İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

BULANIK KARAR ORTAMINDA KARINCA KOLONİSİ OPTİMİZASYONU YÖNTEMİYLE ARAÇ ROTALAMA

DOKTORA TEZİ

Y. Müh. Sezgin KILIÇ

Anabilim Dalı : ENDÜSTRİ MÜHENDİSLİĞİ

Programı : ENDÜSTRİ MÜHENDİSLİĞİ

EYLÜL 2008

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

BULANIK KARAR ORTAMINDA KARINCA KOLONİSİ OPTİMİZASYONU YÖNTEMİYLE ARAÇ ROTALAMA

DOKTORA TEZİ

Y. Müh. Sezgin KILIÇ
(507032106)

Tezin Enstitüye Verildiği Tarih : 13 Mayıs 2008
Tezin Savunulduğu Tarih : 15 Eylül 2008

Tez Danışmanı : Prof.Dr. Cengiz KAHRAMAN

Diğer Jüri Üyeleri : Doç.Dr. Tijen ERTAY (İ.T.Ü)

Prof.Dr. Nahit SERARSLAN (İ.T.Ü.)

Prof.Dr. Ziya ULUKAN (G.S.Ü.)

Prof.Dr. Ertuğrul KARSAK (G.S.Ü.)

EYLÜL 2008

ÖNSÖZ

İletişim ve ulaştırma alanındaki gelişmeler ile ticaretin uluslararası sınırlarının giderek ortadan kalkması bölgeler ve kıtalar arası ürün hareketini son yıllarda önemli ölçüde arttırmıştır. Üretim sistemlerinin her aşamasındaki arz ve talep noktaları giderek dünya coğrafyasının farklı bölgelerine yayılmaktadır.

Bu tez çalışmasında, tam zamanında üretim ve yalın üretim gibi sistemlerinin lojistik anlamda desteklenebilmesi amacıyla ürün ve/veya hizmetlerin her aşamadaki müşterilere istenilen zamanlarda ulaştırılması için gerçek hayattaki belirsizlikleri ve esneklikleri de içeren bir araç rotalama problemi modeli ve çözüm yöntemi önerilmiştir.

Bu çalışma sürecinde değerli katkılarından dolayı Prof. Dr. Nahit SERARSLAN'a, Prof. Dr. Ziya ULUKAN'a ve danışmanım Prof. Dr. Cengiz KAHRAMAN'a ayrıca doktora eğitimim sürecinde desteğinden dolayı Türkiye Bilimsel ve Teknolojik Araştırma Kurumuna teşekkür ederim.

Son olarak, eğitimim sürecinde manevi desteklerini esirgemeyen eşim Gamze ve oğlum Ata Boran'a sevgilerimi sunarım.

MAYIS 2008 Sezgin KILIÇ

İÇİNDEKİLER

KISALTMALAR **v**
TABLO LİSTESİ **vi**
ŞEKİL LİSTESİ **vii**
SEMBOL LİSTESİ **x**
ÖZET **xii**
SUMMARY **xiii**

1. GİRİŞ **1**

2. ARAÇ ROTALAMA PROBLEMİ **6**
2.1 Zaman Aralıklı Araç Rotalama Problemi 9
2.2 Zaman Aralıklı Araç Rotalama Probleminin Formülasyonu 11
2.3 Kesin Çözüm Yöntemleri 13
2.3.1 Sütun oluşturma tabanlı yöntemler 14
2.3.2 Lagrange gevşetmesi tabanlı yöntemler 14
2.4 Sezgisel Çözüm Yöntemleri 15
2.4.1 Klasik sezgisel yöntemler 16
2.4.2 Metasezgisel yöntemler 27
2.5 Örnek Problemler ve Geçmiş Çalışmaların Performansları 35

3. BULANIK KÜME ve OLABİLİRLİK TEORİSİ **42**
3.1 Bulanık Küme 42
3.2 Olabilirlik dağılımı 46
3.3 Olabilirlik teorisi 46
3.4 Bulanık Aralık ve Bulanık Sayı 49
3.5 Bulanık Eşik 52
3.6 Bulanık işlemler 53
3.7 Bulanık öncelik ilişkileri 55
3.8 Bulanık programlama 60

4. KARINCA KOLONİSİ OPTİMİZASYONU **67**
4.1 Gerçek karınca davranışı 67
4.2 KKO algoritmaları 69
4.2.1 Karınca sistemi algoritması 74
4.2.2 MAKS-MİN karınca sistemi 76
4.2.3 Karınca kolonisi sistemi 77
4.3 ARP için KKO algoritmaları 79
4.4 ZA-ARP için Karınca Kolonisi Sistemi tasarımı 82
4.4.1 Ağ yapısının ve başlangıç çözümünün oluşturulması 83
4.4.2 Karıncalar tarafından çözüm oluşturulması 84

4.4.3 İz bırakma ve iz güncelleme 87
4.4.4 Yerel arama 88
4.5 Deneyler 91
4.5.1 Karşılaştırmalar 92
4.5.2 Arama sürecinin incelenmesi 96
5. BULANIK KARAR ORTAMINDA ARAÇ ROTALAMA 101
5.1 Bulanık Yolculuk Süreleri 103
5.2 Esnek Kısıtlar 104
5.3 Bulanık karar ortamında ZA-ARP modelleri 108
5.3.1 Olabilirlik programlama modeli 109
5.3.2 Esnek programlama modeli 117
5.3.3 Gürbüz programlama modeli 120
5.4 Önerilen Modellerin Çözümü ve Deneyler 128
5.4.1 Olabilirlik Programlama Modeli için Deneyler 132
5.4.2 Esnek Programlama Modeli için Deneyler 146
5.4.3 Gürbüz Programlama Modeli için Deneyler 151
6. TARTIŞMA ve SONUÇ 156
KAYNAKLAR 160
EK A 178
EK B 180
ÖZGEÇMİŞ 183

KISALTMALAR

ARP	: Araç Rotalama Problemi
EAEY	: En Az Etki Yerleştirmesi
GA	: Genetik Algoritma
KAS	: Kümülatif Araç Sayısı
KKO	: Karınca Kolonisi Optimizasyonu
KS	: Karınca Sistemi
KTM	: Kümülatif Toplam Mesafe
SARP	: Stokastik Araç Rotalama Problemi
TA	: Tabu Arama
TB	: Tavlama Benzetimi
UB	: Uyumlu bellek
ZA-ARP	: Zaman Aralıklı Araç Rotalama Problemi

TABLO LİSTESİ

Sayfa No

Tablo 2.1: Rotalama ve çizelgeleme probleminin özellikleri....8
Tablo 2.2: Rota oluşturma sezgisellerinin performansı....38
Tablo 2.3: Rota iyileştirme sezgisellerinin performansı....38
Tablo 2.4: Metasezgisellerin performansı....39
Tablo 4.1: KKO uygulama alanları ve geçmiş çalışmalar....73
Tablo 4.2: Dondo ve Cerda (2007) tarafından RC101 problemine bulunan çözüm.93
Tablo 4.3: R110 örneği için bulunan en iyi çözüm....95
Tablo 4.4: Dolaylı iletişimin sonuçlara etkisi....97
Tablo 5.1: Olabilirlik programlama modeli için olası durumlar....116
Tablo 5.2: Esnek programlama modeli için olası durumlar....120
Tablo 5.3: Gürbüz programlama modeli için olası durumlar....127
Tablo 5.4: Müşteriler arası uzaklıklar (depo:0)....134
Tablo 5.5: Müşteriler arası uzaklıklar (devamı)....134
Tablo 5.6: Müşteriler arası yolculuk süreleri....134
Tablo 5.7: Müşteriler arası yolculuk süreleri (devamı-1)....134
Tablo 5.8: Müşteriler arası yolculuk süreleri (devamı-2)....135
Tablo 5.9: Zheng ve Liu (2006) örneği için bulunan en iyi çözüm....135
Tablo 5.10: R1 grubu örnekler için elde edilen sonuçlar....138
Tablo 5.11: RC1 grubu örnekler için elde edilen sonuçlar....140
Tablo 5.12: R110 örneğine $N_{eşik} = 0,80$ durumu için elde edilen en iyi çözüm....143
Tablo 5.13: R103 örneği için sezgisel yöntemlerle bulunmuş en iyi çözüm....144
Tablo 5.14: R1 grubu örnekler için elde edilen sonuçlar....148
Tablo 5.15: RC1 grubu örnekler için elde edilen sonuçlar....149
Tablo 5.16: R101 problemine elde edilen sonuçlar....152
Tablo 5.17: R1 grubu problemlere oluşturulan çözümlerin kümülatif değerleri....154

ŞEKİL LİSTESİ

Sayfa No

Şekil 2.1: Araç rotalama problemi 6
Şekil 2.2: Zaman aralıklı araç rotalama problemi örneği 9
Şekil 2.3: Bir araç rotasında tekrar yerleştirme hareketi 20
Şekil 2.4: Farklı araç rotaları arasında tekrar yerleştirme hareketi 20
Şekil 2.5: Değiştirme hareketi 21
Şekil 2.6: Rota silme hareketi 21
Şekil 2.7: 2-opt değiştirme hareketi 22
Şekil 2.8: Or-opt hareketi 22
Şekil 2.9: 2-opt* hareketi 23
Şekil 2.10: Çapraz-değiştirme hareketi 24
Şekil 2.11: λ-değiş tokuş sezgileli için (2,1) operasyonu örneği 24
Şekil 2.12: Komşuluk bölgesi arama stratejileri 26
Şekil 2.13: Solomon (1987) örnek problemlerindeki müşteri konumları. 36
Şekil 2.14: Klasik sezgisellerin karşılaştırılması 40
Şekil 2.15: Metasezgisellerin karşılaştırılması 40
Şekil 3.1: Klasik bir A kümesi ve v, y, z elemanları 42
Şekil 3.2: Bulanık bir $\tilde{A}$ kümesi ve v, y, z elemanları 44
Şekil 3.3: Kısa, orta ve uzun insan boyu bulanık kümeleri üyelik fonksiyonları 44
Şekil 3.4: $\tilde{A}$ bulanık kümesi ile tanımlanan bir bulanık aralık 50
Şekil 3.5: $\tilde{B}$ bulanık kümesi ile tanımlanan bir bulanık sayı 51
Şekil 3.6: (a) $(-\infty, \tilde{A}]$ bulanık eşiği, (b) $(-\infty, \tilde{A}[$ bulanık eşiği 52
Şekil 3.7: (a) $[\tilde{A}, +\infty)$ bulanık eşiği, (b) $]\tilde{A}, +\infty)$ bulanık eşiği 52
Şekil 3.8: Bulanık maks ve min işlemleri 54
Şekil 3.9: X ve Y değişkenlerini tanımlayan $\tilde{A}$ ve $\tilde{B}$ bulanık aralıkları 56
Şekil 3.10: $\prod(X \geq Y)$ ölçütü 57
Şekil 3.11: $\prod(X > Y)$ ölçütü 57
Şekil 3.12: $\prod(Y \geq X)$ ölçütü 58
Şekil 3.13: $N(Y > X)$ ölçütü değeri 58
Şekil 3.14: $\prod(r \geq x)$ ve $\prod(r \leq X)$ ölçütleri 60
Şekil 3.15: $\prod(r < X)$ ve $\prod(r > X)$ ölçütleri 60
Şekil 4.1: Çift köprü deneyi şeması 68
Şekil 4.2: Başlangıç aşaması 69
Şekil 4.3: Kısa yol üzerinde yoğunlaşma 69
Şekil 4.4: KKO algoritmalarının genel yapısı 72
Şekil 4.5: KS algoritmasının genel yapısı 74
Şekil 4.6: ZA-ARP'nin gezgin satıcı problemine benzetilmesi 84
Şekil 4.7: Karınca tarafından oluşturulabilecek çözümler. 86

Şekil 4.8: ZAARP-KKS algoritmasının temel adımları 90
Şekil 4.9: R110 probleminin depo ve müşteri konumları 94
Şekil 4.10: R110 örneği için elde edilen çözüm 96
Şekil 4.11: Dolaylı iletişimin arama sürecine etkisi 98
Şekil 4.12: Arama sürecinde iz yapısındaki değişim 99
Şekil 5.1: Güvenilirlik düzeyleri ile olabilirlik dağılımı oluşturulması 104
Şekil 5.2: Bulanık yolculuk süresi ($\tilde{t}_{ij}$) 104
Şekil 5.3: Kesin durum için müşteri tatmin düzeyi 106
Şekil 5.4: Esnek en geç teslimat zamanı için müşteri tatmin düzeyi 107
Şekil 5.5: Olabilirlik programlama modeli için örnek durum 112
Şekil 5.6: $N(\tilde{v}_i < b_i)$ ölçütü için örnek durum 113
Şekil 5.7: Teslimat başlangıç zamanı için örnek durum 114
Şekil 5.8: Teslimat başlangıç zamanının elde edilmesi 114
Şekil 5.9: Örnek durum 115
Şekil 5.10: Müşteri tatmin düzeyi 117
Şekil 5.11: Müşteri tatmin düzeyi için örnek durum 118
Şekil 5.12: $\prod(v_i \le \tilde{b}_i)$ değerinin hesaplanması 119
Şekil 5.13: $\tilde{v}_i^g, a_i$ ve $\tilde{b}_i$ için örnek durum 122
Şekil 5.14: $N(\tilde{v}_i^g < b_4^i)$ değerinin hesaplanması 122
Şekil 5.15: $N(\tilde{v}_i^g < \tilde{b}_i)$ ve $N(\tilde{v}_i^g < b_4^i)$ için örnek durum-1 124
Şekil 5.16: $N(\tilde{v}_i^g < \tilde{b}_i)$ ve $N(\tilde{v}_i^g < b_4^i)$ için örnek durum-2 124
Şekil 5.17: $\prod(\tilde{b}_i \ge \tilde{v}_i)$ için örnek durum 125
Şekil 5.18: $\prod(\tilde{b}_i \ge \tilde{v}_i)$ değerinin hesaplanması 125
Şekil 5.19: $N(\tilde{v}_i^g < b_4^i)$ ve $\prod(\tilde{v}_i^g \le \tilde{b}_i)$ hesaplanması 126
Şekil 5.20: ZAARP-KKS algoritması akış şeması 129
Şekil 5.21: Bir karınca için çözüm oluşturma süreci akış şeması 130
Şekil 5.22: Kredibilite fonksiyonu 133
Şekil 5.23: Arama süreci 136
Şekil 5.24: Farklı $N_{\text{eşik}}$ değerleri için elde edilen çözümlerdeki araç sayıları 139
Şekil 5.25: Farklı $N_{\text{eşik}}$ değerleri için elde edilen çözümlerin yolculuk mesafeleri 139
Şekil 5.26: Farklı $N_{\text{eşik}}$ değerleri için elde edilen çözümlerdeki araç sayıları 140
Şekil 5.27: Farklı $N_{\text{eşik}}$ değerleri için elde edilen çözümlerin yolculuk mesafeleri 141
Şekil 5.28: R1 grubu problemler için toplam araç sayısı ve toplam yolculuk mesafeleri 141
Şekil 5.29: RC1 grubu problemler için toplam araç sayısı ve toplam yolculuk mesafeleri 142
Şekil 5.30: $N(\tilde{v}_0^1 < b_0)$ değerinin gösterimi 142
Şekil 5.31: Kesin yolculuk süresinin bulanıklaştırılması. 145
Şekil 5.32: R103 problemi bilinen en iyi çözümü gereklilik değerinin bulanık yolculuk süreleri durumundaki değişimi 145
Şekil 5.33: R207 problemi bilinen en iyi çözümü gereklilik değerinin bulanık yolculuk süreleri durumundaki değişimi 146
Şekil 5.34: Müşteri tatmin düzeyini temsil eden bulanık küme 147
Şekil 5.35: Farklı $\prod_{eşik}$ değerleri için araç sayıları 148

Şekil 5.36: Farklı $\prod_{eşik}$ değerleri için yolculuk mesafeleri 149
Şekil 5.37: Farklı $\prod_{eşik}$ değerleri için araç sayıları 150
Şekil 5.38: Farklı $\prod_{eşik}$ değerleri için yolculuk mesafeleri 150
Şekil 5.39: R1 grubu problemler için toplam araç sayısı ve toplam yolculuk mesafeleri 150
Şekil 5.40: RC1 grubu problemler için toplam araç sayısı ve toplam yolculuk mesafeleri 151
Şekil 5.41: Farklı $N_{eşik}$ ve $\prod_{eşik}$ değerleri için araç sayıları 153
Şekil 5.42: Farklı $N_{eşik}$ ve $\prod_{eşik}$ değerleri için yolculuk mesafeleri 153
Şekil 5.43: R1 grubu problemler için toplam araç sayıları 154
Şekil 5.44: R1 grubu problemler için toplam yolculuk mesafeleri 155

SEMBOL LİSTESİ

a_i	: i müşterisine en erken teslimat başlangıç zamanı
A	: V düğümleri için bağlar kümesi
$\tilde{A}^{\lambda}$	: Bulanık bir $\tilde{A}$ kümesinin λ-kesimi
b_i	: i müşterisine en geç teslimat başlangıç zamanı
$\tilde{b}_i$	: i müşterisine bulanık en geç teslimat başlangıç zamanı
c_{ij}	: (i,j) yolunun kullanım maliyeti
d_{ij}	: (i,j) yolunun mesafesi
G	: V düğümleri ve A bağlarının oluşturduğu ağ yapısı
g	: Araç indisi
h_i^g	: g aracının i müşterisine teslimat başlangıç zamanı
$\tilde{h}_i^g$	: g aracının i müşterisine bulanık teslimat başlangıç zamanı
J_i^k	: i müşterisi üzerinde bulunan k karıncası için aday müşteriler kümesi
i, j	: Müşteri/düğüm indisi
k	: Karınca indisi
K_g	: g aracının kapasitesi
L_k	: k karıncası tarafından oluşturulan çözümün yolculuk mesafesi
L_{min}	: Bulunan en iyi çözümün yolculuk mesafesi
L_0	: Başlangıç çözümünün yolculuk mesafesi
m	: Karınca sayısı
n	: Müşteri sayısı
N	: Müşteriler kümesi, $N = \{1, 2..., i, ...n\}$
$N_{eşik}$	: Kabul edilebilir en düşük gereklilik değeri
P_{ij}^k	: k karıncasının i müşterisinden j müşterisine geçiş olasılığı
q	: Rassal bir sayı
q_0	: geçiş kuralı için eşik değer
q_i	: i müşterisinin talep miktarı
Q	: İz miktarının belirlenmesinde kullanılan sabit sayı
R	: Araçlar kümesi, $R = \{1, 2..., g, ...r\}$
r	: Araç sayısı
s_i	: i müşterisinin servis süresi
t	: Döngü sayacı
t_{ij}	: (i,j) yolu için yolculuk süresi
$\tilde{t}_{ij}$	: (i,j) yolu için bulanık yolculuk süresi

v_i^g : g aracının i müşterisine varış zamanı

$\tilde{v}_i^g$: g aracının i müşterisine bulanık bulanık zamanı

V : Düğümler kümesi, $V = \{0,1,2...n\}$

x_{ij}^g : g aracının (i,j) yolunu kullanıp kullanmadığını gösteren değişken

α : İz miktarının önemini belirleyen sabit

β : Sezgisel bilginin önemini belirleyen sabit

η_{ij} : (i,j) yolu üzerindeki sezgisel bilgi

$\mu_{\tilde{A}}$: $\tilde{A}$ bulanık kümesinin üyelik fonksiyonu

$\pi(u)$: u elemanının olabilirlik derecesi

ρ : Buharlaşma oranı

τ_0 : Başlangıç iz miktarı

τ_{ij} : (i,j) yolu üzerindeki iz miktarı

$\Delta\tau_{ij}$: (i,j) yolu üzerine bırakılacak toplam iz miktarı

$\Delta\tau_{ij}^k$: k karıncası tarafından (i,j) yolu üzerine bırakılacak iz miktarı

ψ^{min} : Bulunan en iyi çözüm

ψ^0 : Başlangıç çözümü

ψ^k : k karıncası tarafından oluşturulan çözüm

∇_{igp} : i müşterisinin g araç rotasındaki p pozisyonuna yerleştirilmesi durumunda oluşacak etki değeri

$\prod_{eşik}$: Kabul edilebilir en düşük müşteri tatmin düzeyi

BULANIK KARAR ORTAMINDA KARINCA KOLONİSİ OPTİMİZASYONU YÖNTEMİYLE ARAÇ ROTALAMA

ÖZET

Bu tez çalışmasında, bulanık kümeler ve olabilirlik teorilerinden faydalanılarak zaman aralıklı araç rotalama problemi için bulanık karar ortamında kullanılabilecek modeller önerilmiştir. Geçmiş çalışmalar incelendiğinde problemin genellikle belirsizliklerin ve tercihlerdeki esnekliklerin göz ardı edilerek modellendiği görülmüştür. Bu tip modeller ile üretilen çözümler uygulama aşamasında genellikle geçerliliklerini yitirmekte ve kullanıcı tarafından elle düzeltilmeleri gerekmektedir. Stokastik modellerin kullanıldığı çalışmalarda ise önerilen yöntemlemlerin çok fazla hesaplama yükü gerektirdiği ve parametrelerin belirlenmesi için problemle ilgili geçmiş verilere ihtiyaç duyulduğu görülmektedir. Bu nedenlerle stokastik modeller de gerçek hayatta karşılaşılan problemlerin çözümünde rahatlıkla kullanılamamaktadır.

Son yıllarda bilgi işlem ve iletişim teknolojilerindeki gelişmelerin yanısıra büyük boyutlu kombinatoryal problemler için iyi sonuçların elde edilebildiği metasezgisel yöntemlerin de ortaya konması, yöneylem araştırması tekniklerinin bu problemler için uygulanabilirliği konusunda klasikleşen, çözüm süresinin uzunluğu ve hesaplama yükünün fazlalığı konularındaki dezavantajları geri plana iterek kullanılan modellerin geçerliliği konusundaki eksikliklerin ön plana çıkmasını sağlamıştır.

Önerilen bulanık programlama modellerinde belirsizliklerin ve esnekliklerin modellenmesi için bulanık kümeler ve bulanık aralıklar kullanılmıştır. Gereklilik ve olabilirlik ölçütleri ile planlayıcının belirlediği en yüksek risk seviyesine ve en düşük müşteri tatmin düzeyine sahip çözümler üretilebilmektedir. Bulanık modeller ile yüksek veri işleme maliyeti düşürülürken modellerin geçerlilikleri de arttırılmıştır. Önerilen modellere çözüm oluşturmak amacıyla karınca kolonisi optimizasyonu tabanlı bir algoritma geliştirilmiştir. Örnek problemler üzerinde gerçekleştirilen deneylerde önerilen modellerin ve çözüm yönteminin bir karar destek sistemi içerisinde kullanımı durumunda planlayıcıların tercih ve önceliklerine göre alternatif çözümlerin üretilebileceği ve oluşturulan çözümler hakkında planlayıcılara ve müşterilere daha fazla bilgi sağlanabileceği değerlendirilmiştir.

VEHICLE ROUTING IN A FUZZY DECISION ENVIRONMENT USING ANT COLONY OPTIMIZATION APPROACH

SUMMARY

In this Ph.D. thesis, the fuzzy set and the possibility theories are utilized in order to propose models for the vehicle routing problem that can be used in uncertain decision environments. Exploring the literature about the problem, it has been seen that the problem is usually modeled by ignoring uncertainties and flexibilities. Whereas, the solutions generated with these models usually become infeasible when implemented and the planners are involved to make corrections by hand. The natural approach to modeling the uncertainty is a stochastic one. Unfortunately, stochastic models are often hard to solve. Moreover, it may be hard or expensive to assume any specific probability distributions for unknown parameters. For these reasons stochastic models are also behind the needs of users.

Operational research methods seem to be inadequate for large sized combinatorial optimization problems up to past decades due to their large computational effort and long solution time. But, recent developments in data processing and communication technologies and recently proposed metaheuristic methods that can generate good solutions to large sized combinatorial problems made these classical disadvantages less important for the researchers. Subsequently, validity of used models become a more important issue for researchers.

The fuzzy programming models proposed in this study exploit fuzzy sets and fuzzy intervals in order to model flexibilities and uncertainties. Using the necessity and the possibility measures, solutions that have the maximum risk level and the minimum customer satisfaction, which are specified by the user, can be generated. Validities of the models are increased while decreasing the computational effort with fuzzy programming models. An ant colony optimization based algorithm for the proposed models is also developed. Results of the experimental studies with benchmark problems indicate that the proposed models can be usable for solving practical problems. The proposed approach can be integrated with a decision support system in order to generate alternative solutions achieving planners' preferences and obtain more information about the solutions both for the planners and the customers.

1. GİRİŞ

Ulaştırma, sosyal ve ekonomik faaliyetlerin gerçekleştirilmesinde ve desteklenmesinde en önemli unsurlardan bir tanesidir. Lojistik maliyetlerin yaklaşık olarak %50'sini ulaştırma maliyetleri oluşturmakta, yiyecek ve içecek gibi bazı özel alanlarda ise bu oran %70'e kadar çıkabilmektedir (De Backer ve diğ., 2000). Ortalama olarak bir ürün toplam maliyetinin %20 kadarını ulaştırma maliyetleri oluşturmaktadır (Reimann ve diğ., 2004). Hem tedarik zincirinin üyeleri arasındaki taşımalar hem de zincir ile son tüketiciler arasındaki taşımalar söz konusu maliyete etki etmektedir.

Uluslararası ticaret hacmindeki büyüme, rekabet avantajı elde edebilmek için farklı coğrafyalarda yeni üretim merkezlerinin oluşturulması, kaliteli ve ucuz ürün veya hammaddenin dünyanın neredeyse her yerinden aranabilir ve temin edilebilir hale gelmesi bölgeler arası eşya hareketini giderek arttırmaktadır. İnternet kullanımının yaygınlaşması ile elektronik alışverişin gelişmesi özellikle küçük eşya trafiğinde önemli bir artışa neden olmuştur. Firmalar ve akademik araştırmacılar etkili dağıtım stratejileri ile ulaştırma maliyetlerini azaltan aynı zamanda da müşteri memnuniyetini arttıran tasarımlar ile bu alandaki iyileştirme potansiyelinin büyüklüğünü yakın tarihte fark etmişlerdir.

Tipik bir dağıtım sisteminde, taşıma araçları kullanılarak, teslimat, müşteri dağıtımı veya tamir ve bakım gibi hizmetler coğrafik olarak değişik konumlarda bulunan müşterilere sunulmaktadır. Bir çok gerçek uygulama incelendiğinde (kargo, yakıt, ilaç vb. malzemelerin dağıtımı, öğrencilerin okul taşıtları ile taşınması vb.) ortak olan amacın, toplam dağıtım maliyetini azaltmak için çeşitli kısıtların da dikkate alınarak en uygun araç rotalarının belirlenmesi olduğu görülmektedir. Dağıtım alanındaki maliyet azaltma problemi geleneksel olarak araç rotalama problemi (ARP) olarak adlandırılmaktadır. Daha açık olarak, ARP; bir araç grubunun, bir müşteri grubuna talep ettikleri emtiayı teslim etmek için merkezi bir depodan başlayarak ve tekrar depoya dönmek üzere verimli bir şekilde rotalandırılmasıdır. Uygulama alanındaki

genişliği ve ekonomik önemi nedeni ile ARP üzerinde yoğunlukla çalışılan bir problemdir.

Modern dünyada ulaştırmanın önemi artık göz ardı edilemez boyuta ulaşmıştır. Ulaştırma, günlük yaşamda ve ekonomik düzende insanların, ürünlerin ve hizmetlerin yer değiştirmesi için temel oluşturmaktadır. İlginç olan ise tasarımcılar ve mimarların yıllardır bilgisayar destekli tasarım sistemleri kullanmaları, firmaların entegre yönetim sistemleri (SAP R/3, Max, Baan) kullanmalarına rağmen ulaştırma alanında bilgisayar destekli gönderim sistemleri pazarının henüz yeterince yaygınlaşmamış olmasıdır. Ulaştırmacılar kendilerini destekleyebilecek bilgisayar sistemleri için standart süreçlere henüz sahip değildirler. Mevcut yazılımların ise firmalara özgü ve özel problemlerin çözümünde kullanılabildiği görülmektedir. Fisher (1995) bu tip yazılımların geliştirilme süreçlerini şöyle özetlemiştir: Öncelikle iyi sonuçlar vereceği tahmin edilen bir sezgisel yöntem seçilir. Test aşamasında, sezgisel yöntemin pek çok durumda çok iyi sonuçlar verdiği ancak bazı örneklerde elde edilen sonuçların kabul edilemeyecek seviyede olduğu görülür. Kullanılan sezgisel yöntem daha iyi sonuçlar verecek şekilde geliştirilir ancak artık daha karmaşık bir yöntem haline gelmiştir. Bu tip bir geliştirme süreci pek çok kez tekrarlanarak istenen durumlar için iyi sonuçlar veren bir algoritma haline getirilir. Ancak artık modeldeki değişikliklere aşırı duyarlı ve çok özel bir yöntem elde edilmiştir. Bu yöntemin farklı bir coğrafik bölgede veya farklı bir firmada uygulanması halinde iyi sonuçlar elde edilemeyecektir. Psinger ve Ropke (2007), farklı tipte rotalama problemlerini çözebilecek genel sezgisel yöntemlerin önemli bir araştırma alanı olduğunu çünkü farklı firmalar için ulaştırma konusunda farklı ihtiyaçların ortaya çıktığını ve farklı tipte rotalama problemlerini çözme gereksinimi duyduklarını belirtmiştir.

Mevcut sistemlerin en önemli eksiklerinden bir tanesi uygulama safhasında her şeyin yapılan planlamaya uygun olacağı varsayımıdır. Örneğin İstanbul-İzmir arasındaki yolculuk süresi yapılan çizelgede teorik olarak 12 saat alınmışsa da uygulama safhasında hava ve yol koşulları nedeniyle bu süre 16 saate kadar da çıkabilmektedir. Bu durum ulaştırma modellerinde *belirsizliklerin* de model içerisinde yer alması gerekliliğini ortaya koymaktadır. Belirsizliklerin modellenmesinde kullanılan en yaygın yaklaşım stokastik modellerdir. Ancak belirsizliklerin stokastik modelerle ele alındığı çalışmalar zaten zor olan rotalama probleminin çözüm süresini uygulamaya

elverişli sınırların oldukça dışına çıkarmaktadır. Ayrıca, stokastik modellerdeki belirsiz parametreler için olasılık dağılımlarının belirlenebilmesi problemle ilgili geçmiş verilerin ulaşabilir olamasını gerektirmekte ve bu durum da genellikle maliyetli ve zor süreçlerle sağlanabilmektedir.

Mevcut sistemlerdeki diğer bir eksiklik ise müşteri tercihlerinin modellenmesinde *esneklik* sağlayamamalarıdır. Müşterilerin genellikle teslimat zamanları ile ilgili tercihlere ve kesin sınırlara sahiptir. Ancak çoğunlukla bu tercihler gözardı edilerek sadece kesin sınırlara sahip bir teslimat zaman aralığı modelde yer almaktadır. Araç, bu zaman aralığı içerisinde teslimat yapabiliyorsa çözüm uygun olarak kabul edilmekte aksi halde reddedilmektedir. Bu tip modellerde iki önemli dezavantaj ortaya çıkmaktadır. Birincisi, dar zaman aralıkları nedeniyle uygun bir çözümün bulunamaması, ikincisi ise geniş zaman aralıkları nedeniyle oluşturulan çok sayıda uygun çözüm içerisinden en fazla müşteri tatmini sağlayan çözümü seçebilmenin mümkün olmamasıdır.

ARP'ni belirsizlikleri ve esneklikleri gözardı ederek ele alan geleneksel yaklaşımlar için en önemli dezavantaj son yıllara kadar çözüm bulma süresinin uygulamaya elverişli sınırların dışında olması olarak görülmekteydi. Ancak yakın zamanda veri işleme yöntemlerinin ve çözüm arama algoritmalarının önemli ölçüde gelişmesiyle daha önce ele alınamayan problemler üzerinde çalışabilmek mümkün olmuştur. Modern işlemcilerin yüksek hızda veri işlemeye olanak sağlaması ve metasezgisel yöntemlerin kullanımı ile büyük boyutlu problemlere uygulamaya elverişli sürelerde iyi çözümler oluşturabilmek artık mümkündür. Ancak bu gelişmeler karar vericiler için ulaştırma alanında yöneylem araştırması yöntemlerinin çekiciliğini yeterli düzeyde arttıramamıştır. Günümüzde mevcut yöntemler için en önemli dezavantaj modelleme aşamasındaki yetersizliktir. Gerçek hayattaki belirsizlikleri ve esneklikleri içermeyen modeller ile oluşturulan çözümlerin uygulama aşamasında genellikle kullanıcı tarafından elle düzeltilmesine ihtiyaç duyulmaktadır (Fortemps, 2000).

Bu tez çalışması, bulanık kümeler ve olabilirlik teorilerinden faydalanılarak zaman aralıklı araç rotalama problemi için geçmiş çalışmalardaki dezavantajların azaltılabileceği öngörüsünü temel almaktadır. Belirsizlikler ve esneklikler model içerisine dahil edilerek hem modellerin gerçek duruma benzerliğinin yani

geçerliliğinin arttırılması hem de stokastik modellerden daha basit ve daha hızlı çözüm üretebilen modellerin ortaya konabilmesi amaçlanmıştır.

Belirsizliğin ve esnekliğin modellenebilmesi için bulanık kümelerin ve olabilirlik teorisinin güçlü bir araç olarak kullanılabileceğini ortaya koyan geçmiş çalışmalar mevcuttur. Bu tip bir yaklaşım yöneylem araştırması alanında henüz çok fazla yaygınlaşmış olmasa da özellikle çizelgeleme problemleri için belirsiz işlem süreleri ve/veya esnek teslimat zamanlarının bulanık kümeler ile temsil edilerek başarılı sonuçların elde edilebildiği görülmüştür (Kılıç ve Kahraman, 2007; Kılıç ve Kahraman, 2006a; Zheng ve Liu, 2006; Dubois ve diğ., 2003a; Chanas ve Kasperski, 2003; Fortemps, 2000; Sakawa ve Kubota, 2000; Slowinski and Hapke, 2000; Dubois ve diğ., 1995).

Bu tez çalışmasında ise araç rotalama probleminin gerçek hayattaki duruma daha benzer olarak bulanık bir karar ortamında ele alınarak modellenmesi ve çözülebilmesi amaçlanmıştır. Geçmiş çalışmalar incelendiğinde bu konu üzerinde çok az sayıda çalışmanın ortaya konduğu görülmektedir. Çalışmada öncelikle ARP belirsiz/eksik veri ve esnek kısıtlar içeren bulanık karar ortamında modellenmiş daha sonra bu modele çözüm oluşturabilecek bir algoritma geliştirilmiştir. Bu çalışmada kullanılan çözüm yönteminden beklentiler ise:

(1) çözümün kısa (uygulamaya elverişli) sürede oluşturulabilmesi,
(2) en iyi olmasa da en iyi çözüme yakın çözümlerin oluşturulabilmesi,
(3) farklı tipte ARP problemlerinin çözümünde kullanılabilir olması yani probleme özgü özel bir yöntem olmamasıdır.

ARP'nin NP-zor sınıfında bir problem olması (1) ve (2) numaralı beklentileri karşılanabilmesi için metasezgisel bir çözüm yönteminin kullanımını gerekli kılmaktadır. Ayrıca metasezgisel bir yöntemin kullanımı ile (3) numaralı beklentinin daha kolay karşılanabilmesi de mümkün olabilecektir. Metasezgisel yöntem olarak yakın bir tarihte ve temel olarak gezgin satıcı probleminin çözümü için önerilmiş olan Karınca Kolonisi Optimizasyonu (KKO) yöntemi kullanılmıştır. Bu tercihin nedeni KKO'nun gezgin satıcı tipindeki problemlerde diğer yöntemlere göre daha başarılı sonuçlar vermesi ve ARP'nin de temelde gezgin satıcı problemine benzer olmasıdır. Geçmiş çalışmalar incelendiğinde de KKO'nun bulanık araç rotalama probleminin çözümüne yönelik kullanıldığı bir çalışmaya rastlanmamıştır.

