştırmışlardır. Sonuçlar diferansiyel gelişim algoritmasının performansının genetik algoritmadan daha iyi olduğunu göstermiştir.

Yapay Arı Kolonisi Algoritması: Arı sistemi kavramı ilk olarak Tomoya Sato ve Masafumi Hagiwara tarafından 1997 yılında ortaya konulmuştur. Yapay arı kolonisi algoritması gerçek arıların davranışlarından esinlenerek oluşturulan bir sezgisel algoritmadır [57]. Arı algoritması 2005 yılında Duc-Truong Pham tarafından geliştirilmiştir ve bal arılarının yiyecek arama davranışını taklit eden popülasyon tabanlı

bir arama algoritmadır [58]. Yapay arı kolonisi algoritması ise 2005 yılında D. Karaboğa tarafından arıların bal yapmak üzere nektar arama davranışını model alarak geliştirilmiştir. Algoritmanın temel aldığı minimal modelde üç çeşit yapay arı bulunmaktadır. Bunlar işçi arılar, gözcü arılar ve kâşif arılardır. Algoritmada öncelikle kâşif arılar probleme yönelik olarak rastgele belirlenen çözüm noktalarına yerleştirilir ve artık birer kaynakları olduğundan işçi arı durumuna geçerler. Her bir işçi arı elindeki çözümden, diğer arılardan aldığı bilgiler ışığında alternatif çözümler üretir. Elindeki çözümden (nektar kaynağı) daha iyi bir çözüm bulduğunda ise diğerini terk edip bu kaynağa yönelir. Daha sonra devreye gözcü arılar girer ve işçi arıların kaynak hakkında verdikleri bilgileri kullanarak bu arıların gittikleri kaynakların etrafındaki kaynaklara (çözümlere) yönelirler. Eğer kaynak (çözüm) işe yaramaz ise yapay kâşif arılar başka kaynaklar (çözümler) bulmak üzere yeniden işbaşı yaparlar. Bu meta-sezgisel yöntem ile kesikli problemlere çözüm aranan çalışmalara çizelgeleme problemleri [59-60] ve makine iş yükleme [61] örnek olarak verilebilir.

Brajevic [62], yapay arı koloni algoritmasını, kapasite kısıtlı araç rotalama problemlerine uygulamış ve bilinen 12 küçük boyutlu problem için çözmüştür. Sonuçlar problemler için bilinen en iyi sonuçlarla kıyaslandığında, önerilen algoritmanın iyi ve umut verici bir yaklaşım olduğunu göstermiştir.

Harmoni Arama Algoritması: Harmoni arama yöntemi, 2001 yılında Geem ve arkadaşları tarafından geliştirilmiştir. Bu sayısal optimizasyon tekniği, bir müzisyenin daha iyi bir harmoni aradığı müzikal performans sürecine benzemektedir. HA, bir orkestradaki müzisyenlerin çaldıkları notalar ile harmonik açıdan en iyi melodinin elde edilmesi prensibine dayanmaktadır. Harmoni araştırması tekniğinde orkestra, ancak tüm orkestra elemanlarının birbirleri ile harmonik açıdan uyumlu bir şekilde çalmaları ile en estetik melodiyi elde edebilirken, optimizasyon modellerinde en iyi çözüm ancak amaç fonksiyonunun küresel optimuma giderek yaklaşması sonucu elde edilebilmektedir. Oluşturulan harmoni, müzisyenlerin hafızalarındaki en kötü harmoniden daha iyi ise bu harmoni en kötü olanla değiştirilmekte ve bu işlem estetik açıdan en güzel harmoni elde edilinceye kadar devam etmektedir. Diğer yandan, optimizasyon problemlerinin çözümü için her bir müzisyenin bir karar değişkeninin yerini alması durumunda, müzisyenlerin hafızalarındaki her bir nota, o karar değişkenine ait farklı bir çözüme karşılık gelmektedir. Bu çözüm hafızadaki en kötü çözümden daha iyi olması durumunda

çözümler yer değiştirilmekte ve en iyi çözüm elde edilinceye kadar bu işleme devam edilmektedir [63]. Harmoni arama algoritmasının kesikli problemlerden akış tipi çizelgeleme [64] ve sırt çantası problemi [65] gibi çeşitli problemlere uygulanmıştır.

Geem'in [66] araç rotalama problemi için harmoni arama yöntemi ile yaptığı çalışmasında, okul servisi problemi için harmonide sürekliliği sağlayan adım, kesikli kısıt varsayılarak algoritma kesikli hale getirilerek çözümlenmiştir. Algoritma kesikli forma getirildikten sonra elde edilen sonuçlar genetik algoritma ile kıyaslanmıştır. Buna göre harmoni arama ile global optimum, ortalama maliyet ve çözüm süresi açısından bir çok çözümde genetik algoritmadan daha iyi çözümler elde edilebildiği görülmüştür.

2. BÖLÜM
AV ARAMA ALGORİTMASI

Son yıllarda, çeşitli optimizasyon problemleri meta-sezgisel algoritmalarla çözülmektedir. Yeni sezgisel algoritmalarla zor ve karmaşık gerçek hayat problemlerine çözümler aranmaktadır.

Av arama optimizasyon algoritması, hayvanların (aslanlar, kurtlar ve yunuslar gibi) grup olarak avlanmasından esinlenilerek geliştirilmiş bir meta-sezgisel algoritmadır. Hayvanların avlanma yollarında farklılıklar olmasına rağmen, hepsinde ortak olarak grupta bir av vardır. Ek olarak, grubun her bir üyesi kendi konumunu ve pozisyonunu diğer üyelerin konumuna göre düzeltmektedir. Eğer av grup içine alınan halkadan kaçarsa, avcılar grubu avı tekrar kuşatmak için organize ederler. Gruptaki avcıların avı kuşatmak ve yakalamak için belli stratejileri vardır. Örneğin kurtlar, sayıca çoğunlukta olarak avlanırlar ve bu sayede kendilerine güvenerek daha büyük ve hızlı hayvanları avlayabilirler. Gruptaki avcılar bir av seçer ve yavaş yavaş ava doğru ilerlerler. Avın onların kokusunu alabileceği, rüzgara doğru durmaktan kaçınırlar. Av arama algoritmasında da yasal alanlardan kaçınmak için bu fikir kullanılır. Algoritmada, her avcı problem için çözüm olarak gösterilir. Algoritmada gruptaki gerçek hayvanlar gibi yapay avcılar, işbirliği içinde avı bulup yakalamaya çalışırlar.

2.1. Av Arama Algoritmasının İşleyişi

Av arama meta-sezgisel algoritması, gerçek hayat grup avlanmasındaki avın yakalanması olayından esinlenilerek geliştirilmiştir. Grup üyelerinin avı sarıp, onu yakalamak için yaptıkları işbirliği, optimizasyon sürecinde amaç fonksiyonunun en iyi çözümü bulması olayına benzerlik gösterir. Her avcının ava göre olan konumu, avın yakalanma şansını belirler. Benzer şekilde, amaç fonksiyonunun değeri her karar

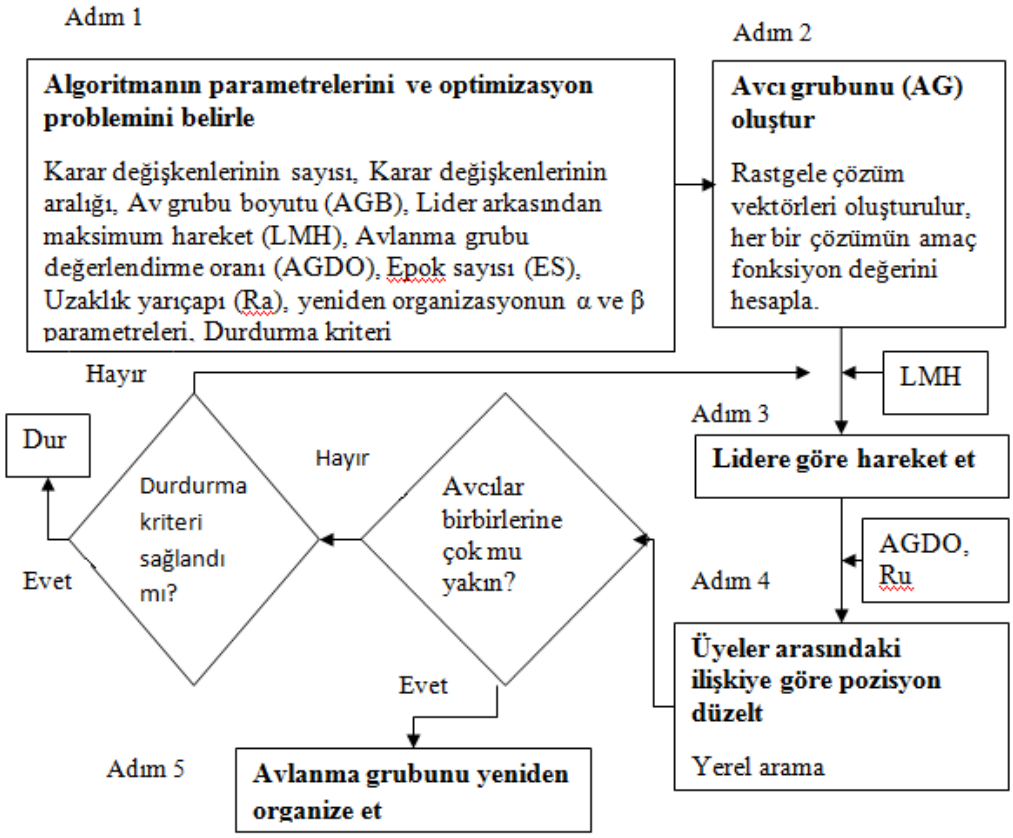
değişkeni için atanan değerler kümesi tarafından belirlenir. AA algoritması, hayvanların yemek ararken grup avlanma modeline dayanarak geliştirilmiştir.

Sürekli optimizasyon problemlerinde çözüm, bir amaç fonksiyonu veya uygunluk fonksiyonu içine karar değişkenlerinin değerleri koyularak tahmin edilmeye çalışılır. Grup avlanmasıyla kıyaslandığında, sürekli optimizasyon problemindeki her bir "avcı" problemdeki "çözüm" ile yer değiştirir (çözüm yapay avcı olarak da adlandırılabilir). Hayvanlardaki grup avlanması ve meta-sezgisel algoritma arasında temel bir farklılık olduğu unutulmamalıdır. Hayvanların grup halinde avlanmasında, avcılar ya avını görür, ya da gece avlandıklarında avlarının kokusunu hissederek onların yerlerine karar verirler. Bunun aksine optimizasyon problemlerinde, optimum çözüme ya da noktaya dair bir işaret yoktur. Hayvanların grup avlanmasında, çözüm (av) dinamiktir ve avcılar avlarının pozisyonlarını doğru tahmin etmelidirler. Optimizasyon probleminde, çözüm statiktir ve arama süreci boyunca pozisyonu değişmez. Gerçek ve yapay avlanma gruplarının farklı zorlukları vardır. Algoritmadaki avlanma sürecinde de bu dinamik yapıya benzer durum vardır, yapay avcılar lidere göre hareket ederler. Lider olan avcı, mevcut durumdaki en iyi pozisyonda yer alır. Aslında, liderin optimum çözümü bulduğu ve diğer bireylerin ona doğru hareket ettiği varsayılır. Mevcut liderden daha iyi noktayı bulan avcı diğer aşamada lider olur.

Gerçekte hayvanlar sadece avlarına doğru ilerlemenin yanı sıra pozisyonlarını da (diğer avcıların ve avın durumlarına göre) düzeltirler. AA algoritmasında ise bu işlem, bir önceki liderin hareketinden sonra gerçekleştirilir ve avcılar diğer üyelerin pozisyonlarına dayanarak pozisyonlarını düzeltirler. Bu yapılandırma avlanma grup değerlendirme oranı (AGDO) ile kurulmuştur.

Gerçek hayvanların grup avlanmasında ayrıca, eğer av halkanın dışına kaçarsa, avcılar avını tekrar halka içine almak için organize olurlar. Avlanma algoritmasında ise, yetenek avcılara verilmiştir, bu yüzden avcılar kuşatma alanının dışında da avlarını arayabilirler. Eğer avcıların pozisyonları birbirlerine çok yakınsa, diğer adımda optimum çözümü bulmak için grup yeniden organize olur.

Şekil 2.1, avlanma algoritmasının prosedürünü, Tablo 2.1 ise adımlarını göstermektedir.



Şekil 2.1. Av arama algoritması optimizasyon süreci

Tablo 2.1. Av arama algoritması adımları

Adım 1. Algoritmanın parametrelerini ve optimizasyon problemini belirle
Adım 2. Avcı grubunu oluştur.
Adım 3. Lidere göre hareket et
Adım 4. Üyeler arasındaki ilişkiye göre pozisyonu düzelt
Adım 5. Avlanma grubunu yeniden organize et
Adım 6. Durdurma kriteri sağlanana kadar adım 3,4 ve 5'i tekrarla

Adım 1. Algoritma parametreleri ve optimizasyon probleminin belirlenmesi

Bu adımda tanımlanan optimizasyon problemini çözmek için, avlanma algoritması parametreleri gereklidir. Avlanma grubu boyutu (avlanma grubundaki çözüm vektörlerinin sayısı), lider arkasından maksimum hareket, avlanma grubu değerlendirme oranı (0 ile 1 aralığında değişir) parametrelerinin değerleri belirlenir.

LMH ve AGDO parametreleri adım 3 ve 4'te tanımlanan avcının pozisyonunun iyileştirilmesinde kullanılmaktadır.

Adım 2. Avcı grubunun oluşturulması.

Avcı sayısına bağlı olarak AG matrisi, rastgele üretilmiş çözüm vektörleri ile doldurulur. Her bir çözüm vektörünün amaç fonksiyonu değeri hesaplanır ve amaç fonksiyonu değerine göre lider tanımlanır.

Adım 3. Lidere göre hareket etmek.

Yeni avcıların pozisyonları $x' = (x'_1, \dots, x'_N)$ lidere göre hareket edilerek üretilir.

$$x'_1 = x_i + rand \times LMH \times (x_i^L - x_i) \tag{2.1}$$

LMH lidere doğru yapılan en büyük harekettir ve 0-1 aralığında rastgele değer alır, x_i^L i. değişken için liderin pozisyon değerini verir.

Her bir avcı için, eğer lidere doğru hareket başarılıysa, avcı yeni pozisyon alır. Ancak eğer hareket başarılı değilse yani yeni pozisyon öncekinden daha kötüyse, avcı önceki pozisyonuna geri döner. Bununla birlikte bazı avantajlar sağlanır. İlk olarak gruptaki en kötü avcı ile diğer avcılar kıyaslanmamış olur, böylelikle diğer çözümleri araştırmak için en zayıf üyeye izin verilir. Böylelikle genetik algoritmadaki gibi daha iyi çözümler bulunabilir. İkinci olarak, önceki pozisyon ile mevcut pozisyonu kıyaslamak avcıların hızlı yakınsamasını önler ve bu şekilde iyi pozisyonlar hiçbir zaman elenemez.

LMH parametresinin değeri probleme göre farklılaşır ve problem boyutuna göre belirlenecek iterasyon sayısına göre belirlenir.

Adım 4. Üyeler arasındaki ilişkiye göre pozisyon düzeltmek.

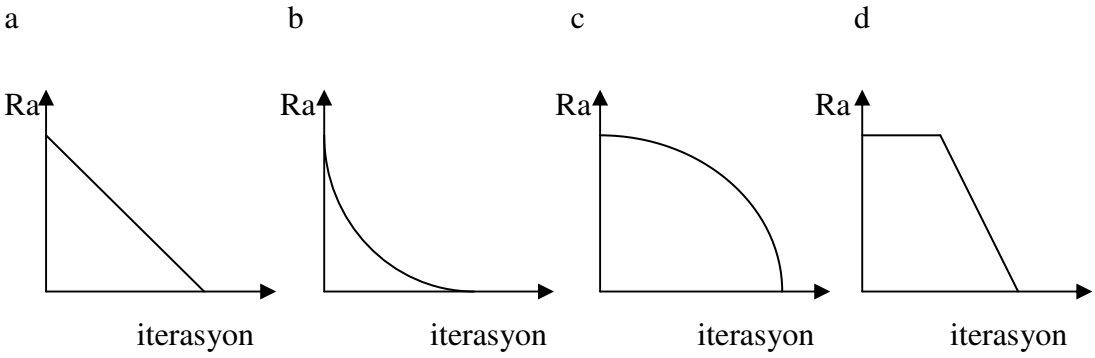
Bu adımda avcılar arasındaki iş birliği, daha etkin bir şekilde avlanmak için modellenmiştir. Lidere doğru hareketten sonra avcılar daha iyi bir çözümler bulmak için farklı bir pozisyon seçerler. Avcılar pozisyon düzeltmeyi iki yolla yaparlar.

1. Gerçek değer düzeltme
2. Sayısal değer düzeltme

Gerçek değer düzeltmede yeni avcının pozisyonu $x' = (x'_1, \dots, x'_N)$ avcı grubundan üretilir. Avcının durumu, pozisyon düzeltmeye bağlıdır. Örneğin, yeni vektörde j. avcı için ilk tasarım değişkeninin değeri avcı grubundan ya da AGDO parametresi kullanılarak düzeltilmiş değerlerden alınabilir. Değişken değerlerinin güncellenmesi ile aşağıdaki durum ortaya çıkar.

$$x_i^{j'} \leftarrow \begin{cases} x_i^{j'} \in \{x_i^1, x_i^2, \dots, x_i^{AGB}\} & AGDO\ olasılığı\ ile \\ x_i^{j'} = x_i^j \pm Ra & 1 - AGDO\ olasılığı\ ile \quad i = 1, \dots, N \quad j = 1, \dots, AGB \end{cases} \tag{2.2}$$

Yukarıdaki eşitlikte, AGDO parametresi avcı grubundan bir değer seçmenin olasılığı ve 1-AGDO pozisyon düzeltmesinin olasılığıdır. Örneğin AGDO değişkeninin 0,3 olması tasarım değişkeni değerinin % 30 olasılıkla avcı grubu değişkenlerinden seçileceğini ve % 70 olasılıkla da pozisyon düzeltmenin yapılacağını ifade etmektedir. AGDO 0,1 ve 0,4 arasında en iyi sonucu verir [67].



Şekil 2.2. (a) Sabit azalan fonksiyon, (b) Sürekli azalan fonksiyon, (c) Üstel azalan fonksiyon, (d) Belirli bir sayıdan sonra azalan fonksiyon

Ra sürekli tasarım değişkeni için rastgele yarıçap mesafesidir, Rand 1 ile 0 arasında değişen uniform bir sayıdır. Ra optimizasyon süreci boyunca azaltılabilir ya da sabitlenebilir. Ra'nın azaltılması için birçok fonksiyon seçilebilir. Bazı yararlı fonksiyonlar Şekil 2.2'de gösterilmiştir. İlk fonksiyon sabit azalan fonksiyondur. Sonuçlar sabit oranda Ra azaltımıyla elde edilir. İkinci fonksiyonda sabit Ra azaltma faktörü kullanılır. Bu seçenek araştırmaların başlangıçta daha hızlı şekilde olmasını ve yerel çözüm arama detaylarında daha uzun zaman harcamak için avcılara izin verir. Üçüncü fonksiyon benzer yaklaşım kullanır ama küresel araştırmaya doğru eğimlidir. Bu yaklaşımda bütün arama uzayının özelliklerini incelemek için avcılar daha fazla

zaman ayırır ve algoritma ömrünün sonuna doğru çözümlerde hızlı yakınsamalara uğrarlar. Dördüncü fonksiyonda, Ra belli iterasyon sayısından sonra küçülmeye başlar. Oftadeh vd. tarafından ortaya atılan av arama algoritmasında kullanılan Ra Eşitlik (2.3)'de verilmiştir. Probleme göre farklı Ra fonksiyonları belirlenebilir.

$$Ra(it) = Ra_{\min(\max(x_i)-\min(x_i))exp}\left(\frac{Ln\left(\frac{Ra_{min}}{Ra_{max}}\right)\times it}{itm}\right) \tag{2.3}$$

Burada, it iterasyon sayısı, max ve min x_i sırasıyla değişken x_i'nin mümkün olan maksimum ve minimum değerleri, Ra_{max} ve Ra_{min} avcının ilişkili arama çapının minimum ve maksimum değeri ve itm ise optimizasyon sürecindeki maksimum iterasyon sayısıdır.

Sayısal değer düzeltmede, her bir değişkenin gerçek değerini kullanmak yerine avcılar her çözüm değişkeninin sayısı tarafından birbirleri ile iletişim halindedir. Örneğin 23.4356 çözüm değişkeninde 6 tane anlamlı sayı vardır. Bu çözüm değişkeni için avcı, avcı grubu değerlendirmesi ya da pozisyon düzeltmesine dayanarak ilk hane için bir değer seçer (mesela 2) . AGDO'nun 0,3 olduğu durumda % 30 olasılıkla avcı grubundan bir sayı seçilir ve % 70 olasılıkla pozisyon düzeltmesi yapılır. Böylece, i. çözüm değişkenindeki k. sayı aşağıdaki eşitlik (2.4)'deki gibi belirlenir.

$$d_{ik}^{j'} \leftarrow \begin{cases} d_{ik}^{j'} \in \{d_{ik}^1, d_{ik}^2, \dots, d_{ik}^{AGB}\} & AGDO\ olasılığı\ ile \\ d_{ik}^{j'} = d_{ik}^1 \pm \alpha & 1 - AGDO\ olasılığı\ ile \end{cases} \quad i = 1, \dots, N \;\; j = 1, \dots, AGB \;\; k = 1, \dots, M \tag{2.4}$$

Burada a, 1 ile 9 arasında bir sayı ya da 1 ile 9 arasında üretilen rastgele değişken olabilir. Oftadeh'in çalışmasındaki sayısal örnekte a 1 alınmıştır [67].

Sayısal pozisyon düzeltmede dikkate alınması gereken bazı noktalar vardır. Grup dikkate alınarak, geçersiz(ya da geçerli olan) ve çözüm değişkeninin değerini kabul edilemez kılan bir başka rakam seçilme şansı yoktur. Örneğin, eğer 423.3423'in 2. sayısını ve 2.5678'in 2. sayısını seçerse bu geçersiz olur, bu çözüm değişkeninin değerini kabul edilemez yapar. Ayrıca pozisyon düzeltmesinde çözüm değişkeninin k. sayısı 9 ya da 0 ise ve sonra 1 sayısı ekleniyor ya da azaltılıyorsa çözüm değişkeninin değerini kabul edilemez yapar (çünkü 10 ya da -1 olur). Böyle durumlarda sayı orijinal değeri ile yer değiştirir.

Sayısal pozisyon düzeltmesinde, her bir çözüm değişkeni için belirli sayıların yeri tekrar düzenlenir. Mevcut çalışmada bütün sayısal örnekler için algoritma 8 ondalık basamağa ayarlanmıştır [67]. AA algoritması, gerçek değer düzeltmesini sayısal değer düzeltmesini ya da her ikisini de kullanabilir. Orijinal AA algoritmasında her ikisi de kullanılmıştır.

Amaç fonksiyonunu değerlendirerek yeni avcı pozisyonunun kalitesine karar verildikten sonra, avcı yeni pozisyonuna hareket eder aksi halde eski pozisyonunda kalır.