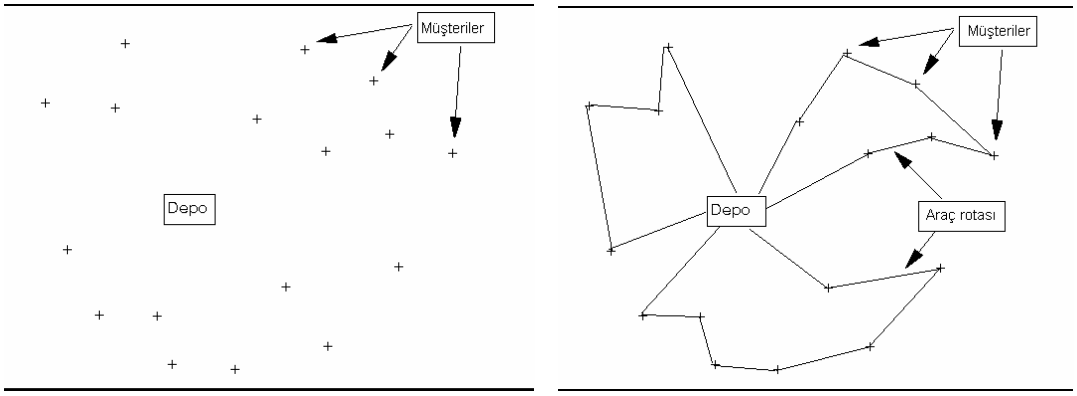
Geçmiş çalışmalar incelendiğinde bulanık karar ortamında ele alınan problemlerde, elde edilen bulanık amaç fonksiyonu değerlerinin birbirleriyle karşılaştırılarak en iyi çözümün arandığı görülmektedir. Bulanık değerlerin birbirleriyle (büyüklük/ küçüklük ilişkisi yönünden) karşılaştırılması için önerilmiş olan çok sayıda yöntem bulunması, elde edilen sonuçların kullanılan yönteme bağımlı olmasına neden olmaktadır. Bu durum, söz konusu yaklaşımlar için önemli bir dezavantaj olarak görülmektedir (Chanas ve Kasperski, 2004). Ayrıca bulanık değerler ile modellenen parametreler için gerçekleşecek değerlerin planlama aşamasında henüz bilinmiyor olması yukarıda belirtilen yaklaşımla elde edilen ve optimal olarak nitelendirilen bir çözümün uygulama aşamasında da aynı sonuçları vereceği veya kısıtlara uygun olacağı konusunda şüphe oluşturmaktadır. Bu nedenle, bulanık karar ortamında elde edilen bir çözümün optimal olarak tanımlanması karar vericiyi yanıltabilir. Bulanık karar ortamında optimal olma durumu da belirsizlik içermektedir.

Bu çalışmada ise olabilirlik teorisine dayanan bir yaklaşım ile yukarıdaki olumsuzluğun giderilebilmesi amaçlanmıştır. Karar verici tarafından planlama öncesinde belirlenecek olan en düşük müşteri tatmin düzeyini ve/veya kısıtlara uygun olma gerekliliğini sağlayan çözümler içerisinde iyi bir çözümün aranmasını içeren bir yaklaşım önerilmiştir. Önerilen yaklaşımla, karar vericiye farklı risk ve müşteri tatmin düzeyleri için oluşturulabilecek alternatif çözümler ve maliyetler arasında seçim yapabilme imkanı sunulmuştur.

Tez çalışması altı bölümden oluşmaktadır. Çalışmanın ikinci bölümünde ARP tanıtılmış ve bu tez çalışmasına temel olan zaman aralıklı araç rotalama problemi (ZA-ARP) için geçmiş çalışmalarda kullanılan çözüm yöntemleri incelenmiştir. Üçüncü bölümde bulanık karar ortamı için önerilen modele temel oluşturması amacıyla bulanık kümeler ve olabilirlik teorisi incelenmiştir. Dördüncü bölümde ise önerilen modelin çözümünde kullanılacak olan karınca kolonisi optimizasyonu (KKO) metasezgiseli tanıtılmıştır. Ayrıca, belirlilik durumunda ZA-ARP'nin çözümünde kullanılabilek bir KKO algoritması geliştirilmiş ve performansı incelenmiştir. Çalışmanın beşinci bölümünde bulanık kümeler ve olabilirlik teorisinden yararlanılarak belirsiz yolculuk süreli ve/veya esneklik içeren teslimat zamanlı ZA-ARP için üç adet model önerilmiştir. Önerilen modellerin KKO algoritması ile çözümü gerçekleştirilmiş ve örnek problemler üzerinde denemeler yapılmıştır. Sonuç ve tartışma kısmı ise altıncı bölümde yer almaktadır.

2. ARAÇ ROTALAMA PROBLEMİ

Malzemenin, insanın veya bilginin depolama biriminden müşterilere dağıtımı ile ilgili alanlarda ortaya çıkan ARP; birbirlerinden farklı yerlerde konumlu olan çok sayıda müşteriye bir veya daha fazla depodan teslimat yapabilmek için diğer yan kısıtların da göz önünde bulundurularak optimal rotaların belirlenmesi olarak tanımlanabilir (Şekil 2.1). Özellikle ulaştırma ve lojistik alanlarında gerçek hayatta sıklıkla karşılaşılan bir problemdir. Okul servislerinin rotalandırılması, posta/kargo dağıtımı, gazete ve dergi dağıtımı, gıda dağıtımı, çöp toplama, akaryakıt dağıtımı gibi örnekler kolayca çoğaltılabilir.



Şekil 2.1: Araç rotalama problemi

İlk defa Dantzig ve Ramser (1959) tarafından bir akaryakıt filosunun kullanımı ile ana akaryakıt deposundan servis istasyonlarına dağıtım yapılması uygulama çalışması ile ortaya konulan problem, Clarke ve Wright (1964) tarafından akademik anlamda tanımlanmış, fiziksel dağıtım ve lojistik alanlarındaki gelişmelerle birlikte önemi devamlı artmıştır. Literatürde ARP konusunda yapılmış çok sayıda çalışma mevcuttur. Yapılan bilimsel çalışmaların neticelerinin yöneticiler tarafından karar verme aşamasında kullanımı ise son yıllarda yoğunluk kazanmıştır.

ARP'nin çözümünde amaç genellikle toplam ulaşım maliyetini en küçük yapacak rotaların bulunmasıdır, öyle ki;

(1) her müşteriye tam olarak bir araç tarafından servis verilmelidir,
(2) her araç rotası depoda başlayıp depoda bitmelidir,
(3) her araç rotasının toplam talebi araç kapasitesini aşmamalıdır.

Oluşturulan çözümlerin maliyeti genellikle toplam yolculuk mesafesi ve/veya kullanılan araç sayısı ile tanımlanmaktadır.

Yukarıdaki ARP tanımı genel bir yaklaşımdır. Uygulama alanına göre farklı kısıtlar probleme dahil edilmektedir. Geçmiş çalışmalarda ele alınan ARP'nin adlandırılmasında amaç fonksiyonunun önemli bir faktör olmadığı, problemin kısıtlar ve veri yapısı özelliklerine (belirli, belirsiz) göre aşağıdaki gibi adlandırıldığı görülmektedir.

- **Kapasite kısıtlı ARP :** Talepleri önceden bilinen bir müşteri kümesine aynı yükleme kapasitesine sahip araçlarla depoda başlayan ve biten rotalar oluşturularak teslimat yapılması.
- **Zaman aralıkları kısıtlı ARP :** Kapasite kısıtlı ARP'ne benzer ancak probleme her müşteriye teslimata başlanabilecek en erken ve en geç zamanı belirten teslimat zaman aralığı kısıtı eklenmiştir.
- **Açık ARP :** Kapasite kısıtlı ARP'ne benzer ancak araçlar depoya geri dönmez. Araç son müşteriye teslimat yapar ve orada kalır.
- **Çok depolu ARP :** Kapasite kısıtlı ARP'ne benzer ancak birden fazla depo vardır.
- **Özel araçlı/teslimatlı ARP :** Kapasite kısıtlı ARP'ne benzer ancak araçlar aynı tipte değildir ve bazı müşterilere belirli tip araçlarla hizmet verilebilir.

Dağıtım ve ulaştırma sistemlerinin verimliliği üzerinde araç filosu planlamasının etkisi oldukça önemlidir. Birçok farklı hususun değerlendirilmesi gerekmektedir. Ancak temel olarak ilgilenilmesi gereken iki husus, belirli bir kapasiteye sahip araçların tüm müşterilere hizmet götürecek şekilde rotalandırılması ve her bir rota için zaman ve öncelikler ile ilgili kısıtlara uygun çizelgenin oluşturulmasıdır. Bu problem pek çok uygulama çalışmasında birleştirilmiş rotalama ve çizelgeleme problemi olarak da adlandırılmaktadır. Bodin ve diğ. (1983), Tablo 2.1'de olduğu gibi ARP ile ilgili farklı özellikleri ve olası durumları özetlemiştir.

Tablo 2.1: Rotalama ve çizelgeleme probleminin özellikleri

Özellik	**Olası durumlar**
Filo büyüklüğü	Tek araç (gezgin satıcı problemi) Çok araç
Filo tipi	Homojen Heterojen
Depo sayısı	Tek depo Çok depo
Talep Yapısı	Deterministik Stokastik
Maksimum tur süresi	Sınırlı Sınırsız
Zaman aralıkları	Sınırsız Tek taraflı Çift taraflı
Şebeke yapısı	Tek yönlü Çift yönlü
Operasyonlar	Sadece alış Sadece teslimat Karışık
Maliyet	Rota maliyeti (rota uzunluğu) Sabit maliyet (araç sayısı) Rota ve sabit maliyet
Amaçlar	Toplam rota maliyeti Toplam sabit maliyet Ceza fonksiyonu (gecikme)

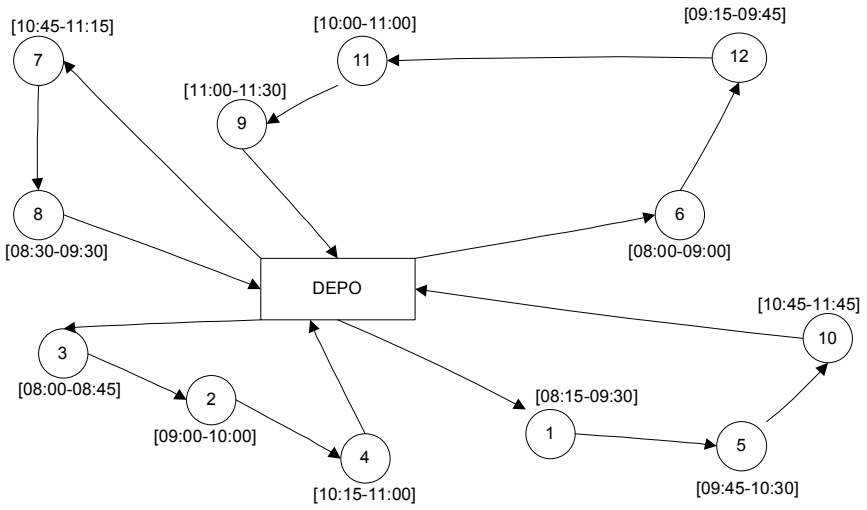
ARP, içerisinde alt bir problem olarak gezgin satıcı problemi içerdiğinden NP-zor bir problemdir. Gezgin satıcı problemine göre çözümü oldukça zor bir problemdir çünkü içerisinde iki alt problem barındırır. Birinci problem müşterileri önceden bilinmeyen sayıda kutuya (rotaya) doldurmayı (kümelendirmeyi) gerektiren kutu doldurma (bin-packing) problemidir. Daha sonra her rota için içerdikleri tüm müşterileri ziyaret eden bir gezgin satıcı problemi çözülmelidir.

ARP, son 40 yıldır üzerinde yoğunlukla çalışılan bir problemdir ancak 50-75 müşteriden daha büyük boyutlu problemlere düzenli olarak çözüm oluşturabilecek bir kesin çözüm yöntemi henüz geliştirilememiştir (Toth ve Vigo, 2002). Problem boyutu arttıkça karmaşıklığı üstel bir biçimde artış göstermektedir. Kesin yöntemlerin, orta ve büyük boyuttaki ARP'lerinin çözümünde yetersiz kalması ve gerçek uygulama alanında amaç fonksiyonu değeri hesaplanmasının da zor olmasından dolayı (problemdeki belirsiz değişkenler nedeniyle, örneğin yolculuk süreleri) uygulama alanında ARP çözümünde yaklaştırma yöntemlerine ihtiyaç duyulmakta ve kullanılmaktadır. Bu durum araştırmaların sezgisel yöntemler üzerinde daha fazla derinleşmesine neden olmuştur. Basit klasik sezgisel yöntemlerin

kullanımı ile başlayan çalışmaların günümüzde daha karmaşık ve kapsamlı çözüm yöntemleri olan metasezgisel yöntemler üzerinde yoğunlaştığı görülmektedir.

2.1 Zaman Aralıklı Araç Rotalama Problemi

Tablo 2.1'de görüldüğü üzere ARP uygulama alanına bağlı olarak farklı kısıtlar içerebilmektedir. Ancak, uygulama alanında en fazla, homojen araçlardan oluşan bir filo, tek bir depo, sınırlı araç kapasitesi ve müşteriler için teslimat zamanı aralıkları kısıtlarının bulunduğu durum ile karşılaşılmaktadır. Bu tip bir problemde her araç için depoda başlayan, farklı müşterilere uğrayan ve tekrar depoya dönen bir rota belirlenerek çözüm oluşturulmaktadır. Araçlar, rotalarında bulunan müşterilere önceden tanımlanmış zaman aralıkları içerisinde teslimata başlayabilmelidirler. Her müşteri için belirlenmiş olan en erken ve en geç teslimat başlangıç zamanları, zaman aralığı kısıtının sınırlarını oluşturmaktadır. Araç müşteriye en erken teslimat anından önce varmış ise en erken teslimat anına kadar bekleme yapmak durumundadır. Aracın müşteriye en geç teslimat anından sonra varış yapması durumunda müşteri teslimatı kabul etmeyeceğinden uygun olmayan bir çözüm oluşturulmuş olacaktır.



Şekil 2.2: Zaman aralıklı araç rotalama problemi örneği

Bir araç rotası üzerinde bulunan müşteri taleplerinin toplamı, ilgili aracın kapasitesi ile sınırlıdır. Problem bu haliyle ZA-ARP olarak adlandırılmaktadır. Şekil 2.2'de dört araç kullanımı ile bir ZA-ARP problemine oluşturulan rotalar gösterilmektedir. Kullanılan araç sayısı sabit maliyeti, tur uzunluğu ise rota

maliyetini oluşturmaktadır. Aracın müşteriye erken varış yapması durumunda oluşacak beklemeler ise bir ceza fonksiyonu ile amaç fonksiyonu içerisine dahil edilebilmektedir. Bazı ZA-ARP'lerinde ise araç müşteriye en geç teslimat anından sonra geldiğinde teslimata izin verilmekte ancak gecikme süresi bir ceza fonksiyonu ile amaç fonksiyonu içerisine dahil edilmektedir.

Gerçek hayatta karşılaşılan dağıtım problemine en çok benzeyen ve son yıllarda yapılan çalışmaların da üzerinde yoğunlaştığı ZA-ARP problemi bu tez çalışmasında temel problem olarak ele alınmıştır. Ancak önerilen çözüm yöntemi ZA-ARP'deki kısıtların kaldırılması durumunda da çözüm oluşturabilir yapıdadır.

Geçmiş çalışmalar incelendiğinde, sezgisel yöntemlerin kullanıldığı çalışmaların hiyerarşik olarak önce araç sayısını sonra yolculuk mesafesini azaltmak üzerinde yoğunlaştığı, kesin yöntemlerin kullanıldığı çalışmaların ise yolculuk mesafesini azaltmak üzerine yoğunlaştığı görülmektedir. Aslında en uygun amaç fonksiyonu ele alınan problemdeki özel kurallara ve uygulama alanının şartlarına bağlıdır. Ancak, özellikle üzerinde en çok çalışmanın yapıldığı Solomon (1987) örnek problemleri için bulunan sonuçlarda, genellikle araç sayılarının aynı olduğu ayırt edici faktörün yolculuk mesafesi olarak öne çıktığı, bazı problemlerde ise kullanılan araç sayısı ile yolculuk mesafesinin birbirleriyle çatışan sonuçlar verdiği görülmektedir.

ZA-ARP'nde araç kapasite kısıtı yer almaz ise (kapasite sonsuz ise), problem özel bir durum olan zaman aralıklı gezgin satıcı problemine dönüşmektedir. Ayrıca, en erken teslimat zamanı sıfır, en geç teslimat zamanı ise sonsuz alınarak problem kolaylıkla kapasite kısıtlı araç rotalama problemine dönüştürülebilir.

ZA-ARP'ndeki müşteri sayısı 10-15 müşteriden 1000'lerce müşteriye kadar çıkabilmektedir. Müşteri sayısındaki artış problemi daha karmaşık hale getirdiğinden küçük boyutlu problemler için önerilen yöntemler genellikle büyük boyutlu problemler için kullanılamamaktadır. Toth ve Vigo (2002) ZA-ARP'nin NP-zor tipinde bir problem olduğunu belirtmişlerdir. Dror (1994) ise ZA-ARP'nin Desrochers ve diğ. (1992)'nin gevşetme yaklaşımı ile ele alındığında bile NP-zor tipinde bir problem olduğunu belirtmiştir. Solomon (1986) ve Lenstra ve Rinooy Kan (1981), farklı sezgisel yöntemlerin en kötü performanslarını incelemişlerdir. Solomon (1986) tüm sezgiseller için ZA-ARP'nin ARP'den daha zor bir problem

olduğunu göstermiş, ayrıca zaman aralıkları kısıtının ZA-ARP'nin karmaşıklığını arttıran en önemli unsur olduğunu belirtmiştir. ZA-ARP için oluşturulan bir çözümün uygun olup olmadığının belirlenebilmesi için müşteriler arası mesafelerin yanısıra müşteriler arasındaki yolculuk süreleri de değerlendirilmelidir. Gerçek hayatta zaman aralıkları kısıtı ile sıklıkla karşılaşılmakta, ARP için önerilmiş olan çözüm yöntemlerinin ZA-ARP'yi çözebilmesi için önemli değişikliklerin yapılması gerekmektedir (Fisher, 1995).

2.2 Zaman Aralıklı Araç Rotalama Probleminin Formülasyonu

N= {1,...,n} müşteriler kümesi, R={1,...,r} ise kullanılabilecek araçlar kümesi olarak tanımlansın. ARP; G=(V,A) ağı üzerinde tanımlanmıştır. V={0, 1, ... , n} düğümler kümesi ve A={(i,j): i,j∈ V, i≠j} bağlar kümesinden oluşur. 0 düğümü dağıtım merkezini veya depoyu temsil etmektedir. Diğer düğümler müşterileri temsil eder. Her V\{0} düğümü negatif olmayan bir q_i talebine sahiptir (q_0=0). Her (i,j) bağı da negatif olmayan bir maliyete, c_{ij}, sahiptir, yolculuk süresi veya yolculuk maliyeti olarak adlandırılabilir. Simetrik durumda c_{ij} ve c_{ji} birbirlerine eşittir. ZA-ARP'de amaç tüm müşterilere teslimatın gerçekleştirilebileceği en düşük maliyetli çözümü oluşturmaktır. Her araç için depoda başlayan ve biten bir tur oluşturulur. Her talep noktasına sadece bir araç servis vermeli ve talebi tam olarak karşılamalıdır. i talep noktasında, q_i miktarındaki talebi boşaltmak/yüklemek için gerekli süre s_i ile gösterilir.

x_{ij}^g aşağıdaki (2.1) denklemi ile tanımlanmış bir karar değişkenidir.

$$x_{ij}^g = \begin{cases} 1, & \text{eğer g aracı } (i,\ j) \text{ yolunu kullanmışsa,} \\ 0, & \text{değilse.} \end{cases} \qquad \textbf{(2.1)}$$

ZA-ARP'nin formülasyonu için daha önce tanımlananlara ilave olarak aşağıdaki simgeler kullanılmıştır.

K_g : araç kapasitesi,

h_i^g : g aracının i müşterisine teslimat başlangıç zamanı,

d_{ij} : i ve j müşterileri arasındaki yolculuk mesafesi,

t_{ij} : i ve j müşterileri arasındaki yolculuk süresi,

$[a_i, b_i]$: i müşterisi için teslimat başlangıç zaman aralığı.

ZA-ARP probleminin matematiksel formülasyonu aşağıdaki gibidir:

$$\min \sum_{g \in R} \sum_{(i,j) \in A} c_{ij} x_{ij}^g \tag{2.2}$$

$$\sum_{g \in R} \sum_{j \in V} x_{ij}^g = 1 \qquad \forall i \in V - \{0\} \tag{2.3}$$

$$\sum_{i \in V - \{0\}} q_i \sum_{j \in V} x_{ij}^g \le K_g \quad \forall g \in R \tag{2.4}$$

$$\sum_{j \in V} x_{0j}^g = 1 \quad \forall g \in R \tag{2.5}$$

$$\sum_{i \in V} x_{i0}^g = 1 \quad \forall g \in R \tag{2.6}$$

$$\sum_{i \in V} x_{iu}^g - \sum_{j \in V} x_{uj}^g = 0 \quad \forall u \in V - \{0\}, \forall g \in R \tag{2.7}$$

$$x_{ij}^g (h_i^g + s_i + t_{ij} - h_j^g) \le 0 \quad \forall (i,j) \in A, \forall g \in R \tag{2.8}$$

$$a_i \sum_{j \in V} x_{ij}^g \le h_i^g \le b_i \sum_{j \in V} x_{ij}^g \quad \forall i \in V, \forall g \in R \tag{2.9}$$

$$x_{ij}^g \in \{0,1\} \quad \forall (i,j) \in A, \forall g \in R \tag{2.10}$$

Amaç fonksiyonu (2.2) ile oluşturulan bir çözümde kullanılan her (*i,j*) yolu nedeniyle oluşan c_{ij} maliyetlerinin toplamı alınarak çözümün maliyetinin minimizasyonu sağlanmaktadır. $\forall i \in V - \{0\}$ için $c_{ij} = d_{ij}$ ve $i = 0$ için H yeterince büyük bir sayı olmak üzere $c_{0j} = d_{0j} + H$ olarak kullanıldığında hiyerarşik olarak önce kullanılan araç sayısını sonra toplam mesafeyi azaltan bir amaç fonksiyonu elde edilmiş olur. (2.3) kısıt kümesi her müşterinin tam olarak bir defa ziyaret edilmesini sağlamaktadır. (2.4) kısıt kümesi ise araçlara kapasitelerinden fazla müşteri

atanmamasını sağlamaktadır. (2.5), (2.6) ve (2.7) kısıt kümeleri her aracın depodan çıkarak kendisine atanan müşterileri ziyaret edip tekrar depoya dönmesini sağlamaktadır. (2.8) kısıt kümesi *i* müşterisinden *j* müşterisine giden *g* aracının *j* müşterisine $(h_i^g + s_i + t_{ij})$ anından önce servise başlayamamasını sağlamaktadır. (2.9) kısıt kümesi her müşteriye kendi zaman aralığında servis verilmeye başlanmasını sağlamaktadır. Modeldeki 0-1 değerli değişkenler (2.10) ile tanımlanmıştır.

ZA-ARP için ilk çalışmalar Pullen ve Webb (1967), Knight ve Hofer (1968) ve Cook ve Russell (1978) tarafından gerçekleştirilmiştir. Geçmiş çalışmalar incelendiğinde kullanılan çözüm yöntemlerinin iki ana başlık altında toplanabileceği görülmektedir. Bunlar; kesin çözüm yöntemleri ve sezgisel yöntemlerdir.

2.3 Kesin Çözüm Yöntemleri

ZA-ARP için kesin çözüm yöntemleri ile kararlı olarak çözülebilen problem büyüklüğü sınırlıdır. Büyük boyutlu bazı problemlere çözüm bulunabilmiştir ancak hâlâ çözülemeyen 50 müşterilik problemler mevcuttur. Örneğin Kallehauge ve diğ. (2006) tarafından önerilen kesin çözüm yöntemi ile 1000 müşterilik bir örnek problem çözülebilmiştir. Ancak bu büyüklükteki bir problemin çözülebilmesi için problem belirli bir yapıda olmalı ve zaman aralıkları oldukça dar olmalıdır.

ZA-ARP, çeşitli tamsayılı programlama yöntemleriyle ele alınmıştır. Cordeau ve diğ. (2002) ve Kallehauge (2008) ZA-ARP için kesin çözüm yöntemlerinin kullanıldığı geçmiş çalışmaları incelemişlerdir.

ZA-ARP için ilk optimizasyon algoritması, durum uzayı gevşetmesi (state space relaxation) yönteminin kullanımı ile Christofides ve diğ. (1981) tarafından önerilmiştir. Durum uzayı gevşetmesi esasen içerisinde dinamik programlama yönteminin kullanıldığı bir dal-sınır algoritmasıdır. Benzer yöntemin kullanıldığı bir çalışma da Kolen ve diğ. (1987) tarafından probleme alt sınır bulunması amacıyla gerçekleştirlimiştir. Müşterilerin araçlara atanması için dal-sınır algoritması kullanılmıştır. Önerilen yöntem ile çözülebilen en büyük boyutlu problem 4 araç ve dar zaman aralıklarına sahip 15 müşteri içermektedir.

Daha sonraki yıllarda, ZA-ARP için iyi sonuçların elde edildiği kesin çözüm yöntemleri, sütun oluşturma tabanlı ve Lagrange gevşetmesi tabanlı yöntemler olmak üzere iki başlık altında toplanabilir. Aslında iki dekompozisyon yönteminde de problemin kısıtları aynı iki kümeye ayrılmaktadır. Böylece aynı kısıtlara sahip en kısa yol problemleri elde edilmekte, elde edilen alt sınırlar da aynı olmaktadır (Kallehauge ve diğ., 2006). Elde edilen alt-problemler NP-zor tipindedir (Dror, 1994).

2.3.1 Sütun oluşturma tabanlı yöntemler

Desrochers ve diğ. (1992) ve Kohl ve diğ. (1999) çalışmaları, Dantzig-Wolfe dekompozisyonu olarak da adlandırılan sütun oluşturma tabanlı yöntemlerin kullanıldığı çalışmalardır. Dantzig-Wolfe dekompozisyonu (Dantzig ve Wolfe, 1960) yönteminde ana problem bir küme-bölümlendirme (set-partitioning) problemidir. Ana problemin gevşetilmesi ile elde edilen doğrusal programlama modelinin çözümü için sütun oluşturma algoritması kullanılmaktadır. En kısa yol problemi, kapasite ve zaman aralıkları kısıtlarıyla çözülerek sütunlar oluşturulmuştur. Dekompozisyon, ana problem ve alt problem üzerine dayalıdır. Ana problem, amaç fonksiyonunu, her müşterinin sadece bir defa ziyaret edilmesi ve iki değerli akış değişkenleri gibi bazı kısıtları içermektedir. Alt problem ise değiştirilmiş amaç fonksiyonunu ve diğer kısıtları içermektedir. Dantzig-Wolf dekompozisyonu ile orijinal ZA-ARP her biri kapasite ve zaman aralıkları kısıtı içeren, kullanılan araç sayısı kadar en kısa yol problemine ayrıştırılmaktadır.

Kohl ve diğ. (1999), 2-yol kesimi (2-path cut) adı altında sütun oluşturma için yeni bir dal-sınır şeması geliştirmiştir. Bu çalışmada bazı 25-50 müşterilik örnek problemler ve bir kaç tane de 100 müşterilik örnek problem için optimal sonuç elde edilebilmiştir. Cook ve Rich (1999), benzer bir yaklaşımla ZA-ARP üzerinde çalışmışlardır. Problemin çözümü için sütun ve kesim oluşturma yaklaşımını kullanmışlardır. Chabrier (2006) ve Irnich ve Villeneueve (2006) sütun oluşturma tabanlı yönetmlerin kullanıldığı yakın tarihli çalışmalardır.

2.3.2 Lagrange gevşetmesi tabanlı yöntemler

ZA-ARP için Lagrange gevşetmesi, zaman aralıkları, kapasite veya bazı karar değişkenleri ile ilgili kısıtlara uygulanmıştır. Lagrange çarpanlarını içeren problem

Lagrange dual problemi olarak adlandırılmaktadır. Desrosiers ve diğ. (1988) tarafından okul servisi planlaması probleminde Lagrange gevşetmesi yöntemi ile filo büyüklüğünün optimizasyonu sağlanmıştır. Kolen ve diğ. (1987) ise yolculuk mesafesinin optimizasyonu için ZA-ARP'nde Lagrange gevşetmesi yönteminin kullanıldığı ilk çalışmadır. Madsen (1990), probleme alt sınır bulunması amacıyla Lagrange gevşetmesi yöntemini kullanmıştır. 31 müşteriye kadar büyüklükteki problemler için optimal sonuçlar bulunmuştur.

Fisher ve diğ. (1997), Lagrange gevşetmesi tabanlı iki yaklaşım ortaya koymuştur; K-ağaç (K-tree) yaklaşımı ve değişken parçalama (variable splitting) yaklaşımı. K-ağaç yaklaşımında, özel bir matematiksel model formülasyonu yapılmakta ve kısıtlara Lagrange gevşetmesi uygulanmaktadır. Değişken parçalama yaklaşımında ise bir atama problemi ve zaman aralıkları ve kapasite kısıtlı en kısa yol problemleri elde edilmektedir.

Kohl ve Madsen (1997), her müşterinin sadece bir defa ziyaret edilmesini gerektiren kısıtın gevşetilmesini içeren bir yaklaşım önermişlerdir. Ana problem optimal Lagrange çarpanlarının bulunması, alt problem ise zaman aralıklı ve kapasite kısıtlı en kısa yol problemidir. Problemin çözümü için altgradyan algoritması ve hızlandırılmış sütun oluşturma algoritması kullanmışlardır.

Kallehauge ve diğ. (2006), ZA-ARP'nin atama kısıtlarının gevşetilmesiyle oluşturulan Lagrange dual problemini çözmek için doğrusal programlama çatısı altında bir kesen düzlem algoritması geliştirmişlerdir. Kesen düzlem algoritmasını bir dal-sınır algoritması içerisinde kullanarak geliştirdikleri Lagrange dal-kesim-maliyet algoritması ile çözüm oluşturma zamanlarında önemli iyileştirmeler sağlamışlardır.

2.4 Sezgisel Çözüm Yöntemleri

ZA-ARP'nin oldukça karmaşık bir problem olması kesin çözüm yöntemlerine göre oldukça düşük hesaplama yükü ile iyi kalitede çözümler oluşturabilen etkin sezgisel yöntemlerin geliştirilmesini gerekli kılmıştır. Çünkü, endüstriyel rotalama problemi büyük boyutlu örnekleri kararlı bir şekilde çözebilecek algoritmalara ihtiyaç duymaktadır. ZA-ARP konusundaki araştırmalar optimal olması gerekli olmayan

ancak uygulanabilir sürelerde iyi kalitede çözümler üreten sezgisel yöntemler üzerinde yoğunlaşmıştır.

ZA-ARP için pek çok sezgisel yöntem geliştirilmiştir. Laporte ve Semet (2002) sezgisel yöntemleri klasik sezgisel ve metasezgisel (modern sezgisel) yöntemler olmak üzere iki gruba ayırmıştır. Cordeau ve diğ. (2002) ve Bräysy ve Gendreau (2005a, 2005b) çalışmalarında ZA-ARP için sezgisel yöntemlerin kullanıldığı çalışmalar detaylı olarak incelenmiştir.

2.4.1 Klasik sezgisel yöntemler

Yoğunlukla 1960-1990 yılları arasında geliştirilen klasik sezgiseller; kısa sürede iyi çözümlere ulaşabilmek için arama uzayının dar bir alanını kullanan özel amaçlı yöntemlerdir. Solomon (1987), ARP için geliştirilmiş olan sezgisel yöntemleri ZA-ARP için genelleştirmiştir. ARP için önerilmiş olan klasik sezgisel yöntemler zaman aralıkları ve kapasite kısıtlarının da dikkate alınmasıyla ZA-ARP için de kullanılabilmektedir. Geçmiş çalışmalarda ZA-ARP için çok sayıda klasik sezgisel yöntem kullanılmıştır. Bu yöntemler temel olarak rota oluşturma (constructive) sezgiselleri ve rota iyileştirme (improvement) sezgiselleri olarak iki sınıfa ayrılabilir.

2.4.1.1 Rota oluşturma sezgiselleri

Rota oluşturma sezgiselleri, çözümün oluşturulacağı ağ yapısı üzerinde düğümlerin veya bağların sıralı olarak uygun bir çözüm elde edilene kadar seçilmesini içeren yöntemlerdir. Her adımda, mevcut parça çözüm üzerine yeni bir rotalanmamış müşteri eklenir. Kapasite ve zaman aralıkları kısıtlarına uygun olarak belirlenen aday müşteriler kümesi içerisinden seçim işlemi, genellikle bir maliyet fonksiyonu kullanılarak gerçekleştirilir. Süreç tamamlanmadığı sürece tam bir çözüm oluşmaz. Çoğunlukla başka bir yaklaşımla çözülecek problem için başlangıç çözümünü oluşturmak için kullanılırlar.

Solomon (1986), ZA-ARP için önce rotala sonra kümele yaklaşımını önermiştir. İlk aşamada kapasite ve zaman aralıkları kısıtları gözardı edilerek tüm müşterileri içeren büyük bir araç rotası oluşturulmaktadır. Daha sonra büyük rota parçalara ayrılarak kapasite ve zaman aralıkları kısıtlarına uyan araç rotaları elde edilmektedir.

Solomon (1987), daha önce ARP için önerilmiş sezgiselleri temel alarak ZA-ARP için birçok sezgisel yöntem önemiştir. Önerilen yöntemlerden bir tanesi Clarke ve Wright (1964) tarafından ARP için önerilmiş olan tasarruf sezgiselini temel almaktadır. Tasarruf sezgiseli, araç sayısının karar değişkeni olduğu simetrik ve asimetrik problemlerde iyi sonuçlar veren bir yöntem olup aşağıdaki adımları takip ederek çözüm oluşturmaktadır.

I. Başlangıç çözümü olarak her biri (0, i, 0) olmak üzere n adet araç turu oluşturulur.

II. $s_{ij} = d_{i0} + d_{0j} - d_{ij} \quad i, j = 1, ..., n$ ve $i \neq j$ olmak üzere i ve j müşterilerini içeren rotaların birleştirilmesi halinde oluşacak olan tasarruf değerleri hesaplanır.

III. Tasarruf değerleri büyükten küçüğe sıralanarak liste oluşturulur.

Tasarruf algoritmasının paralel ve sıralı olmak üzere iki versiyonu vardır. Paralel versiyonunda listenin başındaki s_{ij} ele alınır. Mevcut turlar içerisinden iki adet birleştirilmeye uygun tur seçilir, seçilen turlardan bir tanesinin başlangıcı (0, j) bağı diğerinin bitişi ise (i, 0) bağı olmalıdır. (0, j) ve (i, 0) bağları silinerek ve (i, j) bağı eklenerek bu iki tur birleştirilir. Örnek olarak, s_{12} değeri için (0, 1, 0) turu ile (0, 2, 0) turu birleştirilerek (0, 1, 2, 0) turu oluşturulur. Sıralı versiyonda ise ele alınan (0, i, ..., j, 0) turu için ilk s_{ki} veya s_{jl} değerleri için (k, 0) ile biten veya (0, l) ile başlayan birleştirilmeye uygun tur aranır. Algoritma uygun birleşme kalmayana kadar devam eder. Solomon (1987), tur birleştirmelerinin zaman aralıkları kısıtına uygun olmasını sağlayarak ve araçların müşteri önünde bekleme sürelerine bir üst sınır getirerek tasarruf sezgiselini ZA-ARP için uyarlamıştır.

Flood (1956) tarafından önerilen en yakın komşu sezgiseli, ilk defa Solomon (1987) tarafından ZA-ARP için kullanılmıştır. Basitliği ve hesaplama hızı en önemli avantajlarındandır. En yakın komşu sezgiseli, depodan başlayarak daha önce ziyaret edilmemiş, kapasite ve zaman aralıkları kısıtlarına uygun müşterilerden, mevcut müşteriye en yakın olanını rotaya eklemek suretiyle çözüm oluşturmaktadır. Mevcut rotaya eklenecek uygun bir müşteri bulunamadığında yeni bir araç rotası oluşturulmaktadır. En yakında bulunan müşterinin tespiti için süre ve/veya mesafe değişkenlerini içeren özel bir maliyet fonksiyonu kullanılmaktadır.