Adım 5. Avlanma grubunu yeniden organize etmek

Arama süreci devam ettikçe avcıların yerel minimumda takılmış olma ihtimalleri vardır. Bu durumda avcıların optimum noktayı bulmaları için kendilerini yeniden organize etmeleri gerekir. Avlanma grubunun yeniden organizasyonu iki şekilde gerçekleştirilir. Eğer gruptaki en kötü avcı ve liderin amaç fonksiyonu değeri önceden belirlenen ϵ_1 değerinden daha küçük olursa ve durdurma kriteri sağlanamazsa her bir avcı için algoritma avcı grubunu yeniden organize eder. Alternatif olarak belli sayıda arama yapıldıktan sonra da yeniden organizasyon gerçekleştirilir. Avcı grubu yerel minimum noktada takılınca veya arama belli bir sayıya ulaşınca eşitlik (2.5) kullanılarak yeniden organizasyon yapılır. Yeniden organizasyonda lider mevcut pozisyonunda kalır ve diğer avcılar arama uzayında liderin pozisyonuna göre rastgele seçilirler.

$$x_i' = x_i^L \neq rand \times (\max(x_i) - \min(x_i)) \times \alpha \exp(-\beta \times ES) \quad (2.5)$$

Burada x_i^L i. değişken için liderin pozisyon değeridir. Rand 0 ile 1 arasında düzgün dağılımlı rastgele bir sayıdır. $\max(x_i)$ ve $\min(x_i)$, x_i değişkeninin maksimum ve minimum değerleridir. ES, epok olarak ifade edilen yeniden organizasyon sayısıdır. Algoritma ilerlediğinde çözüm adım adım optimum noktaya doğru yakınsar. α ve β değerleri pozitif gerçek değerlerdir. Bunlar algoritmanın küresel yakınsama oranına karar verir. α'nın büyük değerleri ve β'nın küçük değerleri algoritmanın yavaş bir şekilde yakınsamasına sebep olur. Bu değerler, büyük boyutlu optimizasyon problemleri için ya da birkaç yerel optimuma sahip problemler için tavsiye edilir. Bunun aksine küçük α değerli ile büyük β değerleri algoritmanın yakınsamasını hızlandırır. Bu ise, az sayıdaki tasarım değişkenleri ile küçük optimizasyon problemleri için önerilir.

Pratik olarak yeniden organizasyonun dört avantajı vardır. İlk olarak tasarım uzayını aramak için yeni bir fırsat doğar. İkinci olarak lider kendi pozisyonunu koruduğu için, diğer avcıların yeniden organize edilmesinden sonra daha iyi bölgelere doğru yön verilir. Böylece bütün tasarım uzayı gereksiz yere aranmaz. Her yeniden organizasyonda, lider kendi pozisyonunu koruduğu için avcılar arama süreci boyunca en iyi çözümü kaybetmezler. Son olarak ve en önemlisi ES arama süreci boyunca arttıkça avcılar optimum noktayı bulmak için tasarım uzayında daha fazla yerel arama yaparlar. Algoritma ilerlediğinde çözüm yavaş yavaş optimum noktaya doğru yaklaşır.

Adım 6. Durdurma kriteri sağlanana kadar adım 3- 5'i tekrarlamak.

Adım 6'da durdurma kriteri sağlanınca hesaplama durdurulur. Eğer sağlanmazsa adım 3-5 tekrarlanır. Durdurma kriteri maksimum arama sayısı olarak tanımlanabilir. Alternatif olarak, eğer lidere göre fonksiyon yeniden organize edildiğinde ya da en kötü avcı ve liderin amaç fonksiyonu değeri önceden belirlenen $\in$ değerinden daha küçük olduğunda arama süreci biter.

Av arama algoritması önceki optimizasyon algoritmalarıyla karşılaştırıldığında, bu yeni algoritmanın etkin ve sağlam sonuçlar verdiği görülmüştür [67]. Av arama algoritmasında kullanılan test örnekleri, çeşitli optimizasyon problemlerine uygulanabilen global arama problemleridir. Av arama algoritması kullanılarak elde edilen sonuçlar diğer algoritmalardan daha iyi sonuçlar vermiştir [68]. Buna ek olarak diğer optimizasyon problemleri sınırlandırılmıştır. Ancak av arama algoritmasında avcılar tasarım uzayına çıkabilir ve yasak bölgeleri araştırabilir böylelikle daha uygun alanlar gerçek hayattaki gibi bulunabilir. Ayrıca bu yeni algoritma kesikli ve kombinatoryal optimizasyon problemlerine algoritma üzerinde gerekli değişiklikler yapılarak uygulanabilir.

3. BÖLÜM

ZAMAN PENCERELİ ARAÇ ROTALAMA PROBLEMİ

Zaman Pencereli Araç Rotalama Problemi, depodan hareket eden araçların müşterileri belli zaman aralığı içinde ziyaret etme zorunluluğu olan (zaman penceresi kısıtı) özel bir araç rotalama problemi türüdür. Yapısı nedeniyle okul otobüsü 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 daha uyumludur. Bu nedenlerden dolayı diğer araç rotalama problemlerine göre hakkında daha çok araştırma yapılmıştır [69].

ZPARP, bilgi teknolojisindeki gelişmeler sonucunda firmaların verimlilik ve zaman odaklı tedarik zincirine yönelmeleri ile tedarik zinciri için vazgeçilmez bir araç haline gelmiştir. Ayrıca ZPARP ile sezgisellerin tasarımı ve optimum yaklaşımların geliştirilmesi için önemli ilerlemeler sağlanmaktadır.

Literatürde zaman pencereli ARP'nin, kapasite kısıtlı ARP'ye göre daha karmaşık olduğu ifade edilmektedir. Bu yüzden zaman pencereli ARP'lere daha çok sezgisel ve meta-sezgisel yöntemler ile çözümler aranmaktadır.

ZPARP, belirli bir ceza maliyeti ile müşterilere ilgili zaman pencerelerinin dışında hizmet verilebilmesine olanak sağlayan Esnek ZPARP ve müşterilere söz konusu zaman aralığının dışında hizmet verilememesi durumunu söz konusu olduğu Sıkı ZPARP olarak ikiye ayrılmaktadır. Tez çalışmamızda SZPARP ele alınmıştır.

3.1. Sıkı Zaman Pencereli Araç Rotalama Problemi Matematiksel Formülasyonu

Sıkı zaman pencereli ARP algoritmasında, genel ARP algoritmasından farklı olarak zaman kısıtları eklenmiştir. Model aşağıdaki gibidir;

$$x_{ijk} = \begin{cases} 1, eğer\ k\ nolu\ araç\ i\ müşterisinden\ j\ müşterine\ gidiyorsa \\ 0, diğer\ durumda \end{cases}$$

$c_{ij} = i$ müşterisi ile j müşterisi arasındaki mesafe $(maliyet)$

$Q = bir\ araca$ yüklenebilecek toplam ürün miktarı

$q_i = i$ müşterisinin talep miktarı

$y_i = alt\ turları\ engellemek\ için\ kullanılan\ rastgele\ değişken$

$b_j^k: k$ aracının j müşterisine servise başlama zamanı

$t_{ij}: i$ müşterisine servis zamanı

$e_i: i$ müşterisine en erkan varış zamanı

$l_i: i$ müşterisine en geç varış zamanı

$$Min\ Z = \sum_{i=0}^{N} \sum_{j=0}^{N} (c_{ij} \sum_{k=1}^{M} x_{ijk}) \tag{3.1}$$

$$\sum_{i=0}^{N} \sum_{k=1}^{M} x_{ijk} = 1, \quad j = 1, \dots N \tag{3.2}$$

$$\sum_{i=0}^{N} x_{ipk} - \sum_{j=0}^{N} x_{pjk} = 0, \quad k = 1, \dots, M \quad p = 0, \dots, N \tag{3.3}$$

$$\sum_{i=1}^{N} (q_i \sum_{j=0}^{N} x_{ijk}) \leq Q, \quad k = 1, \dots, M \tag{3.4}$$

$$\sum_{j=1}^{N} x_{0jk} = 1, \quad k = 1, \dots, M \tag{3.5}$$

$$x_{ij}^k \left(b_i^k + t_{ij}\right) \leq b_j^k \quad k = 1, \dots, M \tag{3.6}$$

$$e_i \leq b_j^k \leq l_i \tag{3.7}$$

$$x_{ijk} \in \{0,1\}, \forall, i, j, k \quad (3.8)$$

Burada eklenen (3.6) kısıtı, k aracının i müşterisinden j müşterisine $b_i^k + t_{ij}$ süresi dolmadan geçmemesini sağlar, (3.7) kısıtı ise müşteriye istenilen zaman aralığında servisin yapılmasını sağlar [70].

3.2. ZPARP Literatürü

ZPARP deterministik–tam çözüm algoritmaları ile çözülebilmekle beraber, NP-zor türünde bir problem olması nedeniyle bu çözümler uzun işlem zamanı gerektirmektedir [71]. Bu nedenle yerel arama algoritmaları ve/veya meta-sezgiselleri kullanarak yaklaşık çözüm arayan birçok çalışma literatürde mevcuttur. Bu üç türde yapılan araştırmalar izleyen bölümlerde bahsedilmektedir.

3.2.1. Deterministik – Tam Çözüm Algoritmaları

Deterministik-tam çözüm algoritmaları ZPARP için optimum çözüm bulabilen yöntemlerdir. Ancak çözüm süresi ve bilgisayar kapasiteleri göz önüne alındığında büyük boyutlu problemlerde yetersiz kalmaktadırlar.

Bu konulardaki ilk çalışmalardan birini Desrosiers v.d. [72] m-GSP probleminin zaman pencere kısıtlı durumu için çalışmışlardır. Çözümlerinde simpleks ve dal-sınır algoritması kullanmışlardır. Küçük boyutlu problemlerde optimal sonuca ulaşmışlardır. Desrochers v.d. [73], küme kaplama formülasyonu için sütun yaratma yaklaşımı kullanan bir optimizasyon algoritması geliştirmişler. Bu algoritma literatürde belirtilmiş optimal algoritmalara göre 6 kat daha büyük problemleri optimal olarak çözme yeteneğine sahiptir. Kallehauge v.d. 2001 [74] yılında yaptıkları çalışmalarında, ZPARP ile bağlantılı diferansiyellenemeyen optimizasyon yönteminin uygulama sonuçlarını ele almışlardır. ZPARP'nin en kısa yol ayrışması, lagranj çarpanlarının optimal bulunmasını gerektirmektedir. Bu problem konveks ve diferansiyellenemez bir problemdir. Optimal çarpanları bulmak için, bir güvenilir-bölge dengeleyici araçlı kesme-düzlemi algoritmasının geliştirilmesi amaçlanmıştır. Bu algoritma dal-sınır şemada tam sayılı çözümü bulmak için kullanılan Dantzig-Wolfe algoritması ile birleştirilmiştir. Bu yaklaşım Solomon' un örnek problemleriyle test edilmiştir. 14 adet optimal çözülmemiş

büyük boyutlu problemlerde sonuca ulaşılmıştır. Çözüm zamanları dikkate değer şekilde azalmıştır.

3.2.2. Yerel Arama Sezgiselleri

Yerel arama sezgiselleri, GSP için geliştirilen algoritmaların ZPARP' ne uyarlanması ile gerçekleştirilmiştir. Bu çalışmalardan, [75] çalışmasında, literatürdeki beş sezgisel yaklaşım ZPARP'ye uygulamıştır. Bu algoritmalar daha önce GSP için geliştirilmiş tasarruf sezgiseli (savings heuristics), zaman yönelimli en yakın komşuluk sezgiseli (time-oriented, nearest-neighbor heuristic), ekleme sezgiseli (insertion heuristics), zaman yönelimli tarama sezgiselidir (time-oriented sweep heuristic). Çalışmanın sonunda ekleme tipi sezgisellerin daha iyi sonuç verdiği anlaşılmıştır. Cordone ve Calvo'nun [76], ZPARP için geliştirdikleri model, üç basit prosedürün bir kombinasyonudur. Bunlardan klasik k-opt değişimleri çözümü iyileştirmekte, ad hoc (özel amaç) prosedürü araç sayısını azaltmakta ve ikinci bir amaç fonksiyonu da aramayı yerel optimumdan uzaklaştırmayı amaçlamaktadır. Braysy [77] tarafından ZPARP çözümü için deterministik yerel arama yöntemleri sunulmuştur. Yeni algoritma üç aşamalı olup, birinci aşamada iki tür kurma algoritmasından biri ile başlangıç çözümü elde edilmekte, ikinci aşamada zincir atma (ejection chain) tabanlı yerel algoritma ile rota sayıları azaltılmakta, son aşamada ise or-opt değişimleri ile rotalar kısaltılmaya çalışılmaktadır. Deney sonuçları bu algoritmanın diğer çalışmalara göre hem rekabetçi, hem de daha hızlı olduğunu göstermiştir.