Solomon (1987) tarafından I1, I2, I3 olarak adlandırılan 3 adet sıralı yerleştirme sezgiseli önerilmiştir. Geçmiş çalışmalarda en fazla kulllanılan ve en başarılı rota oluşturma sezgisellerinden biri olarak görülen I1 sıralı yerleştirme sezgiseli öncelikle kök müşteri olarak belirlenen bir müşteriyi içeren başlangıç rotası oluşturmaktadır. Daha sonra, kısıtlara uygun olarak henüz rotalanmamış müşterilerin başlangıç rotasına yerleştirilmesi yapılır. Tüm mümkün yerleştirmeler yapıldıktan sonra rotalanmamış müşteri varsa kalan müşteriler arasından yeni bir kök müşteri belirleme ve yerleştirme süreçleri tekrarlanır. Bu süreç tüm müşteriler rotalanana kadar devam eder. Kök müşterinin seçiminde depoya en uzak mesafede bulunmak veya en erken servis başlangıç zamanına sahip olmak kriterleri kullanılabilir. I1 sezgiseli iki kriter kullanarak yerleştirilecek olan müşteriyi ve seçilen müşterinin yerleştirileceği pozisyonu belirlemektedir. İlk kriter, henüz ziyaret edilmemiş her bir müşteri için mevcut rota veya rotalara kısıtlara uygun bir şekilde yerleştirilebileceği en düşük maliyetli pozisyonu belirlemektedir. $C_1(i,u,j)$ fonksiyonu ile u müşterisinin en az maliyetle ardışık i ve j müşterileri arasına yerleştirilebileceği belirlenir.

$$C_1(i,u,j) = \min_{(i,j)} \{\alpha_1 C_{11}(i,u,j) + \alpha_2 C_{12}(i,u,j)\} \quad \textbf{(2.11)}$$

$$C_{11}(i,u,j) = d_{iu} + d_{uj} - \alpha_3 d_{ij} \quad \alpha_3 \geq 0 \quad \textbf{(2.12)}$$

$$C_{12}(i,u,j) = h_j^{yeni} - h_j \quad \textbf{(2.13)}$$

$$\alpha_1 + \alpha_2 = 1 \quad \textbf{(2.14)}$$

İkinci kriter, $C_2(i,u,j)$ fonksiyonu ile yerleştirme durumundaki tasarruf değerlerini hesaplayarak, yerleştirilecek müşteriyi (u^*) ve yerleştirme pozisyonunu (i^*, j^* arasına) belirlemektedir.

$$C_2(i^*,u^*,j^*) = \underset{u}{\text{maks}} \{\alpha_4 d_{0u} - C_1(i,u,j)\}, \quad \alpha_4 \geq 0 \quad \textbf{(2.15)}$$

$C_2(i,u,j)$ fonksiyonu aslında yeni bir rota oluşturma ile mevcut rotaya müşteri ekleme maliyetleri arasındaki farkı hesaplamaktadır.

Solomon (1987) tarafından önerilen I2 yerleştirme sezgiseli, müşteri seçiminde toplam rota uzunluğunu veya toplam rota süresini en az arttıracak bir yerleştirme maliyeti kullanılır. I3 yerleştirme sezgiseli ise müşteri seçiminde kriter olarak müşteriye servis verilmesinin aciliyetini kullanmaktadır. Potvin ve Rousseau (1993), I1 sezgiselinin parallel versiyonunu önermişlerdir. Çok sayıda kök müşteri belirlenerek müşteri yerleştirmesi yapılmıştır. Dullaert (2000) ve Dullaert ve Bräysy (2003), I1 sezgiselindeki C_{12} ölçütüne toplam rota süresindeki değişimi de ekleyerek daha iyi sonuçlar elde etmişlerdir.

Gillet and Miller (1974) tarafından önerilen tarama (sweeping) sezgiseli düzlemsel boyuttaki ARP örneklerinde kullanılan bir algoritmadır. Öncelikle depoyu merkez alan bir ışının döndürülmesi ile uygun kümeler belirlenir. Daha sonra her küme en kısa yol problemi olarak ele alınır. Solomon (1987), tarama sezgiselini temel alarak ZA-ARP için zamana yönelik tarama sezgiselini (time-oriented sweep heuristic) önermiştir. Müşterilerin rotalara yerleştirilmesinde I1 sezgiseli kullanılmıştır.

Ioannou ve diğ. (2001), Solomon (1987) tarafından önerilen sıralı yerleştirme yapısını farklı kriterlerle kullanan bir sezgisel yöntem önermişlerdir. Rotalanmamış bir müşterinin bir araç rotasına yerleştirilmesi için rotada bulunan müşterilerin mevcut durumlarına en az düzeyde etki, seçim kriteri olarak kullanılmıştır.

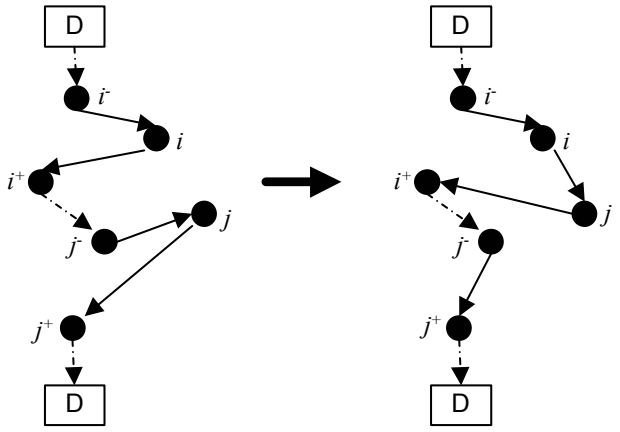
2.4.1.2 Rota iyileştirme sezgiselleri

Rota iyileştirme sezgiselleri, genellikle rota oluşturucu bir sezgisel tarafından oluşturulan uygun bir başlangıç çözümü üzerinde seri değişiklikler ile (hareket olarak adlandırılır) çözümün kalitesini arttıran yerel arama sezgiselleridir. Bir rota geliştirici sezgiselin tasarımı için; başlangıç çözümünün nasıl elde edileceği, hangi hareket mekanizmasının kullanılacağı, kabul kriteri ve durma kriteri tanımlanmalıdır. Bazı rota iyileştirme sezgiselleri mevcut bir çözüm yerine kendi oluşturduğu bir çözümü, başlangıç çözümü olarak kullanmaktadır. Yani aşamalı olarak önce rotalar oluşturulmakta sonra da oluşturan rotalar iyileştirilmektedir. Bu tip yaklaşımlar iki aşamalı sezgiseller olarak da adlandırılmaktadır.

Hareketler, geliştirilecek olan çözüm için bir komşuluk bölgesini tanımlamaktadır. Hareketler araç rotalama probleminin tanımlanmış olduğu ağ yapısı üzerindeki düğümler ve/veya bağlar üzerinde yapılacak olan işlemler ile tanımlanır. İyileştirme

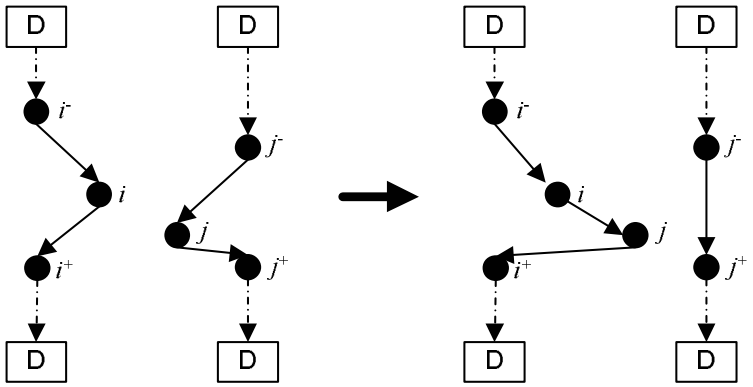
sadece her aracın belirlenen rotasını daha iyi hale getirmek veya çoklu rotaların iyileştirilmesi şeklinde olabilir. ARP için geçmiş çalışmalarda sıklıkla kullanımı görülen hareketler aşağıda tanıtılmıştır (i^+ ve i^-, i müşterisinden bir sonraki ve bir önceki müşterileri temsil etmektedir):

Tekrar yerleştirme (relocation) hareketi : Mevcut bir rotada bulunan i ve j müşterileri için, j müşterisi i'nin hemen sonrasına yerleştirilir. Bir araç rotası içerisinde veya farklı araç rotaları arasında gerçekleştirilebilir.



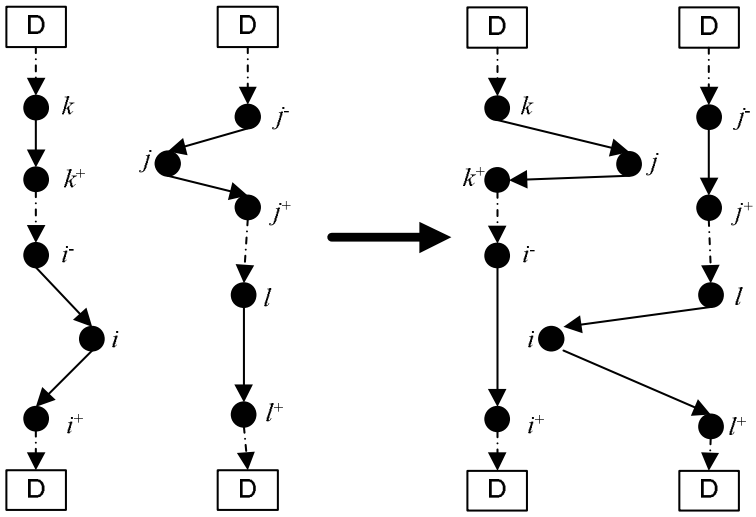
Şekil 2.3: Bir araç rotasında tekrar yerleştirme hareketi

Bir araç rotası için, Şekil 2.3'te görüldüğü gibi (i, i^+), (j^-, j), (j, j^+) bağları silinmesi ve (i, j), (j, i^+),(j^-, j^+) bağlarının eklenmesiyle gerçekleştirişir. Farklı araç rotaları arasında tekrar yerleştirme hareketi, Şekil 2.4'te görüldüğü gibi (i, i^+), (j^-, j), (j, j^+) bağları silinmesi ve (i, j), (j, i^+),(j^-, j^+) bağları eklenmesiyle gerçekleştirilir.



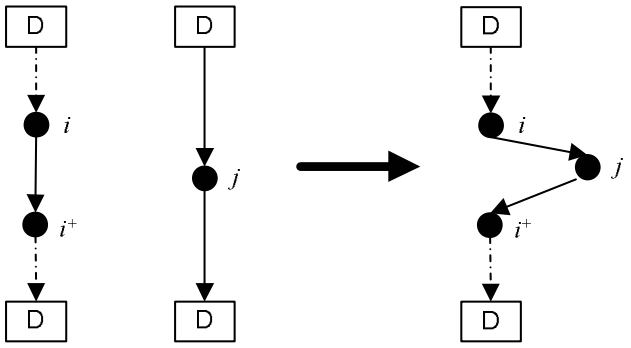
Şekil 2.4: Farklı araç rotaları arasında tekrar yerleştirme hareketi

Değiştirme (exchange) hareketi : Farklı iki araç rotasında bulunan *i* ve *j* müşterilerinin rotaları değiştirilir. Müşteriler birbirlerinin pozisyonlarına veya farklı pozisyonlara yerleştirilebilir. Şekil 2.5'te yer değiştirmenin farklı pozisyonlara yapıldığı örnek bir durum gösterilmiştir: *i* müşterisinin *l* müşterisinin önüne, *j* müşterisinin ise *k* müştersinin önüne yerleştirilmesi için (k, k^+), (i^-, i), (i, i^+), (j^-, j), (j, j^+), (l, l^+) bağları silinip, (k, j), (j, k^+), (i^-, i^+), (j^-, j^+), (l, i), (i, l^+) bağları eklenmektedir.



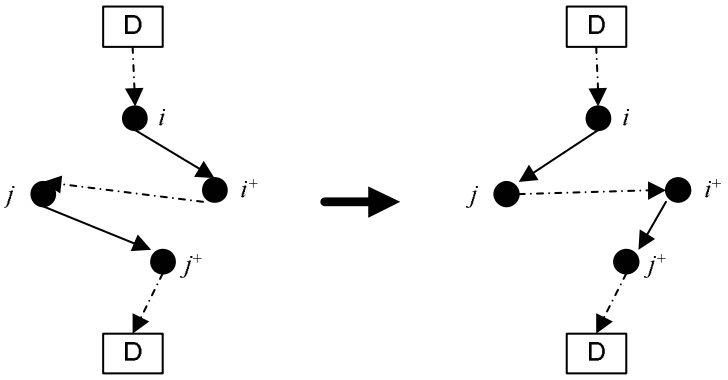
Şekil 2.5: Değiştirme hareketi

Rota silme hareketi : Mevcut bir çözümde kullanılmakta olan araç sayısını azaltmak amacıyla kullanılır. Bir rota iptal edilerek müşterilerinin diğer rotalara yerleştirilmesi yapılır. İptal edilecek rotanın belirlenmesinde genellikle müşteri sayısının az olması kriteri kullanılır. İptal edilen rotada bulunan müşterilerin diğer rotalara yerleştirilmesinde ise farklı seçim ve yer belirleme kriterleri kullanılmaktadır. Şekil 2.6'da rota silme hareketi için örnek bir durum gösterilmiştir.

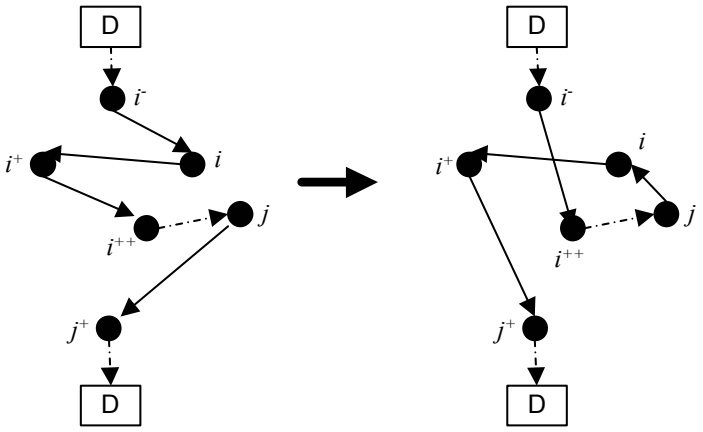


Şekil 2.6: Rota silme hareketi

Rota iyileştirme sezgiselleri genellikle Lin (1965) ve Or (1976) tarfından ortaya konan *k*-opt ve Or-opt bağ değiştirme (edge exchange) süreçlerini temel almaktadır. *k*-opt sezgiseli, mevcut rotada bulunan *k* adet bağı başka bir *k* adet bağ ile değiştirmektedir, bu hareket *k*-değiştirme olarak adlandırılır. Herhangi bir *k*-değiştirme hareketi ile geliştirilemeyen bir rota *k*-optimal olarak adlandırılır. Sezgiselin karmaşıklığı *k* değerinin büyüklüğüne bağlıdır. Büyük *k* değerleri için sezgisel daha güçlü hale gelmektedir ancak hesaplama yükü de artmaktadır. Şekil 2.7'de 2-değiştirme veya 2-opt olarak adlandırılan hareket gösterilmektedir. (i, i^+) ve (j, j^+) bağları (i, j) ve (i^+, j^+) bağları ile değiştirilmiş, i^+ ve j arasında bulunan müşterilerin sırası ters çevrilmiştir.



Şekil 2.7: 2-opt değiştirme hareketi

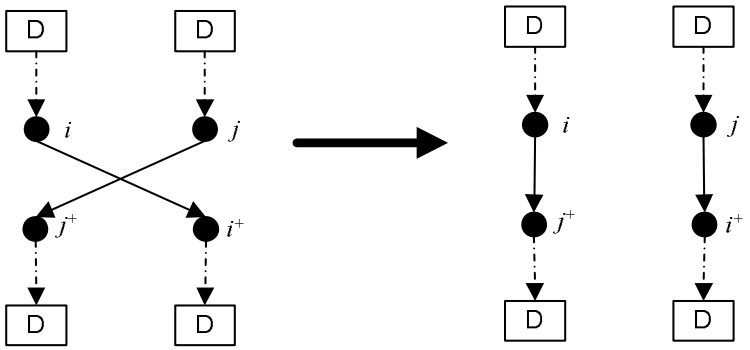


Şekil 2.8: Or-opt hareketi

Or-opt hareketi ise en fazla üç müşteriden oluşan bir zinciri mevcut rotadan çıkarmakta ve bu zinciri tüm olası yerlere eklemeye çalışmaktadır. Zincir içerisinde bulunan müşterilerin sıralaması değiştirilmemektedir. Şekil 2.8'de Or-opt hareketi gösterilmiştir. (i^-, i), (i^+, i^{++}), (j, j^+) bağları (i^-, i^{++}), (j, i), (i^+, j^+) bağları ile

değiştirilerek i ve i^+ müşterileri j ve j^+ müşterileri arasına yerleştirilmiştir. Baker ve Schaffer (1986), Savelsbergh (1985), Solomon ve Desrosiers (1988) çalışmalarında k-opt ve Or-opt temelli sezgiseller ortaya konmuştur.

Potvin ve Rousseau (1995) çalışmasında ZA-ARP için 2-opt, 3-opt ve Or-opt sezgisellerini karşılaştırılmış ayrıca 2-opt* değiştirme sezgiseli önerilmiştir. 2-opt* değiştirme sezgiselinde iki farklı rota karıştırılmaktadır. Karıştırma esnasında öncelikle her iki rotadan birer bağ silinmekte, daha sonra mevcut rotalarda silinen bağlardan sonraki müşteriler, kendi aralarındaki sıralamaları değiştirilmeden, rotalar arasında yer değiştirilmektedir.

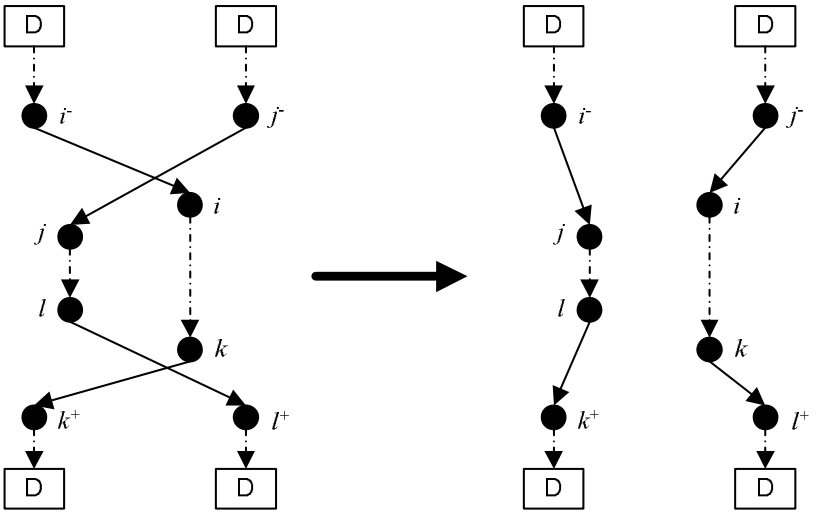


Şekil 2.9: 2-opt* hareketi

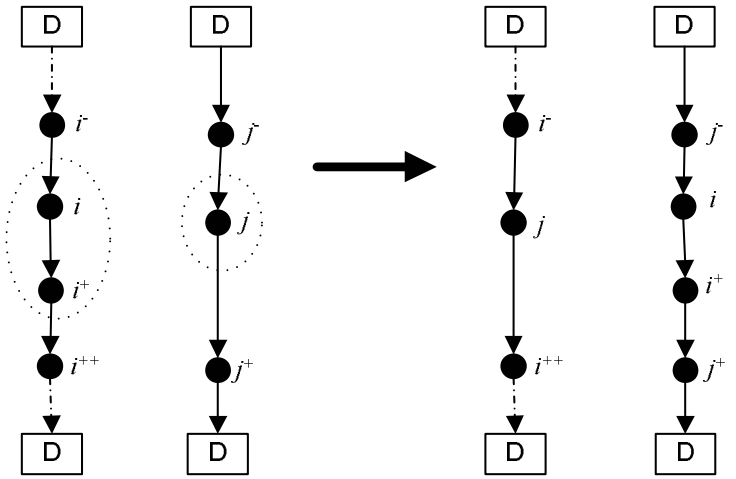
Şekil 2.9'daki örnek durumda görüldüğü gibi 2-opt* değiştirme hareketi, farklı rotalarda bulunan i ve j müşterileri için (i, i^+) ve (j, j^+) bağlarının silinmesi ve (i, j^+) ve (j, i^+) bağlarının rotalara eklenmesiyle gerçekleştirlebilir.

Taillard ve diğ. (1997), 2-opt* ve Or-opt değiştirmelerini genelleştiren çapraz-değiştirme (CROSS-exchange) sezgiselini önermiştir. Çapraz değiştirme hareketi ile birinci rotadan (i^-, i) ve (k, k^+) bağları, ikinci rotadan ise (j^-, j) ve (l, l^+) bağları silinerek farklı sayılarda müşteri içeren i-k ve j-l yolları rotalar arasında yer değiştirilmekedir (Şekil 2.10).

Osman (1993) λ-değiş tokuş (λ-interchange) sezgiselini önermiştir. Bu sezgisel 2-opt hareketlerini, farklı rotalara düğümlerin tekrar atanmasını ve iki rota arasında düğüm değişimlerini içeren kombine bir yöntemdir. λ-değiş tokuş sezgiseli, mevcut bir çözümdeki rotaların tüm ikili kombinasyonları için gerçekleştirilir. Rotalar arasında λ ve daha küçük boyuttaki tüm alt yolların yer değiştirmesi denenir. Yer değiştirmeler operasyon olarak tanımlanır.



Şekil 2.10: Çapraz-değiştirme hareketi



Şekil 2.11: λ-değiş tokuş sezgiseli için (2,1) operasyonu örneği

Şekil 2.11'de λ-değiş tokuş sezgiselinde λ=2 için (2,1) operasyonunun gösterimi yapılmıştır. (2,1) operasyonunun anlamı birinci rotadan iki ardışık müşterinin ikinci rotadan bir müşteri ile yer değiştirilmesidir. Şekil 2.11'deki durumda λ=2 için gerçekleştirilebilecek tüm operasyonlar; (0,1), (0,2), (1,0), (1,1), (1,2), (2,0), (2,1), (2,2) olarak sıralanabilir. (1,0) operasyonunun anlamı birinci rotadan bir müşterinin ikinci rotaya kaydırılacağıdır.

Glover (1992), GENI-değiştirme hareketini kullanan püskürtme (ejection) zinciri sezgiselini önermiştir. GENI-değiştirme hareketi, mevcut rotasından çıkarılan bir müşterinin başka bir rotaya yerleştirilmesini içerir. Yerdeğiştirme hareketinden farklı olarak öncelikle müşterinin yerleştirileceği hedef rotadan bir müşteri çıkarılarak

yerleştirilmek istenen müşteriye yer açılması sağlanır. Böylece her hareket sonunda bir müşteri açığa çıkacaktır. GENI-değiştirme hareketi, hedef rotadan bir müşteri çıkarılmadan yerleştirilme yapılana kadar tekrar etmektedir.

Antes ve Derigs (1995), rota oluşturma ve rota iyileştirme süreçlerini içeren paralel bir sezgisel kullanmışlardır. Rotalanmamış müşteriler için I1 sezgiselindeki maliyet fonksiyonu ile yerleştirilebilecekleri rotalar için maliyetler belirlenmekte, daha sonra rotalar kendileri için en az maliyetli yerleştirmeyi kabul etmektedir. Uygun bir çözüm elde edildiğinde, çözüm içerisinde bulunan en verimsiz müşterilerin başka rotalara kaydırılması ile iyileştirme yapılmaktadır.

Russel (1995) çalışmasında öncelikle Potvin ve Rousseau (1993) çalışmasındaki rota oluşturma sezgiselineine benzer bir yöntemle rotalar oluşturulmakta, daha sonra ardışık dört müşterinin mevcut rotadan silinerek başka rotalara yerleştirilmesiyle çözümlerde iyileşme sağlanmaktadır.

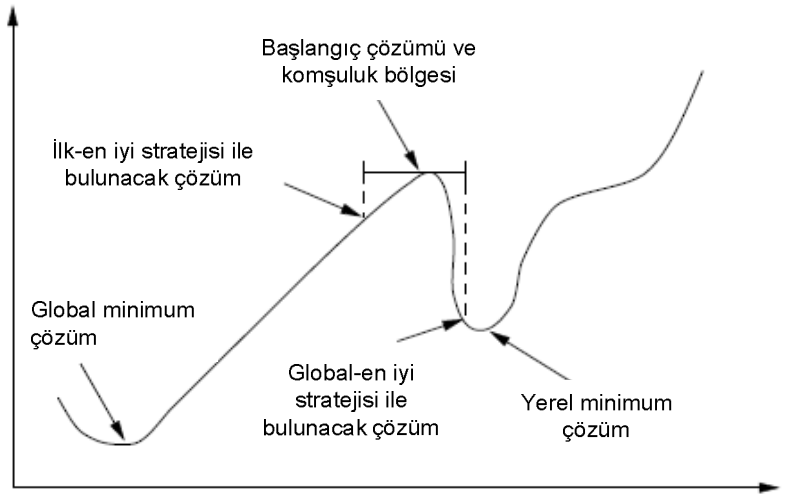
Shaw (1997) ve Shaw (1998) çalışmalarında, kısıt tatmini teknikleri ile mevcut rotalardaki müşterileri tekrar çizelgeleyen bir büyük komşuluk arama sezgiseli kullanılmıştır. Rassal olarak seçilen müşterilerin kısıt tatmini ve dal-sınır yöntemleri ile optimal olarak yerleştirilmesi yapılmıştır. Çok fazla hesaplama yükü gerektirdiğinden rotalarda bulunan müşteri sayılarının az olduğu problemlere uygulanabilmiştir.

Caseau ve Laburthe (1999), özellikle büyük boyutlu rotalama problemlerinde kullanılmak üzere bir sezgisel geliştirmişlerdir. Çözüm oluşturma aşamasında, her müşteri yerleştirmesinden sonra 2-opt* hareketi uygulanmaktadır. Açıkta bulunan bir müşteri için uygun yerleştirme yapılamadığında mevcut parça çözüme farklı hareketler uygulanarak müşteriye yer açılmaktadır.

Cordone ve Wolfer-Calvo (2001), *k*-opt değişimi hareketlerini kullanan deterministik bir sezgisel önermişlerdir. Arama sürecinin yerel optimumlara yakınsamasını önlemek için değişmeli olarak yolculuk mesafesinin veya süresinin minimizasyonu amaçlı arama yapılmaktadır. Başlangıç çözümlerini oluşturmak için I1 sezgiseli kullanılmıştır. Başlangıç çözümlerinin her birine yerel arama (2-opt veya 3-opt) uygulandıktan sonra en iyi olanı seçilmektedir.

Bräysy (2002), yeni bir yaklaşımla ZA-ARP için sezgisel bir yöntem geliştirmiştir. Üç aşamadan oluşan yaklaşımın birinci aşamasında rota oluşturma sezgiselleri farklı parametre kombinasyonları ile kullanılarak çok sayıda başlangıç çözümü üretilmektedir. Rota oluşturma sezgiselleri, Solomon (1987) ve Russel (1995) çalışmalarını temel almaktadır. İkinci aşamada, tasarruf sezgiselini (Clarke ve Wright, 1964) ve püskürtme sezgiselini (Glover, 1992) temel alan yeni bir sezgisel ile rota sayısının azaltılması amaçlanmıştır. Üçüncü aşamada ise toplam mesafeyi azaltmak içi Or-opt sezgiseli kullanılmıştır.

Mevcut bir ψ çözümü için $N(\psi)$, ψ çözümünün komşuluk bölgesini tanımlamaktadır. Yani, ψ çözümü üzerinde yukarıda tanımlanan hareket opertörlerinin kullanılmasıyla ulaşılabilecek çözümlerin oluşturduğu kümeyi tanımlamaktadır. Yerel arama sezgiselleri tanımlanan komşuluk bölgesinde gerçekleştirilen arama sürecinin sona erdirilmesi için iki farklı strateji kullanmaktadırlar. Birinci strateji, mevcut çözümün komşuluk bölgesinde ilk daha iyi çözümün bulunduğu anda aramanın sona erdirilmesini sağlayan ilk en iyi (First-Best) stratejisidir. İkincisi ise mevcut çözüm komşuluk bölgesinin tamamen aranmasından sonra aramanın sona erdirilmesini sağlayan global en iyi (Global-Best) stratejisidir.



Şekil 2.12: Komşuluk bölgesi arama stratejileri

İlk en iyi stratejisinde komşuluk bölgesindeki aramanın yönü (sırası) performans üzerinden etkilidir. Global en iyi stratejisinde ise komşuluk bölgesindeki tüm çözümler kontrol edileceğinden aramanın yönü önemli değildir. Doğal olarak global en iyi stratejisinin hesaplama yükü ve süresi daha fazladır. Genellikle global en iyi

stratejisi daha iyi sonuçlara ulaşılmasını sağlayabilir ancak Şekil 2.12'de görüldüğü gibi ilk en iyi stratejisinin daha iyi performans göstereceği durumlar da olabilir. Şekil 2.12'de global en iyi stratejisi ile algoritma yerel bir minimum çözüme yakınsayarak durgunluk haline geçebilir. İlk en iyi stratejisinin kullanımı durumunda ise algoritma global minimum çözüme ulaşabilecektir.

Yerel arama yöntemleri ile elde edilen bir çözüm, optimal çözümün çok uzağında da olabilir. Yerel arama yöntemleri, hareket operatörü ile tanımlanan sıralama ile amaç fonksiyonunda iyileşme sağlayan çözümleri yeni başlangıç çözümü kabul ederek aramayı sürdürdüklerinden performansları genellikle başlangıç çözümüne ve kullanılan komşuluk bölgesine oldukça bağımlıdır.

2.4.2 Metasezgisel yöntemler

İlk defa Glover (1986) tarafından kullanılan *metaheuristic* kelimesi aslında Yunanca daha büyük, daha üst seviye anlamındaki $\mu\varepsilon\tau\alpha$ (meta) ile aramak anlamındaki $\varepsilon\upsilon\rho\iota\sigma\kappa\omega$ (heuristic) kelimelerinin birleşiminden oluşmaktadır. Türkçe kaynaklarda ise metasezgisel veya metahöristik olarak kullanıldığı görülmektedir. Stützle (1998a) metasezgisel yöntemleri şöyle tanımlamıştır; *"metasezgiseller, probleme özgü sezgisellere performanslarını arttırmak için rehberlik eden üst düzey stratejilerdir. İterasyonlarla ilerleyen yerel aramanın yerel bir optimum noktada saplanıp kalmasına engel olmak temel amaçlarıdır. Bunun için aramanın daha kötü çözümlere geçiş yapmasına izin verebilir veya yerel arama için oluşturulan başlangıç çözümlerinin tamamen rassal olarak değil de daha zeki bir yöntemle oluşturulmasını sağlayabilirler. Çoğu metasezgisel yöntem, yüksek kalitede çözümleri çabukça bulabilen eğilimleri öneren yöntemler olarak yorumlanabilir. Önerilen eğilim değişik formlarda olabilir. Amaç fonksiyon değerini temel alan azaltma eğilimi, daha önceki kararları temel alan bellek eğilimi veya geçmiş performanslara dayalı deneyim eğilimi gibi örnekler verilebilir. Metasezgisellerin çoğu arama esnasında olasılıklara dayanan seçimler yaparlar. Ancak şansın önemli olduğu rassal arama süreçlerinden farklı olarak seçim olasılıkları zeki bir seçim yapılabilmesini sağlayacak şekilde oluşturulur."*

Yoğunlukla 1990 - günümüz arasında geliştirilmiş olan metasezgiseller; komşuluk arama kurallarını birleştiren, hafıza yapıları kullanan ve geçmişte oluşturdukları çözümlerin kombinasyonunu ile arama sürecini zekileştiren genel amaçlı

mekanizmalardır. Yoğunlaştırma (intensification) ve çeşitlendirme (diversification) stratejilerini bir arada kullanırlar. Arama sürecine, çözüm uzayının uygun çözümler bulunmayan bölgelerini de dahil edebilirler. Gendreau ve diğ. (2002)'e göre ARP için en etkin ve umut vaat eden yöntemlerdir. Metasezgisel yöntemler, zor kombinatoryal optimizasyon problemler için iyi kalitede çözümlere ulaşabilmek için sezgisel yaklaşımları kullanan/yöneten tasarımlar olarak da adlandırılabilir. Blum ve Roli (2003), metasezgisellerin temel özelliklerini aşağıdaki gibi genelleştirmiştir:

- Aramayı optimal veya optimala yakın çözümlere doğru yönlendirerek arama uzayını etkin bir şekilde araştırırlar.
- Arama uzayının dar alanlarında sıkışıp kalmayı önleyecek mekanizmalara sahiptirler.
- Geçmiş aramalarda elde ettikleri tecrübeyi kullanırlar.
- Arama alanına yönelik bilgiyi (probleme özel bilgiyi) kullanmak için klasik sezgiselleri bir kontrol mekanizması ile kullanırlar. Kontrol mekanizması genellikle dinamiktir. Geçmiş aramalardan elde edilen bilginin kullanımı ile sağlanan ve aramayı iyi çözümler etrafına yönelten yoğunlaştırma stratejisi ile aramayı daha önce incelenmemiş bölgelere yönlendiren çeşitlendirme stratejileri arasındaki denge, kontrol mekanizması ile sağlanır.

Bir çok optimizasyon yaklaşımında, metasezgiseller, klasik sezgisel yöntemlerin dezavantajlı yönlerini ortadan kaldırmak amacıyla kullanılmıştır. Metasezgisellerin zeki olarak tanımlanabilecek arama stratejileri, klasik sezgisellerin yerel optimum çözümlerde takılıp kalmalarını engellemektedir.

ZA-ARP için metasezgisel yöntemler genellikle rota oluşturma ve rota iyileştirme sezgisellerine rehberlik etme amacıyla kullanılmaktadır. Rota oluşturma sezgiseli kullanılarak probleme yerel bir çözüm (genellikle başlangıç çözümü) oluşturulmakta daha sonra metasezgisel yöntemle yerel çözüm geliştirilmektedir. Rota iyileştirme sezgiseli ise mevcut çözümü veya çözümleri iyileştirmek için metasezgisel yöntemin içinde yer almaktadır.

ZA-ARP alanındaki güncel çalışmaların metasezgisel yöntemler üzerinde yoğunlaştığı görülmektedir. Bräysy ve Gendreau (2005b) bu konuda yapılan geçmiş çalışmaları incelemişlerdir. Tabu arama, tavlama benzetimi, genetik algoritmalar ve KKO metasezgiselleri ZA-ARP için en fazla kullanılan metasezgisel yöntemlerdir.

2.4.2.1 Tavlama Benzetimi Algortiması

Kirkpatrick ve diğ. (1983) tarafından önerilen tavlama benzetimi (TB) algoritması fiziksel bir süreç olan ve tavlama olarak adlandırılan, kararlı bir madde elde edilebilmesi için metallerin soğutulma stratejisinden esinlenerek geliştirilmiştir. Tavlama süreci ile kombinatoryal optimizasyon problemlerinin çözümüne yönelik bir süreç oluşturulması için aşağıdaki benzerlikler kullanılmıştır:

- Kombinatoryal optimizasyon problemindeki mevcut çözüm ve komşu çözümleri fiziki sistemin durumlarına benzetilmiştir. Mevcut çözüm ile komşu çözüm arasında küçük farklılıklar vardır.
- Amaç fonksiyonu enerjinin durumuna benzemektedir.
- Tavlama sürecindeki ısı bir kontrol parametresi ile modellenmiştir. Isı yüksek olduğunda büyük değişimlere izin verilmekte, ısı düştüğünde daha küçük değişimlere izin verilmekte, sıfıra yaklaştığında ise değişikliklere izin verilmemektedir.

TB algoritmasında, komşu bir çözüm mevcut çözümden daha iyi bir amaç fonksiyonu değerine sahipse tavlama süreci komşu çözümü yeni mevcut çözüm olarak kabul etmektedir. Diğer durumda ise (daha kötü bir amaç fonksiyon değerine sahip komşu çözüm bulunması) komşu çözümün yeni çözüm olarak kabul edilebilmesi için bir olasılık hesaplanmakta ve bu olasılık değerine uygun olarak çözümün kabulüne veya reddedilmesine karar verilmektedir.

Osman (1993), iki rota arasında λ adet müşteriye kadar değişime olanak sağlayan, λ-değişim komşuluk yapısı kullanan bir TB önermiştir. İlk iyileşme sağlayan çözüm komşu çözüm olarak belirlenmektedir. Algoritmanın soğuma çizelgesine uygun olarak, arama sürecinde ısı sürekli düşmektedir. Mevcut çözüm bir komşusu ile değiştirildiğinde ise ısı yarıya inmektedir.

Chiang ve Russel (1996), ZA-ARP için Russel (1995)'in paralel çözüm oluşturucu sezgiselini yönlendirmek amacıyla TB kullanmıştır. Başlangıç çözümleri oluşturulurken TB periyodik olarak devreye girmektedir.

Thangiah ve diğ. (1994) çalışmasında, ZA-ARP için önerilmiş olan metasezgisel yöntemlerin başlangıç çözümlerinin iyileştirilmesi aşamasında karma TB yöntemleri kullanılmıştır.