3.2.3. Meta-Sezgiseller

Yerel arama algoritmalarının yerel minimumda takılmasını engellemek ve global optimuma daha yakın sonuç bulmak amacıyla meta-sezgiseller ZPARP' de de çok sık kullanılmıştır. Kullanılan algoritmalara tabu arama, diferansiyel gelişim, parçacık sürü optimizasyonu, tavlama benzetimi, genetik ve karınca koloni optimizasyonu algoritmaları örnek verilebilir.

Zaman pencereli araç rotalama problemlerine meta-sezgisellerin uygulandığı ulaşılabilen ilk algoritma Thangiah v.d.'ne [78] aittir. Thangiah v.d., ZPARP için bir genetik algoritma geliştirmiştir. GIDEON adlı bu genetik algoritmayı Solomon [75]

örnek problemlerine uygulayarak literatürdeki çözümlerden daha iyi sonuçlar bulmuşlardır. Czech ve Czarnas [41], paralel benzetim tavlaması algoritmasını ZPARP' ye uygulamışlardır. Çalışmada, süreçlerin işbirliği yöntemine dayanan tavlama benzetimi algoritması ile teslimat soruna en iyi çözüm aranmıştır. Paralellik kullanılarak en iyi çözümler elde edilmesi amaçlamıştır. Solomon [75] örnek problemleriyle yapılan denemelerde bilinen çözümlere göre çok iyi sonuçlar alınmıştır. Tan v.d. 2005 [79] yılındaki çalışmalarında ZPARP'ni, karınca koloni sistemi ile çözümlemişlerdir. Algoritmalarının temel fikri, optimum çözümü elde etmek için iki ayrı karınca kolonisi kullanmak olmuştur. Algoritma iki aşamada uygulanmış olup, birinci adımda karınca kolonisi mümkün olduğunca az araç kullanarak çözüm bulmakta, ikini aşamada ise karıncalar başka kolonilerle ZPARP'de en az taşıma süresini elde etmek için işbirliği yapmaktadır. ZPARP çözerken iki kolonin iş birliği yapması çok iyi sonuçlar elde edilmesini sağlamıştır. Bu yöntemle elde edilen sonuçlar diğer yöntemlere ait sonuçlarla karşılaştırıldığında, karınca koloni sisteminin ZPARP için en iyi çözümleri bulmayı sağladığı görülmüştür. Wang v.d.'nin 2008 [80] yılındaki çalışmalarında araç sayısının sınırlı sayıda olduğu ZPARP için yine Genetik algoritma uygulanmıştır. Çözüm gösterimi olarak talep noktasının hizmet göreceği araç ve hangi sırada göreceğini belirten sıra numarasının olduğu bir kodlama kullanılmıştır. Berger ve Barkaoui [81] ZPARP için iki melez genetik algoritmanın birlikte çalıştığı bir model geliştirmişlerdir. Aynı anda çalışan GA'lar farklı amaç fonksiyonlarına sahiptir; ilk algoritmada toplam mesafe en azlanmaya çalışılırken ikincisinde ise olurlu çözüm elde etmek için kısıt ihlali en azlanmaya çalışılır. Önerilen bu algoritma sonuçlarının, en iyi bilinen sonuçlarla rekabet edebilecek çözümler verdiği kanıtlanmıştır. Alvarenga v.d. [82]'de genetik algoritma ve küme bölme yaklaşımını bir arada kullandıkları çalışmalarında ZPARP için toplam mesafenin en küçüklenmesini temel amaç alan melez bir sezgisel model ortaya koymuşlardır. Genetik algoritma kullanarak birbirinden bağımsız koşumlar gerçekleştirilerek çözüm kümeleri elde edilmiş, daha sonra da bu çözüm kümelerinin kombinasyonları oluşturularak, küme bölme için en iyi kalitede çözüm elde edilmesi için rota kümesi elde edilmiştir. Çalışma, literatürdeki en iyi bilinen sonuçlarla kıyaslandığında, minimal hareket mesafesi açısından en iyi sonuçların elde edildiğini göstermiştir. Nazif ve Lee'nin [83] çalışmalarında yine genetik algoritma, optimize çaprazlama operatörü (OCGA) ile ZPARP'ye uygulanmıştır. Çalışmada optimize çaprazlama operatörü kullanılarak dizayn edilen yönsüz ikili grafiklerle, tüm kısıtları

sağlayan ve en iyi toplam maliyeti veren rotalar önerilmiştir. Ayrıca algoritmaya çeşitli tekniklerle yapılan eklemeler sonucunda çözüm kalitesi artırılmaya çalışılmıştır. OCGA Solomon'un [75] 100 müşterili problem setine uygulanmış ve toplam seyahat mesafesi bakımından oldukça iyi sonuçlar elde edilmiştir. Bent ve Hentenryck'in 2006 [84] yılındaki çalışmasında eş zamanlı topla-dağıt araç rotalama problemi zaman penceresi kısıtı ile iki aşamalı melez bir algoritma ile çözümlenmiştir. İlk aşamada toplam yol sayısını azaltmak için probleme tavlama benzetimi algoritması uygulanmıştır. İkinci aşama da ise geniş komşuluk arama yerel sezgiseli ile toplam seyahat maliyetinin en aza indirilmesi amaçlanmıştır. Elde edilen sonuçlar 100, 200 ve 600 müşteri ile ilgili problemlerde oldukça iyi sonuçlar elde edildiğini göstermiştir. Gambardella v.d. [85], ZPARP için uyguladıkları çoklu karınca koloni algoritması konulu çalışmalarında, hiyerarşik yapay karınca kolonileri oluşturarak birden çok amaç fonksiyonunu optimize etmeye çalışmışlardır. Böylece ilk koloni araç sayısını minimize ederken, ikinci koloni toplam seyahat mesafesini minimize etmektedir. Çözüm sonuçları, algoritmanın çözüm kalitesi ve hesaplama süresi açısından iyi bilinen mevcut yöntemleri ile rekabet edebilecek düzeyde olduğunu göstermiştir. Ayrıca literatürdeki problemler için bilinen en iyi çözümlerinde elde edilmesini sağlamıştır. Erbao v.d. [86], eş zamanlı topla-dağıt ve zaman pencereli ARP için melez diferansiyel gelişim ve genetik algoritma ile yaptıkları çalışmada, ilk olarak desimal kodlama kromozom yapısını kullanmışlardır. Daha sonra mutasyon operatörü, çeşitli genetik algoritma teknikleri ve diferansiyel gelişim algoritmasıyla problemin ana optimizasyon çizelgesi sonuçları geliştirilmiştir. Diferansiyel gelişim ve genetik algoritmadan oluşan bu melez algoritma, büyük boyutlu lineer olmayan problemlerin çözümünde oldukça etkilidir. Sayısal sonuçlar bu melezalgoritma ile yalnızca genetik algoritma ya da diferansiyel gelişim algoritması uygulanarak elde edilen sonuçlardan daha iyi sonuçlar alındığını göstermiştir. Ye'nin 2011 [87] yılındaki çalışmasında, ZPARP parçacık sürü optimizasyon algoritması ile çözümlenmiştir. Çalışmada, farklı parçacıklar belirli görevlere atanır. Buna göre, iyi parçacıklar için küçük eylemsizlik ağırlıkları verilirken, büyük parçacıklar için büyük eylemsizlik ağırlıkları verilir. Sonrasında parçaların atalet ağırlıklarına göre uygunluk fonksiyonu hesaplanır. Karşılaştırmalı sonuçlar, geliştirilmiş parçacık sürüsü algoritmasının ZPARP için oldukça etkili olduğunu göstermiştir. Nasir ve Masrom'un [88] çalışmalarında ZPARP'ye meta-sezgisel algoritmalar kullanılarak çözüm aranmıştır. Çalışmada genetik algoritma ve parçacık sürü optimizasyon algoritması

seçilmiştir. Sonuçlar kümelenmiş tabanlı müşteri dağılımı açısından parçacık sürüsü optimizasyon algoritmasının genetik algoritmadan daha iyi sonuçlar verdiğini göstermiştir. Diğer taraftan rastgele müşteri dağılımlarında da genetik algoritma, parçacık sürüsü optimizasyon algoritmasından daha üstündür. İşlem süresi açısından da parçacık sürüsü optimizasyon algoritmasının performansı, genetik algoritmanınkinden daha iyidir.

4. BÖLÜM
AV ARAMA ALGORİTMASI İLE SZPARP'NİN ÇÖZÜMÜ

4.1. Problem Tanımı

Tez çalışmasında sıkı zaman pencereli araç rotalama problemine yeni geliştirilmiş olan av arama algoritması ile çözüm aranmıştır. Uygulama verisi olarak, literatürde Solomon verileri olarak yer alan ve Solomon'un 1967 yılında yaptığı çalışmada belirttiği verilerden yararlanılmıştır. Solomon problemleri toplam 56 adettir ve bu problemlerin hepsi ZPARP olup, çeşitli parametreleri ile birbirlerinden ayrılmaktadır. Bu çalışmada Solomon problemlerinden C101, C102, C103, C104, C105, C201, C202, C203, C204, C205, R101, R102, R103, R104, R105, RC101, RC102, RC103, RC104 ve RC105, 25 müşteri sayısı ile ele alınmıştır.

Problemlerde bir depo ve 25 müşteri bulunmaktadır. Müşteriler ve depoların koordinatları belirli ve sabittir. Depo ve müşteriler arası uzaklık öklit uzaklık ile hesaplanmaktadır. Sıkı zaman pencereli araç rotalama probleminde, müşterilere servis yapılabilecek zaman aralığı kısıtı ve araçların kapasite kısıtı sağlanacak şekilde en kısa toplam yol ile tüm müşterilere hizmet verilmelidir. Av arama algoritması ile kaç adet araç (rota) gerektiği ve her bir rotada hangi müşterilerin, hangi sırada yer alacağının cevabı aranmaktadır.

SZPARP için önerilen algoritmada *0* depoyu ve $1 \leq i \leq n$ için *i* müşteriyi göstermek üzere yönsüz grafı ifade etmektedir. Ele alınan sabit ve değişkenler ise şunlardır;

- c_{ij}: (i,j) kenarına ait maliyet ($c_{ij} = c_{ji}$)
- a_i: aracın i müşterisine ulaşma zamanı
- t_i: i müşterisine hizmet zamanı
- q_i: i müşterisinin talebi

- e_i: i müşterisine en erken varış zamanı
- l_i: i müşterisine en geç varış zamanı
- L: her bir aracın maksimum işlem zamanı
- Q: araç kapasitesi
- R_i: bir rotaya ait sıra ($i_1, i_2, \dots, i_k$; $i_j \in k$)
- W_{ij}: i aracının j müşterisinde bekleme zamanı

R_i $(i_1, i_2, \dots, i_k)$ rotası için ulaşım zamanı a_{ij} şu şekilde ifade edilir (i_{j-1} müşterisine varış zamanı, artı i_{j-1} müşterisinde hizmet zamanı ve i_{j-1} müşterisinden i_j müşterisine seyahat zamanı);

$$a_{ij} = a_{ij-1} + t_{j-1} + c_{ij-1,ij} \tag{4.1}$$

En erken varış zamanı kısıtını sağlamak için (4.2) eşitliği şöyle düzenlenebilir;

$$a_{ij} = \max\{e_{ij}, a_{ij-1} + t_{ij-1} + c_{ij-1,ij}\} \tag{4.2}$$

i aracının j müşterisinde bekleme zamanı (4.3) eşitliğiyle ifade edilir;

$$W_{ij} = \max\{0, e_{ij} - a_{ij-1} - t_{ij-1} - c_{ij-1,ij}\} \tag{4.3}$$

Aracın işlem zamanı kısıtı ise aşağıdaki eşitlikle sağlanır;

$$a_{ik} + t_{ik} + c_{ik,0} \leq L \tag{4.4}$$

R_i rotası için toplam seyahat zamanı (4.5) eşitliği ile tanımlanır;

$$C(R_i) = c_{0,i1} + \sum_{j=2}^{k}\left(c_{ij-1,ij}\right) + \sum_{j=1}^{k} W_{ij} + \sum_{j=1}^{k} t_{ij} + c_{ik,0} \tag{4.5}$$

Bir S çözümü, bir müşterinin sadece bir rotaya atandığı $R_1, R_2, \dots, R_l$ rotalarının birleşimidir. SZPARP'de optimizasyon problemi (4.6) eşitliği ile formüle edilir;

$$min_s C(S) = \sum_{i=1}^{l} C(R_i) \tag{4.6}$$

4.2. Dizi Gösterimi

Dizi gösterimi, meta-sezgisellerin performansını etkileyen etmenlerden biridir. Bu nedenle öncelikle problem yapısına uygun, etkin bir dizi gösterimine karar vermek gerekir.

SZPARP için önerilen av arama algoritmasında rassal anahtarlama ve permütasyon kodlamaya dayanan bir yapı kullanılmıştır. Öncelikle her müşteriyi içeren tam bir tur elde edilmiştir. Bu amaçla her bir müşteriye, Bean'in 1994 yılında geliştirmiş olduğu rassal anahtarlamaya dayanarak 0 ile 1 arasında rassal bir sayı atanmış ve bu sayılar küçükten büyüğe doğru sıralanmıştır. Bu sıralamanın indis değerleri, müşterilerin ziyaret sırası olarak elde edilmiştir. Tablo 4.1'de 10 müşteri için örnek bir sıralama yer almaktadır.

Tablo 4.1. Müşteri sırası oluşturma

1	2	3	4	5	6	7	8	9	10
0,123	0,052	0,368	0,458	0,222	0,897	0,321	0,111	0,852	0,732
Sıralama: 2-8-1-5-7-3-4-10-9-6									

Müşterilerin sıralanmasından oluşan tam bir tur elde edildikten sonra, Prins'in 2004 [89] yılında ARP için ortaya koyduğu ayırma algoritması SZPARP için uyarlanarak tam tur, kapasite kısıtına uygun alt turlara ayrılmıştır.

4.2.1. Prins'in Ayırma Algoritması

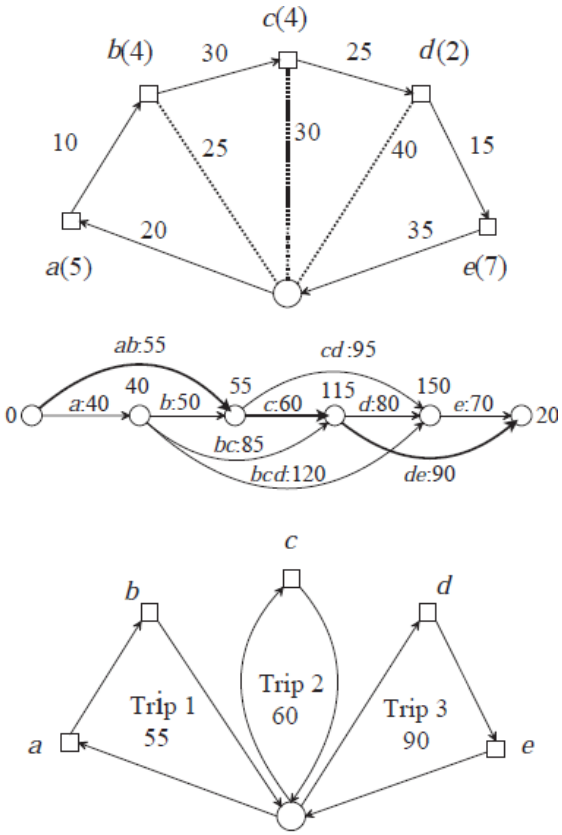
Prins [89], bir dizinin birçok farklı rotaya ayrılabileceği düşüncesinden hareketle, toplam maliyeti minimize edecek, en iyi ayırmayı bulmakta bir algoritma geliştirmiştir. Algoritmanın temel mantığı şu şekildedir:

S=(1,2,…,n) bir çözümü göstersin. $\bar{V} = \{1,2,\dots,n\}$ olmak üzere, $H = (\bar{V}, \bar{E})$ yardımcı grafı ele alınsın. Problemdeki müşteri sayısı *N* olmak üzere, bu graftaki düğüm sayısı N+1'dir. Düğümler kümesindeki ilk düğüm depoyu temsil eden aylak düğümdür ve "0" ile gösterilir; diğerleri ise 1 ile N arasındaki sayılardan oluşan ve müşterileri gösteren düğümlerdir. *(i,j)* arkı,

$$\overline{E_{ij}} = c_{0,i+1} + \sum_{k=i+1}^{j-1}\left(t_k + C_{k,k+1}\right) + t_j + c_{j,0} \leq L: \sum_{k=i+1}^{j} q_k \leq Q \quad (4.7)$$

eşitliği sağlanıyorsa $\bar{E}$'nin elemanıdır. Bu durumda, $\overline{E_{ij}}$ *(i+1, i+2, …, j)* rotası için toplam ulaşım maliyetidir. *S* için optimal ayrım, *H*'da 0 köşesinden n köşesine en kısa yol *P*'ye karşılık gelir.

Aşağıdaki örnek [89], kurulumu anlatmaktadır. Şekildeki ilk graf S=(a,b,c,d,e) sırasını göstermektedir. Her bir kenardaki sayı, ulaşım zamanıdır. Servis süresi 0 varsayılmakta ve talepler sırasıyla 5, 4, 4, 2 ve 7 olarak verilmektedir. Aracın kapasitesi 10 ve maksimum işlem zamanı 90'dır. Bu durumda yardımcı graf *H*, şekildeki 2. graf olacaktır. *H*'daki 55 ağırlıklı kenar (0,a,b,0) yolunun ulaşım zamanını ifade etmektedir. Diğer kenarlarla ilgili ağırlıklar benzer şekilde ifade edilmiştir. 0'dan e'ye en kısa yol, minimum toplam ulaşım zamanı 205 ile (0,b,c,e)'dir. Son graf, üç rota ile optimal sonucu göstermektedir.



Şekil 4.1. Çözüm değerlendirmeye örnek

Yardımcı graf, verilen *S* kromozomunu optimal rotalara nasıl ayıracağımızı anlamamıza yardımcı olur. Fakat gerçekte, bu tip bir grafı oluşturmamıza gerek yoktur. Bu işlem bir

etiketleme algoritması ve bir ayırma prosedürü ile yapılabilir [89]. *S=(1,2,….,n)* bir diziyi göstermek üzere, S'deki her *j* köşesi için V_j ve P_j değerleri hesaplanır. V_j , *H*'daki 0 düğümünden j düğümüne en kısa yolun maliyeti ve P_j ise *j*'nin bu yoldaki öncülüdür. Minimum maliyet V_n şeklinde elde edilir. Ele alınan herhangi bir *i* için, *L* veya *Q* aşıldığında *j* artışı son bulur. Prins'in etiketleme algoritması şöyledir;

Tablo 4.2. Prins'in etiketleme algoritması

$V_0=0$
Tüm i'ler için (i=1,2,…,n) $V_i=+\infty$
Tüm i'ler için yap
  Maliyet=0, yük=0, j=i
  Tekrarla
    $Yük=yük+q_{sj}$
    Eğer i=j ise maliyet=$c_{0,sj}+t_{sj}+c_{sj,0}$
    Aksi halde maliyet=maliyet-$c_{sj-1,0}+c_{sj-1,sj}+t_{sj}+c_{sj,0}$
    Eğer (maliyet≤L) ve (yük<Q) ise
      Eğer V_{i-1}*+maliyet<*V_j *ise* $V_j=V_{i-1}$*+Maliyet,* $P_j=i-1$
      $j=j+1$
  (j>n) veya (maliyet>L) veya (yük>Q) olana kadar

4.2.2. SZPARP İçin Uyarlama

SZPARP için (4.7) eşitliğinde verilen *(i,j)* arkı için toplam seyahat zamanı, en erken ve en geç varış zamanı kısıtları dikkate alınarak eşitlik (4.8) şeklinde düzenlenir:

$$\overline{E_{ıJ}} = c_{0,i+1} + \sum_{k=i+1}^{j-1} \left(W_i + t_k + C_{k,k+1}\right) + W_j + t_j + c_{j,0} \leq L$$

$$: \sum_{k=i+1}^{j} q_k \leq Q, \quad \overline{E_{ıJ}} - c_{j,0} - t_j \leq l_j \tag{4.8}$$

S için optimal ayrım, *H*'da 0 köşesinden n köşesine en kısa yol *P*'ye karşılık gelir. Bu değişim sonucu ayırma algoritması Tablo 4.3 şeklinde düzenlenir.

Tablo 4.3. SZPARP için etiketleme algoritması

$V_0=0$
Tüm i'ler için (i=1,2,…,n) $V_i=+\infty$
Tüm i'ler için yap
  Maliyet=0, yük=0, j=i
  Tekrarla
    $Yük=yük+q_{sj}$
    Eğer i=j ise $maliyet=maks(c_{0,sj}, e_{sj})+t_{sj}+c_{sj,0}$
    Aksi halde $maliyet=maks(maliyet-c_{sj-1,0}+c_{sj-1,sj}, e_{sj})+t_{sj}+c_{sj,0}$
    Eğer $(maliyet \leq L)$ *ve* $(yük<Q)$ *ve* $(maliyet-c_{sj,0}-t_{sj} \leq l_j)$ *ise*
      Eğer $V_{i-1}+maliyet<V_j$ *ise* $V_j=V_{i-1}+Maliyet$, $P_j=i-1$
      $j=j+1$
  $(j>n)$ *veya* $(maliyet>L)$ *veya* $(yük>Q)$ *veya* $(maliyet-c_{sj,0}-t_{sj}>l_j)$ *olana kadar*
Tüm i'ler için $rota(i)=\emptyset$, $t=0$, $j=N$
Tekrarla
  $t=t+1$, $i=P_j$
  K=i+1'den j'ye kadar $rota(t)=rota(t) \cup \{t_k\}$
  $j=i$
i=0 olana kadar

Algoritmanın Tekrarla döngüsünde tüm uygun alt rotalar incelenmektedir. İlgilenilen bir *(i,j)* hattı ile j düğümüne daha az bir maliyetle ulaşılabiliniyorsa; V_j'ye ilgili maliyet değeri, P_j'ye ise bu maliyeti veren rotayı tutacak şekilde i değeri atanmaktadır. Bu şekilde her döngüde; mevcut rotaya sıradaki t_j müşterisi eklenerek veya rota boyunca taşınan yük miktarının araç kapasitesini geçmesi halinde veya son tarih kısıtlarının aşılması durumunda, yeni rota açılarak *H* üzerindeki tüm uygun rotalar gözden geçirilmektedir. Algoritmada elde edilen son V_N değeri *s* çözümünün amaç fonksiyonu değerini vermektedir. Bu maliyete sahip çözümün rotaları ise ikinci tekrarla döngüsü ile *P* vektöründen yararlanılarak bulunmaktadır.

4.3. SZPARP için Önerilen AA Algoritmasının Temel İşleyişi

Başlangıç Avcı Grubunun Oluşturulması: Avlanma grup boyutu büyüklüğünde avcı grubu, başlangıçta rastgele oluşturulur. Bu amaçla bölüm 4.2'de bahsedilen rassal anahtarlama ve ayırma algoritması, oluşturulan ve 0-1 arası rassal değerler içeren her bir avcıya (diziye) uygulanarak rotalar belirlenir. Avcıların maliyetleri hesaplanarak en düşük maliyete sahip avcı lider olarak atanır.