Bent ve Van Hentenryck (2004) ise ZA-ARP için TB'nin kullanıldığı yakın tarihli bir çalışmadır. İki aşamalı bir karma metsezgisel kullanmışlardır. İlk aşamada TB ile araç sayısının azaltılmasına çalışılmakta ikinci aşamada ise büyük ölçekli bir komşuluk arama sezgiseli ile yolculuk mesafesinin azaltılması sağlanmaktadır.

2.4.2.2 Tabu Arama

Fred Glover tarafından 1986 yılında ortaya konan tabu arama (TA) metasezgiseli, bellek tabanlı bir yöntemdir (Glover, 1986; Glover ve Laguna, 1997). TA yönteminin her iterasyonunda bir hareket operatörü (2-değiştirme gibi) ile mevcut çözümün komşuluk bölgesinde bulunan çözümler oluşturulur. Bu çözümler içerisinde en iyi olan, yeni mevcut çözüm olur. Bu süreç bir durma kriteri şartı sağlanana kadar devam eder. Komşuluk bölgesinde bulunan en iyi çözüm, mevcut en iyi çözümden daha kötü olsa bile seçilmektedir. Arama sürecinin yakın geçmişinde elde edilen çözümlere ulaştıran hareketler yasaklanarak benzer çözümlerin tekrar oluşturulması engellenir. Yasaklanan hareketler tabu hareket olarak adlandırılır ve bir tabu listesinde (yasaklılar listesi) saklanır. Tabu listesi dinamiktir, yeni hareketler listeye girdikçe eskileri silinir, bu nedenle kısa dönemli bellek (short-term memory) olarak da adlandırılmaktadır. Ayrıca arama sürecini yönlendirebilmek amacıyla orta ve uzun dönemli bellek yapılarının da kullanımı mümkündür. Farklı yoğunlaşma ve çeşitlendirme stratejilerinin kullanımı ve uygun bir yasaklı liste yapısı ile iyi sonuçlara ulaşılabilmektedir.

ARP için en başarılı ve ilk TA uygulamalardan biri Osman (1993) tarafından ortaya konmuştur. Komşuluk yapısı λ-değişim (λ=2) olarak tanımlanmıştır. İlk en iyi veya global en iyi stratejileri ile komşuluk bölgesinde arama yapılmaktadır.

ZA-ARP için TA yönteminin kullanıldığı ilk çalışma ise Garcia ve diğ. (1994) tarafından gerçekleştirilmiştir. Bu çalışmada başlangıç çözümünü oluşturmak için I1 yerleştirme sezgiseli, mevcut çözümü iyileştirmek için ise 2-opt* değiştirme ve Or-opt değiştirme sezgiselleri kullanılmıştır. Daha sonraki tarihlerde yapılan çalışmalarda ise özel çeşitlendirme ve yoğunlaşma stratejilerinin kullanımı, araç sayısını azaltmak için farklı stratejilerin kullanımı, karmaşık yeniden optimizasyon tekniklerinin kullanımı, tavlama benzetimi veya genetik algoritma gibi diğer arama yöntemleriyle birlikte karma uygulamalar ve paralel uygulamalar görülmektedir.

Chiang ve Russell (1997) arama esnasında tekrarları önlemek için parametrelerini dinamik olarak değiştirebilen bir tepkili tabu arama (reactive tabu search) yöntemini ZA-ARP için önermişlerdir.

Schulze ve Fahle (1999) çalışmasında Clarke ve Wright (1964) tarafından geliştirilen tasarruf sezgiseli, Cordeau ve diğ. (2001) çalışmasında ise değiştirilmiş bir tarama sezgiseli başlangıç çözümünün oluşturulması amacıyla kullanılmıştır.

Rochat ve Taillard (1995), ZA-ARP için uyumlu bellek (adaptive memory) olarak adlandırdıkları bir stokastik çeşitlendirme ve yoğunlaşma stratejisi önermişlerdir. Daha sonra Taillard ve diğ. (1997) ve Schulze ve Fahle (1999) benzer stratejiyi farklı komşuluk yapıları ile kullanmışlardır. Uyumlu bellek (UB), çözüm kümesi içerisindeki "en iyi" elemanların (parça çözümlerin) izini takip edebilen özel bir veri yapısıdır. UB oluşturulduktan sonra "en iyi" elemanlar birleştirilerek geçici bir çözüm elde edilir. Geçici çözüm yerel arama ile geliştirilerek UB güncellenir. UB'nin başlangıç safhalarında değişik çözümlerin farklı elemanları birleştirilerek çeşitlendirme sağlanırken süreç ilerledikçe UB elemanları sınırlı sayıda arama bölgesinden çözüm kümesi elemanları içermeye başlar, böylece yoğunlaşma sağlanır. Ho ve Hougland (2004) ve De Backer ve diğ. (2000) ise ZA-ARP için yakın tarihli TA çalışmalarındandır.

2.4.2.3 Genetik Algoritmalar

J. Holland tarafından 1975 yılında geliştirilen genetik algoritma (GA)'ların karmaşık problemlerin çözümünde pratik anlamda kullanılabilirliği Goldberg (1989) tarafından gösterilmiştir. GA, doğal evrim sürecinden esinlenilerek oluşturulmuştur. Aday çözümlerden oluşan popülasyon nesiller boyunca doğum, ölüm ve mutasyon mekanizmalarının etkisinden geçirilerek iyileşme sağlanır. Basit bir GA'da aday bir çözüm bir sayı dizisi ile temsil edilmek üzere kodlama yapılır. Genellikle ikili (binary) kodlama ile 0-1 değerlerinden oluşan diziler ile çözümler temsil edilir. Her aday çözüm için uygunluk fonksiyonu ile bir uygunluk değeri hesaplanır. Yeni nesilleri oluşturacak ebeveynlerin seçiminde uygunluk değeri yüksek olan çözümlerin seçilme şansı daha fazladır. İki ebeveyn birey çeşitli mekanizmalarla birleştirilerek yeni nesilin çocuk bireyleri oluşturulur. Bu işlem çaprazlama olarak adlandırılır. Ayrıca mutasyon ile bazı mevcut bireylerin bazı özelliklerinde rassal

değişiklikler yapılarak yeni nesile geçişleri sağlanır. Yeni nesil, çocuk bireylerden, mutasyona uğramış bireylerden ve eski nesilin uygunluk değeri yüksek olan bireylerinden oluşur. Genellikle nesillerde bulunan birey (aday çözüm) sayısı sabittir. Böylece eski nesillerdeki uygunluk değeri düşük bireylerin yeni nesillere geçişi engellenirken çaprazlama ile aramanın iyi çözümler bölgesinde yoğunlaşması, mutasyon ile aramanın çeşitlendirilmesi sağlanır.

GA'nın ZA-ARP için ilk uygulaması Thangiah ve diğ. (1991) tarafından gerçekleştirilmiştir. Thangiah (1995) çalışmasında daha detaylı olarak ele alınan uygulama çalışması, önce kümele sonra rotala stratejisi içermektedir. GA kümeleme aşamasında kullanılmıştır.

Blanton ve Wainwright (1993) çalışmasında müşteri sıralamaları kromozomlar ile temsil edilmiştir. Arama süreci, müşteriler için uygun bir sıralamanın bulunmasına yönlendirilmiştir.

Potvin ve Bengio (1996), iki ayrı ebeveyn çözümden alınan rota parçalarını birleştirerek çocuk birey oluşturan çaprazlama operatörü ve çözümdeki rota sayısını azaltmayı amaçlayan bir mutasyon operatörü ile ZA-ARP için GA kullanmışlardır.

Berger ve diğ. (1998), ZA-ARP için belirli müşterileri mevcut rotalarından çıkarıp, yapıcı bir sezgisel ile tekrar çizelgeleyen hibrid bir GA yaklaşımı önermişlerdir. Berger ve Barkaoui (2000) geliştirdikleri yeni çaprazlama ve mutasyon operatörlerini kullanarak Berger ve diğ. (1998) çalışmasını geliştirmişlerdir.

Homberger ve Gehring (1999), önerdikleri GA yaklaşımında 3 adet rota iyileştirme sezgiseli (Or-opt değişim, λ-değiş tokuş, 2-opt değişim) kullanmışlardır.

Tan ve diğ. (2001), ZA-ARP için düzensiz GA (messy GA) kullanmışlardır. I1 yerleştirme sezgiselini kullanarak aday çözümleri oluşturmuşlardır. Gehring ve Homberger (2001), Homberger ve Gehring (2005) ve Alvarenga ve diğ. (2007) ZA-ARP için GA yaklaşımını kullanan yakın tarihli çalışmalardır.

2.4.2.4 Karınca Kolonisi Optimizasyonu

KKO, yapay karıncaların yiyecek arama davranışlarından esinlenerek oluşturulmuş bir metasezgiseldir. Kombinatoryal problem için oluşturulan ağ yapısı üzerinde,

doğal karıncalarının yiyecek arama sürecini taklit eden yapay karıncalar ile en kısa yolların bulunması amaçlanmaktadır.

Gambardella ve diğ. (1999a), ZA-ARP için MACS-VRPTW olarak adlandırdıkları bir çoklu karınca koloni sistemi algoritması önermişlerdir. MACS-VRPTW isimli KKO algortiması çözüm kalitesi ve çözüm süresi yönünden ZA-ARP için hâlâ en başarılı metasezgisel çözüm yöntemlerinden biri olarak görülmektedir (Donati ve diğ., 2008). MACS-VRPTW, etkileşimli olarak çalışan iki adet karınca kolonisi içeren bir algoritmadır. Bir koloni kullanılan araç sayısını düşürmeyi amaçlarken diğer koloni belirli bir araç sayısı için toplam yolculuk süresini azaltmayı amaçlamaktadır.

Bu tez çalışmasında da kullanılan çözüm yöntemi olan KKO metasezgiseli dördüncü bölümde detaylı olarak tanıtılmıştır.

2.4.2.5 Diğer metasezgiseller

Bu bölümde, ZA-ARP için yukarıda ele alınan ve en fazla bilinen dört metsezgiselden farklı metasezgisellerin kullanıldığı çalışmalar özetlenmiştir.

Kontoravdis ve Bard (1995), ZA-ARP için iki aşamalı bir açgözlü rassal uyumlu arama süreci (greedy randomized adaptive search procedure) önermişlerdir. Algoritmanın başlangıç aşamasında teslimat zaman aralıkları daha dar olan veya diğer müşterilere daha uzakta bulunan kök müşteriler belirlenerek başlangıç rotaları oluşturulmaktadır. Daha sonra henüz rotalanmamış müşterilerin yerleştirilebilecekleri en iyi pozisyonlar için bir ceza değerleri hesaplanmakta ve ceza değeri en büyük olan müşterilerden oluşturulan bir kümeden rastgele seçilen bir müşteri mevcut rotaya yerleştirilmektedir. Her beş yerleştirme hareketinden sonra mevcut çözüme püskürtme ve 2-opt hareketleri içeren yerel arama uygulanmaktadır.

Potvin ve Robillard (1995) ve Potvin ve diğ. (1996), kök müşterilerin belirlenmesi için yapay sinir ağları yöntemini kullanmışlardır. Her araç için bir ağırlık vektörü oluşturulmuştur. Başlangıç aşamasında tüm ağırlık vektörleri rassal olarak depoya yakın bölgelere yerleştirilmiştir. Daha sonra her adımda bir müşteri seçilerek tüm vektörlere olan uzaklığı hesaplanmıştır. Müşteriye en yakın konumda bulunan ağırlık vektörü müşteriye yakınlaştırılarak güncellenmiştir. Tüm müşteriler için çok defa

tekrarlanan bu süreç sonunda ağırlık vektörlerine en yakında bulunan müşteriler kök müşteri olarak belirlenmiştir.

Kilby ve diğ. (1999), ZA-ARP için rehberli yerel arama (guided local search) yöntemini kullanmışlardır. Voudoris ve Tsang (1998) tarafından geliştirilmiş olan rehberli yerel arama yöntemi daha önce bulunmuş olan yerel minimum çözümleri cezalandırılarak tekrar oluşturulmasını engellemektedir. Oluşturulan çözümlerin iyileştirilmesi için 2-opt, yerdeğiştirme, bağ değiştirme ve 2-opt* sezgiselleri kullanılmıştır.

Liu ve Shen (1999), rotalar ile müşteriler arasındaki ilişkiler üzerine odaklanan yeni bir komşuluk yapısı kullanarak iki aşamalı bir metasezgisel yöntem önermişlerdir. Kök müşterilerin seçiminde, teslimat zaman aralıkları ve daha önce seçilen kök müşterilere göre konumları etkili olmuştur. Müşterinin herhangi bir rotaya yerleştirilmesi mümkün olmadığında püskürtme hareketi kullanılmaktadır. Uygun bir çözüm elde edildiğinde λ-değiş tokuş sezgiseli ile yerel arama yapılmaktadır.

Bräysy (2003), ZA-ARP için 4 aşamadan oluşan yeni bir metasezgisel yöntem önermiştir. Önerilen yöntem daha önce Mladenovic ve Hansen (1997) tarafından önerilmiş olan değişken komşuluk arama (variable neighborhood search) yaklaşımını temel almaktadır. İlk aşamada başlangıç çözümü oluşturulmaktadır. Rotalara her *k* adet müşteri eklenmesinden sonra Or-opt sezgiseli uygulanmaktadır. İkinci aşamada, püskürtme hareketi tabanlı bir sezgisel ile rota sayısının azaltılması amaçlanmıştır. Üçüncü aşamada ise oluşturulan çözümlerin yolculuk mesafesi çapraz değiştirme ve yerleştirme tabanlı dört yerel iyileştirme sezgiselinin kullanımını içeren değişken komşuluk arama yöntemiyle azaltılmaktadır. Bräysy ve diğ. (2004) ise Bräysy (2003) çalışmasındaki rota oluşturma ve rota iyileştirme sezgisellerini değiştirerek ve tavlama benzetimi temelli bir çözüm iyileştirme süreci kullanarak daha iyi sonuçlar elde etmişlerdir.

Ibaraki ve diğ. (2005), ZA-ARP için rassal olarak oluşturulan başlangıç çözümlerini geliştirmek için üç metasezgisel yöntem önermiştir. Çok başlangıçlı yerel arama (multistart local search), iterasyonlu yerel arama (iterated local search) ve uyumlu çok başlangıçlı yerel arama (adaptive multistart local search) olarak adlandırılan

yöntemler temel olarak rassal olarak oluşturan belirli sayıda başlangıç çözümü üzerinde yerel arama gerçekleştirmektedir.

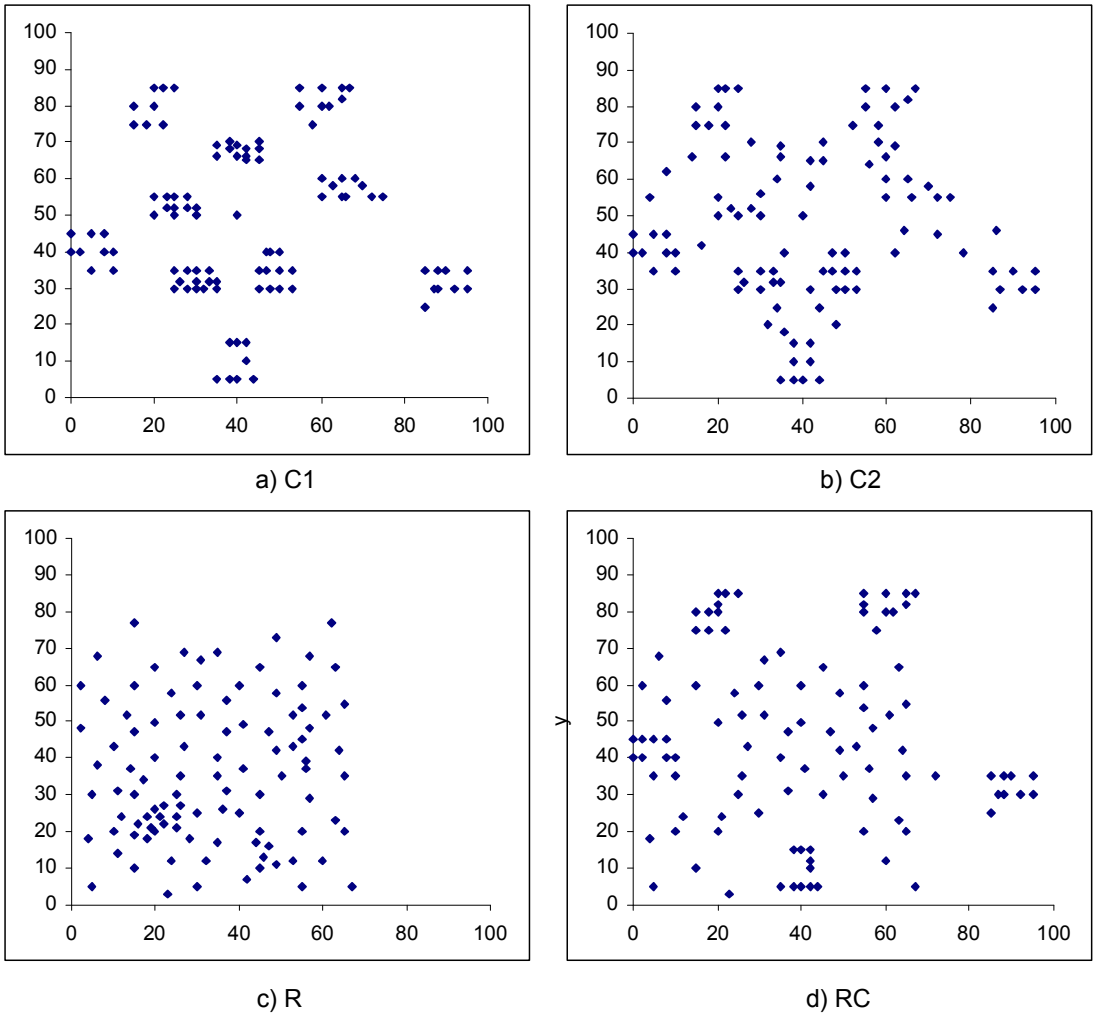
Psinger ve Ropke (2007), uyumlu büyük komşuluk arama (adaptive large neighborhood search) metasezgiselini kullanmışlardır. Uyumlu yapı, arama sürecini çeşitlendirmek veya yoğunlaştırmak amacıyla farklı sezgisellerin kullanımını içermektedir.

2.5 Örnek Problemler ve Geçmiş Çalışmaların Performansları

ZA-ARP ile ilgili geçmiş çalışmlarada en fazla kullanılan test problemi Solomon (1987) tarafından ortaya konan örnekler kümesidir. Solomon örnekleri; müşterilerin coğrafi yerleşimlerine göre, düzgün dağılıma uygun olarak rassal (R), kümelenmiş (C) ve yarı-kümelenmiş (RC) olarak, zaman aralıkları ve araç kapasitelerine göre ise dar zaman aralıklı ve düşük araç kapasiteli (1), geniş zaman aralıklı ve yüksek araç kapasiteli (2) olarak gruplandırılmıştır. Örneğin C2 tipinde bir problemde müşteriler belirli coğrafi bölgelerde gruplar halinde bulunmakta, teslimat zaman aralıkları geniş ve araç kapasitesi yüksektir.

Toplam 6 grup (R1, R2, C1, C2, RC1, RC2) örnek veri seti vardır. Toplam örnek problem sayısı ise 56'dır. Müşterilerin konumları yatay ve dikey eksenler üzerlerindeki yerleri ile tanımlanmıştır. Şekil 2.13'te Solomon (1987) örnek gruplarındaki müşterilerin konumları gösterilmiştir. R1, R2 ve RC1, RC2 gruplarındaki örnek problemlerde müşteri konumları aynıdır. Müşteriler arasındaki mesafeler Öklid uzaklıkları ile belirlenmektedir. Yolculuk süreleri ise bir birim uzaklığın bir birim zamanda gidilebildiği varsayımıyla hesaplanmaktadır. Yani yolculuk süreleri müşteriler arasındaki mesafeye eşit bir zaman birimindedir. Geçmiş çalışmalarda örnekler 25, 50, 75 ve 100 müşterilik durumları için ele alınmıştır. Solomon örneklerine ve güncel en iyi sonuçlara "http://www.top.sintef.no" adresinden ulaşılabilir.

ZA-ARP için sezgisel yöntemlerin kullanıldığı çok sayıda çalışma yapılmıştır ancak bu çalışmalar içerisinde en iyi olarak nitelendirilebilecek bir çalışma öne çıkmamaktadır, farklı çalışmalar farklı yönlerden birbirlerine üstünlük sağlayabilmektedir.



Şekil 2.13: Solomon (1987) örnek problemlerindeki müşteri konumları.

Sezgisel yöntem içeren algoritmaların performans değerlendirmesi koşum süresi, çözüm kalitesi, uygulama kolaylığı, gürbüzlük ve esneklik gibi birden çok kriterin karşılaştırılmasını gerektirmektedir (Barr ve diğ. 1995, Cordeau ve diğ. 2002). Sezgisel yöntemler uygulama alandaki problemleri çözmeye yönelik olarak geliştirildiklerinden esneklik önemli bir kriterdir. Sezgisel algoritma modeldeki, kısıtlardaki veya amaç fonksiyonundaki değişiklere kolaylıkla uyum sağlayabilmelidir. Gürbüzlük anlamında, sezgisel bir algoritma problemin karakteristik özelliklerindeki farklılıklara karşı aşırı duyarlı olmamalıdır. Gürbüz bir algoritma hiçbir örnek problem üzerinde zayıf bir performans göstermemelidir. Sezgisel algoritmalar genellikle rassal süreçler içerdiğinden (rassal olarak oluşturulan parametre değerleri veya rassal seçimler) aynı problem için çözüm denemeleri farklı

sonuçlar verecektir. Ancak her deneme sonucunda iyi kalitede çözümlerin elde edilebilmesi önemli bir performans kritedir. Sezgisel yöntemlerdeki rassallık sonuçların analizini ve karşılaştırılmasını zorlaştırmaktadır. Farklı denemeler sonucunda elde edilen en iyi sonuçların birbiriyle karşılaştırılması, geçmiş çalışmalarda sıklıkla yapılmış olmasına rağmen, algoritmaların gerçek performansı hakkında yanıltıcı değerlendirmeler neden olmaktadır. Ortalama değerler üzerinden performans karşılaştırması daha anlamlı ve doğru değerlendirmelerin yapılabilmesine imkan sağlayabilir. Sezgisel yöntemlerin performans değerlendirmesi ile ilgili detaylı bir çalışma Cordeau ve diğ. (2002) tarafından gerçekleştirilmiştir. Sezgisel bir yöntemin iyi sonuçlara ulaşma zamanı ve elde edilen en iyi çözümün amaç fonksiyonu değeri, farklı yöntemler arasından tercih yapılabilmesi için önemlidir. Çözüm kalitesinin değerlendirilmesi için genellikle optimal çözüme yakınlık standart bir ölçüt olarak kullanılmaktadır. Genellikle çözüm süresi ve çözüm kalitesi arasında ters bir ilişki vardır. Çözüm süresi (döngü sayısı) arttıkça elde edilen çözümlerin kalitesi de artmaktadır. Burada önemli olan pratik anlamda uygun sürelerde iyi kalitede çözümlerin üretilebilmesidir. Sezgisel yöntemlerin karşılaştırılmasında karşılaşılan diğer bir zorluk çalışmaların farklı kişiler tarafından farklı hesaplama ortamlarında gerçekleştirimiş olmasından kaynaklanmaktadır. Kullanılan hesaplama araçlarının performansları arasındaki farklar, kişilerin programlama becerileri arasındaki farklar ve kullanılan kaynaklar arasındaki farklar genellikle gözardı edilmek durumundadır (Hooker, 1995). Dongarra (1998), kullanılan bilgisayarlar arasındaki hesaplama sürelerinin karşılaştırılabilmesi amacıyla kullanabilecek katsayılar önermiştir. Bazı çalışmalarda sadece elde edilen en iyi sonuçlar verilirken süreler hakkında bilgi verilmemektedir. Bu tip durumlarda karşılaştırma sadece en iyi sonuçlar üzerinden yapılabilmektedir.

Tablo 2.2 - Tablo 2.4, rota oluşturma sezgisellerinin, rota iyileştirme sezgisellerinin ve metasezgisellerin kullanıldığı bazı çalışmalarda Solomon (1987) örnek problemleri için elde edilen sonuçlar yer almaktadır. Soldaki ilk sütun çalışmanın yazarlarını ve yılını, sonraki 6 sütün ise her problem grubu için elde edilen çözümlerdeki araç sayılarının (ilk satır) ve toplam mesafelerin (ikinci satır) grup ortalamalarını, son sütun ise 56 örnek problem için elde edilen çözümlerdeki kümülatif araç sayısını (KAS) ve kümülatif toplam mesafeyi (KTM) vermektedir. Bazı çalışmalada mesafe değerleri tamsayılara yuvarlatılmış olarak kullanıldığından

KTM değerleri tüm çalışmalar için tamsayılara yuvarlatılmıştır. Tabloların altlarında ilgili çalışmalarda yazarlar tarafından açıklandığı kadarıyla; kullanılan bilgisayar, bağımsız koşum sayısı ve ortalama çözüm süresi bilgileri verilmiştir.

Tablo 2.2 rota oluşturma sezgisellerinin kullanıldığı bazı çalışmaları içermektedir. (1) ve (2) nolu çalışmalarda birinci öncelikli amaç araç sayısını azaltmak, (3) nolu çalışma ise tek amaç yolculuk mesafesini azaltmaktır.

Tablo 2.2: Rota oluşturma sezgisellerinin performansı

Yazarlar	R1	R2	C1	C2	RC1	RC2	KAS KTM
(1) Solomon (1987)	13,58	3,27	10,00	3,13	13,50	3,88	453
	1436,7	1402,4	951,9	692,7	1596,5	1682,1	73004
(2) Potvin ve Rousseau (1993)	13,33	3,09	10,67	3,38	13,38	3,63	453
	1509,04	1386,67	1343,69	797,59	1723,72	1651,05	78834
(3) Ioannou ve diğ. (2001)	12,67	3,09	10,00	3,13	12,50	3,50	429
	1370	1310	865	662	1512	1483	67891

(1) Dec 10, 1 koşum, 0,6 dak.; (2) IBM PC, 1 koşum, 19,6 dak.; (3) Intel Pentium 133 MHz, 1 koşum, 4,0 dak.

Tablo 2.3 rota iyileştirme sezgisellerinin kullanıldığı bazı çalışmaları içermektedir. Tüm çalışmalarda birinci öncelikli amaç araç sayısını azaltmaktır. İkinci öncelikli amaç ise (2) ve (3) nolu çalışmalarda yolculuk süresini azaltmak, diğerlerinde yolculuk mesafesini azaltmaktır.

Tablo 2.3: Rota iyileştirme sezgisellerinin performansı

Yazarlar	R1	R2	C1	C2	RC1	RC2	KAS KTM
(1) Potvin ve Rousseau (1995)	13,33 1381,9	3,27 1293,4	10,00 902,9	3,13 653,2	13,25 1545,3	3,88 1595,1	448 69285
(2) Russel (1995)	12,66 1317	2,91 1167	10,00 930	3,00 681	12,38 1523	3,38 1398	424 65827
(3) Antes ve Derigs (1995)	12,83 1386,46	3,09 1366,48	10,00 955,39	3,00 717,31	12,50 1545,92	3,38 1598,06	429 71158
(4) Shaw (1997)	12,31 1205,06	-	10,00 828,38	-	12,00 1360,40	-	-
(5) Shaw (1998)	12,33 1201,79	-	10,00 828,38	-	11,95 1364,17	-	-
(6) Caseau ve Laburthe (1999)	12,42 1233,34	3,09 990,09	10,00 828,38	3,00 599,63	12,00 1403,74	3,38 1220,99	420 58927
(7) Schrimpf ve diğ. (2000)	12,08 1211,53	2,82 949,27	10,00 828,38	3,00 589,86	11,88 1361,76	3,38 1097,63	412 56830
(8) Cordone ve Wolfler (2001)	12,50 1241,89	2,91 995,39	10,00 834,05	3,00 591,78	12,38 1408,87	3,38 1139,70	422 58481
(9) Bräysy (2002)	12,17 1253,24	2,82 1039,56	10,00 832,88	3,00 593,49	11,88 1408,44	3,25 1244,96	412 59945

(1) Sparc iş istasyonu, - , 3,0 dak. (2) PC/486/DX2 66 MHz, 3 koşum, 1,4 dak. (3) RS6000/530, 4 koşum, 3,6 dak. (4) Dec Alpha, 3 koşum, 2 saat, (5) Sun Ultra Sparc 143 MHz, 6 koşum, 1 saat, (6) Pentium 300 MHz, 1 koşum, 5 dak., (7) RS6000, - , 9 dak., (8) Pentium 166 MHz, 1 koşum, 15,7 dak., (9) Pentium 200 MHz, 1 koşum, 4,6 dak.

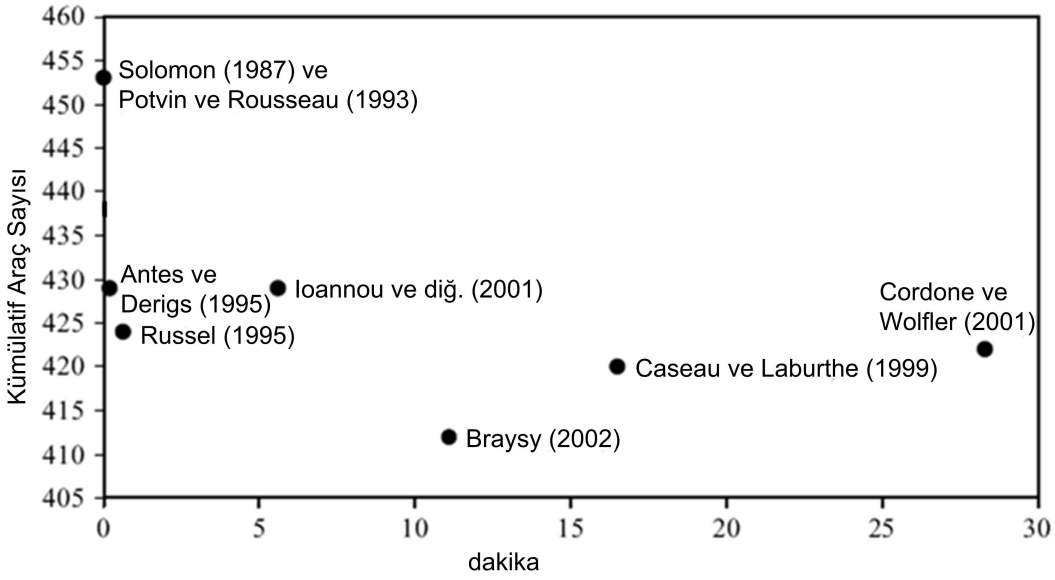
Tablo 2.4 metasezgisellerin kullanıldığı bazı çalışmaları içermektedir. (10) nolu çalışmada tek amaç toplam yolculuk mesafesinin azaltılmasıdır. Diğer tüm çalışmalarda hiyerarşik olarak birinci öncelikli amaç araç sayısının azaltılması, ikinci öncelikli amaç ise (1) ve (7) nolu çalışmalarda yolculuk süresinin azaltılması diğerlerinde yolculuk mesafesinin azaltılmasıdır.

Tablo 2.4: Metasezgisellerin performansı

Yazarlar	R1	R2	C1	C2	RC1	RC2	KAS KTM
(1) Garcia ve diğ. (1994)	12,92 1317,7	3,09 1222,6	10,00 877,1	3,00 602,3	12,88 1473,5	3,75 1527,0	436 65977
(2) Kontoravdis ve Bard (1995)	12,58 1325,44	3,09 1164,27	10,00 827,3	3,00 589,65	12,63 1500,94	3,50 1414,21	427 64196
(3) Rochart ve Taillard (1995)	12,25 1208,50	2,91 961,72	10,00 828,38	3,00 589,86	11,88 1377,39	3,38 1119,59	415 57231
(4) Potvin ve Bengio (1996)	12,58 1296,83	3,00 1117,64	10,00 838,11	3,00 590,00	12,13 1446,25	3,38 1368,13	422 62634
(5) Taillard ve diğ. (1997)	12,17 1209,35	2,82 980,27	10,00 828,38	3,00 589,86	11,50 1389,22	3,38 1117,44	410 57523
(6) Homberger ve Gehring (1999)	11,92 1228,06	2,73 969,95	10,00 828,38	3,00 589,86	11,63 1392,57	3,25 1144,43	406 57876
(7) Schulze ve Fahle (1999)	12,25 1239,15	2,82 1066,68	10,00 828,94	3,00 589,93	11,75 1409,26	3,38 1286,05	414 60346
(8) Gambardella ve diğ. (1999)	12,00 1217,73	2,73 967,75	10,00 828,38	3,00 588,98	11,63 1382,42	3,25 1129,19	407 57525
(9) Liu ve Shen, (1999)	12,17 1249,57	2,82 1016,58	10,00 830,06	3,00 591,03	11,88 1412,87	3,25 1204,87	412 59318
(10) Kilby ve diğ. (1999)	12,67 1200,33	3,00 966,56	10,00 830,75	3,00 592,24	12,13 1388,15	3,38 1133,42	423 57423
(11) Bräysy (2003)	11,92 1222,12	2,73 975,12	10,00 828,38	3,00 589,86	11,50 1389,58	3,25 1128,38	405 57710
(12) Bräysy ve diğ. (2004)	12,00 1214,69	2,73 960,44	10,00 828,38	3,00 589,86	11,50 1389,20	3,25 1124,14	406 57422
(13) Homberger ve Gehring (2005)	11,92 1212,73	2,73 955,03	10,00 828,38	3,00 589,86	11,50 1386,44	3,25 1108,52	405 57192
(14) Ibaraki ve diğ. (2005)	11,92 1217,40	2,73 959,11	10,00 828,38	3,00 589,86	11,50 1391,03	3,25 1122,79	405 57444

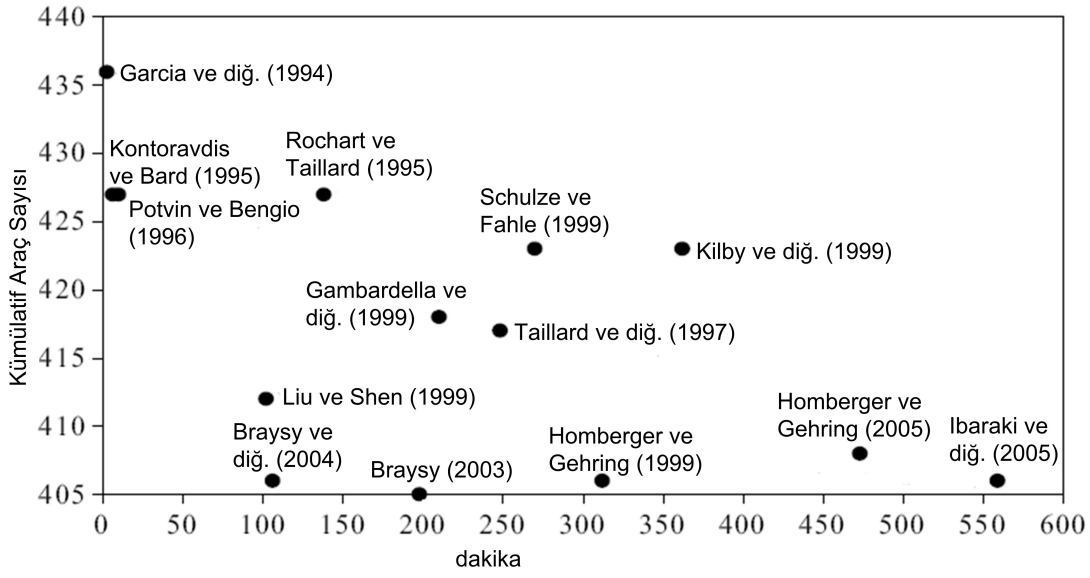
(1) Sun Sparc 10, -, 20 dak.; (2) Sun Sparc 10, 5 koşum, 1,2 dak.; (3) Slicon Graphics 100 MHz, 1 koşum, 92,2 dak.; (4) Sun Sparc 10, -, 25 dak.; (5) Sun Sparc 10, 1 koşum, 248 dak.; (6) Pentium 200 MHz, 10 koşum, 13 dak.; (7) Motorola PowerPC 604, 5 koşum, 8,3 dak (8) Sun Ultra Sparc 1 167 MHz, -, -; (9) HP 9000/720, 3 koşum, 20 dak.; (10) DEC Alpha, 3 koşum, 48,3 dak.; (11) Pentium 200 MHz, 1 koşum, 82,5 dak.; (12) AMD 700 MHz, 30 koşum, 2,7 dak.; (13) Pentium 400 MHz, -, -; (14) Pentium III 1 GHz, 1 koşum, 250 dak.