Lider Arkasından Hareket: Avcıların yeni pozisyonları lider peşinden hareket ederek belirlenir. Bu amaçla Bölüm 3'de verilen aşağıdaki fonksiyondan yararlanılır. Burada x_i^L , liderin i. değişken için pozisyon değeridir.

$$x_i' = x_i + rand \times LMH \times (x_i^L - x_i) \tag{4.9}$$

(4.9) fonksiyonu sonucu değerler reel çıkacağı için yine her bir avcıya rassal anahtarlama ve ayrıştırma algoritması uygulanır. Maliyet hesaplanarak, lider arkasından hareketin avantajlı olup olmadığına bakılır. Eğer lider arkasından hareket, maliyette azalış sağlıyorsa avcı yeni pozisyona geçer, aksi halde eski pozisyonunda kalır.

Pozisyon Düzeltme: Avcılar lider arkasından hareket ettikten sonra, daha iyi pozisyonlar bulmak için pozisyon düzeltme işlemi gerçekleştirirler. Bu amaçla avlanma grup değerlendirme oranına göre gerçek değer düzeltme uygulanır. Bu işlem avcıların işbirliği yaptığı grup değerlendirme içeren yerel bir aramadır. Yeni avcı pozisyonları eşitlik (4.10)'a göre oluşturulur.

$$x'' = \begin{cases} x_i'' \in \{x_i^1, x_i^2, \dots, x_i^{AGB}\} & AGDO\ olasılığı\ ile \\ x_i'' = x_i' \pm Ra_{min}(\max(x_i) - \min(x_i))\exp\left(\frac{Ln\left(\frac{Ra_{min}}{Ra_{max}}\right) \times it.no}{ES \times EIS}\right) & (1 - AGDO) olasılığı\ ile \end{cases}$$

(4.10)

Buna göre yeni değer, avlanma gurup değerlendirme oranı olasılığı ile avcı grubundan *(1-AGDO)* olasılığı ile pozisyon düzeltmeden elde edilir. *Ra* sürekli tasarım değişkeni için rastgele yarıçap mesafedir ve azaltılmasında Eşitlik (2.3)'de verilen üstel fonksiyon kullanılmıştır. Eşitlik (4.10)'da Ra_{min} ve Ra_{max} avcının ilişkili arama çapının sırasıyla minimum ve maksimum değerlerini, *it.no* o andaki iterasyon sayısını, $max(x_i)$ ve $min(x_i)$ değişken x_i'nin sırasıyla mümkün maksimum ve minimum değerlerini $ES \times EIS$ ise maksimum iterasyon sayısını göstermektedir. Her bir avcının pozisyon düzeltme sonucu yeni maliyeti hesaplanarak eski konumunun maliyeti ile karşılaştırılır. Eğer daha düşük bir maliyet elde ediliyorsa avcı yeni pozisyona ilerler aksi halde eski pozisyonunda kalır.

Yeniden Organizasyon: Avcıların yerel optimuma takılı kalmalarını engellemek için avlanma grubu yeniden organize edilir. Bu amaçla EIS kadar arama yapıldıktan sonra

avcılar yeniden organize olur. Yeniden organizasyon sırasında liderin konumu sabit kalır ve diğer avcılar konumlarını Eşitlik (4.11) ile yeniden düzenlerler.

$$x_i''' = x_i^L \pm rand(10) \times (\max(x_i) - \min(x_i)) \times \alpha \exp(-\beta \times ES) \quad (4.11)$$

Burada x_i^L liderin i. değişken için pozisyon değeri, $max(x_i)$ ve $min(x_i)$, x_i değişkeni için olası maksimum ve minimum değerlerdir. α ve β değerleri pozitif gerçek değerlerdir. Bunlar algoritmanın küresel yakınsama oranına karar verir.

Durdurma Koşulu: Algoritmada epok sayısı kadar yeniden organizasyon gerçekleştirilir. Her bir epokta iterasyon sayısı kadar arama yapıldıktan sonra ve epok sayısı tamamlandıktan sonra algoritma sonlandırılır. Dolayısıyla algoritma epok sayısı (ES) ile iterasyon sayısının (EIS) çarpımı kadar (ES×EIS) çalıştırıldıktan sonra durdurulur.

Özet olarak önerilen algoritmanın adımları Tablo 4.4'de ve Şekil 4.2'de verilmektedir.

Tablo 4.4. SZPARP için önerilen AA algoritması

Adım 1: *Müşteri sayısı (N), araç kapasitesi (Q), avlanma grup boyutu (AGB), lider arkasından maksimim hareket (LMH), avlanma grup değerlendirme oranı (AGDO), epok sayısı (ES), epokta iterasyon sayısı (EIS), karar değişken aralıkları, α, β, Ra_{max} ve Ra_{min} parametrelerinin değerlerinin atanması.*

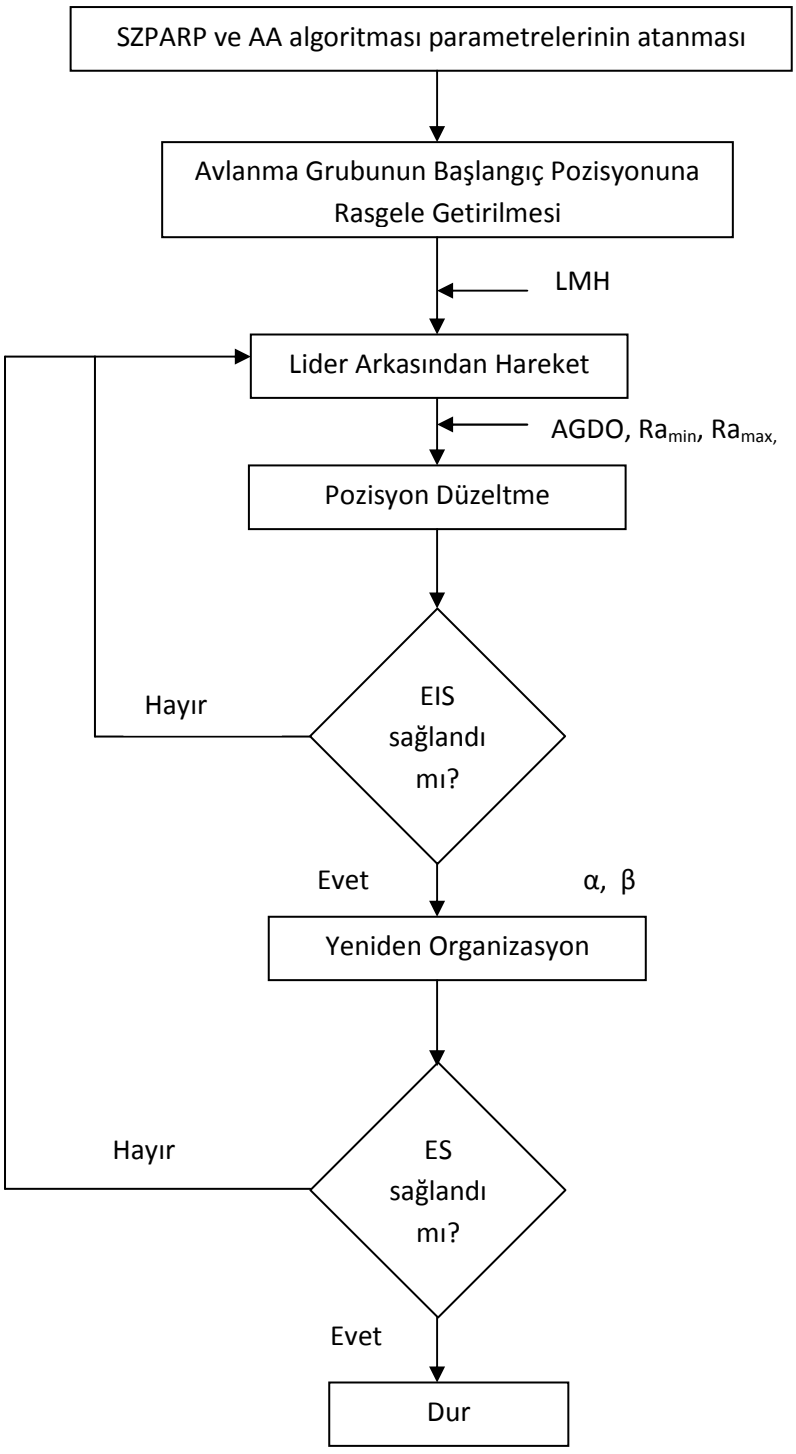
Adım 2: *Avlanma grubunun oluşturulması.*
Her bir avcının pozisyonun rasgele belirlenmesi, rassal anahtarlama ve ayırma algoritması kullanılarak rotaların oluşturulması, maliyetin hesaplanması. Maliyeti en düşük olan avcının lider olarak atanması.

Adım 3: *Avcıların yeni pozisyonlarının (4.9) eşitliğine göre lider arkasından hareket ederek belirlenmesi.*
Her bir avcıya rassal anahtarlama ve ayırma algoritmasının uygulanması. Her bir avcının yeni konumunun maliyetinin hesaplanması. Avcının yeni konumunun maliyeti eski konumundan iyi ise yeni konuma hareket ettirilmesi, aksi halde eski konumunda kalması.

Adım 4: *Avcıların pozisyonlarının, lider arkasından hareket ettikten sonra (4.10) eşitliği ile düzeltilmesi.*
Düzeltme sonucu elde edilen pozisyon değerinin maliyeti hesaplanarak eskisi ile karşılaştırılarak, maliyetin daha düşük olması durumunda yeni pozisyona avcının hareket ettirilmesi.

Adım 5: *EIS'ye ulaşana kadar Adım 3 ve 4'ün tekrarlanması. Epokta iterasyon sayısı tamamlandığında AG'nun yeniden organize edilmesi.*
Yeniden organizasyonda lider mevcut konumunda kalır ve diğer avcılar yeni konumlarını (4.11) eşitliğine göre belirlerler.

Adım 6: *ES'ye ulaşana kadar Adım 3-5'in tekrarlanması.*



Şekil 4.2. SZPARP için AA algoritmasın işleyişi

4.4. Deneysel Çalışmalar

Sıkı zaman pencereli araç rotalama problemi için önerilen av arama algoritması Intel® Core ™ 2 Duo CPU 2,4 GHz 2 GB RAM bilgisayar kullanılarak "DELPHİ 7" programlama dilinde kodlanmıştır. Algoritmanın etkinliğini değerlendirmek için Solomon problemleri üzerinde deneysel çalışmalar gerçekleştirilmiştir. Önerilen algoritmanın parametreleri Tablo 4.5'de verilmektedir.

Tablo 4.5. Önerilen AA algoritması parametre değerleri

Parametre	Kısaltma	Değer
Avlanma Grup Boyutu	AGB	500
Lider Arkasından Maksimum Hareket	LMH	0,3
Avlanma Grup Değerlendirme Oranı	AGDO	0,8
Epok Sayısı	ES	10
Epokta İterasyon Sayısı	EİS	1000
Alfa	α	0,1
Beta	β	-1
Maksimum Ra Değeri	Ra_{max}	0,01
Minimum Ra Değeri	Ra_{min}	0,000001

4.4.1. Sayısal Sonuçlar

Bu çalışmada Solomon problemlerinden C101, C102, C103, C104, C105, C201, C202, C203, C204, C205, R101, R102, R103, R104, R105, RC101, RC102, RC103, RC104 ve RC105 25 müşteri sayısı ile ele alınmıştır. Önerilen av arama algoritması ile kaç adet araç (rota) gerektiği ve her bir rotada hangi müşterilerin, hangi sırada yer alacağı belirlenmiştir. Ele alınan SZPARP problemlerinin her biri için AA algoritması 5 kez çalıştırılmıştır. Tablo 4.6 AA algoritması ile ele alınan problemler üzerinde elde edilen sayısal sonuçları göstermektedir. Tablonun birinci sütunu ele alınan problemi, ikinci sütun elde edilen araç sayısını, üçüncü sütun önerilen algoritma ile bulunan maliyeti, dördüncü sütun problem için optimum maliyeti ve son sütun % hatayı göstermektedir. % hata şu şekilde hesaplanmaktadır.

$$\%\, Hata = \frac{AA\ Maliyeti - Optimum\ Maliyet}{Optimum\ Maliyet} \times 100 \qquad (4.12)$$

Tablo 4.6. Önerilen AA algoritmasıyla elde edilen sayısal sonuçlar

Problem	Araç Sayısı	AA Maliyeti	Optimum Maliyet	% Hata
C101	3	191,8	191,3	0,26
C102	3	190,6	190,3	0,16
C103	3	190,7	190,3	0,21
C104	3	187,4	186,9	0,26
C105	3	191,8	191,3	0,26
C201	2	215,5	214,7	0,37
C202	2	215,4	214,7	0,33
C203	2	217,4	214,7	1,26
C204	2	213,9	213,1	0,38
C205	2	215,5	214,7	0,37
R101	8	618,3	617,1	0,19
R102	7	548,1	547,1	0,18
R103	5	454,6	454,6	0
R104	4	418,9	416,9	0,48
R105	6	531,5	530,5	0,19
RC101	4	462,2	461,1	0,24
RC102	3	352,7	351,8	0,26
RC103	3	333,9	332,8	0,33
RC104	3	306,6	306,6	0
RC105	4	412,4	411,3	0,27

Tablo 4.6'dan görülebileceği gibi AA algoritması 25 müşterili SZPARP problemleri için iyi sonuçlar vermektedir. % hatalar optimum sonuca oldukça yakın ve iki problem için de optimumdur. Algoritmanın beş kez çalışmasında genellikle aynı veya yakın sonuçlar elde edilmiştir.

Elde edilen sonuçlar literatürde yer alan Amini'nin 2011 çalışması ile karşılaştırılmıştır. Amini [90] ZPARP için parçacık sürü optimizasyonuna (PSO) dayanan bir algoritma geliştirmiştir ve algoritmanın performansını tez çalışması kapsamında ele alınan problemler üzerinde değerlendirmiştir. Geliştirilen AA algoritması ve PSO algoritması karşılaştırma sonuçları Tablo 4.7'de verilmektedir. Tablodan da görüldüğü gibi AA

algoritması üç problemde daha iyi sonuçlar elde etmiş ve bunların ikisinde de optimum sonucu yakalamıştır. PSO algoritmasının ise iki problemde elde ettiği sonuçlar daha iyidir ancak optimum değildir.

Tablo 4.7. Önerilen AA algoritmasıyla PSO algoritması sonuçlarının karşılaştırılması

Problem	AA Algoritması			PSO Algoritması		
	Araç Sayısı	Maliyet	% Hata	Araç Sayısı	Maliyet	% Hata
C101	3	191,8	0,26	3	191,8	0,26
C102	3	190,6	0,16	3	190,6	0,16
C103	3	190,7	0,21	3	190,7	0,21
C104	3	**187,4**	**0,26**	3	192,1	2,78
C105	3	191,8	0,26	3	191,8	0,26
C201	2	215,5	0,37	2	215,5	0,37
C202	2	215,4	0,33	2	215,4	0,33
C203	2	217,4	1,26	2	**215,5**	**0,37**
C204	2	213,9	0,38	2	213,9	0,38
C205	2	215,5	0,37	2	215,5	0,37
R101	8	618,3	0,19	8	618,3	0,19
R102	7	548,1	0,18	7	548,1	0,18
R103	5	**454,6**	**0**	5	454,7	0,02
R104	4	418,9	0,48	4	**418,1**	**0,29**
R105	6	531,5	0,19	6	531,5	0,19
RC101	4	462,2	0,24	4	462,2	0,24
RC102	3	352,7	0,26	3	352,7	0,26
RC103	3	333,9	0,33	3	333,9	0,33
RC104	3	**306,6**	**0**	3	307,1	0,16
RC105	4	412,4	0,27	4	412,4	0,27

5. BÖLÜM
TARTIŞMA-SONUÇ VE ÖNERİLER

5.1. Tartışma-Sonuç ve Öneriler

Günümüzde şirketler arasındaki giderek artan rekabet, çok detaylı stratejik kararlardan etkilenebilecek hassasiyettedir. Bunun sebebi bilgi ve teknolojinin çok daha hızlı yayılabilmesidir. Özellikle dağıtım sektörünü ilgilendiren ve lojistiğin önemli bir bileşeni olan araç rotalama, söz konusu rekabetin öne çıktığı en önemli alanlardan birisidir. Bu nedenle şirketlerin ve araştırmacıların araç rotalama üzerindeki ilgisi artmış, buda araç rotalamasının sürekli gelişimini sağlamıştır.

Bu çalışmada, lojistik sisteminin temelini oluşturan araç rotalama problemlerinden, sıkı zaman pencereli araç rotalama problemi, meta-sezgisel algoritmalardan yeni geliştirilen av arama algoritması ile çözümlenmiştir.

Geliştirilen algoritmayı, kesikli formdaki problemlere uygulayabilmek için rassal anahtarlama ile tam bir tur elde edilmiştir. Ardından araç rotalama problemleri için geliştirilen ayırma algoritması, sıkı zaman pencereli araç rotalama problemine uyarlanmıştır. Bu yolla geliştirilen SZPARP için av arama algoritması, Solomon problemlerinden 25 müşterili C101, C102, C103, C104, C105, C201, C202, C203, C204, C205, R101, R102, R103, R104, R105, RC101, RC102, RC103, RC104 ve RC105 problemlerine uygulanmıştır.

Elde edilen sonuçlar incelendiğinde, av arama algoritması ile bazı problemlerde optimum çözüme ulaşıldığı görülmüştür. Ayrıca geliştirilen bu algoritma, literatürdeki aynı problemler üzerine uygulanan parçacık sürüsü optimizasyon algoritması ile kıyaslandığında, iyi sonuçlar elde edilebildiğini göstermiştir.

Ancak geliştirilen av arama algoritmasının temel sorunu büyük boyutlu problemlerde iyi sonuçların elde edilememesidir. Bu nedenle çalışmanın bundan sonraki aşamalarında, başlangıç çözümü için bir sezgisel kullanımı ve av arama algoritmasının seçilecek diğer bir meta-sezgisel algoritma ile melezleştirilmesi planlanmaktadır. Böylece büyük boyutlu problemlerde de iyi sonuçlar elde edilebileceği ön görülmektedir.

Sonuç olarak bu tezde, SZPARP'nin çözümü için ilk kez av arama algoritması uygulanmış, müşteri sayısının az olduğu problemlerde literatürdeki diğer problemlerle rekabet edebilecek düzeyde çözümlere ulaşılmıştır.

KAYNAKÇA

1. Emel, G., G., Taşkın, Ç., 2005. Araç rotalama problemlerinin iki aşamalı çözümünde genetik algoritma kullanımı. **Gazi Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi, 7**(1): 1-17.
2. Yurtkuran, A., 2009. Araç Rotalama Problemlerinin Çözümü İçin Yeni Bir Meta-Sezgisel Yaklaşım: Elektromanyetik Algoritma. Uludağ Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, Bursa, 68 s.
3. Aydemir, E., 2006. Esnek Zaman Pencereli Araç Rotalama Problemi ve Bir Uygulama. Gazi Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, Ankara, 68 s.
4. Dantzig, G., B., Ramser, J., H., 1959. The truck dispatching problem. **Management Science, 6**(1): 80-91.
5. Toth, P., Vigo, D., 2002. models relaxations and exact approaches for the capacitated vehicle routing problem. **Discrete Applied Mathematics, 123**(1-3): 487-512.
6. Dursun, P., 2009. Zaman Pencereli Araç Rotalama Problemi'nin Genetik Algoritma İle Modellenmesi. İstanbul Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, İstanbul, 75s.
7. Dantzig, G., Fulkerson, R., 1954. Johnson, S., Solution of a large-scale travelling salesman problem. **Operations Research, 2**: 393–410.
8. Clarke, G., Wright, J., W., 1964. Scheduling of vehicles from a depot to a number of delivery points. **Operations Research, 12**: 568–581.
9. Goldberg, D., E., 1989. Genetic Algorithms In Search, Optimization And Machine Learning. Addison-Wesley Publishing Company, USA, pp. 372, 1989.
10. Levin, A., 1972. Scheduling and fleet routing models for transportation systems. **Transportation Science, 5**(3): 232–256.
11. O'Connor, A., D., Dewald, C., A., 1970. A sequential deletion algorithm for the design of optimal transportation networks. 37th National Meeting Of The Operations Research Society of America, Bulletin of the Operations Research Society of America, 18(1).

12. Marks, D., H., Sticker, R., 1970. Routing for public service vehicles. **ASCE Journal Of The Urban Planning and Development Division, 97**(UP2): 165–178.

13. Liebman, J., C., 1970. Mathematical models for solid waste collection and disposal. 38th National Meeting Of The Operations Research Society of America, Bulletin of the Operations Research Society of America, 18(2).

14. Solomon, M., 1983. Vehicle Routing And Scheduling With Time Window Constraints: Models And Algorithms, pp. 42-83. Technical Report, College of Business Admin, Northeastern University.

15. Şeker, Ş., 2007. Araç Rotalama Problemleri ve Zaman Pencereli Stokastik Araç Rotalama Probleminde Genetik Algoritma Yaklaşımı. Yıldız Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, İstanbul, 125 s.

16. Keskintürk, T., 2009. Araç Rotalama Problemlerinin Global Karınca Koloni Optimizasyonu İle Çözümü. İstanbul Üniversitesi, Soysal Bilimler Enstitüsü, Doktora Tezi, İstanbul, 112s.

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

18. Christofides N., Mingozzi A., Toth P., 1981. Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxations. **Mathematical Programming, 20**(1): 255-282.

19. Göksal, F., P., 2010. Eşzamanlı Topla-Dağıt Araç Rotalama Problemi İçin Sezgisel Yaklaşımlar: Genetik Algoritma ve Kuş Sürüsü Eniyileme. Gazi Üniversitesi, Fen Bilimleri Enstitüsü,Yüksek Lisans Tezi, Ankara, 73s.

20. Chen, A., Yang, G., Wu, Z., 2006. Hybrid discrete particle swarm optimization algorithm for capacitated vehicle routing problem. **Journal Of Zhejiand University Science, 7**(4), 607-614.

21. Ralphs, T., K., Kopman, L., Pulleyblank, W., R., Trotter, L., E., 2001. On the capacitated vehicle routing problem. **Mathematical Programming, 94**(2-3): 343-359.

22. Brandao, J., 2006. A new tabu search algorithm for the vehicle routing problem with backhauls. **European Journal of Operational Research, 173**: 540–555.

23. Gendreau, M., Guertin, F., Potvin, J., Se'guin, R., neighborhood search heuristics for a dynamic vehicle dispatching problem with pick-ups and deliveries. **Transportation Research Part C, 14**: 157–174.

24. Bianchessi, N., Righini, G., 2007. Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery. **Computers & Operations Research, 34** (2): 578-594.

25. Çetin, S., Gencer, C., 2010. Kesin zaman pencereli - eş zamanlı dağıtım toplamalı araç rotalama problemi: matematiksel model, **Gazi Üniversitesi Mühendislik ve Mimarlık Fakültesi Dergisi, 25**(3): 579-585.

26. Desrochers, M., Verhoog, T., W., 1989. A matching based savings algorithm for the vehicle routing problem. **Les Cahiers du GERAD, 89**(04):1-16.

27. Altınkemer, K., Gavish, B., 1991. Parallel savings based heuristics for the delivery problem. **Operations Research, 39** (3): 456-469.

28. Thompson, H., N., Psaraftis, 1993. Cyclic transfer algorithms for the multivehicle routing and scheduling problems. **Operations Research, 41**(5): 935-946.

29. Kinderwater, M., W., Savelsbergh P,. 1997. Vehicle routing: handling edge exchanges. Local Search in Combinatorial Optimization Wiley, Chichester.

30. Fisher, M., L., Jaikumar, R., 1981. A generalized assignment heuristic for vehicle routing. **Networks, 11**(2): 109-124, 1981.

31. Ryan, D., M., Hjorring, C., Glover, F., 1993. Extensions of the petal method for vehicle routing. **Journal Of The Operational Research Society, 44**(3): 289-296.

32. Gillet, B., Miller, L., 1974. A heuristic algorithm for the vehicle dispatch problem. **Operations Research, 22**(2): 340-349.

33. Değertekin, S., Ö., Ülker, M., 2006. Hayalioğlu, M., S., Uzay çelik çerçevelerin tabu arama ve genetik algoritma yöntemleriyle optimum tasarımı, **İMO Teknik Dergisi, 259**: 3917-3934.