Bräysy ve Gendreau (2005a), ZA-ARP için önceki çalışmalarda kullanılan klasik sezgisel ve metasezgisel yöntemlerin performansınlarını kabaca karşılaştırabilmek için ilgili çalışmalardaki çözüm sürelerini Dongarra (1998) tarafından belirlenen çarpanlarla Sun Sparc 10 bilgisayarına göre normalize ederek aşağıdaki Şekil 2.14 ve Şekil 2.15'te yer alan grafikleri oluşturmuştur.



Şekil 2.14: Klasik sezgisellerin karşılaştırılması

Şekil 2.14 rota oluşturma ve rota iyileştirme sezgisellerinin kullanıldığı çalışmalar için oluşturulmuştur. Rota oluşturma sezgiselleri, rota iyileştirme sezgisellerine göre daha hızlıdırlar ancak ancak Şekil 2.14'te görüldüğü gibi elde edilen çözümlerin kalitesi daha kötüdür. Antes ve Derigs (1995), Russel (1995) ve Bräysy (2002) çalışmalarında elde edilen sonuçlar, çözüm kalitesi ve çözüm süresi kriterlerine göre Pareto optimal olarak değerlendirilebilir.



Şekil 2.15: Metasezgisellerin karşılaştırılması

Şekil 2.15 metasezgisellerin kullanıldığı çallışmaları içermektedir. Garcia ve diğ. (1994), Kontoravdis ve Bard (1995), Bräysy ve diğ. (2004) ve Bräysy (2003) çalışmalarında elde edilen sonuçlar, çözüm kalitesi ve çözüm süresi kriterlerine göre Pareto optimal olarak değerlendirilebilir.

Şekil 2.14 ve Şekil 2.15 karşılaştırıldığında, metasezgisel yöntemler ile klasik sezgisellere göre daha uzun çözüm sürelerinde daha kaliteli sonuçların elde edilebildiği görülmektedir. ZA-ARP için geçmiş çalışmalarda önerilmiş ve kullanılmış olan çok sayıda sezgisel yöntem içerisinden diğerlerinin tümüne tüm yönlerden üstünlük sağlayabilen bir yöntem öne çıkmamaktadır.

3. BULANIK KÜME ve OLABİLİRLİK TEORİSİ

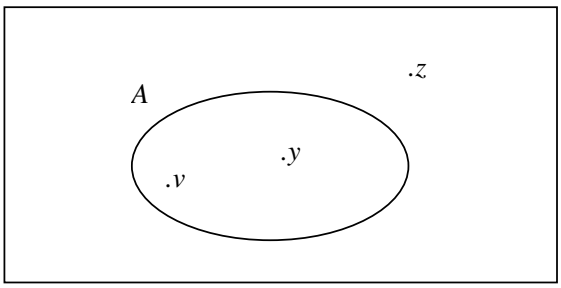
Bu bölümde bulanık küme ve olabilirlik teorisi ile ilgili tanımlar ve kavramlara yer verilmiştir. Konu kapsamı, sonraki bölümlerde önerilecek olan modellerin oluşturulmasında ihtiyaç duyulacak olan tanımlar, kavramlar ve araçlarla sınırlandırılmıştır. Bulanık küme ve olabilirlik teorisi ile ilgili detaylı bilgilere Dubois ve Prade (1980a, 1988 ve 2000) çalışmalarından ulaşılabilir.

3.1 Bulanık Küme

U tanım kümesi içerisindeki bir *u* elemanının klasik bir *A* kümesine üyelik derecesi (3.1) eşitliğinde gösterilen karakteristik fonksiyon ($\mathrm{X}_A(u)$) ile tanımlanmaktadır;

$$\mathrm{X}_A(u) = \begin{cases} 1, & u \in A \\ 0, & u \notin A \end{cases} \qquad \textbf{(3.1)}$$

Tanım kümesi içerisindeki herhangi bir elemanın klasik bir *A* kümesi için karakteristik fonksiyon değeri söz konusu elemanın *A* kümesine ait olup olmamasına bağlı olarak sadece 0 veya 1 değerlerini alabilmektedir. Şekil 3.1'de de görüldüğü gibi klasik bir kümede kümeye dahil olan ve olmayan elemanlar arasında kesin bir ayırım vardır. Şekil 3.1'de gösterilen *A* kümesi için karakteristik fonksiyon değerleri; $v \in A,\ y \in A$ *ve* $z \notin A$ olduğundan $X_A(v) = X_A(\mathrm{y}) = 1$ ve $X_A(\mathrm{z}) = 0$ olacaktır.



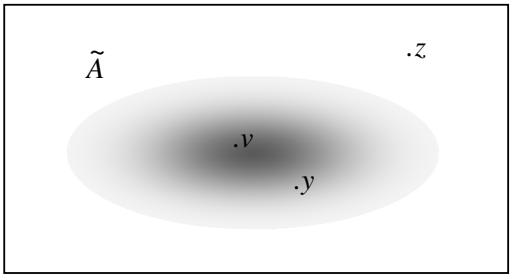
Şekil 3.1: Klasik bir *A* kümesi ve *v, y, z* elemanları

Bu tip bir kabul teorik düzeyde genellikle geçerliliğini koruyabilmektedir. Örneğin, *A* çift sayılar kümesi olarak tanımlanmış ise evrensel kümenin tam sayılardan oluştuğu bir düzlemde her tam sayı için (3.1) eşitliği ile 0 veya 1 olmak üzere karakteristik fonksiyon değeri belirlenebilir.

Ancak, gerçek hayatta karşılaşılan durumlar ele alındığında klasik küme kavramı çoğu zaman model oluşturmada yetersiz kalmakta, çok fazla varsayım ve kabul ile gerçek hayatın modellenmesi için kullanılabilmektedir. Özellikle günlük hayatta kullandığımız uzun, kısa, soğuk, sıcak, az, çok vb. sözel değişkenler söz konusu olduğunda klasik bir küme tanımı yetersiz kalabilmektedir. Örneğin, boyu uzun olan insanlardan oluşan klasik bir *A* kümesini tanımlamak için bu kümenin kesin sınırlarının belirlenmesi gereklidir. Uzun boylu insanlar kümesini boyu 180 cm ve üstü insanlar olmak üzere tanımlamış olsun. Bu durumda boyu 179,99 cm olan bir insanı uzun boylu olarak kabul etmemek ne kadar doğrudur? Ayrıca, boyu 195 cm olan bir insan ile boyu 181 cm olan bir insanı aynı derecede uzun boylu kabul etmek ne kadar doğrudur? Buna benzer durumlar için oldukça fazla örnek verilebilir. Zadeh (1965) tarafından önerilen bulanık kümeler işte bu tip durumlarda, gerçek hayattaki belirsizlik içeren algılamanın modellenebilmesi amacıyla kullanılabilmektedir. Bulanık kümeler, klasik kümeler gibi kesin sınırlara sahip değildir ve herhangi bir *u* elemanının bulanık bir kümeye üyelik derecesi, üyelik fonksiyonu ($\mu(u)$) ile tanımlanır. $\mu(u)$ üyelik fonksiyonu, [0,1] kapalı aralığında değer alır. *U* tanım kümesi içinde tanımlanan bulanık bir $\tilde{A}$ kümesi, (3.2) denklemi ile tanımlanan çiftler ile liste halinde gösterilebilir. $\mu_{\tilde{A}}(u)$, *u* elemanının $\tilde{A}$ kümesine üyelik derecesidir. $\mu_{\tilde{A}}(u)$ değeri ne kadar büyükse *u* elemanının kümeye ait olma derecesi de o kadar büyüktür.

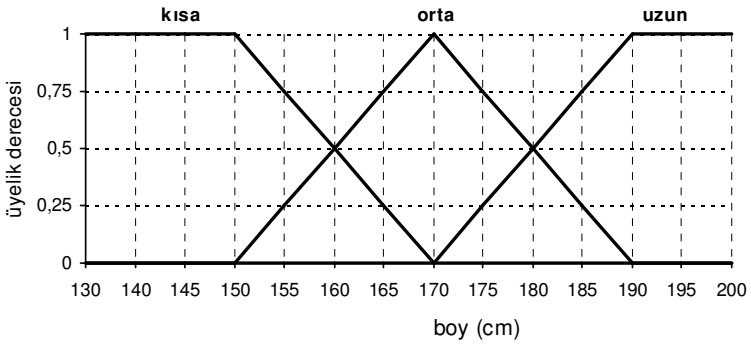
$$\tilde{A} = \left\{ \left(u, \mu_{\tilde{A}}(u) \right) \middle| u \in U \right\} \qquad \textbf{(3.2)}$$

Şekil 3.1'de klasik kümeler için kullanılan Venn diyagramı gösterimi, Şekil 3.2'de görüldüğü gibi küme sınırı belirsizleştirilerek bulanık kümeler için kullanılabilir. Bu durumda $\mu_{\tilde{A}}(v) = 1, 0 < \mu_{\tilde{A}}(y) < 1, \mu_{\tilde{A}}(z) = 0$ olarak üyelik dereceleri belirlenebilir.



Şekil 3.2: Bulanık bir $\tilde{A}$ kümesi ve *v, y, z* elemanları

Bulanık kümelerin gösterimi için genellikle üyelik fonksiyonlarının grafikleri kullanılır. Klasik küme yaklaşımıyla kesin sınırları tam olarak belirlenemeyen uzun insan boyu kümesi ve benzer şekilde kısa ve orta insan boyu kümeleri bulanık kümelerle tanımlanırsa üyelik fonksiyonları Şekil 3.3'te olduğu gibi gösterilebilir.



Şekil 3.3: Kısa, orta ve uzun insan boyu bulanık kümeleri üyelik fonksiyonları

Şekil 3.3'te görüldüğü gibi boyu 185 cm olan bir insan 0,75 derece ile uzun boylu insanlar kümesine aitken aynı zamanda 0,25 derece ile de orta boylu insanlar kümesine aittir. Klasik küme teorisi ile böyle bir modellemenin yapılabilmesi mümkün değildir.

U tanım kümesi içerisinde tanımlanmış olan bulanık bir $\tilde{A}$ kümesine üyelik derecesi 1 olan elemanlardan oluşturulan klasik bir küme, $\tilde{A}$ kümesinin çekirdeği, $c(\tilde{A})$, olarak adlandırılır (3.3).

$$c(\tilde{A}) = \{u \,|\, \mu_{\tilde{A}}(u) = 1\} \qquad \textbf{(3.3)}$$

$\tilde{A}$ kümesine üyelik derecesi en büyük olan elemanın üyelik derecesi, $\tilde{A}$ kümesinin yüksekliği, $h(\tilde{A})$, olarak adlandırılır. $h(\tilde{A})$ değeri 1'den küçük ise, $c(\tilde{A})$ kümesi boştur. $c(\tilde{A})$ boş küme değilse, $\tilde{A}$ normal bulanık kümedir. Normal olmayan bulanık kümeler alt-normal bulanık küme olarak adlandırılır.

Bulanık $\tilde{A}$ kümesi (3.4) eşitsizliğindeki koşulu sağlıyor ise dışbükeydir.

$$\mu_{\tilde{A}}(z) \geq \min\{\mu_{\tilde{A}}(x), \mu_{\tilde{A}}(y)\} \qquad \forall x, y, z \quad z \in [x, y] \tag{3.4}$$

Bulanık bir $\tilde{A}$ kümesinin λ-kesimi ($\tilde{A}^{\lambda}$), (3.5) denklemi ile klasik bir küme olarak elde edilir. $\tilde{A}^{\lambda}$, $\tilde{A}$ bulanık kümesine üyelik derecesi en az λ kadar olan elemanlardan oluşur. Dışbükey bulanık kümelerde $\tilde{A}^{\lambda} = [\underline{a}(\lambda), \overline{a}(\lambda)]$ kapalı aralığı ile gösterilebilir.

$$\tilde{A}^{\lambda} = \{u \,|\, \mu_{\tilde{A}}(u) \geq \lambda\}, \qquad \lambda \in (0,1]. \tag{3.5}$$

Bulanık bir $\tilde{A}$ kümesine üyelik derecesi 0'dan büyük elemanlardan oluşan klasik bir küme, $\tilde{A}$ kümesinin desteği, $s(\tilde{A})$, olarak adlandırılmaktadır (3.6).

$$s(\tilde{A}) = \{u \,|\, \mu_{\tilde{A}}(u) > 0\} \tag{3.6}$$

Bulanık bir $\tilde{A}$ kümesinin tümleyeni, $\tilde{A}^c$ olarak gösterilen bulanık bir kümedir. $\tilde{A}^c$ kümesindeki elemanların üyelik dereceleri $\mu_{\tilde{A}^c}(u) = 1 - \mu_{\tilde{A}}(u)$ olarak hesaplanabilir.

Bulanık kümelerin birleşimi veya kesişimi için tanım kümesindeki her elemanın her bir bulanık kümedeki üyelik derecelerinin en büyüğü veya en küçüğü alınır.

$$\mu_{\tilde{A} \cup \tilde{B}}(x) = \text{maks}\{\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)\} \tag{3.7}$$

$$\mu_{\tilde{A} \cap \tilde{B}}(x) = \min\{\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)\} \tag{3.8}$$

3.2 Olabilirlik dağılımı

Olabilirlik dağılımı (Zadeh, 1978), *U* kümesindeki her *u* elemanına olabilirlik dereceleri ($\pi(u) \in [0, 1]$) atamaktadır. Olabilirlik dağılımı *U* kümesi içerisinde farklı değerler alabilen *X* değişkeni hakkındaki belirsizliği temsil etmektedir.

X değişkeninin olabilirlik dağılımı, bulanık bir $\tilde{A}$ kümesinin üyelik fonksiyonu ile tanımlanmaktadır. $\pi_X = \mu_{\tilde{A}}$ ile gösterilir. Gerçekleşmesi farklı düzeylerde makul (plausible) görülebilen *X* değişkenine ait üyelik fonksiyonudur. *X* değişkeninin olabilirlik dağılımı ile temsil edilen değerleri birbirleriyle bağdaşmaz çünkü *X* değişkeni için bu değerlerden sadece bir tanesi gerçek olacaktır. Bazı *u* değerleri için $\pi_X(u) = 0$ değerini almaktadır, bunun anlamı *X*=*u* durumunun imkansız olduğudur. $\pi_X(u) = 1$ için ise *X*=u durumunun bir sürpriz olmayacağı anlaşılmaktadır, olasılığın 1 olması durumuna göre oldukça zayıf bir ifadedir. Olabilirlik dağılımı, olasılık dağılımına benzemektedir, ancak $\pi_X(u) = 1$'in anlamı *X*=*u* durumunun makul olduğu ve hariç tutulamayacağıdır.

X değişkeni için *U* kümesindeki elemanlardan bir tanesi gerçek değer olacağından en az bir u elemanı için $\pi_X(u) = 1$'dir ve bu durum normalizasyon koşulu olarak adlandırılır. En az bir değerin tam anlamıyla olabilir olduğunu ortaya koymaktadır. $\forall u \in U, \pi_X(u) < 1$ ifadesi mantıksal olarak tutarlı değildir çünkü bu durumda *U* kümesindeki her eleman *X* değişkeni için kısmi olarak imkansızlık içerecektir.

Dubois ve Prade (1988) çalışmasında, eksik bilgi durumunda olabilirlik dağılımlarının oluşturulması ve yorumlanması konuları detaylı olarak incelenmiştir.

3.3 Olabilirlik teorisi

İlk olarak L. Zadeh tarafından 1978 yılında bulanık kümeler teorisi ve bulanık mantığın geliştirilmesiyle ortaya konulan olabilirlik teorisi, birtakım belirsizliklerin ele alındığı matematiksel bir teoridir. Olabilirlik teorisi, temel olarak olabilirlik dağılımına dayanmaktadır. Olasılık teorisinin tamamlayıcısı olarak düşünülebilir. D. Dubois ve H. Prade (1988) olabilirlik teorisinin gelişimine önemli katkılarda bulunmuşlardır.

Olabilirlik teorisi tam olmayan bilgiyi kodlamaktadır, olasılık teorisi ise rastgele, gerçekten gözlenmiş olaylara dayanmakta veya subjektif bir iddiayı yansıtmaktadır. Olabilirlik teorisinde bulanık bir X değişkeni hakkında tamamen bilgisizlik durumunda her $u \in U$ değeri için $\pi_X(u) = 1$'dir (Dubois ve diğ. 2003a). Çünkü aksi yönde bir bilgi mevcut değildir.

X değişkeninin gerçekleşecek değerinin U tanım kümesinin bir alt kümesi olan E kümesine ait olması ($X \in E$) olayının olabilirliği, $\prod(X \in E)$ olarak gösterilir. X değişkeninin olabilirlik dağılımını temsil eden bulanık $\tilde{A}$ kümesi ile E kümesinin olabilirlik dağılımlarının kesişimlerinin en büyük yüksekliği $\prod(X \in E)$ değerini verir (3.9).

$$\prod(X \in E) = \sup_u \min\left(\pi_X(u), \mu_E(u)\right) \qquad \textbf{(3.9)}$$

(3.9) eşitliği, E kümesinin de bulanık bir küme olabilmesi durumunu kapsamaktadır. E kümesinin klasik bir küme olduğu biliniyorsa (3.9) yerine (3.10) eşitliği kullanılabilir.

$$\prod(X \in E) = \sup_{u \in E} (\pi_X(u)) \qquad \textbf{(3.10)}$$

$\prod(X \in E)$ değeri; $X \in E$ olayının gerçekleşebilirliğinin bir ölçütü veya $X \in E$ önermesinin, X değişkeninin olabilirlik dağılımını temsil eden bulanık $\tilde{A}$ kümesinin elde edilmesinde kullanılan bilgiye dayanılarak ifade edilen tutarlılığıdır.

U kümesinin birer alt kümesi olan E ve F kümeleri için aşağıdaki (3.11) eşitliği geçerlidir;

$$\prod(X \in (E \cup F)) = \text{maks}\,(\prod(X \in E), \prod(X \in F)) \qquad \textbf{(3.11)}$$

(3.11)'de $X \in E$ olayının doğruluğu kesin olarak biliniyor ise ve E olayı bir [a,b] kesin aralığını tanımlıyor ise belirsizlik X değişkeninin [a, b] aralığında hangi değeri alacağının bilinmemesindedir. X değeri [a, b] aralığında tek bir değer için gerçek olacaktır. Bu durumda $\prod(X \in (E \cup F))$ değeri (3.12) ile hesaplanabilir.

$$\prod(X \in (E \cup F)) = \begin{cases} 1, & \text{eğer } E \cap F \neq \{\}, \\ 0, & \text{değilse (E ve F ayrık kümelerdir).} \end{cases} \quad \textbf{(3.12)}$$

Olasılık teorisi bir olayın gerçekleşme beklentisini bir sayı kullanarak açıklar. Olabilirlik teorisinde ise olayların gerçekleşme beklentisi için olabilirlik ve gereklilik (kesinlik) olmak üzere iki ölçüt kullanılmaktadır.

"$X \in E$" olayının gerekliliği $N(X \in E)$ ile gösterilir ve X değişkeninin olabilirlik dağılımını temsil eden bulanık $\tilde{A}$ kümesinin ne kadarının E kümesinin çekirdek kısmı içerisinde bulunduğunun bir ölçütüdür. Daha net olarak, $\tilde{A}$ bilgisinin kullanımıyla $X \in E$ önermesinin kesinlikle doğru olması durumunun değerlendirmesidir.

$$N(X \in E) = \inf_{u} \text{ maks}\left(1 - \pi_X(u), \mu_E(u)\right) = \inf_{u \notin E}\left(1 - \pi_X(u)\right) = 1 - \prod(X \in E^c) \quad \textbf{(3.13)}$$

E^c, E kümesinin tümleyenidir. Bu durumda $N(X \in E) = 1$ olabilmesi için $\tilde{A}$ bulanık kümesinin destek kısmının tamamıyla E kümesinin çekirdek kısmı içerisinde olması gerekir. Yani, X değişkeni için olabilirliği bulunan tüm değerlerin E kümesine üyelik derecesi 1 olmalıdır. $X \in E$ olayının kesinlikle doğru olabilmesi için $X \notin E$ imkansız olmalıdır.

U kümesinin birer alt kümesi olan E ve F kümeleri için $N(X \in (E \cap F))$ (3.14) ile hesaplanabilir.

$$N\left(X \in (E \cap F)\right) = \min\left(N(X \in E), N(X \in F)\right) \quad \textbf{(3.14)}$$

$X \in E$ olayı için $N(X \in E) > 0$ ise $\prod(X \in E) = 1$'dir.

$N(X \in E) = 1$ ise $X \in E$ olayının gerçekleşmesi kesindir ve $\prod(X \in E) = 1$ olmasını gerektirir.

$\prod(X \in E) = 0$ ise $X \in E$ olayı gerçekleşemez ve $N(X \in E) = 0$ olmasını gerektirir.

$\prod(X \in E) = 1$ ise $X \in E$ olayının gerçekleşmesi sürpriz olmayacaktır. $N(X \in E)$ değeri için herhangi bir çıkarım yapılamaz.

Olasılık teorisindeki durumdan farklı olarak olabilirlik ve gereklilik kendi içinde dual değildir.

$$\prod(X \in E) + \prod(X \in E^c) \geq 1 \qquad \textbf{(3.15)}$$

$$N(X \in E) + N(X \in E^c) \leq 1 \qquad \textbf{(3.16)}$$

Olabilirlik ile olasılık kavramları arasındaki en büyük farklardan biri burada ortaya çıkmaktadır. Bir olayın olasılığı aynı zamanda tümleyen olayın olasılığını da belirlerken, olabilirlik veya gereklilik ölçütlerinde olay ve tümleyen olay arasında daha zayıf bir ilişki vardır.

3.4 Bulanık Aralık ve Bulanık Sayı

Bulanık aralık, başlangıç ve bitiş noktaları belirsizlik içeren bir gerçel sayı aralığını modellemektedir. Gerçek değeri kesin olarak bilinemeyen bir X değişkeninin olabilirlik dağılımı, bulanık bir $\tilde{A}$ kümesinin üyelik fonksiyonu ile tanımlansın. $\mu_{\tilde{A}} : \square \rightarrow [0,1]$ fonksiyonu üstten yarı-sürekli (upper semi-continuous) bir fonksiyon ise, normal ve dışbükey olan $\tilde{A}$ bulanık kümesi bulanık aralık olarak adlandırılabilir.

Bulanık aralık modellediği parametre için belirsizliği tanımlamaktadır. Bilinmeyen parametre değerinin alabileceği değerler için olabilirlik dağılımını oluşturmaktadır. X değişkeni ve $\tilde{A}$ bulanık aralığı için "X, $\tilde{A}$'dır" ifadesi aşağıdaki (3.17) denklemi ile X değişkeni için olabilirlik dağılımını oluşturur:

$$\prod(X = u) = \mu_{\tilde{A}}(u) \qquad u \in \square \qquad \textbf{(3.17)}$$

$[a,b] \subset \square$ olduğu varsayıldığında, $X \in [a,b]$ olayının olabilirlik ve gereklilik ölçütleri (3.18) ve (3.19) denklemleri ile tanımlanabilir:

$$\prod\left(X \in [a,b]\right) = \sup_{u \in [a,b]} \mu_{\tilde{A}}(u) \qquad \textbf{(3.18)}$$

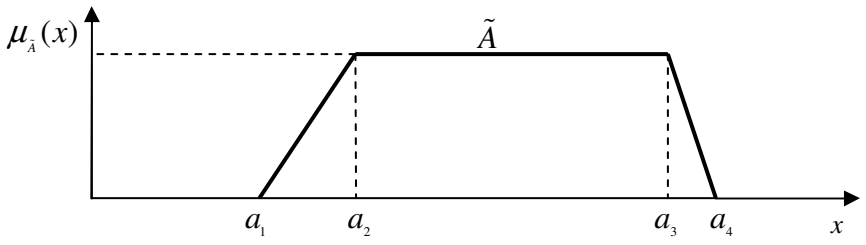
$$N\left(X \in [a,b]\right) = \inf_{u \notin [a,b]} (1 - \mu_{\tilde{A}}(u)) \qquad \textbf{(3.19)}$$

Eğer tüm $u < 0$ değerleri için $\mu_{\tilde{A}}(u) = 0$ ise *X* değişkeni negatif değildir.

Uygulama alanında, belirsizlik içeren bir değişkenin alacağı değer ile ilgili tahmin yapılmasının gerekmesi sıklıkla karşılaşılan bir durumdur. Klasik yaklaşımlarda bu tahminler genellikle iyimser ve kötümser durumlar için ayrı ayrı yapılır. Kötümser tahmin durumunda aralıklar geniş olacağından birbirini takip eden hesaplamalar sonucunda elde edilecek sonuçlar kullanıcıya kullanışlı bir bilgi sağlayamayacaktır. İyimser tahmin durumunda ise aralıklar dar olacağından ilgili değişkenin aralık dışında değer alması durumunda elde edilecek sonuçlar yanıltıcı olacaktır. Bulanık aralık, tanımladığı değişken için, en kötümser tahmin ile en iyimser tahmin aralığındaki tüm bilgiyi modelleyebilir. Bulanık aralık, belirsiz bir niceliğin daha fazla bilgi ile temsil edilebilmesine olanak sağlar. Kaufmann ve Gupta (1985), bulanık aralık ile güven aralığı kavramı arasındaki ilişkiyi ortaya koymuştur. Bulanık aralık "güven aralığı kavramının genelleştirilmesi olarak" ele alınabilmektedir.

Bulanık bir aralık, bulanık kümenin farklı λ-kesimlerini temsil eden doğru parçalarından doğrusal enterpolasyonla elde edilebilir.

$\lambda = \left\{ \lambda_1 = e \to 0, \lambda_2 = \frac{1}{n-1}, ..., \lambda_k = \frac{k-1}{n-1}, ..., \lambda_{n-1} = \frac{n-2}{n-1}, \lambda_n = 1 \right\}$ kesimleri için belirlenen doğru parçalarından herhangi bir $\tilde{A}$ bulanık aralığının elde edilebilmesi için $A^{\lambda_k} \supseteq A^{\lambda_{k+1}}$ şartı sağlanmalıdır. Böylece bulanık aralığın dışbükey bir üyelik fonksiyonuna sahip olması sağlanır. En basit (en az bilgi gerektiren) bulanık bir aralık n=2 için λ={*e*, 1} kesimleri kullanılarak yamuk şekilli bir bulanık küme olarak oluşturulabilir. Şekil 3.4'te $A^e = [\underline{a}(e), \overline{a}(e)] = [a_1, a_4]$ ve $A^1 = [\underline{a}(1), \overline{a}(1)] = [a_2, a_3]$ λ-kesimleri kullanılarak elde edilen $\tilde{A}$ bulanık aralığının üyelik fonksiyonu gösterilmiştir.

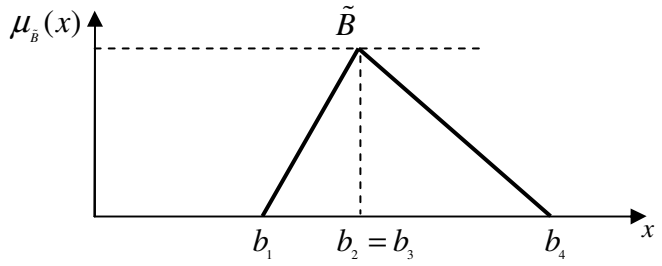


Şekil 3.4: $\tilde{A}$ bulanık kümesi ile tanımlanan bir bulanık aralık

Şekil 3.4'te yer alan yamuk şekilli $\tilde{A}$ bulanık aralığının üyelik fonksiyonu (3.20) ile elde edilebilir. Yamuk şekilli bir $\tilde{A}$ bulanık aralığı, n=2 için (a_1, a_2, a_3, a_4) dörtlüsü ile tanımlanabilir.

$$\mu_{\tilde{A}}(x) = \begin{cases} 1, & x \in [a_2, a_3], \\ 1 - \dfrac{a_2 - x}{a_2 - a_1}, & x \in [a_1, a_2], \\ 1 - \dfrac{x - a_3}{a_4 - a_3}, & x \in [a_3, a_4], \\ 0, & x \in (-\infty, a_1) \cup x \in (a_4, +\infty). \end{cases} \quad \textbf{(3.20)}$$

Bulanık sayı kavramı ise özel bir bulanık aralığı tanımlamaktadır. Bulanık bir $\tilde{B}$ aralığının üyelik fonksiyonu tek bir mod değerine sahipse, yani bulanık kümenin yüksekliğini temsil eden $h(\tilde{B})$ değerini 1 yapan tek bir b gerçel sayısı varsa, $\tilde{B}$ bulanık aralığı bulanık bir sayı olarak tanımlanabilir. Bu durumda $\tilde{B}$ bulanık sayısı "yaklaşık b" ifadesini modellemek için kullanılabilir.



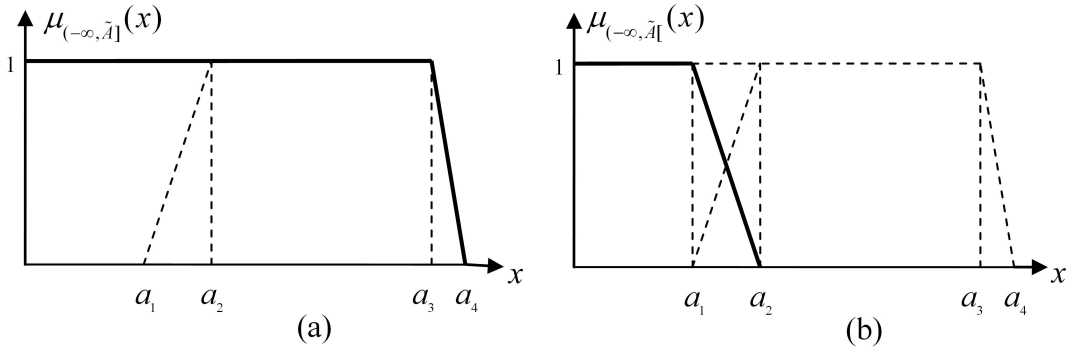
Şekil 3.5: $\tilde{B}$ bulanık kümesi ile tanımlanan bir bulanık sayı

En basit (en az bilgi gerektiren) bulanık bir sayı n=2 için λ={e,1} kesimleri kullanılarak üçgen şekilli bir bulanık küme olarak oluşturulabilir (Şekil 3.5). Üçgen şekilli bir $\tilde{B}$ bulanık aralığı, n=2 için $(b_1, b_2 = b_3, b_4)$ üçlüsü ile tanımlanabilir.

Kesin bir $u \in \mathbb{R}$ sayısı, $\mu_u(u) = 1$ ve $x \neq u$ için $\mu_u(x) = 0$ olmak üzere bulanık bir sayı olarak tanımlanabilir. Bulanık sayılar kesin sayıları kapsamaktadır.

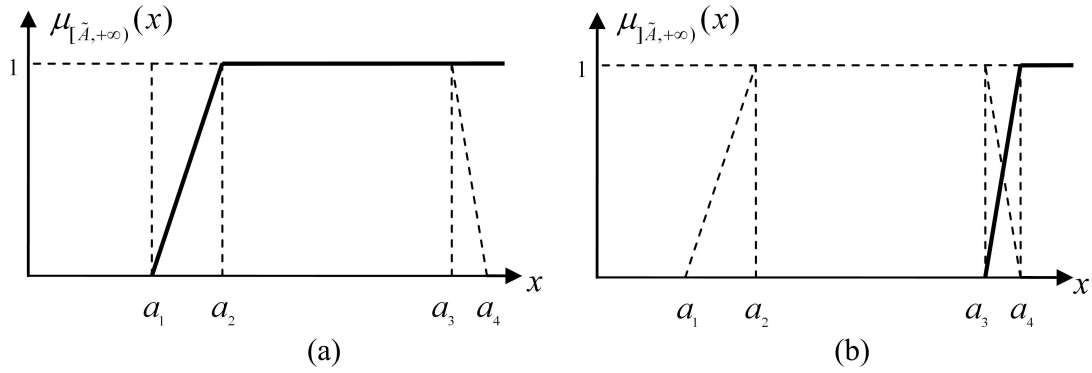
3.5 Bulanık Eşik

Bulanık bir aralık, artan ($a_3 = a_4 = +\infty$) veya azalan ($a_1 = a_2 = -\infty$) bir üyelik fonksiyonuna sahip ise bulanık eşik olarak adlandırılır. Bulanık eşikler genellikle bulanık aralık ile tanımlanan bir değişkenden büyük veya küçük olan değerleri modellemek için kullanılırlar. X değişkenini temsil edilen $\tilde{A}$ bulanık aralığı için $(-\infty, \tilde{A}]$, $(-\infty, \tilde{A}[$, $[\tilde{A}, +\infty)$, $]\tilde{A}, +\infty)$ bulanık eşikleri Şekil 3.6 ve Şekil 3.7'de gösterilmiştir (Dubois, 2003).



Şekil 3.6: (a) $(-\infty, \tilde{A}]$ bulanık eşiği, (b) $(-\infty, \tilde{A}[$ bulanık eşiği

Şekil 3.6 (a)'da gösterimi yapılan $(-\infty, \tilde{A}]$ bulanık eşiği, $\tilde{A}$ bulanık aralığı ile temsil edilen X değişkenine eşit veya daha küçük olabilecek sayıların bulanık kümesini, Şekil 3.6 (b) ile gösterimi yapılan $(-\infty, \tilde{A}[$ bulanık eşiği ise X değişkeninden küçük olabilecek sayıların bulanık kümesini temsil etmektedir. Benzer ilişkiler Şekil 3.7'de büyüklük durumu için gösterilmiştir.



Şekil 3.7: (a) $[\tilde{A}, +\infty)$ bulanık eşiği, (b) $]\tilde{A}, +\infty)$ bulanık eşiği

3.6 Bulanık işlemler

X ve *Y* değişkenlerini temsil eden $\tilde{A}$ ve $\tilde{B}$ bulanık aralıkları kullanılarak $f(\tilde{A},\tilde{B})$ fonksiyonu ile *Z* değişkenine ait olabilirlik dağılımı elde edilecek olsun. Bu durumda $Z = f(\tilde{A},\tilde{B})$'nin üyelik fonksiyonu Zadeh (1975) tarafından bulanık kümeler için ortaya konulan yayılım prensibi ile elde edilir. $f(\tilde{A},\tilde{B})$ fonksiyonu ile tanımlanan herhangi bir * işlemi için yayılım prensibi kullanılarak işlem sonucunda elde edilecek *Z* bulanık değişkenin üyelik fonksiyonu (3.21) denklemi ile belirlenebilir.

$$\mu_{\tilde{A}*\tilde{B}}(z) = \sup_{x,y:x*y=z} \min\left\{\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(y)\right\}, z \in \square \qquad \textbf{(3.21)}$$

$\tilde{A}$ ve $\tilde{B}$ bulanık aralıklarının dört işlemi yayılım prensibi ile (3.22) - (3.25) denklemleri ile tanımlanabilir (Dubois ve Prade, 1988).

$$\mu_{\tilde{A}+\tilde{B}}(z) = \sup\left\{\min\left(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(z-x)\right) \middle| x \in \square\right\} \qquad \textbf{(3.22)}$$

$$\mu_{\tilde{A}-\tilde{B}}(z) = \sup\left\{\min\left(\mu_{\tilde{A}}(z+x), \mu_{\tilde{B}}(x)\right) \middle| x \in \square\right\} \qquad \textbf{(3.23)}$$

$$\mu_{\tilde{A}.\tilde{B}}(z) = \begin{cases} \sup\left\{\min\left(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(z/x)\right) \middle| x \in \square - \{0\}\right\}, & \text{eğer z} \neq 0, \\ \text{maks}\left(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(z/x)\right), & \text{eğer z=0.} \end{cases} \qquad \textbf{(3.24)}$$

$$\mu_{\tilde{A}\div\tilde{B}}(z) = \sup\left\{\min\left(\mu_{\tilde{A}}(x.z), \mu_{\tilde{B}}(x)\right) \middle| x \in \square\right\} \qquad \textbf{(3.25)}$$

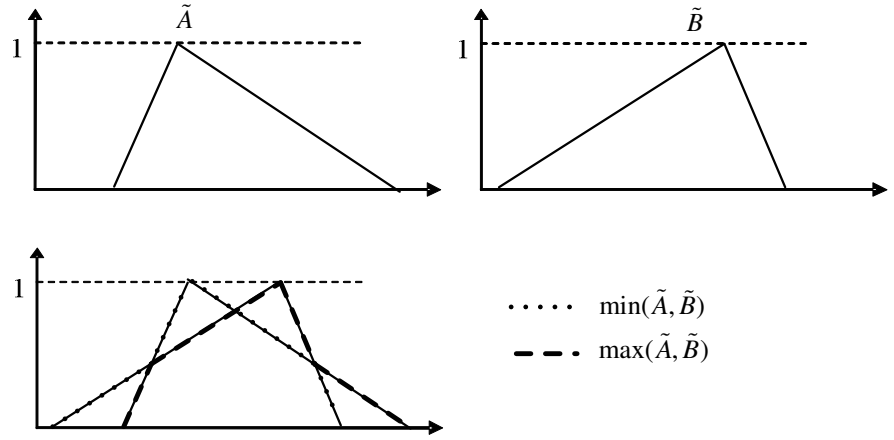
X ve *Y* tanım aralıklarının sürekli olduğu durumlarda (3.22) - (3.25) denklemleri ile sonsuz defa hesaplama yapılması gerekebilir. Bu nedenle bulanık aralık işlemleri için işlem gören bulanık aralıkların şekillerine göre genellikle 2-21 kesimlik λ-kesimleri kullanılmaktadır. Örneğin, işlem gören bulanık aralıkların her biri üçgen veya yamuk şekilli ise aritmetik işlemler için λ={e,1} olmak üzere 2 düzeyde kesim yapılması yeterli olmaktadır.

$\tilde{A}$ ve $\tilde{B}$ bulanık aralıklarının toplama ve çıkarma işlemler için, $\lambda \in (0,1]$ olmak üzere aşağıdaki (3.26) ve (3.27) denklemleri kullanılabilir.

$$(\tilde{A}+\tilde{B})^{\lambda}=[\underline{a}(\lambda)+\underline{b}(\lambda),\ \overline{a}(\lambda)+\overline{b}(\lambda)] \quad \textbf{(3.26)}$$

$$(\tilde{A}-\tilde{B})^{\lambda}=[\underline{a}(\lambda)-\overline{b}(\lambda),\ \overline{a}(\lambda)-\underline{b}(\lambda)] \quad \textbf{(3.27)}$$

X ve Y değişkenlerinin büyüklük (maks) ve küçüklük (min) işlemleri $\tilde{A}$ ve $\tilde{B}$ bulanık aralıkları arasındaki ilişki ile incelenir. Yayılım prensibi kullanılarak $\tilde{A}$ ve $\tilde{B}$ bulanık aralıklarından büyük veya küçük olacak bir bulanık aralık için olabilirlik dağılımı elde edilir. Şekil 3.8'de görüldüğü gibi $\text{maks}(\tilde{A},\tilde{B})$ ve $\min(\tilde{A},\tilde{B})$ ile elde edilen olabilirlik dağılımlarının şekli $\tilde{A}$ veya $\tilde{B}$'den farklı olabilmektedir.



Şekil 3.8: Bulanık maks ve min işlemleri

$\tilde{A}$ ve $\tilde{B}$ bulanık aralıklarının maks ve min işlemler için, $\lambda\in(0,1]$ olmak üzere aşağıdaki (3.28) ve (3.29) denklemleri kullanılabilir.

$$\left(\text{maks}\{\tilde{A},\tilde{B}\}\right)^{\lambda}=\left[\text{maks}\{\underline{a}(\lambda),\ \underline{b}(\lambda)\},\ \text{maks}\{\overline{a}(\lambda),\overline{b}(\lambda)\}\right] \quad \textbf{(3.28)}$$

$$\left(\min\{\tilde{A},\tilde{B}\}\right)^{\lambda}=\left[\min\{\underline{a}(\lambda),\underline{b}(\lambda)\},\ \min\{\overline{a}(\lambda),\overline{b}(\lambda)\}\right] \quad \textbf{(3.29)}$$

Bulanık aralıkların büyükten küçüğe veya tersine sıralandırılmaları için birbirleriyle karşılaştırılmaları ile min ve maks operatörlerinin kullanımıyla belirlenen olabilirlik dağılımları birbirlerinden farklı konulardır ve karıştırılmamaları gerekir.

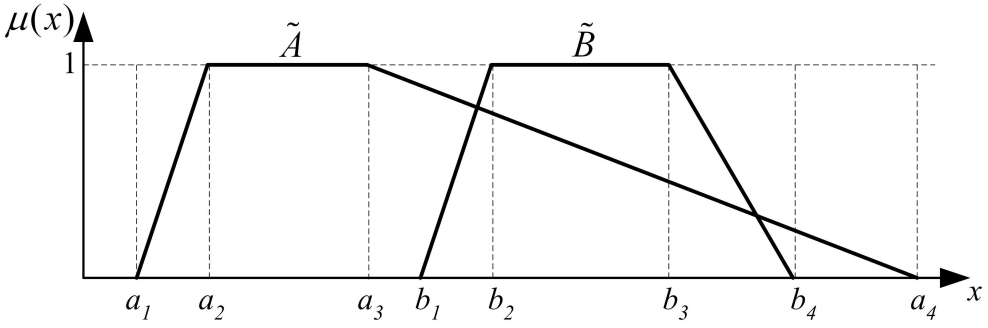
Örneğin, bir çizelgeleme probleminde, bir işe ait operasyonun bir makineye atanması durumunda operasyonun makinedeki en erken başlangıç zamanını tespit edebilmek

için; söz konusu işin bir önceki operasyonunun bitiş zamanı ile atama yapılacak olan makinedeki en son operasyonun bitiş zamanlarının karşılaştırılması gerekmektedir. İşlem sürelerinin kesin olarak bilindiği modellerde iki kesin sayı içerisinden büyük olanı tespit etmek oldukça kolaydır. Ancak sürelerdeki belirsizliklerin bulanık aralıklarla modellendiği bir problemde karşılaştırılmasına ihtiyaç duyulan zamanlara maks operatörü uygulanarak atanması yapılmak istenen operasyonun en erken başlangıç zamanının da bir bulanık aralık olarak belirlenmesi uygun bir yaklaşım olacaktır. Aynı problemin farklı çözümleri için amaç fonksiyonu değerlerinin birbirleriyle karşılaştırılarak en iyi çözümün belirlenmesi durumunda ise eğer amaç fonksiyonu değeri bir bulanık aralık olarak elde ediliyorsa söz konusu aralıkların birbirleriyle karşılaştırılarak küçükten büyüğe veya büyükten küçüğe sıralandırılmaları gerekmektedir. Bulanık aralıkların sıralandırılması için genellikle durulaştırma yöntemi kullanılmaktadır. Durulaştırma, bulanık değerlerin kesin değerlere dönüştürülmesidir. Durulaştırma işlemi için önerilmiş olan çok sayıda yöntem mevcuttur. Durulaştırma işlemi sonunda elde edilen kesin sayı değeri bulanık değerlerin sıralandırılması için kullanılmaktadır. Dubois ve diğ. (2003a), durulaştırma işlemi için önerilen yöntemleri incelemişler, Fortemps ve Roubens (1996) tarafından önerilen AC (area compensation) yönteminin diğerlerine göre insan muhakemesine en yakın olmakla birlikte toplanabilme ve doğrusallık özelliklerine de sahip olduğunu belirtmişlerdir. Bulanık bir $\tilde{A}$ değeri, AC yöntemi ile (3.30) denklemi kullanılarak durulaştırılabilir. $[\underline{a}(\lambda), \overline{a}(\lambda)]$, $\tilde{A}$'nın λ-kesimidir.

$$AC(\tilde{A}) = \frac{1}{2}\int_0^1 (\underline{a}(\lambda) + \overline{a}(\lambda))d\lambda \qquad \textbf{(3.30)}$$

3.7 Bulanık öncelik ilişkileri

Bulanık aralıklarda öncelik ilişkileri, bulanık aralıkların gerçekleşecek değerlerinin öncelikleri ile tanımlanmaktadır. *X* ve *Y* değişkenlerinin olabilirlik dağılımları $\tilde{A}$ ve $\tilde{B}$ bulanık aralıkları ile tanımlanmış olsun (Şekil 3.9).



Şekil 3.9: *X* ve *Y* değişkenlerini tanımlayan $\tilde{A}$ ve $\tilde{B}$ bulanık aralıkları

X değişkeninin gerçekleşecek değeri ile *Y* değişkeninin gerçeklececek değeri arasındaki öncelik ilişkileri için Dubois ve Prade (1988) aşağıdaki olabilirlik ve gereklilik indislerini önermişlerdir;

$$\prod(X \geq Y) = \sup_{x \geq y} \min\left\{\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(y)\right\} \tag{3.31}$$

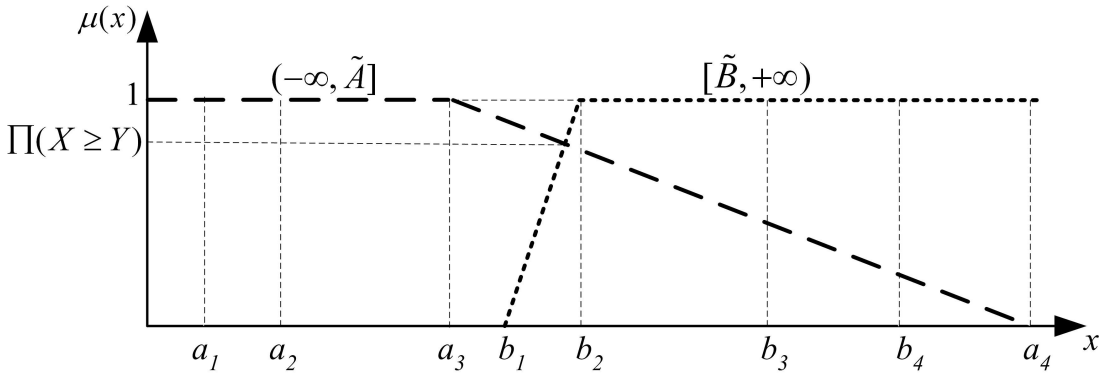
$$\prod(X > Y) = \sup_{x} \inf_{y \geq x} \min\left\{\mu_{\tilde{A}}(x), 1 - \mu_{\tilde{B}}(y)\right\} \tag{3.32}$$

$$N(X > Y) = 1 - \prod(Y \geq X) \tag{3.33}$$

$\prod(X \geq Y)$, *X* değişkeninin gerçekleşecek değerinin *Y* değişkeninin gerçekleşecek değerine eşit veya daha büyük olma olayının olabilirliğini temsil etmektedir. Bu olay *X* değişkenin, $[\tilde{B}, +\infty)$ bulanık eşiğine ait olması veya *Y* değişkenin, $(-\infty, \tilde{A}]$ bulanık eşiğine ait olması durumlarını içermektedir. $\prod(X \geq Y)$, (3.34) denklemi kullanılarak hesaplanabilir.

$$\begin{aligned}
&\prod(X \geq Y) = \prod\left(X \in [\tilde{B}, +\infty) \cup Y \in (-\infty, \tilde{A}]\right) \\
&= \text{maks}\left\{\sup_{x} \min\left(\mu_{\tilde{A}}(x), \mu_{[\tilde{B}, +\infty)}(x)\right), \sup_{y} \min\left(\mu_{\tilde{B}}(y), \mu_{(-\infty, \tilde{A}]}(y)\right)\right\} \\
&= \sup_{x} \min\left(\mu_{(-\infty, \tilde{A}]}(x), \mu_{[\tilde{B}, +\infty)}(x)\right)
\end{aligned} \tag{3.34}$$

$\prod(X \geq Y)$ değerinin hesaplanması Şekil 3.10'da gösterilmiştir.

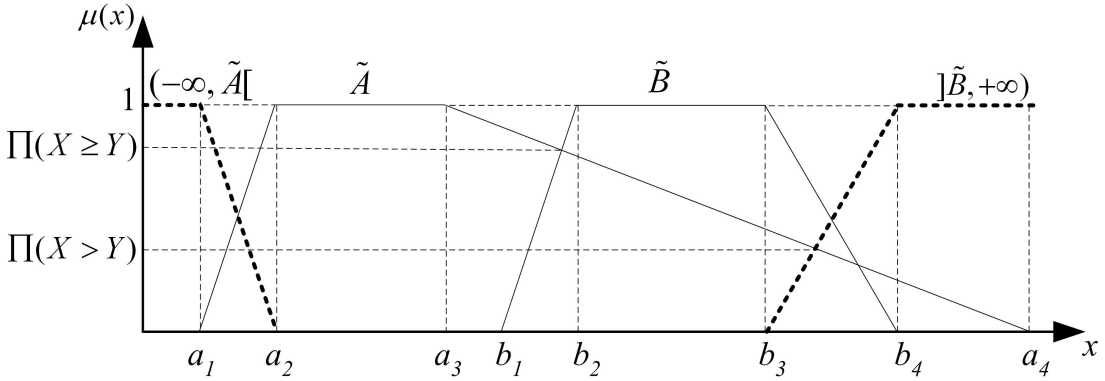


Şekil 3.10: $\prod(X \geq Y)$ ölçütü

$\prod(X > Y)$ ise *X* değişkeninin gerçekleşecek değerinin, *Y* değişkeninin gerçekleşecek değerinden büyük olabilirliğini temsil etmektedir. Bu olay *X* değişkenin gerçekleşecek değerinin, *Y* değişkeni değeriden büyük olan sayıların bulanık kümesini temsil eden $]\tilde{B}, +\infty)$ bulanık eşiğine ait olmasını veya *Y* değişkenin gerçekleşecek değerinin, *X* değişkeni değerinden küçük olan sayıların bulanık kümesini temsil eden $(-\infty, \tilde{A}[$ bulanık eşiğine ait olması durumlarını içermektedir, (3.35) eşitliği ile hesaplanabilir.

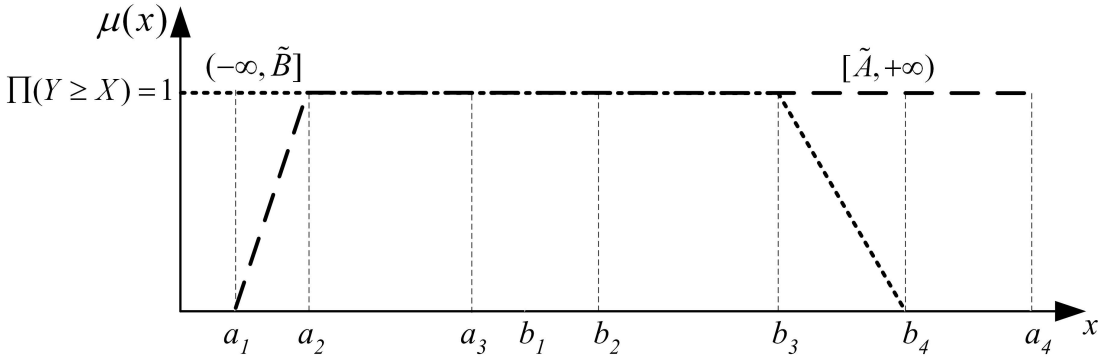
$$\begin{aligned} \prod(X > Y) &= \prod\left(X \in]\tilde{B}, +\infty) \cup Y \in (-\infty, \tilde{A}[\right) \\ &= \text{maks}\left\{ \sup_x \min\left(\mu_{\tilde{A}}(x), \mu_{]\tilde{B}, +\infty)}(x)\right), \sup_y \min\left(\mu_{\tilde{B}}(y), \mu_{(-\infty, \tilde{A}[}(y)\right)\right\} \end{aligned} \quad \textbf{(3.35)}$$

$\prod(X > Y)$ ve $\prod(X \geq Y)$, Şekil 3.11'de gösterilmiştir.



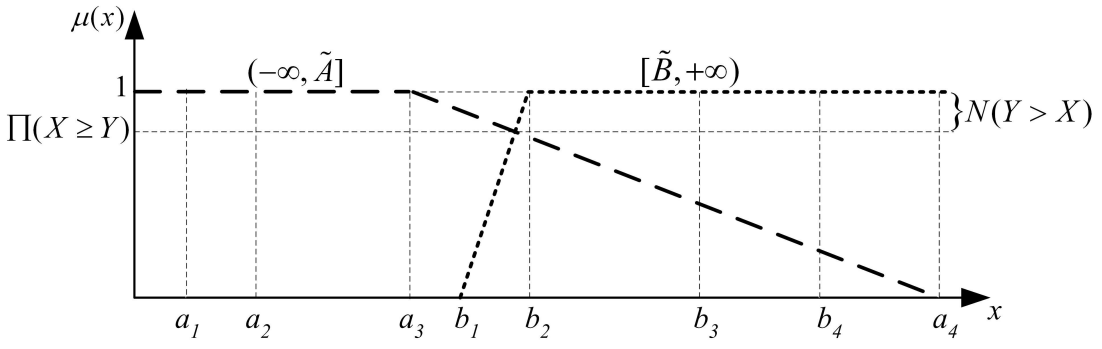
Şekil 3.11: $\prod(X > Y)$ ölçütü

$N(X > Y)$ ölçütü, X değişkenin gerçekleşecek değerinin Y değişkenin gerçekleşecek değerinden büyük olması olayının gerekliliğini ifade etmektedir. Daha önce belirtildiği gibi, gereklilik ölçütü ilgilenilen olayın gerçekleşmemesi yönündeki bilgi kullanılarak hesaplanmaktadır. Ancak ve ancak ilgilenilen olayın gerçekleşmemesi (tümleyen olayın gerçekleşmesi) yönünde herhangi bir bilgi yoksa gereklilik ölçütü 1 değerini alabilmektedir. $N(X > Y) = 1 - \prod(Y \geq X)$ değerinin grafiksel gösterimi Şekil 3.12'de yer almaktadır.



Şekil 3.12: $\prod(Y \geq X)$ ölçütü

Şekil 3.12'de görüldüğü gibi $\prod(Y \geq X) = 1$ olduğundan $N(X > Y) = 0$'dır. Şekil 3.13'te $\prod(X \geq Y)$ ölçüt değeri kullanılarak $N(Y > X)$ ölçütü değeri gösterilmiştir.



Şekil 3.13: $N(Y > X)$ ölçütü değeri

(a_1, a_2, a_3, a_4) ile tanımlanan yamuk şekilli $\tilde{A}$ bulanık aralığı ve (b_1, b_2, b_3, b_4) ile tanımlanan yamuk şekilli $\tilde{B}$ bulanık aralığı için öncelik ilişkileri, olabilirlik ve gereklilik ölçütü değerleri ile (3.36) – (3.38) denklemleri kullanılarak hesaplanabilir.

$$\prod(X \geq Y) = \text{maks}\left(0, \min\left(1, 1 + \frac{a_3 - b_2}{(a_4 - a_3) + (b_2 - b_1)}\right)\right) \quad \textbf{(3.36)}$$

$$\prod(X > Y) = \text{maks}\left(0, \min\left(1, \frac{a_4 - b_3}{(a_4 - a_3) + (b_4 - b_3)}\right)\right) \quad \textbf{(3.37)}$$

$$N(X > Y) = \text{maks}\left(0, \min\left(1, \frac{a_2 - b_3}{(a_2 - a_1) + (b_4 - b_3)}\right)\right) \quad \textbf{(3.38)}$$

Her durumda $\prod(X \geq Y) \geq \prod(X > Y) \geq N(X > Y)$ geçerlidir.

X değişkeninin alacağı değer ile *Y* değişkeninin alacağı değerlerin birbirine eşit olma olayının olabilirliği (3.39) ile hesaplanabilir.

$$\prod(X = Y) = \min\{\prod(X \geq Y), \prod(Y \geq X)\} \quad \textbf{(3.39)}$$

X ve *Y* değişkenlerinin alacağı değerlerin birbirinden farklı olması olayının gerekliliği (3.40) denklemi kullanılarak hesaplanabilir.

$$N(X \neq Y) = 1 - \prod(X = Y) \quad \textbf{(3.40)}$$

X değişkeni için $\tilde{A}$ bulanık kümesi ile tanımlanan $[\tilde{A}, +\infty)$ ve $]\tilde{A}, +\infty)$ bulanık eşikleri için kesin bir *r* sayısının bu eşiklerden büyük/küçük olma durumlarının olabilirliği ve gerekliliği *r* sayısı $r_1 = r_2 = r_3 = r_4 = r$ olmak üzere bulanık bir (r_1, r_2, r_3, r_4) aralığı yapısında ele alınarak hesaplanabilir.

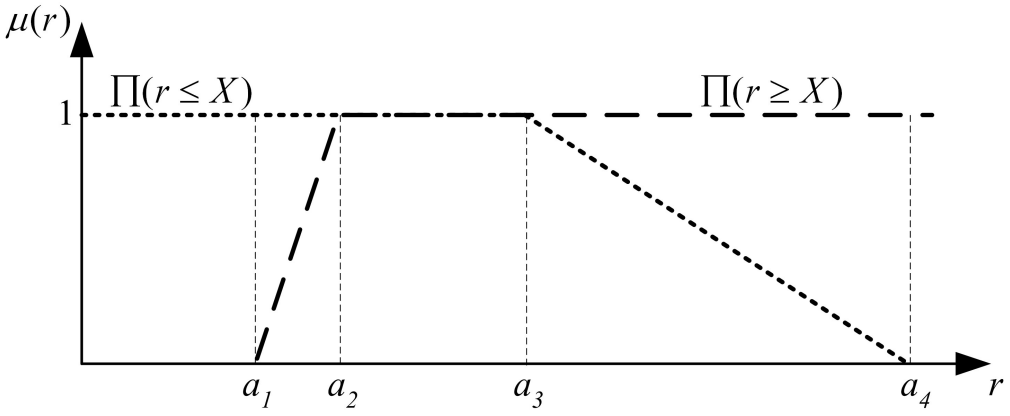
$$\prod(r \geq X) = \mu_{[\tilde{A}, +\infty)}(r) = \text{maks}\left(0, \min\left(1, 1 + \frac{r - a_2}{(a_2 - a_1)}\right)\right) \quad \textbf{(3.41)}$$

$$\prod(r > X) = \sup \mu_{]\tilde{A}, +\infty)}(r) = \text{maks}\left(0, \min\left(1, \frac{r - a_3}{(a_4 - a_3)}\right)\right) \quad \textbf{(3.42)}$$

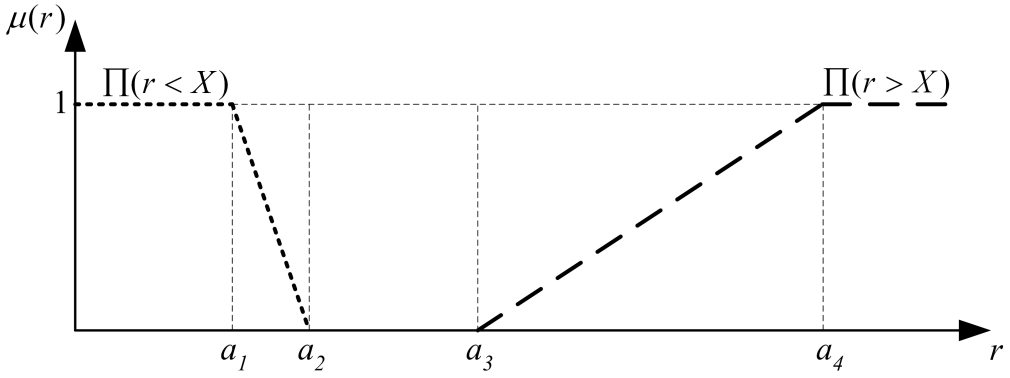
$$N(r > X) = 1 - \prod(r \leq X) = \prod(r > X) \quad \textbf{(3.43)}$$

$$N(r \geq X) = 1 - \prod(r < X) = \prod(r \geq X) \quad \textbf{(3.44)}$$

Kesin bir *r* sayısı ile $\tilde{A}$ bulanık sayısı ile tanımlanmış *X* değişkeni arasındaki öncelik ilişkileri Şekil 3.14 ve Şekil 3.15'te gösterilmiştir.



Şekil 3.14: $\prod(r \geq x)$ ve $\prod(r \leq X)$ ölçütleri



Şekil 3.15: $\prod(r < X)$ ve $\prod(r > X)$ ölçütleri

3.8 Bulanık programlama

20. yüzyılın ikinci yarısından itibaren üretim planlama ve çizelgeleme, yerleştirme, ulaşım, finans, tasarım vb. konularda yoğunlaşan optimizasyon çalışmalarının büyük bir kısmında göz ardı edilmiş olsa da gerçek hayattaki birçok parametre bilgi eksikliği veya doğal nedenlerden dolayı belirsizlik içermektedir. Belirsizlik altında optimizasyonun en önemli zorluğu ise belirsizlik uzayının büyüklüğü nedeniyle küçük boyutlu problemlerin bile sıklıkla büyük boyutlu optimizasyon problemine dönüşmesinden kaynaklanmaktadır. Ayrıca, ilgilenilen problem için oluşturulan modelin tamsayılı değişkenler içermesi durumunda problemin zorluğu daha da

artmaktadır. Yarım yüzyıldan fazla bir süredir üzerinde çalışılan ve hem teori hem de algoritma konusunda önemli gelişmelerin ortaya konduğu belirsizlik altında optimizasyon konusu halen en önemli açık problemlerden biri olarak değerlendirilmektedir.

Belirsizliklerin modellenmesi için kesikli ve sürekli olasılık dağılımlarının kullanıldığı stokastik programlama yaklaşımı geçmiş yıllarda en fazla kullanılan yöntem olmuştur. Daha yakın tarihli çalışmaların ise bulanık programlama yaklaşımı üzerinde yoğunlaştığı görülmektedir. Bulanık programlama kabaca stokastik olmayan belirsizliklerin bulunduğu matematiksel programlama olarak tanımlanabilir (Dubois ve Prade, 1988). Bulanık programlamada belirsiz parametreler bulanık aralıklar veya sayılar ile belirsiz kısıtlar ise bulanık kümeler ile modellenmektedir. Gerçek hayat problemleri incelendiğinde genellikle iki tip belirsizlik ile karşılaşıldığı görülmektedir.

Birinci tip belirsizlik, hemen hemen tüm nesnelerin değişimleri sürekli olan fiziki özelliklere sahip olmasından kaynaklanmaktadır. Aslında hiç kimse tam olarak 175 cm uzunluğunda değildir, ancak aşağı yukarı veya kabaca 175 cm uzunluğunda olan pek çok insan vardır. Ölçümleri anlamlı bir doğrulukla yuvarlamak genellikle mantıklı ve yararlı olmasına rağmen 1 veya 0'a başka bir deyişle yanlış veya doğruya yuvarlama yapılması durumunda önemli zorluklar ortaya çıkmaktadır. Bazı insanlar için "uzun boyludur" veya "kısa boyludur" ifadelerini kullanmak çok kolaydır ve ifadenin doğruluğu konusunda herhangi bir belirsizlik söz konusu değildir. Ancak uzun veya kısa boylu olmasına kolayca karar verilemeyen bir çok insan vardır. Bu insanlar uç/sınır veya arada kalma durumlarını temsil etmektedir ve kesin sınırların belirlenememesi/esneklik içerme (vagueness) anlamındaki belirsizliğe temel oluşturmaktadır. Kesikli olmama, ilgilenilen alan sınırlarının tam olarak belirlenememesi veya kesin ayrımların yapılamaması durumunu ifade etmektedir. Bulanık kümeler bu tip belirsizliklerin modellenebilmesine imkan sağlamaktadır. Bulanık programlamada, gerçek hayatta sürekli değerler alan değişkenler amaç fonksiyonu değeri kabul sınırı (aspirasyon kriteri) olarak veya kısıtların sağ tarafında bulanık kümeler ile modellenerek kullanılabilmektedir. Örnek olarak, bir iş çizelgeleme probleminde J_i işi için teslim tarihi müşteri tarafından "saat 10 gibi"

olarak ifade ediliyorsa, oluşturulan modelde J_i işinin teslim tarihi ile ilgili sağ taraf kısıtı müşteri tatmin düzeyini temsil eden bir bulanık küme ile modellenebilmektedir.

İkinci tip belirsizlik ise muğlaklık veya şüpheli olma (ambiguity) anlamında ortaya çıkan belirsizliktir. Birden fazla tercih arasındaki seçim kriteri tam olarak ortaya konmamıştır. Olasılık anlamındaki belirsizliğin genelleştirilmiş hali olarak da düşünülebilir. Bir elemandan birden çok elemana ilişki içerir. Yukarıdaki örnekte J_i işinin işlem süresi için "yaklaşık 2 dakika" ifadesi ile henüz bilinmeyen ancak uygulama anında kesinleşecek bir değer için muğlaklık anlamında belirsizlik ortaya konmaktadır. 2 civarında bir değerin gerçekleşeceği doğrudur ancak kesin olarak bilinmemektedir. Muğlaklık anlamındaki belirsizlik bilgi eksikliğinden veya doğadaki rassallıktan kaynaklanabilir. Bulanık programlamada muğlak parametreler bulanık aralıklar ile modellenmektedir.

Bulanık programlama modelleri içerdikleri belirsizliklere göre üç gruba ayrılabilir (Inuiguchi ve Ramik, 2000):

- Esneklik içeren bulanık programlama
- Muğlaklık içeren bulanık programlama
- Esneklik ve muğlaklık içeren bulanık programlama

Esneklik içeren bulanık programa modeli ilk olarak Bellman ve Zadeh (1970) tarafından ortaya konmuştur. Daha sonra Tanaka ve diğ. (1974) ve Zimmermann (1976, 1978) tarafından geliştirilmiştir. Bulanık hedefler ve kısıtlar (sağ taraf) durumunda karar vermeyi gerektiren bir problemdir. Bulanık hedefler ve kısıtlar amaç fonksiyonu hedef değeri için esnekliği, kısıtlar için elastikiyeti/toleransı temsil etmektedir. Bu nedenle *esnek programlama* olarak da adlandırılmaktadır. Klasik bir doğrusal programlama modeli (3.45) ile gösterilebilir;

$$\begin{aligned} \min \quad & c^T x \\ & Ax \le b \end{aligned} \qquad \textbf{(3.45)}$$

Bu problemde karar değişkeni vektörü olan *X*, $Ax \le b$ kısıtını sağlamalıdır. Ancak e'nin en küçük pozitif değerleri için bile $b < Ax \le b + e$ durumları aday bir çözüm olarak değerlendirilmemektedir. Uygulama alanında ise sağ taraf değerlerinin tam bir kesinlikte kısıtlayıcı olmadığı, karar verici için sol taraf denklemlerinden elde

edilebilecek farklı sonuçların $Ax \le b + e$ durumunda kabul edilebilir olması sıklıkla karşılaşılan bir durumdur. Bu tip durumlarda, sağ taraf kısıtlarının tam bir kesinlikle modellenmesi yerine daha zayıf veya esnek bir ifadeyle modellenmesine ihtiyaç duyulmaktadır. Bulanık programlama ile bu tip kısıtlar modellenebilmektedir. $B = \{r : r \le b\}$ olarak tanımlanırsa yukarıdaki klasik programlama modelindeki kısıt denklemi yerine $Ax \in B$ kullanılabilir. Bulanık programlamada, B klasik kümesi yerine, "takriben b'den küçük değerler"i temsil eden $\tilde{B}$ bulanık kümesi kullanılarak ve $Ax \in \tilde{B}$ kısıt denklemi ile $\mu_{\tilde{B}}(Ax)$ üyelik dereceleri kontrol edilerek esnek/zayıf kısıtlar modellenebilmektedir. $\tilde{B}$ kümesi bulanık kısıt olarak adlandırılmaktadır.

Bulanık programlamada ikinci kategori olan muğlaklık içeren modeller ise amaç fonksiyonu katsayıları ve kısıt denklemi katsayıları için muğlak değerlerin kullanıldığı durumdur. Bu modellerde bulanık hedefler ve kısıtlar yer almamaktadır. Dubois ve Prade (1980b) tarafından ilk olarak doğrusal denklemler için muğlak katsayıların kullanımı ile ortaya konmuştur. Daha sonra Dubois (1987) bulanık katsayılı matematiksel programlama problemlerindeki bulanık aralıklar için olabilirlik teorisini temel alarak dört eşitsizlik indisi önermiştir. Böylece bulanık katsayılar, katsayı değerleri için olabilirlik dağılımları olarak tanımlanmıştır. Bu tip matematiksel programlama modelleri ise *olabilirlik programlama* olarak adlandırılmıştır. Yukarıdaki (3.45) klasik doğrusal programlama modelinde c ve A katsayılarının gerçel sayı olarak bilindiği varsayılmaktadır. Ancak, uygulama alanında karşılaşılan problemlerde bu katsayıların gerçel bir sayı olarak belirlenmesi için yeterli bilgi bulunmamakta veya rassallık nedeniyle genellikle "c_i yaklaşık 3'tür" gibi bir bilginin modellenmesine ihtiyaç duyulmaktadır. Bu tip durumlarda c ve A katsayılarının dahil olabileceği aralıklar $\tilde{c}$ ve $\tilde{A}$ bulanık kümeleri ile temsil edilebilir. Ax'in ve c'nin gerçekleşebileceği aralıklar ve olabilirlikleri $\tilde{A}x$ ve $\tilde{c}$ bulanık kümeleri ile modellenebilir. $B \in \tilde{A}x$ kısıt denklemi ile $\mu_{\tilde{A}x}(B)$ üyelik dereceleri kontrol edilerek muğlak kısıtlar modellenebilmektedir.

Üçüncü tip bulanık programlama modeli ise sağ taraf kısıtlarında esnekliğin, amaç fonksiyonu ve kısıt katsayılarında ise muğlaklığın birarada bulunduğu durumdur. Negoita ve diğ. (1976) tarafından ilk defa bulanık doğrusal programa problemi

olarak ortaya konmuştur. Karar vericinin tercihi bulanık bir tatmin alanı ile modellenmiştir. Bu tip bulanık programlama modelleri ise *gürbüz programlama* olarak adlandırılmıştır. Bulanık programlama modellerinin en genel halidir. $\tilde{A}x \in \tilde{B}$ kısıt denklemi ile $\mu_{\tilde{A}x}(\tilde{B})$ üyelik dereceleri kontrol edilerek muğlak kısıtlar modellenebilmektedir.

Geçmiş çalışmaların çoğunda bulanık programlama problemleri olabilirlik teorisi temel alınarak şans kısıtlı programlama problemi olarak formüle edilmişlerdir. Bulanık programlama modellerinde oluşturulan bir çözümün uygun bir çözüm olup olmayacağı konusunda karar vericiye olabilirlik ve gereklilik ölçütleri kullanılarak bilgi sağlanmaktadır. Bulanık matematiksel programlama problemleri ile ilgili geçmişte fazla sayıda çalışma yapılmış, özellikle bulanık doğrusal programlama problemlerinin çözümü için oldukça başarılı yöntemler önerilmiştir. Bulanık doğrusal programlama problemi; amaç fonksiyon değerinin belirli bir kesin değerden büyük/küçük olmasına ilişkin gereklilik ölçütünün maksimum yapılması (modal, minimum risk, aspirasyon kriteri yaklaşımları) şeklinde veya belirli bir gereklilik ölçütü eşik değeri belirlenerek bu eşik değerin üstünde gereklilik ile amaç fonksiyonu değerinin maksimum yapılması şeklinde (fraktal yaklaşım) ele alınabilmektedir. Kısıtların sağlanması için ise minimum gereklilik ölçeği sınır değeri belirlenerek ve güçlü varsayımlar kullanılarak gürbüz programlama modelleri klasik doğrusal programlama modellerine dönüştürülebilmektedir (Inguiguchi ve Ramik 2000).

Yakın zamanda ise araştırmaların bulanık kombinatoryal problemler üzerinde yoğunlaşmaya başladığı görülmektedir. Bazı araştırmacılar genetik algortima, tavlama benzetimi, tabu arama ve karınca kolonisi optimizasyonu gibi popüler metasezgisel yöntemlerle çizelgeleme ve proje seçimi gibi problemler için çalışmalar ortaya koymuştur (Ishibuchi ve diğ., 1994; Tsujimura ve diğ., 1995; Hapke ve Slowinski, 1996; Sakawa ve diğ., 1997; Zheng ve Liu, 2006; Kılıç ve Kahraman, 2006a; Kılıç ve Kahraman 2007). Öte yandan bulanık kombinatoryal problemler üzerine teorik yaklaşımların geliştirildiği çalışmaların da devam ettiği gözlenmektedir.