34. Homberger, J., Gehrıng, H., 1999. Two evolutionary metaheuristics for the vehicle routing problem with time windows. **Information Systems and Operational Research 37**: 297-318.

35. Cordeau, J., F., Laporte, G., Mercıer, A., 2001. A unified tabu search heuristic for vehicle routing problems with time windows. **Journal of the Operational Research Society, 52**: 928–936.

36. Taillard, E., Badeau, P., Gendreau, M., Guertin, F., Potvin, J., Y., 1997. A tabu search heuristic for the vehicle routing problem with soft time windows. **Transportation Science, 31**:170–186.

37. Chiang, W., C., Russell, R., A., 1996. Simulated annealing metaheuristics for the vehicle routing problem with time windows. **Annals of Operations Research, 63**: 3–27.

38. Metropolis, N., Rosenbluth, A., W., Rosenbluth, M., N., Teller, A., H., Teller, E., 1953. Equations of state calculations by fast computing machines**, Journal of Chemical Physics, 21**(6), 1087-1092.

39. Kirkpatrick, S., Gelatt, C., D., Vecchi, M., P., 1983. Optimization by simulated annealing**. Science New Series, 22**(4598): 671-680.

40. Gülsün, B., Tuzkaya, G., Bildik, E., 2008. Tersine lojistik ağ tasarımı: bir tavlama benzetimi yaklaşımı. **Sigma, 26**(1): 68-80.

41. Czech, Z., J., Czarnas, P., 2002. Parallel simulated annealing for the vehicle routing problem with time windows, pp. 376-383. 10th Euromicro Workshop on January 9-11, Spain.

42. Alaykıran, K., Engin, O., 2005. Karınca kolonileri meta-sezgiseli ve gezgin satıcı problemleri üzerine bir uygulaması. **Gazi Üniversitesi Mühendislik ve Mimarlık Fakültesi Dergisi, 20** (1): 69-76.

43. Yıldırım, U., M., Çatay, B., 2009. An ant colony algorithm for time-dependent vehicle routing problem with time windows. **Operations Research Proceedings, 6**: 337-342.

44. Ombuki, B., Ross, B., J., Hanshar, F., 2006. Multi-objective genetic algorithms for vehicle routing problem with time windows. **Applied Intelligence, 24**: 17-30.

45. Kennedy, J., Eberhart, R., C., 1995. Particle Swarm Optimization, pp. 1942-1948. IEEE International Conference on, November 27- December 1, 1995, Piscataway, NJ.

46. Tamer, S., Karakuzu, C., 2006. Parçacık sürüsü optimizasyon algoritması ve benzetim örnekleri. Kocaeli Üniversitesi Mühendislik Fakültesi, Elektronik ve Haberleşme Mühendisliği Bölümü, Kocaeli, İzmit.

47. Chu, S., C., Chen, Y., T., Ho, J., H., 2006. Timetable scheduling using particle swarm optimization, pp. 324-327. International Conference on Innovative Computing, Information and Control, September 1, 2006, Beijing, China.

48. Pang, W., Wang, K., Zhou, C., Dong, L., 2004. Fuzzy discrete paricle swarm optimization for solving travelling salesman problem. Forth International Conferance On Computer and Information Technology.

49. Shi, X., H., Liang, Y., C., Lee, H., P., Lu, C., Wang, Q., X., 2007. Particle swarm optimization-based algorithms for TSP and generalized TSP. **Information Processing Letters, 103**: 169-176.

50. Zhu, Q., Yingchun, L., Zhu, S., 2006. An improved particle swarm optimization algorithm for vehicle routing problem with time windows, pp. 1386-1390. IEEE Congress On Evolutionary Computation, September 11, 2006, Canada.

51. Storn, R., Price, K., 1995. Differential evolution: a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report, Berkley, CA.

52. Keskintürk, T., 2006. Diferansiyel gelişim algoritması. **İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi, 5**(9): 85-99.

53. Deng, C., Zhao, B., Yang, Y., Deng, A., 2009. Novel binary dıfferential evolution algorithm for discrete optimization, pp. 346-349. 5th International Conferance On Natural Computation, 14-16 August, 2009, Jıjiang, China.

54. Zhang, J., Avasarala, V., Sanderson, A., C., Mullen, T., 2008. Diffrential Evolutıon For Discrete Optimization: An Experimental Study On Combinatorial Auction Problems, pp. 2794-2800. Congress On Evolutionary Computation, June 1-6, 2008, Hong Kong.

55. Prado, R., S., Silva, R., C., P., Guimaraes, F., G., Neto, O., M., 2010. Using differential evolution for combinatorial optimization: a general approach, 11-17 s. Systems Man and Cybernetics, 10-13 October, 2010, Preto, Brazil.

56. Mingyong, L., Erbao, C., 2010. An improved differential evolution algorithm for vehicle routing problem with simultaneous pickups and deliveries and time windows. **Engineering Applications of Artificial Intelligence, 23**(2): 188-195.

57. Kıran, M., S., Gündüz, M., Sahman, M., A., 2009. Arı kolonisi optimizasyon algoritması kullanarak en kısa yol bulma, 11-14 s. EMO IV. Öğrenci Kurultayı Sempozyumu, 15-16 Ekim, 2009, Adana, Türkiye.

58. Pham, D., T., Ghanbarzadeh, A., Koc., E., Otri, S., Rahim, S., Zaidi, M., 2006. The bees algorithm, pp. 454-459. Intelligent Production Machines and Systems, 3-14 Jully, 2006, Cardiff, UK.

59. Pham, D., T., Koç, E., Lee, J., Y., Phrueksanant, J., 2007. Using the bees algorithm to schedule jobs for a machine. Manufacturing Engineering Centre, Cardiff, UK.

60. Pan, Q., Tasgetiren, M., F., Suganthan, P., N., Chua, T., J., 2011. A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem, **Information Sciences, 181**(12): 2455-2468.

61. Sabar, N., R., Ayob, M., Kendall, G., 2009. Solving examination timetabling problems using honey-bee mating optimization, pp. 399-407. Multidisciplinary International Conference on Scheduling: Theory and Applications, August 10-12, 2009 Dublin.

62. Brajevic, I., 2011. Artificial bee colony algorithm for the capacitated vehicle routing problem. **European Journal of Operational Research, 215**(1): 126-135.

63. Ayvaz, M., T., Karahan, H., H., Gurarslan, G., 2007. Su dağıtım şebekelerinin armoni araştırması optimizasyon tekniği ile optimum tasarımı, 188-202 s. 5. Kentsel Altyapı Ulusal Sempozyumu, 1-2 Kasım, 2007, Hatay, Türkiye.

64. Wang, L., Pan, Q., Tasgetiren, M., F., 2010. Minimizing the total flow time ın a flow shop with blocking by using hybrid harmony search algorithms. **Expert Systems With Applications, 37**(12): 7929-7936.

65. Zou, D., Gao, L., Li, S., Wu, J., 2011. Solving 0-1 knapsack problem by a novel global harmony search algorithm. **Applied Soft Computing, 11**(2): 1556-1564.

66. Geem, Z., W., Kim, J., H., Loganathan, G., V., 2001. A new heuristic optimization algorith: harmony search. **Simulation, 76**(2): 60–68.

67. Oftadeh, R., Mahjoob, M., J., Shariatpanahi, M., 2010. A novel meta-heuristic optimization algorithm ınspired by group hunting of animals:hunting search. **Computer and Mathematics with Applications, 60**(7): 2087-2098.

68. Oftadeh, R., Mahjoob, M., J., 2009. A new meta-heuristic optimization algortihm:hunting search, pp. 1-5. 5th International Conference on, September 2-4, 2009, Famagusta, Grecee.

69. Ioannou, G., Kritikos, M., 2004. A synthesis of assignment and heuristic solutions for vehicle routing with time windows. **Journal of the Operational Research Society, 55**(1): 2-11.

70. Chen, C., H., Ting, C., J., 2005. A hybrid ant colony system for vehicle routing problem with time windows. **Journal Of The Eastern Asia Society For Transportation Studies, 6**: 2822-2836.

71. Tokaylı, M., A., 2005. Zaman Pencereli Araç Rotalama Problemi İçin Bir Karar Destek Sistemi. Gazi Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, Ankara, 129 s.

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

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

74. Kallehauge, B., Larsen, J., Madsen, O., B., G., 2001. Lagrangean Duality Applied On Vehicle Routing With Time Windows Experimental Results. IMM, 26 s.

75. Solomon, M., M., 1987. Algorithms for the vehicle routing and scheduling problems with time window constraints. **Operations Research, 35**(2): 254-265.

76. Cordone, R., Calvo, R., W., 2001. A heuristic for the vehicle routing problem with time windows. **Journal of Heuristics, 7**: 107-129.

77. Braysy, O., 2003a. Fast local searches for the vehicle routing problem with time windows. **Infor, 40**(4): 319-330.

78. Thangiah, S.R., Nygard, K.E., Juell, P.L., 1991. GIDEON: A genetic algoritm system for vehicle routing with time windows, pp. 322-328. Proceedings Seventh IEEE Conference on Artificial Intelligence Applications, IEEE Computer Society Press, Los Alamitos, CA.

79. Tan, X., Luo, X., Chen, W., N., Zhang, J., 2005. Ant colony system for optimizing vehicle routing problem with time windows, MIEE Departmant Of Computer Science, China.

80. Wang, X., P., Xu, C., L., Hu, X, P., 2008. Genetic algorithm for vehicle routing problem with time windows and a limited number of vehicles, pp. 128-133. 15th International Conference on Management Science & Engineering, 10-12 September, 2008.

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

82. Alvarenga, G., B., Mateus, G., R., Tomi, G., 2007. A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows. **Computers & Operations Research, 34**: 1561-1584.

83. Nazif, H., Lee, L., S., 2010. Optimized crossover genetic algorithm for vehicle routing problem with time windows. **American Journal of Applied Sciences, 7**(1): 95-101.

84. Bent, R., Hentenryck, P., V., 2006. A two-stage hybrid algorithm for pickup and delivery vehicle routing problems with time windows. **Computer & Operations Research, 33**: 875-893.

85. Gambardella, L., M., Taillard, E., Agazzi, G., 1999. A multiple ant colony system for vehicle routing problems with time windows. Technical Report Idsia, Switzerland.

86. Erbao, C., Mingyong, L., Kai, N., 2008. A differential evolution & genetik algortihm for routing problem with simultaneous delirvery and time windows. Proceedings of the 17th World Congress The International Federation of Automatic Control Seoul, July 6-11, 2008, Korea.

87. Ye, A., 2011. Study of the vehicle routing problem with time windows based on improved particle swarm optimization algortihm. College of Information Science and Engineering; Zhejiang Normal University; Jinhua, Zhejiang.

88. Nasir, A., M., Masrom, S., 2011. Evaluation of vehicle routing problem with time windows by using metaheuristics algorithm, Recent Researches in Mathematical Methods in Electrical Engineering and Computer Science.

89. Prins, C., 2004. A simple and effective evolutionary algorithm for the vehicle routing problem. **Computer and Operations Research, 31**: 1985-2002.

90. Amini S., 2011. A novel PSO for solving the VRPTW with real case study, pp. 562-567. Proceedings of the 2011 International Conference on Industrial Engineering and Operations Management, January 22-24, 2011, Kuala Lumpur, Malaysia.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı, Soyadı: Nihan GÜLSOY
Uyruğu: Türkiye (TC)
Doğum Tarihi ve Yeri: 7 Kasım 1987, Kayseri
Medeni Durumu: Bekar
email: nihangulsoy@hotmail.com

EĞİTİM

Derece	Kurum	Mezuniyet Tarihi
Lisans	EÜ MF Endüstri Mühendisliği	2009
Lise	Özel Erciyes Lisesi, Kayseri	2004

İŞ DENEYİMLERİ

Yıl	Kurum	Görev
2010- Halen	Halkbank Kredi ve Proje Değerlendirme Daire Başkanlığı	Uzman Yardımcısı

YABANCI DİL

İngilizce