Bulanık karar ortamında ele alınan çizelgeleme problemleri incelendiğinde, işlem veya yolculuk sürelerindeki belirsizliklerin olabilirlik teorisi ile modellenerek tanımlandığı görülmektedir. Olabilirlik teorisi ile olasılık teorisine göre daha basit modellerle ve daha az veriye ihtiyaç duyularak belirsizlik içeren veriler ile çalışılmıştır. Problemlerdeki tercihler veya öncelikler için uzlaşma sağlayan en iyi çözümün bulunması amacıyla yapılan çalışmalarda ise tercihlerin veya önceliklerin esnek kısıtlarla modellenebildiği görülmüştür. Belirsiz verilerin bulanık sayılarla, müşteri tatmin düzeylerinin ise bulanık kümelerle ele alındığı çalışmaların iş çizelgeleme ve sıralama problemleri üzerinde yoğunlaştığı görülmektedir. Ishii ve diğ. (1992), Han ve diğ. (1994) ve Tanaka ve Vlach (1997) bulanık teslimat zamanlı tek makina problemi üzerinde çalışmışlardır. Itoh ve Ishii (1999), Wang ve diğ. (2002), Sung ve Vlach (2003) ve Chanas ve Kasperski (2001) bulanık işlem zamanlı tek makina problemi üzerinde çalışmışlardır. Bu çalışmalarda çözümün optimalliği için farklı kıstaslar kullanılmıştır. Chanas ve Kasperski (2004), belirsizlik durumunda elde edilen bir çizelgenin optimalliği için olabilirlik ve gereklilik indislerini kullanmıştır. Önerdikleri yaklaşımda optimal olma durumu [0,1] arasında bir değerle temsil edilen ve belirsizlik içeren bir değerdir. Kasperski (2005), iş sıralama problemi için olabilirlik teorisi temelli bir yaklaşım önermiştir. Problemde yer alan ve belirsizlik içeren her parametre bir olabilirlik dağılımına sahiptir. Optimal sonucun bulunması amacıyla Dubois ve Prade (1988) tarafından önerilen indisler kullanılmıştır. Kılıç ve Kahraman (2007) çalışmasında bulanık işlem süreli ve teslimat zamanlı akış tipi çizelgeleme problemi için elde edilen çözümlerin uygulama aşamasında kısıtlara uygun olup olmayacakları gereklilik ölçütü ile ele alınmıştır.

Bulanık küme teorisi, belirsiz bilginin modelllenebilmesi için istatistiksel yöntemleri kullanmadan geçerli ve güçlü modellerin oluşturulabilmesine olanak sağlamıştır. Bulanık modeller ile hem istatistiksel modellerdeki gerçeklik kaybının önüne geçilebilmekte hem de yüksek veri işleme maliyeti düşürülmektedir (Rommelfanger, 1995). Etkileşimli çözüm sürecinin başlangıç aşamasında, bulanık sistem, sadece karar verici tarafından yüksek maliyet gerektirmeden sağlanan veri ile modellenebilir. Uzlaşık çözüm elde edildikten sonra ilerleyen aşamalar için hangi bilgilere ihtiyacı duyulacağına, bu bilgilerin elde edilme maliyeti ile bu bilgiler elde edildiğindeki kazanımların neler olabileceği karar verici tarafından değerlendirilebilir. Bu süreç devam ettirilerek uzlaşık çözümler aşamalı olarak

geliştirilebilir. Böylece sadece ihtiyaç duyulan bileşenler için veri elde edilerek ve sadece ihtiyaç duyulan kadarıyla verinin işlenmesi sağlanarak çözüm sürecinin maliyeti azaltılabilir.

4. KARINCA KOLONİSİ OPTİMİZASYONU

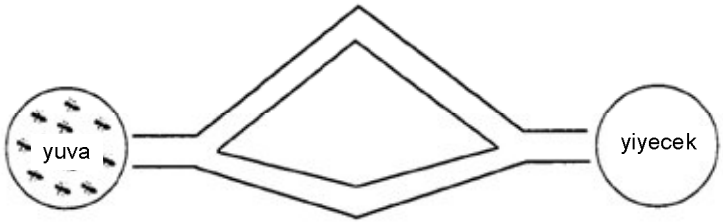
4.1 Gerçek karınca davranışı

Aralarında belirli bir iş paylaşımı olan ve doğada topluluk halinde yaşayan hayvanlar sosyal hayvanlar olarak adlandırılmaktadır. Karıncalar, arılar ve kuşlar gibi bir çok hayvan türü bu sınıfa girmektedir. Bu türler içerisindeki bireyler genellikle zeki olarak tanımlanamasa da grup olarak birbirleriyle ve doğa ile etkileşimleri sonucunda ortaya zeki davranışlar çıkabilmektedir. Genellikle ortak bir çevrede yaşayan, basit yapılara sahip ve basit davranışlar gösteren bireylerden oluşan bu tip sistemler, karmaşık ve bütünsel nitelikte davranışlar üretebilmektedir. Bu sistemlerdeki dikkat çekici unsur bireyselliğin öne çıkarılmadan ve herhangi bir liderliğe ihtiyaç duyulmadan sadece bireylerin kendilerine ait basit görevleri gerçekleştirmesi neticesinde karmaşık problemlere kısa sürede çözüm bulabilmeleridir. Bu tip sistemler incelendiğinde, zor problemlerin çözümünde kullanılabilecek bir çok özelliğin bulunduğu görülmektedir. Sosyal hayvan sınıfına giren karıncaların yiyecek arama ve yuvaya taşıma davranışları da özellikle kombinatoryal tip problemlerin çözümü için başarılı sonuçların elde edildiği bir yöntem olan KKO için esin kaynağı olmuştur.

KKO yönteminin esin kaynağı, Arjantin karıncası olarak adlandırılan bir karınca türünün yiyecek arama ve yuvaya taşıma davranışı üzerinde Deneubourg ve diğ. (1990) tarafından gerçekleştirilen gözlem ve deneylerin sonuçlarına dayanmaktadır. Çalışmada, karıncanın yiyecek ararken ve bulduğu yiyeceği yuvasına taşırken hangi yöne gideceğine nasıl karar verdiği sorusuna yanıt aranmıştır. Sonuç olarak, karıncayı yönlendiren bir dış etkinin bulunmadığı, bir iletişim ortamının kullanımıyla, kendini örgütleme (self-organization) süreci ile kendi kendini yönlendiren bir mekanizmanın bulunduğu gözlenmiştir. Kendini örgütleme, içinde bulunulan sistemin iç yapısının, sistem dışı bir yönlendirmeye veya etkiye maruz kalmadan gelişme göstermesidir. Arjantin karınca sistemi kendini örgütleyen doğal bir sistemdir. Karıncalar kendi aralarında dolaylı yoldan kurdukları etkileşim ile

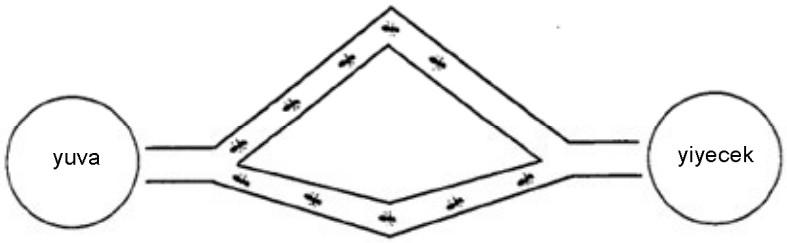
yüksek düzeyde bir organizasyon yapısı oluşturmaktadırlar. Karıncalar arasında görülen bu iletişim Türkçe bir sözcük ile karşılığı bulunamayan *stigmergy* terimi ile tanımlanmaktadır. Stigmergy, içinde bulunulan çevrenin kullanımıyla kurulan dolaylı bir iletişim türüdür. Çevrede değişiklik yapılmasıyla veya çevreye bazı maddelerin bırakılmasıyla sağlanır. Örneğin bir karınca bulunduğu çevrede herhangi bir değişiklik yapıyorsa ve başka bir karınca bu değişikliğe tepki veriyorsa bu iki karınca arasında stigmergy tipinde bir iletişim kurulmuştur. Kendini örgütleme ve stigmergy'nin beraber görüldüğü Arjantin karıncalarının yiyecek arama ve yuvaya taşıma davranışları ilginç bir süreçtir.

Arjantin karıncası yuvadan yiyecek kaynağına giderken veya bulduğu yiyeceği yuvaya taşırken yüzeye feremon adında kimyasal bir madde bırakmakta, diğer karıncalar da bu feremon izini takip etme eğilimi göstermektedirler. Karınca kolonisinin bu davranışı yiyecek kaynağı ile yuva arasındaki en kısa yolu bulmalarını sağlayabilmektedir. Karınca, feremon izi bırakarak çevrede değişiklik yaratmaktadır. Feremon izinin miktarı ise izi bulan karıncanın izi takip etme veya etmeme olasılığını belirlemektedir.



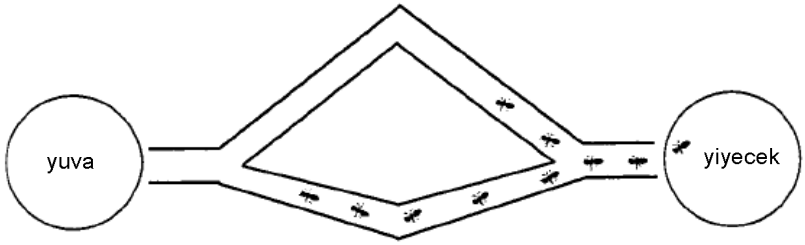
Şekil 4.1: Çift köprü deneyi şeması

Deneubourg ve diğ. (1990), Arjantin karıncasının yiyecek arama ve yuvaya taşıma davranışını incelemek için çift köprü deneyini gerçekleştirmişlerdir. Şekil 4.1'de şeması görülen düzeneği kullanmışlardır. Düzenekte, karıncanın yuva ile yiyecek kaynağı arasında kullanabileceği iki adet farklı uzunlukta yol (köprü) vardır. Deneyin başlangıcında tüm karıncalar yuvadadır ve serbest bırakılırlar. Aramanın başlangıç safhasında kullanılabilecek yollar üzerinde herhangi bir madde olmadığından, karınca rasgele seçimler yapacaktır. Üst yolun ve alt yolun yaklaşık olarak eşit sayıda karınca tarafından seçildiği başlangıç durumu Şekil 4.2'de gösterilmiştir. Karıncalar yürürken kullandıkları yol üzerine feremon izi bırakmaktadırlar.



Şekil 4.2: Başlangıç aşaması

Şekil 4.2'de görüldüğü gibi, alt yol daha kısa olduğundan, alt yolu kullanan karıncalar yiyecek kaynağına daha kısa sürede ulaşacak ve yuvaya geri dönüş için yola çıktıklarında alt yol üzerinde daha fazla iz miktarı algılayacaklarından tekrar alt yolu seçme olasılıkları daha fazla olacaktır. Bu süreç güçlü bir geri besleme döngüsü oluşturacak, kısa yol üzerindeki iz miktarı hızla artmaya başlayacak ve ilerleyen zamanda neredeyse tüm karıncaların artık kısa yolu kullanmaya başladıkları görülecektir (Şekil 4.3).



Şekil 4.3: Kısa yol üzerinde yoğunlaşma

4.2 KKO algoritmaları

Arjantin karıncalarının yiyecek arama davranışını temsil eden doğal sistemde yiyecek arama süreci kendini örgütleyen bir sistemin oluşmasını sağlamıştır. Yollar üzerine iz bırakılması ile çevrede yapılan değişiklik ve karıncaların bu değişikliğe tepki vermesi stigmergy için iyi bir örnektir. Kendini örgütleme ve stigmergy'nin birlikte görüldüğü bu doğal sistem, kombinatoryal problemlerin çözümü için esin kaynağı olmuştur. Karıncaların yiyecek arama davranışını temel alan bir yöntem için aşağıdaki unsurların üzerinde durulması gerekmektedir:

- **Olumlu geri besleme:** İyi çözümlerin desteklenmesi için gereklidir. Feremon birikmesi ile sağlanabilir. Çift köprü deneyinde gösterildiği gibi,

kısa yollar üzerindeki iz miktarı artmakta ve kısa yollar daha fazla karınca tarafından tercih edilmekte, bu durum kısa yol üzerindeki iz miktarını daha da arttırmaktadır.

- **Olumsuz geri besleme:** Erken yakınsamayı önlemek için ihtiyaç duyulur. Feremon izini buharlaştırarak sağlanabilir.
- **Değişikliklerin desteklenmesi:** Daha iyi çözümlerin elde edilebilmesi için rassal faktörler önemlidir. Bazen karıncanın üzerinde daha az iz bulunan veya hiç iz bulunmayan yolları da tercih edebilmesine olanak sağlayarak daha kısa yollar keşfedilebilir.
- **Çoklu etkileşim:** Karıncalar feremon izi ile iletişim kurmaktadır. Çevrede bulunan feremon izi aslında geçmişte yapılan arama süreci hakkında bilgi depolayan bir bellektir.

KKO metasezgiselinde yapay karıncaların, kombinatoryal problem için oluşturulan ağ yapısı üzerinde, doğal karıncalarının yiyecek arama sürecini taklit ederek en kısa yolları bulması amaçlanmaktadır. Bu tip algoritmalar için KKO algoritmaları (veya KKO metasezgiseli) tanımı ilk defa Dorigo ve Di Caro (1999a,b) çalışmalarında önerilmiştir. Dorigo ve diğ. (1991) tarafından önerilen Karınca Sistemi (KS) algoritması ise KKO metasezgiseli olarak adlandırılabilecek ilk algoritmadır.

KS, doğal karıncalara göre ilave yeteneklere sahip yapay karıncaların kullanımı ile gezgin satıcı probleminin çözümü için önerilmiştir. Yapay karıncaların sahip olduğu ilave yetenekler aşağıda sıralanmıştır:

- Ziyaret ettikleri noktaları belleklerinde tutarlar.
- Tamamen kör değildirler, aday noktalar hakkında sezgisel bilgiye sahiptirler.
- Zamanı kesikli bir değişken olarak kullanırlar.
- Elde edilen çözümün kalitesi ile orantılı olarak çözümü oluşturan bağlar üzerine iz bırakırlar.

KKO algoritmalarının doğal karıncaların yiyecek arama davranışlarından esinlendiği üç temel unsur aşağıdaki gibidir:

- Karıncaların üzerinde daha fazla iz bulunan yolu seçme olasılığı daha yüksektir.
- Kısa yollar üzerinde bulunan iz miktarındaki artış oranı daha yüksektir.

- Yollar üzerinde bulunan izler karıncaların dolaylı olarak iletişim kurmasını sağlar.

KKO algoritmalarında yapay karıncalar arasındaki işbirlikçi etkileşim yüksek kalitede çözümlerin oluşmasını sağlamaktadır. KKO metasezgiselinin birçok kombinatoryal problemin çözümünde kullanıldığı ve genellikle başarılı sonuçların elde edildiği görülmektedir (Dorigo ve Stützle, 2004).

KKO metasezgiselinin temelinde çok sayıda karıncanın yiyecek ararken gösterdiği davranışların yapay bir karınca sürüsü ile simülasyonu bulunmaktadır. Yapay karıncalar, yapay feremon izini ve probleme özgü sezgisel bilgiyi kullanarak stokastik çözüm oluşturucu bir sezgisel ile çözüm oluşturmaktadır. Çözüm oluşturucu sezgiseldeki stokastik unsur ile olasılıklara dayanan seçimler yapılarak stokastik olmayan bir sezgisele göre daha fazla çözümün araştırılabilmesine imkan sağlanmaktadır. Aynı zamanda, sezgisel bilginin kullanımı ve karıncalar arasındaki kollektif etkileşim ile karıncalar arama uzayının umut vaat eden bölgelerine yönlendirilmektedir. Çözüm oluşturma aşamasında yollar üzerine bırakılan iz miktarı aslında karıncaların arama uzayı hakkında elde ettikleri bilgiyi depolamaktadır, bir öğrenme süreci olarak da görülebilir. Karıncalar çözüm oluşturmak üzere serbest bırakıldıkları ağ yapısı üzerindeki düğümlerde hangi tarafa gideceklerini aday yollar üzerindeki iz miktarlarını ve aday yolların uzunluklarını değerlendirerek karar vermektedir. Kullandıkları geçiş kuralı, kısa ve iz yoğunluğu fazla olan yolu daha avantajlı kılmaktadır. Karıncalar çözüm oluşturduklarında daha önce tanımlanmış olan bir iz güncelleme kuralı ile kullandıkları yollar üzerine iz bırakmaktadırlar. Bu kural, bulunan çözümün kalitesine bağlı olarak kullanılan yollar üzerine bırakılacak olan iz miktarını da belirlemektedir. İyi çözümler için daha fazla iz bırakılmaktadır. Ayrıca bazı yollar üzerinde aşırı iz yoğunluğu oluşmasını önlemek için izlerin buharlaşması sağlanmaktadır. Böylece aramanın erken safhalarında alt-optimal bir çözüme yakınsama olması engellenmekte, arama uzayının yeni bölgelerinin keşfedilmesine olanak sağlanmaktadır.

KKO algoritmalarında, her yapay karınca, problem için oluşturulan ağ yapısı üzerinde bağımsız hareket ederek probleme bir çözüm oluşturmaktadır. Ağ yapısında bağlarla birbirine bağlı bulunan düğümler, karınca için oluşturmakta olduğu çözüme

ekleyebileceği aday elemanları temsil etmektedir. Her bağ üzerinde iki farklı türde bilgi depolanmaktadır:

- *i* düğümünden *j* düğümüne gitmenin sezgisel önceliğini belirlemede kullanılan bilgi. Bu bilgi arama süreci boyunca değişmemektedir.
- *i* düğümünden *j* düğümüne gitmenin öğrenilmiş çekiciliğini temsil eden, bağ üzerindeki yapay feremon iz miktarı ile temsil edilen bilgi. Bu bilgi arama süreci boyunca periyodik olarak güncellenmektedir.

Prensip olarak KKO algoritmaları bir çözüm oluşturma mekanizmasının tanımlanabildiği tüm kombinatoryal problemlerde uygulanabilir olarak değerlendirilmektedir. KKO algoritmalarının genel yapısı Şekil 4.4'te gösterilmiştir.

```
parametre değerlerini gir
başlangıç izlerini oluştur
while durma kriteri sağlanana kadar do
    çözümleri oluştur
    yerel arama-iyileştirme sezgiseli kullan (opsiyonel)
    çözümleri değerlendir
    izleri güncelle
end while
```

Şekil 4.4: KKO algoritmalarının genel yapısı

Kombinatoryal optimizasyon problemleri temelde gezgin satıcı problemi yapısında olmalarına rağmen ilave kısıtlar içermektedirler. Dolayısıyla bu tip problemlerde orijinal KS algoritmasının kullanımı ile iyi çözümler elde edilememektedir. Bu nedenle, KKO algoritmalarının farklı kombinatoryal problemlerin çözümünde kullanılması durumunda, yapay karıncaların uygun çözümler bölgesinde arama yapmalarına rehberlik edebilecek yerel sezgisel yöntemler kullanılmaktadır.

KS algoritması ile başarılı sonuçların elde edilmesi, sonraki yıllarda farklı uygulama alanlarındaki problemlerin çözümüne yönelik olarak farklı isimlere sahip birçok KKO algoritmasının önerilmesinde etkili olmuştur. Önerilen algoritmaların performansı genellikle diğer metasezgiel yöntemlerle baş edebilir veya daha iyi seviyede olmuştur. Geçmiş yıllarda KKO metasezgiseli ile farklı uygulama alanlarında gerçekleştirilen çalışmalar Tablo 4.1'de özetlenmiştir.

Tablo 4.1: KKO uygulama alanları ve geçmiş çalışmalar

Uygulama Alanı	Yazar(lar)	Algoritma ismi	Yıl
Gezgin Satıcı Problemi	Dorigo ve diğ.	AS	1991, 1996
	Gambardella ve Dorigo	Ant-Q	1995
	Dorigo ve Gambardella	ACS	1997
	Bullnheimer ve diğ.	AS_{rank}	1999a
	Stützle ve Hoos	MMAS	2000
	Cheng ve Mao	ACS-TSPTW	2007
Kuadratik Atama Problemi	Gambardella ve diğ.	AS-QAP	1999b
	Maniezzo ve Colorni	ANTS-QAP	1999
	Stützle ve Hoos	MMAS-QAP	2000
	Talbi ve diğ.	Parallel Ant Colonies	2001
	Solimanpur ve diğ.	ACO	2004
Çizelgeleme problemi	Colorni ve diğ.	AS-JSP	1994
	Stützle	AS-FSP	1998b
	McMullen	ACO	2001
	T'kindt ve diğ.	ACO	2002
	Gravel ve diğ.	ACO	2002
	Kılıç ve Kaylan	AS-Schedule	2003
	Ying ve Liao	ACS	2004
	Shyu ve diğ.	ACO	2004
	Blum	Beam-ACO	2005
	Kılıç ve Kahraman	MMAS	2006a
	Kılıç ve Kahraman	Fuzzy ACO	2006b
	Kılıç ve Kahraman	MMAS for FPFSP	2007
Araç Rotalama problemi	Bullnheimer ve diğ.	AS_{rank}-CVRP	1999b
	Gambardella ve diğ.	MACS-VRPTW	1999a
	Reimann ve diğ.	AS_{rank}-$CVRP_{sav}$	2002
	Reimann ve diğ.	D-ants	2004
	Bell ve McMullen	ACO	2004
	Donati ve diğ.	TDVRPTW	2008
Şebeke Rotalama	Schoonderwoerd ve diğ.	ABC	1996
	Di Caro ve Dorigo	AntNet	1998
Ardışık sıralama	Gambardella ve Dorigo	HAS-SOP	2000
Ağ boyama	Costa ve Hertz	ANTCOL	1997
Kısıt Tatmini	Solnon	Ant-P-solver	2000
Sınıflandırma	Shelokar ve diğ.	ACO classifier system	2004a
Kümeleme	Shelokar ve diğ.	ACO	2004b
	Kuo ve diğ.	Ant *k*-means	2005
	Yang ve Kamel	Multi-ant colonies	2006

Aşağıda orijinal KS algoritması ve en başarılı KKO algoritmaları olarak kabul görmüş olan MAX-MIN karınca sistemi algoritması ile karınca koloni sistemi algoritması ayrıntılı olarak tanıtılmıştır. Algoritmalar arasındaki farklılıkların anlaşılabilmesi için her üç algoritma da gezgin satıcı probleminin çözümüne yönelik olarak tanıtılmıştır.

4.2.1 Karınca sistemi algoritması

Gezgin satıcı probleminin çözümü için Dorigo ve diğ. (1991) tarafından önerilen KS algoritmasında, yapay karıncalar, gezgin satıcı probleminin ağ yapısı üzerinde bir düğümden başka bir düğüme hareket etmek için olasılıklı bir geçiş kuralı kullanmaktadırlar. Başlangıç aşamasında her karınca ağ üzerinde rasgele bir düğüm üzerine yerleştirilir. Daha sonra her karınca daha önce ziyaret etmediği düğümlere hareket ederek bütün düğümleri içeren bir çözüm oluşturur. Şekil 4.5'te *n* şehir içeren bir gezgin satıcı problemini *m* adet yapay karınca kullanarak çözen bir KS algoritmasının genel yapısı verilmiştir.

```
başlangıç izlerini oluştur
m adet karıncayı rasgele seçilmiş düğümlere yerleştir
t=0
while durma kriteri sağlanana kadar do;
t=t+1
    for k=1 to m do
        repeat
        geçiş kuralı ile sonraki düğümü seç (Denklem 4.1)
        until  k karıncası tüm düğümleri dolaşana kadar
        if daha iyi bir çözüm bulunduysa then
        en iyi çözümü güncelle
        end if
    end for
    iz güncelle (Denklem 4.2)
end while
bulunan en iyi çözümü göster
```

Şekil 4.5: KS algoritmasının genel yapısı

k karıncası, *i* düğümü üzerinde iken gidebileceği *j* düğümlerine ait geçiş olasılıkları ($P_{ij}^{k}(t)$), aşağıdaki (4.1) denklemi ile hesaplanmaktadır.

$$P_{ij}^{k}(t) = \frac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{u \in J_i^k} [\tau_{iu}(t)]^{\alpha}[\eta_{iu}]^{\beta}} \qquad \textbf{(4.1)}$$

t değeri döngü indeksidir (kesikli zaman değişkeni). $\tau_{ij}(t)$, *t* döngüsünde (*i,j*) bağı üzerinde bulunan feremon izi miktarıdır. η_{ij} ise (*i,j*) bağı üzerindeki sezgisel bilgidir, yapay karıncanın *i* düğümünden *j* düğümünü görebilme ölçüsüdür, genellikle (*i,j*)

yolu uzunluğunun tersi olarak hesaplanır. J_i^k ise i düğümü üzerinde bulunan k karıncasının gidebileceği, henüz ziyaret etmediği düğümler kümesidir. α ve β ise feremon iz miktarı ile sezgisel bilginin olasılık değeri belirlemedeki etkilerini belirleyen parametrelerdir. Geçiş kuralı, bir sonraki düğümün (4.1) denklemi ile hesaplanan olasılıklara uygun olarak seçimini sağlamaktadır.

Yollar üzerinde bulunan $\tau_{ij}(t)$ miktarındaki feremon izleri, tüm karıncalar tarafından çözümler oluşturulduğunda aşağıdaki (4.2) denklemi ile güncellenmektedir.

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t) \qquad \textbf{(4.2)}$$

ρ parametresi, [0,1) aralığında değer alabilen buharlaşma oranıdır. $\Delta\tau_{ij}(t)$ değeri, t döngüsü sonunda (i,j) yoluna eklenecek toplam iz miktarıdır. Oluşturdukları çözümde (i,j) yolunu kullanan tüm karıncaların bırakacakları iz miktarlarının toplamı olarak (4.3) denklemi ile hesaplanmaktadır.

$$\Delta\tau_{ij}(t) = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t) \qquad \textbf{(4.3)}$$

$\Delta\tau_{ij}^{k}(t)$ ise t döngüsü sonunda k karıncası tarafından (i,j) yoluna bırakılacak iz miktarıdır. (4.4) denklemi ile hesaplanır.

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \dfrac{Q}{L_k}, & \text{eğer } \{(i,j) \in \psi_k\}, \\ 0, & \text{değilse.} \end{cases} \qquad \textbf{(4.4)}$$

Q sabit bir değerdir. L_k ise k karıncası tarafından oluşturulan ψ_k turunun uzunluğudur.

(4.2) denklemi ile feremon izlerinin her döngü sonunda güncellenmesi ile iyi kalitede çözümlere ait olan yollardaki iz miktarları artarken düşük kalitede çözümlere ait olan yollardaki iz miktarları azalacaktır. Bu durumda artan döngü sayısı ile birlikte (4.1) denklemi ile oluşturulan geçiş olasılıkları, iyi kalitede çözümlere ait olan yolların seçilme olasılıklarını arttıracaktır.

Orjinal KS algoritması için 3 farklı iz bırakma ve güncelleme stratejisi ile önerilmiştir. Yukarıda anlatılan ve karıncaların sadece tam bir çözüm oluşturduktan sonra bırakılacak iz miktarlarının (4.4) denklemi ile, iz güncellemesinin ise (4.2) denklemi ile yapıldığı algoritma karınca-döngü (ant-cycle) olarak adlandırılmıştır.

Karınca-yoğunluk (ant-density) algortiması olarak adlandırılan ikinci algoritmada ise karıncalar (*i,j*) yolu üzerinden geçiş yaptıktan hemen sonra (*i,j*) yolu üzerine (4.5) denklemi ile hesaplanan $\Delta\tau_{ij}^{k}$ miktarında iz bırakmaktadırlar.

$$\Delta\tau_{ij}^{k}(t)=\begin{cases}Q, & \text{eğer karınca } (i,j) \text{ yolunu kullanmışsa,}\\ 0, & \text{değilse.}\end{cases} \qquad \textbf{(4.5)}$$

Karınca-miktar (ant-quantity) algoritması olarak adlandırılan üçüncü algoritmada ise karınca-yoğunluk modelinde olduğu gibi karıncalar her hareket sonrasında iz bırakmaktadır, bırakılacak olan iz miktarı ise (4.6) denklemi ile hesaplanmaktadır. d_{ij}, (*i,j*) yolunun uzunluğudur.

$$\Delta\tau_{ij}^{k}(t)=\begin{cases}\dfrac{Q}{d_{ij}}, & \text{eğer karınca } (i,j) \text{ yolunu kullanmışsa,}\\ 0, & \text{değilse.}\end{cases} \qquad \textbf{(4.6)}$$

Geçmiş çalışmalarda, gezgin satıcı problemi için karınca-döngü algoritmasının diğerlerinden daha iyi performans gösterdiği görülmüştür. Ancak karınca-miktar ve karınca-yoğunluk algoritmalarındaki stratejiler daha sonra farklı problemler için geliştirilmiş olan KKO algoritmalarında farklı aday seçim stratejileri ile birlikte kullanılmıştır.

4.2.2 MAKS-MİN karınca sistemi

Stützle ve Hoos (2000) tarafından geliştirilen MAKS-MİN karınca sistemi algoritmasının orijinal KS algoritmasından temel farkı izlerin güncelleme sürecindedir. Her döngü sonunda sadece en iyi çözümü bulan karınca tarafından iz güncellenmesi yapılmaktadır. Ayrıca, yollar üzerinde bulunabilecek iz miktarları da sınırlandırılmıştır. İz güncellemesi aşağıdaki (4.7) eşitliği ile yapılmaktadır.

$$\tau_{ij}(t+1)=\left[(1-\rho)\tau_{ij}(t)+\Delta\tau_{ij}^{\min}(t)\right]_{\tau_{min}}^{\tau_{maks}} \qquad \textbf{(4.7)}$$

τ_{maks} ve τ_{min} bir yol üzerinde bulunabilecek iz miktarı için alt ve üst sınırlardır. (4.7) eşitliğinde kullanılan $[x]_b^a$ işlemi (4.8) eşitliği ile tanımlanmıştır.

$$[x]_b^a = \begin{cases} a, & \text{eğer } x > a, \\ b, & \text{eğer } x < b, \\ x, & \text{değilse.} \end{cases} \quad \textbf{(4.8)}$$

$\psi^{\min}$ en iyi çözüm ve $L_{\min}$ değeri de bu çözümün uzunluğu olmak üzere $\Delta\tau_{ij}^{\min}$ değeri (4.9) eşitliği ile tanımlanmıştır.

$$\Delta\tau_{ij}^{\min} = \begin{cases} 1/L_{\min}, & \text{eğer } \{(i,j) \in \psi_{\min}\}, \\ 0, & \text{değilse.} \end{cases} \quad \textbf{(4.9)}$$

En iyi çözüm olarak; mevcut döngü içerisinde bulunan en iyi çözüm, algoritmanın başlangıcından itibaren bulunmuş olan en iyi çözüm veya her iki stratejinin karışımı kullanılabilir.

4.2.3 Karınca kolonisi sistemi

Dorigo ve Gambardella (1997) tarafından gezgin satıcı problemi için önerilen karınca kolonisi sistemi (KKS) algoritmasının, KS algoritmasından iki önemli farkı vardır.

Birinci fark, yeni bir geçiş kuralının tanımlanmasıyla sağlanmıştır. KKS algoritmasında kullanılan geçiş kuralı parametrelerinin ayarlanmasıyla yoğunlaşmayı veya çeşitlendirmeyi destekleyebilecek niteliktedir. KKS algoritmasındaki geçiş kuralı (4.10) denklemi ile geçiş olasılıklarını belirlemektedir.

$$P_{ij}^k(t) = \begin{cases} 1, & \text{eğer } \{q \le q_0 \ ve \ j = j^*\}, \\ 0, & \text{eğer } \{q \le q_0 \ ve \ j \ne j^*\}, \\ \dfrac{[\tau_{ij}(t)]^\alpha [\eta_{ij}]^\beta}{\sum_{u \in J_i^k} [\tau_{iu}(t)]^\alpha [\eta_{iu}]^\beta}, & \text{değilse.} \end{cases} \quad \textbf{(4.10)}$$

$$j^* = \arg \text{maks}_{u \in J_i^k} ([\tau_{iu}(t)]^\alpha [\eta_{iu}]^\beta) \quad \textbf{(4.11)}$$

q değeri [0,1] aralığında, her aday düğüm seçimi öncesinde rasgele belirlenen bir sayıdır. q_0 ise eşik değeri olarak adlandırılan parametredir. Bu durumda, q değeri q_0'a eşit veya daha küçük ise karınca j^* düğümüne harekete etmeye karar verecektir. j^* düğümü, (4.11) ile belirlenen ve aday düğümler arasında en iyi olasılık değerine sahip olan düğümdür. Böylece, büyük q_0 değerleri için aramanın iyi çözümler etrafında yoğunlaşması sağlanmaktadır. Eğer q değeri q_0'dan büyük ise KS algoritmasında olduğu gibi her aday düğüm için hesaplanan geçiş olasılıklarına uygun olarak bir seçim yapılacaktır. Böylece, küçük q_0 değerleri için aramanın çeşitlendirilmesi sağlanmaktadır. Geçmiş çalışmalarda q_0 parametresi için en iyi değer 0.9 olarak bulunmuştur (Donati ve diğ., 2006).

KKS ile KS algoritmaları arasındaki ikinci önemli fark ise feremon izlerinin güncellenmesi ile ilgilidir. KKS algoritmasında yerel ve global olarak adlandırılan iki farklı iz güncelleme süreci kullanılomaktadır.

Yerel iz güncelleme : Karınca, her hareketinden sonra geçiş yaptığı yol üzerindeki iz miktarını (4.12) denklemini kullanarak güncellemektedir.

$$\tau_{ij}(t) = (1-\rho)\tau_{ij}(t) + \rho\tau_0 \quad \textbf{(4.12)}$$

τ_0, başlangıç iz miktarıdır. Algoritmanın başlangıç aşamasında en yakın komşuluk sezgiseli gibi çözüm oluşturucu bir sezgisel yöntemle başlangıç çözümü oluşturulur. Daha sonra tüm bağlar üzerine $\tau_0 = (nL_0)^{-1}$ miktarında iz bırakılır (L_0:başlangıç çözümünün tur uzunluğu). Yerel iz güncelleme ile eğer (*i,j*) yolu bir karınca tarafından kullanılmış ise yol üzerindeki iz miktarı azalmakta ve diğer karıncalar için bu yol daha az çekici hale getirilmektedir. Böylece karıncaların daha önce ziyaret edilmemiş düğümlere gitmesi desteklenmektedir. Yerel iz güncelleme ile algoritmanın durgunluk haline geçmesi önlenmeye çalışılmaktadır. Durgunluk hali, bazı yollarda aşırı iz birikimi nedeniyle tüm karıncaların aynı yolları seçmekte olduğu durumu tanımlamaktadır. Yerel iz güncellemesi ile bir döngü içerisinde tüm karıncaların aynı çözümleri oluşturmasının önüne geçilmesi amaçlanmıştır.

Global iz güncelleme : Her döngü sonunda, algoritmanın başlangıç aşamasından beri oluşturulmuş en iyi tura ait yollardaki iz miktarları (4.13) denklemi ile güncellenir.

$$\tau_{ij}(t+1) = \begin{cases} (1-\rho)\tau_{ij}(t) + \rho(1/L_{\min}), & \text{eğer } \{(i,j) \in \psi_{\min}\}, \\ \tau_{ij}(t), & \text{değilse.} \end{cases} \quad \textbf{(4.13)}$$

$L_{\min}$, algoritmanın başlangıç aşamasından beri oluşturulmuş en kısa toplam yolculuk mesafeli tur olan $\psi^{\min}$ çözümünün toplam mesafesidir. En iyi tura ait olan yollardaki iz miktarının arttırılmasıyla aramanın en iyi çözüm etrafında yoğunlaşması sağlanmaktadır.

KKS algortimasının diğer KKO algoritmalarına göre en önemli avantajı, yerel iz güncelleme ile daha önce kullanılmış olan yollardaki iz miktarı azaltılarak daha önce kullanılmamış olan yolların da kullanılması yönünde yönlendirme yapılıken, global iz güncelleme ile karıncaların o ana kadar bulunmuş en iyi çözüm etrafında arama yapmasının da desteklenebilmesidir.

4.3 ARP için KKO algoritmaları

Geçmiş çalışmalar incelendiğinde ARP için KKO algoritmasının kullanıldığı çok sayıda çalışma olmadığı, önerilen algoritmaların KS veya KKS algoritmalarını temel aldığı görülmüştür. Bu bölümde geçmişte ARP için KKO algoritması yöntemi kullanılarak yapılan çalışmalar incelenmiştir.

Bullnheimer ve diğ. (1999b), gezgin satıcı problemi için önerilmiş olan AS_{rank} algoritmasını modifiye ederek kapasite kısıtlı ARP'ne uygulamışlardır. Önerdikleri algoritmaya AS_{rank}-CVRP ismini vermişlerdir. Bu çalışma daha sonra Reimann ve diğ. (2002) tarafından geliştirilmiş ve AS_{rank}-$CVRP_{sav}$ algoritması önerilmiştir. ZA-ARP için geçmiş çalışmalar incelendiğinde Gambardella (1999a) tarafından önerilen MACS-VRPTW isimli KKO algortimasının çözüm kalitesi ve çözüm süresi yönünden hala en başarılı yaklaşık çözüm yöntemlerinden biri olduğu görülmektedir (Donati, 2008). MACS (multi ant colony system) birden fazla karınca kolonisi içeren bir algoritma kullanıldığını ifade etmektedir.

MACS-VRPTW, paralel olarak çalışan karınca kolonilerinden oluşmaktadır. Temel fikir, ACS-VEI adlı karınca kolonisinin kullanılacak araç sayısını en küçüklemek için çalıştırılırken, ACS-TIME adlı karınca kolonisinin belirli sayıda araç için yolculuk süresinin en küçüklemesi için çalıştırılmasıdır. Algoritmalar paralel çalışmaktadır, ACS-TIME ile v_{min} adet araç için arama yapılırken ACS-VEI, v_{min}-1 adet araç için uygun çözüm aramaktadır. ACS-VEI algoritması ile v_{min}-1 adet araç için uygun çözüm bulunduğu an algoritmalar yeni v_{min} değeri için tekrar çalıştırılmaktadır. ARP ve ZA-ARP için önerilmiş olan söz konusu algoritmaların temel mekanizmaları aşağıda özetlenmiştir.

Ağ yapıları : AS_{rank}-CVRP ve AS_{rank}-$CVRP_{sav}$ algoritmaları, her müşteri için bir düğüm ve bir adet de depo düğümünden oluşan ağ yapısı üzerinde çalışmaktadır. MACS-VRPTW algoritması ise her müşteri için bir düğüm ve birden fazla depo düğümünden oluşan bir ağ yapısı üzerinde çalışır. Gerçek depodan uzaklıkları sıfır olan yapay depolar oluşturulur. Toplam depo sayısı kullanılan araç sayısı kadardır.

Kısıtlar : ARP'nde her müşteri tam olarak bir kez ziyaret edilmeli ve araç kapasiteleri aşılmamalıdır. ZA-ARP için ise ilave olarak müşterilere servis başlangıç zamanı için alt ve üst sınırlar vardır.

İz yapıları : Üç algoritmada da izler bağlar/yollar ile ilişkilendirilmiştir. *i* ve *j* düğümleri arasındaki bağda bulunan iz miktarı *i* müşterisinden sonra direkt olarak *j* müşterisine gidilmesinin çekiciliğini ifade eder. ACS-VEI ve ACS-TIME farklı iz kümelerini kullanır.

Sezgisel bilgi : AS_{rank}-CVRP ve AS_{rank}-$CVRP_{sav}$ algoritmaları sezgisel bilgi olarak tasarruf algoritmasındaki tasarruf değerlerini kullanır. MACS-VRPTW algoritmasında kullanılan sezgisel bilgi ise c_{ij} yolculuk maliyetinin, $[a_i, b_i]$ zaman aralığının ve n_j değerinin bir fonksiyonudur. n_j değeri, *j* düğümünün geçmiş döngülerde kaç defa bir karınca çözümü içerisinde yer almadığıdır.

Çözüm oluşturma : AS_{rank}-CVRP algoritması, karınca sistemindeki olasılıklı seçim kuralını kullanır. *n*/4 uzunluğunda bir aday listesinden uygun bir müşteriyi seçer. Eğer oluşturulan turu uygun bir şekilde devam ettirecek bir müşteri bulunamaz ise depoya dönülerek tur kapatılmış olur. Eğer servis verilmemiş müşteriler mevcut ise yeni bir tur için başlangıç yapılır. AS_{rank}-$CVRP_{sav}$ ise tasarruf algoritmasındaki temel

adımları kullanır. Başlangıçta n adet tek şehirlik turlar oluşturulur. Daha sonra bu turlar birleştirilmeye çalışılır. Tasarruf algoritmasından farkı, tur birleştirme işleminin deterministik olarak değil de karınca sistemindeki olasılıklı seçim kuralı ile gerçekleştirilmesidir. Tur oluşturmanın her aşamasında en büyük tasarruf değerlerine sahip $n/4$ adet aday (i,j) çifti arasından seçim yapılarak ilerlenir. MACS-VRPTW algoritmasında, ACS-TIME ve ACS-VEI aynı çözüm oluşturma sürecini kullanırlar. Bir karınca rasgele seçtiği bir kukla depodan başlamak üzere tur oluşturmaya başlar. Her adımda zaman aralıkları ve kapasite kısıtlarını ihlal etmeden oluşturacağı tura bir müşteri ekleyebilir veya depoya geri döner (yani karınca oluşturduğu tura henüz ziyaret edilmemiş bir müşteri ekleyebileceği halde henüz ziyaret edilmemiş bir depoya dönüş yapabilir). Eğer kısıtları ihlal etmeden oluşturmakta olduğu tura müşteri ekleyemiyorsa zorunlu olarak herhangi bir kukla depoya geri döner. Çözüm oluşturma işlemi bittiğinde (örneğin tüm depolar ziyaret edildiğinde) eğer rotalanmamış müşteriler var ise bu müşteriler yerleştirme sezgiseli kullanılarak oluşturulmuş olan çözüme eklenmeye çalışılır.

İz güncelleme : AS_{rank}-CVRP ve AS_{rank}-$CVRP_{sav}$ algoritmaları karınca sistemindeki iz güncelleme sürecini kullanmaktadır. ACS-TIME ve ACS-VEI ise karınca kolonisi sistemindekine benzer olarak yerel ve global iz güncelleme kurallarını kullanır.

Yerel Arama : AS_{rank}-CVRP karıncalar tarafından oluşturulan rotaları iyileştirmek için 2-opt yerel aramasını gezgin satıcı problemi için kullanır. AS_{rank}-$CVRP_{sav}$ önce iki ayrı tur arasında yer değiştirme hareketi ile yerel arama yapar. Daha sonra oluşan turlar üzerinde 2-opt uygular. MACS-VRPTW ise daha karmaşık bir yerel arama süreci kullanır. Uyguladıkları süreç, Taillard ve diğ. (1997) tarafından önerilmiş olan çapraz değiştirme hareketini temel almaktadır.

AS_{rank}-CVRP ve AS_{rank}-$CVRP_{sav}$ algoritmaları sadece kapasite kısıtlı ARP'ye uygulanmıştır. Genellikle AS_{rank}-$CVRP_{sav}$ performans olarak AS_{rank}-CVRP'den oldukça üstündür. Reimann ve diğ. (2002) ve Reimann ve diğ. (2004) çalışmalarında AS_{rank}-$CVRP_{sav}$ ile pek çok tabu arama algoritmasının karşılaştırılması yapılmış ve AS_{rank}-$CVRP_{sav}$'ın daha önceki tabu arama algoritmalarından daha iyi bir performansa sahip olduğu gösterilmiştir.

MACS-VRPTW ise halen zaman aralıklı ARP için en iyi algoritmalardan biri konumundadır. Pek çok örnek problem için yeni en iyi çözümleri ortaya çıkarmıştır.

Reimann ve diğ. (2004), daha önce Reimann ve diğ. (2002) çalışmasında önerdikleri AS_{rank}-$CVRP_{sav}$ yöntemini temel alarak D-Ants algoritmasını önermişlerdir. Önerdikleri yeni algoritmanın temel avantajı kullandığı "böl ve ele geçir" konsepti olmuştur. Büyük boyutlu problemlerin çözümü için birbirlerine fiziksel olarak yakın olan turların kümelendirilerek asıl problemin alt problemlere ayrıştırılması stratejisi izlenmiştir. Alt problemlerin çözümü için ise AS_{rank}-$CVRP_{sav}$ kullanılmıştır. Örnek problemler üzerinde gerçekleştirilen çalışmalarda AS_{rank}-$CVRP_{sav}$ ile D-Ants arasındaki performans farkının 75 ve üstü boyutlu problemlerde ortaya çıktığı görülmüştür. Küçük boyutlu problemlerde D-Ants kullanımı gereksiz yere süre kaybına neden olmaktadır.

ZA-ARP için KKO metasezgiselinin kullanıldığı en yakın tarihli çalışma ise Donati ve diğ. (2008) tarafından gerçekleştirilmiştir. Müşteriler arasındaki yolculuk sürelerinin trafik yoğunluğuna bağlı olarak gün içinde farklı değerler alması durumunu modellemişlerdir. Hiyerarşik olarak araç sayısını ve yolculuk süresini azaltmayı amaçlamışlardır. Sabit yolculuk süreleri durumunda elde edilen çözümlerin, değişken yolculuk süreleri durumunda uygun bir çözüm olamayabileceğini ve yolculuk sürelerindeki değişkenlik arttıkça söz konusu çözümlerin uygun olmama derecelerinin de arttığını göstermişlerdir. Zaman aralıkları kısıtının bulunmadığı durumlarda ise sabit yolculuk süresi durumunda elde edilen optimal çözümlerin değişken yolculuk süreleri durumunda alt-optimal olabileceğini göstermişlerdir. Önerdikleri modelin çözümü için bazı küçük değişiklikler ile MACS-VRPTW algoritmasını kullanmışlardır.

4.4 ZA-ARP için Karınca Kolonisi Sistemi tasarımı

Bu tez çalışmasında bulanık karar ortamında modellenen ZA-ARP'nin, KKO metasezgiseli ile çözülmesi amaçlanmıştır. Ancak geliştirilecek bir algoritmanın doğruluğunun ve performansının ortaya konabilmesi için karşılaştırma yapılabilecek benzer bir çalışma bulunmamaktadır. Bu nedenle, geliştirilen algoritma öncelikle belirlilik ortamında gerçekleştirilen geçmiş çalışmalar kullanılarak test edilmiş,

tatmin edici sonuçların elde edilmesinden sonra bulanık karar ortamındaki modellerin çözümü için kullanılmıştır.

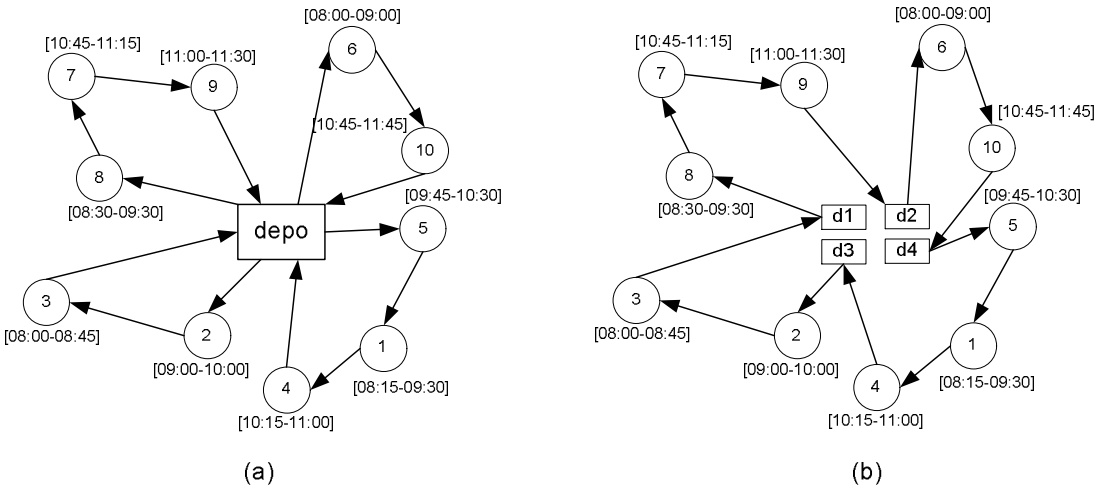
Bu bölümde, Dorigo ve Gambardella (1997) tarafından gezgin satıcı probleminin çözümü için önerilen KKS algoritması, Gambardella ve diğ. (1999a) tarafından geliştirilen MACS-VRPTW algoritması ve Donati ve diğ. (2008) çalışmalarında kullanılan mekanizmalardan yararlanılarak belirlilik ortamındaki ZA-ARP için bir KKO algoritması geliştirilerek doğruluğu ve performansı incelenmiştir. Önerilen algoritmanın bulanık karar ortamında kullanımı ise beşinci bölümde ele alınmıştır.

Belirlilik ortamında ZA-ARP için sezgisel yöntemlerin kullanıldığı geçmiş çalışmalar incelendiğinde, genellikle Solomon (1987) test problemleri üzerinde denemeler yapılarak performanslarının ortaya konduğu görülmektedir. Bu çalışmalarda birinci öncelikli amaç kullanılan araç sayısını azaltmak ikinci öncelikli amaç ise toplam mesafeyi azaltmaktır. Önerilen yöntemin performansının incelenebilmesi için amaç fonksiyonu aynı yaklaşımla kullanılmıştır. Önerilen algoritma, bulanık karar ortamında ZA-ARP için geliştirilmekte olduğundan performans değerlerinin belirlilik ortamında yapılmış geçmiş çalışmalara yakın olması bu aşamada yeterli görülmüştür. Önerilen algoritma, zaman aralıklı araç rotalama probleminin çözümü için karınca kolonisi sistemi tabanlı bir yöntem olduğundan ZAARP-KKS olarak adlandırılmış, temel süreçleri aşağıda tanıtılmıştır.

4.4.1 Ağ yapısının ve başlangıç çözümünün oluşturulması

Algoritmanın başlangıç aşamasında ZA-ARP'nin üzerinde tanımlanmış olduğu $G(V,A)$ ağına kukla depolar eklenerek, kullanılabilecek araç sayısı kadar kukla depo oluşturulur. Kukla depolar gerçek depo ile aynı bağlara sahiptir ve depolar arasındaki uzaklıklar sıfırdır.

Her yapay karınca, ağ üzerinde tam bir tur oluşturmakla görevlidir. Bu yapılandırma ile araç rotalama problemi klasik gezici satıcı problemine benzetilmiş olur. Şekil 4.6 (a)'da 10 müşteri içeren bir ZA-ARP'ne 4 araç kullanımı ile elde edilebilecek bir çözümünün orijinal $G(V,A)$ ağ yapısı üzerinde gösterimi yer almaktadır. Aynı problemin güncellenmiş ağ yapısı ile çözümü Şekil 4.6 (b)'de görüldüğü gibi gezgin satıcı problemi yapısındadır.



Şekil 4.6: ZA-ARP'nin gezgin satıcı problemine benzetilmesi

Ağ yapısının oluşturulmasından sonra, en yakın komşuluk sezgiseli ile oluşturulan başlangıç çözümü kullanılarak tüm yollar üzerine (4.14) denklemi ile τ_0 miktarında başlangıç izi bırakılmaktadır. En yakın komşuluk sezgiseli, rasgele bir kukla depodan başlayarak ve aday müşterilerden devamlı en yakın mesafede olanı seçerek çözüm oluşturur. Kapasite veya zaman aralıkları kısıt nedeniyle seçebileceği aday müşteri kalmadığında kukla depolardan birine döner, kapasite ve zaman değişkenlerini güncelleyerek çözüm oluşturmaya devam eder.

$$\tau_0 = \frac{1}{n.L_0} \qquad \textbf{(4.14)}$$

L_0, en yakın komşuluk sezgiseli ile bulunan çözüm olan ψ^0'ın toplam yolculuk mesafesidir. Bu çözüm henüz oluşturulmuş en iyi çözüm olduğundan en iyi çözüm (ψ^{min}) olarak da adlandırılır.

4.4.2 Karıncalar tarafından çözüm oluşturulması

ZAARP-KKS algoritmasındaki yapay karıncaların çözüm oluşturma süreci tek bir karınca için tanıtılmıştır. Bu süreç, her döngü içersinde tüm karıncalar tarafından uygulanmaktadır.

Başlangıçta, rasgele seçilen kukla depolardan bir tanesinin üzerine karınca yerleştirilir. Karınca her adımda aday müşterilerden birine hareket etmek suretiyle çözüm oluşturmak üzere görevlendirilmiştir. Her hareket öncesinde aday müşteriler

kümesi belirlenir. *i* müşterisi üzerinde bulunan *k* karıncasının hareket edebileceği aday müşteriler kümesini temsil eden J_i^k, henüz ziyaret edilmemiş, gidildiği takdirde talebi tam olarak karşılanabilecek ve zaman aralıkları içerisinde teslimata başlanılması mümkün olabilen şehirler ve karıncanın daha önce ziyaret etmediği kukla depolardan oluşmaktadır. Ancak, karınca mevcut durumda bir depo üzerinde ise bir sonraki müşteri olarak tekrar bir deponun seçimine izin verilmez (depodan depoya harekete izin verilmez).

i müşterisi üzerinde bulunan *k* karıncası, J_i^k kümesi içerisinden müşteri seçimini orijinal KKS algoritmasında kullanılan (4.10) denklemi ile hesaplanan geçiş olasılıklarına uygun olarak yapar. $P_{ij}^k(t)$, *t* döngüsünde *i* müşterisi üzerinde bulunan *k* karıncasının *j* müşterisine gitme olasılığıdır. Karınca tarafından seçim aşamasında kullanılan sezgisel bilgiyi temsil eden η_{ij} değeri (4.15) denklemi ile hesaplanmaktadır. Sezgisel bilgi, sonraki düğümün seçimi esnasında aday düğümlerin çekiciliğini tanımlamaktadır.

$$\eta_{ij} = \frac{1}{(d_{ij}).(b_j - v_j)} \qquad \textbf{(4.15)}$$

v_j, karıncanın *j* müşterisine varış zamanıdır. b_j, *j* müşterisi için en geç teslimat zamanıdır. d_{ij} değeri ile daha yakın mesafede bulunan müşteriye avantaj sağlanmaktadır. (b_j-v_j) değeri ile müşterinin en geç teslimat zamanı ile karıncanın varış zamanı arasındaki fark daha az olan müşteriye avantaj sağlanmaktadır. Yani, oluşturulmakta olan rotaya daha sonra eklenmesi mümkün olmayacak müşterilerin rotalanmasına öncelik verilmektedir.

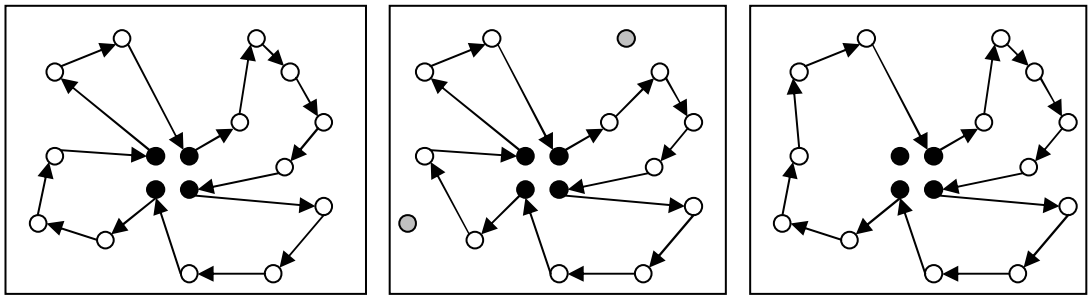
Aday müşteriler kümesinde kukla depolar da yer alacağından karınca bazen kapasite ve zaman aralıkları kısıtları nedeniyle zorunlu olarak bazen de henüz ziyaret edebileceği müşteriler varken olasılıklı seçim kuralı nedeniyle bir sonraki müşteri olarak depoyu seçebilir. Bir sonraki müşteri olarak deponun seçimi ile bir araç rotası oluşturmuş olur. Karınca depodan geçtiğinde kapasite ve zaman ile ilgili değişken değerleri uygun şekilde güncellenir. Bu süreç, aday müşteri kümesinde müşteri kalmayana kadar devam eder. Süreç tamamlandığında 3 farklı tipte çözüm elde edilebilir;

(I) tüm düğümleri içeren tam bir tur oluşturulabilir,

(II) oluşturulan turda hizmet verilemeyen müşteri olabilir,

(III) oluşturulan turda ziyaret edilmeyen depo(lar) olabilir.

Şekil 4.7'de 13 müşteri ve 4 adet kullanılabilir araç için bir karınca tarafından oluşturulabilecek farklı tipte çözümler gösterilmiştir. Depolar siyah noktalarla müşteriler beyaz noktalarla temsil edilmiştir. Aslında tüm kukla depolar aynı koordinat değerlerine sahiptir ancak şeklin kolaylıkla anlaşılabilmesi için birbirleri üzerine yerleştirilmemişlerdir. Şekil 4.7'de görüldüğü gibi (I) ve (III) nolu çözümler uygun, (II) nolu çözüm ise uygun olmayan bir çözümdür. Kukla depo oluşturmanın bir avantajı da müşterileri depoya bağlayan yollar üzerindeki iz miktarlarının tek depo durumuna göre daha az olmasıdır (iz güncelleme kuralları nedeniyle). Böylece karıncaların daha az araç kullanımı ile uygun bir çözüm oluşturma olasılığı arttırılmış olur.



Şekil 4.7: Karınca tarafından oluşturulabilecek çözümler.

(II) tipinde bir çözüm elde edildiğinde, açıkta kalan müşteriler bir yerleştirme sezgiseli ile mevcut tura eklenmeye çalışılır. Yerleştirme sezgiseli, açıkta kalan müşterileri rassal bir sırayla mevcut rotalardan birine yerleştirmeye çalışır. Yerleştirme yapılacak rotalar, yerleştirilecek müşteriye olan toplam uzaklıklarına göre sıralanır. En yakın toplam uzaklığa sahip olan rotadan başlanmak üzere uygun bir yerleştirme aranır. Yerleştirme yapılabilirse, açıkta kalan diğer müşterilerin de yerleştirilmesi denenir. Tüm müşteriler tura eklenebilirse uygun bir çözüm elde edilmiş olur. Aksi durumda açıkta müşteri kalacağından oluşturulan karınca tarafından uygun olmayan bir çözüm elde edilmiş olacaktır.

Uygun bir çözüm elde edildiğinde yerel arama ile çözümü iyileştirilmeye çalışır. Yerel arama süreci sonunda mevcut en iyi çözümden daha iyi bir çözüm elde edilmiş ise ψ^{min} güncellenir.

Daha önce belirtildiği gibi araç sayısının azaltılması öncelikli amaçtır ve daha az araç kullanan bir çözüm bulunduğunda kullanılabilir araç sayısı (kukla depoların sayısı) yeni bulunan çözümdeki araç sayısına eşitlenerek ağ yapısı güncellenir. Bu aşamadan sonra karıncalar güncellenen araç sayısından daha fazla araç kullanımı gerektiren çözümler için arama yapamayacaktır.

4.4.3 İz bırakma ve iz güncelleme

ZAARP-KKS algoritmasında yerel ve global olmak üzere iki tip iz güncellemesi yapılmaktadır. Yerel iz güncelleme ile çözüm oluşturma esnasında seçilen yollar üzerindeki iz miktarları azaltılarak aramanın çözüm uzayının farklı bölgelerine yönlenebilmesi sağlanır. Global iz güncellemede ise aramayı iyi çözümler etrafında yoğunlaştırmak için en iyi çözümde kullanılan yollardaki iz miktarları arttırılır.

Yerel iz güncellemesi her karınca hareketinden sonra gerçekleştirilir. Bir karınca *i* düğümünden *j* düğümüne hareket ettiğinde *i* ile *j* düğümleri arasında bulunan τ_{ij} izi orijinal KKS algoritmasında kullanılan (4.12) denklemi kullanılarak güncellenir. Sonraki karıncalar güncellenmiş iz değerleri ile geçiş olasılıklarını belirleyeceklerdir. Bir kaınca tarafından *i* müşterisinden *j* müşterisine gidildiğinde, (*i, j*) bağında bulunan iz miktarı başlangıç iz miktarına yaklaştırılarak azaltılmaktadır. Böylece, daha sonra *i* müşterisine gelen karıncaların *j* müşterisine gitme olasılıkları azalmış olacağından arama sürecinin çeşitlendirilmesi sağlanmaktadır.

Global iz güncelleme işlemi ise her döngü sonunda (tüm karıncalar tarafından çözümler oluşturulduktan ve yerel arama yapıldıktan sonra) orijinal KKS algoritmasında kullanılan (4.13) denklemi ile gerçekleştirilir. Global iz güncelleme işleminde, sadece o ana kadar bulunmuş olan en iyi çözümde (ψ^{min}) yer alan yollar üzerindeki iz miktarları güncellenerek mevcut en iyi çözümde kullanılan bağlar üzerindeki iz miktarları arttırılmaktadır. Böylece, sonraki döngülerde aramanın mevcut en iyi çözüm etrafında daha fazla yoğunlaşması sağlanmaktadır.

4.4.4 Yerel arama

ZAARP-KKS algoritmasında, karıncalar tarafından elde edilen uygun çözümlerin iyileştirilmesi (araç sayısının veya toplam mesafenin azaltılması için) geçmiş çalışmalarda ARP için önerilmiş yerel arama sezgiselleri uyarlanarak kullanılmıştır.

Uygun bir çözümdeki toplam mesafenin azaltılması için tekrar yerleştirme operatörü kullanılmıştır. Müşteriler sırayla bulundukları konumdan çıkarılarak yerleştirilebilecekleri diğer konumlara (rota içi ve rotalar arası), diğer müşterilerin bağıl sıralamaları aynı kalacak şekilde yerleştirilmeleri durumunda, çözümde iyileşme sağlanıp sağlanmadığı kontrol edilmektedir. İyileşme sağlanan ilk yerleştirmede süreç durdurulmaktadır.

Uyun bir çözümde kullanılan araç sayısının azaltılması için en az etki yerleştirmesi (EAEY) olarak adlandırılan bir rota silme sezgiseli önerilmiştir. EAEY sezgiseli mevcut çözümdeki bir araç rotasını iptal etmekte, açıkta kalan müşterileri ise diğer rotalara yerleştirmeye çalışmaktadır. İptal edilecek araç rotasının seçimi için her araç rotasına bir seçilme olasılığı atanmaktadır. Seçilme olasılıkları rotalardaki müşteri sayılarına bağlı olarak değişmektedir. Örnek olarak 4 araç rotasından oluşan bir çözümde rotalardaki müşteri sayıları sırayla (4, 3, 5, 3) ise 1 nolu araç rotasının iptal edilme olasılığı;

$$\frac{1/4}{1/4+1/3+1/5+1/3}=\frac{0,25}{1,116}=0,22$$

olarak hesaplanır. Benzer şekilde 2 ve 4 nolu araç rotalarının iptal edilme olasılıkları 0,30, 3 nolu araç rotasının iptal edilme olasılığı ise 0,18 olarak hesaplanır. Böylece oluşturulan olasılık dağılımına uygun olarak iptal edilecek araç rotasının seçimi gerçekleştirilir.

İptal edilecek araç rotasının belirlenmesinden sonra, bu rotada bulunan müşteriler rotada bulundukları sırayla, kapasite ve zaman aralıkları kısıtları dikkate alınarak yerleştirme işlemi yapılır. Müşterinin yerleştirilebileceği tüm pozisyonlar için etki değeri ($\nabla_{i,g,p}$) hesaplanır. (4.16) eşitliği ile hesaplanan $\nabla_{i,g,p}$ etki değeri, açıkta bulunan i müşterisinin g araç rotasında p pozisyonuna yerleştirilmesi durumunda

kendinden sonra gelen i^+ müşterisinin mevcut durumu ile yeni durumu arasındaki fark için hesaplanan sayısal bir değerdir.

$$\nabla_{i,g,p} = \begin{cases} 0 & \text{eğer } v_{i^+}^{yeni} \le a_{i^+} \\ \dfrac{v_{i^+}^{yeni} - a_{i^+}}{b_{i^+} - a_{i^+}} & \text{değilse} \end{cases} \quad \textbf{(4.16)}$$

Yeni durumda aracın i^+ müşterisine varış zamanı ($v_{i^+}^{yeni}$), en erken teslimat başlangıç zamanına eşit veya daha küçük ise *i* müşterisinin ilgili pozisyona yerleşmesi sadece *g* aracının i^+ müşterisi önündeki bekleme süresini azaltacaktır. Söz konusu yerleştirmenin i^+ müşterisi teslimat zamanına ve *g* araç rotasında bulunan diğer müşterilerin teslimat zamanlarına herhangi bir etkisi olmayacaktır. Bu nedenle etki değeri 0 olarak hesaplanmaktadır. Yeni durumda $a_{i^+} < v_{i^+}^{yeni}$ ise, i^+ müşterisinden sonra gelen müşteriler için de teslimat zamanları artabilecektir. Bu durumda etki değeri, $v_{i^+}^{yeni}$ zamanının i^+ müşterisinin teslimat zaman aralığına bölünmesiyle hesaplanır. Hesaplanan $\nabla_{i,g,p}$ değeri (0,1] aralığında ise *j* müşterisinden sonraki tüm müşterilerin teslimat zaman aralığı kısıtlarına uygunluğu kontrol edilmelidir. $\nabla_{i,g,p} > 1$ ise i^+ müşterisine teslimat yapılamayacağından yerleştirme uygun olmayacaktır.

Uygun yerleştirmeler için hesaplanan $\nabla_{i,g,p}$ değerlerinin en küçüğü için *i* müşterisinin g^* araç rotasının p^* pozisyonuna yerleştirilmesi yapılır (4.17).

$$(g^*, p^*) = \arg\min_{g,p}(\nabla_{i,g,p}) \quad \textbf{(4.17)}$$

İptal edilen rotadaki tüm müşterilerin yerleştirilmesi yapılana kadar yukarıda tanımlanan süreç devam ettirilir. Herhangi bir müşteri için yerleştirme yapılamadığında EAEY sezgiseli sona erdirilir.

Şekil 4.8'de belirli karar ortamında ZA-ARP için önerilen KKS algoritmasının temel adımları gösterilmiştir. ZAARP-KKS algoritmasının 1-3 adımlarında öncelikle kukla depoların eklenmesiyle rotalama problemi gezgin satıcı problemine benzetilmiştir.

Daha sonra başlangıç çözümü oluşturulmakta ve ağ yapısı içinde bulunan tüm bağlara başlangıç izi bırakılmaktadır.

müşteri sayısı = n
kullanılabilir toplam araç sayısı= r
karınca sayısı = m
durma kriteri : $t = t_{max}$ (t: döngü sayacı)

1: kukla depolar ekleyerek karıncaların çözüm oluşturacağı ağ yapısını kur
2: en yakın komşuluk sezgiseli ile ψ^0 çözümü oluştur
3: tüm yollar üzerine τ_0 miktarlarında iz bırak
4: **while** durma kriteri sağlanana kadar **do**;
5: t=1
6: **for** k=1 to m **do**
7: rasgele bir depoyu başlangıç düğümü olarak seç
8: **repeat**
9: aday düğümler kümesini oluştur
10: **if** mevcut düğüm = depo **then**
11: aday düğümler kümesinden depoları çıkar
12: **end if**
13: geçiş kuralı ile sonraki düğümü belirle (Denklem 4.10)
14: k karıncasını sonraki düğüme hareket ettir
15: mevcut düğüm ← sonraki düğüm
16: k karıncasının durumunu güncelle (kapasite, teslim zamanı, vb.)
17: kullanılan yol için yerel iz güncelleme uygula (Denklem 4.12)
18: **until** aday düğümler kümesi ={ }
19: **if** ziyaret edilmeyen müşteri düğümleri varsa **then**
20: yerleştirme sezgiseli kullan
21: **end if**
22: **if** uygun bir çözüm elde edilemediyse **then**
23: **break** (sonraki karınca ile aramaya başlamak üzere adım 6'ya dönülür)
24: **end if**
25: yerel arama sezgisellerini kullan
26: **if** mevcut en iyi çözümden daha iyi bir çözüm bulunduysa **then**
27: en iyi çözümü güncelle
28: **if** daha az araç kullanan bir en iyi çözüm bulunduysa **then**
29: kullanılmayan araç sayısı kadar depoyu ağ yapısından sil
30: **end if**
31: **end if**
32: **end for**
33: en iyi çözüm ile global iz güncelleme uygula (Denklem 4.13)
34: t=t+1
35: **end while**

Şekil 4.8: ZAARP-KKS algoritmasının temel adımları

Algoritmanın 4-35 adımları bir döngüyü temsil etmektedir. Bir döngü içerisinde kullanılan karınca sayısı kadar çözüm oluşturulabilmektedir. 6-32 adımları bir karınca tarafından çözüm oluşturulması, oluşturulan çözüme açıkta kalan

müşterilerin eklenmesi, yerel arama uygulanması ve yerel iz güncelleme süreçlerini içermektedir. 6-32 adımları kullanılan karınca sayısı kadar tekrarlanmaktadır. 6-32 adımları içerisinde sadece karıncanın davranışları ile ilgili olan kısım 7-18 adımlarıdır. 19-21 adımlarında; eğer karıncanın oluşturduğu çözüm tüm müşterileri içermemesi nedeniyle uygun değilse, eksik müşterilerin çözüme uygun bir şekilde yerleştirilmesi amaçlanmıştır. Eğer eksik müşterilerin mevcut çözüme eklenemiyorsa 22-24 adımları ile işlem yapmakta olan karıncanın devre dışı bırakılarak bir sonraki karıncanın göreve başlaması sağlanmaktadır. 25. adıma gelinmesi uygun bir çözüm elde edildiği anlamı taşımaktadır. Bu çözüm üzerinde yerel arama sezgiselleri kullanılarak çözümün iyileştirilmesine çalışılmaktadır. 26-31 adımları mevcut en iyi çözümün hafızada tutulmasını sağlamaktadır. Daha önce belirtildiği gibi birinci öncelikli amaç kullanılan araç sayısını azlatmaktır. Yani daha az araç kullanan bir çözüm, yolculuk mesafesi daha fazla olsa bile daha iyi bir çözüm olarak kabul edilmiştir. 33. adımda ise global iz güncellemesinin yapılması sağlanmıştır.

4.5 Deneyler

Tasarlanan ZAARP-KKS algoritmasındaki parametre değerleri için Dorigo ve Gambardella (1997) çalışmasında önerilen değerler kullanılmıştır. Geçmiş çalışmalarda her probleme özgü parametre değerlerinin bulunması ile daha iyi sonuçların elde edilebilirliği gösterilmiştir. Ancak bu tip bir çözüm yaklaşımı probleme özgü olmakta ve ek bir hesaplama yükü getirmektedir. Sabit olarak kullanılan parametre değerleri; $\alpha = 1,\ \beta = 2,\ \rho = 0,1,\ q_0 = 0,9,\ m = 10$.

ZA-ARP için geçmişte yapılan çalışmaların çoğunda Solomon (1987) örnekleri kullanıldığından tasarlanan algoritmanın sonuçlarının karşılaştırılabileceği birçok çalışma mevcuttur. Ancak yapılan karşılaştırmaların aşağıda sıralanan tutarsız faktörlerden olumsuz olarak etkilendiği dikkate alınmalıdır.

- Sabit parametre değerlerinin kullanılmaması
- Sonuçları elde edebilmek için gerçekleştirilen koşum sayısı
- Kullanılan bilgisayar, işletim sistemi ve hesaplama süreleri
- Kullanılan amaç fonksiyonu

Önerilen algoritma MATLAB 7.0 programında kodlanmış ve AMD Athlon 64 X2 4000 2,10 GHz işlemci ve 1,75 GB RAM kullanan bir bilgisayarda çalıştırılmıştır.

4.5.1 Karşılaştırmalar

Bu aşamada, tasarlanan ZAARP-KKS algoritmasının ZA-ARP'ni çözme potansiyelinin gösterilebilmesi amaçlanmıştır. Bu amaçla, Solomon (1987) örneklerinin kullanıldığı geçmiş çalışmalarda elde edilen çözümler ile ZAARP-KKS algoritması ile elde edilen çözümler karşılaştırılmıştır. Böylece bulanık verilerle çalışmaya geçmeden önce önerilen yöntemin doğruluğu ortaya konmuş ayrıca iyi bir arama performansı gösterip göstermediği incelenmiştir.

Kesin çözüm yöntemleri ve sezgisel çözüm yöntemleri ile elde edilen çözümlerin karşılaştırılmasında, kullanılan verilerin hassaslık düzeyine dikkat edilmesi gerekmektedir. Çünkü, kesin çözüm yöntemleri genellikle kaba yuvarlamalar yapılan verileri kullanmaktadır. Örneğin Kohl ve diğ. (1999) ve Kallehauge ve diğ. (2006) çalışmalarında ilk ondalık basamağı tabana yuvarlanan veriler kullanılmıştır; 10,66 değeri 10,6 olarak hesaplamalarda yer almıştır. Dondo ve Cerda (2007) çalışmasında ise ilk ondalık basamak tavana yuvarlanarak 10,66 değeri 10,7 olarak hesaplamalarda kullanılmıştır. Bu nedenle çözümlerin tam anlamıyla karşılaştırılabilmesi için sadece amaç fonksiyonu değerinin değil oluşturulan rotaların da kontrol edilmesi gerekmektedir. Örneğin Kallehauge ve diğ. (2006) çalışmasında toplam mesafesi 190,3 olan bir çözüm için Dondo ve Cerda (2007) çalışmasında toplam mesafe 190,7 olarak hesaplanmaktadır. KKS-ZARP algoritmasında sayısal değerler IEEE 754 standartına uygun olarak 10^{-16} hassasiyete sahip çift duyarlıklı (double precision) değişkenlerle kodlanmıştır.

Kodlama aşamasındaki hataların gözden kaçırılmaması için önerilen algoritma ile elde edilen çözümler geçmiş çalışmalarda verilen çözümler ile detaylı olarak karşılaştırılmıştır. Bu esnada, Dondo ve Cerda (2007) çalışmasında araç varış zamanlarının ve teslimat başlangıç zamanlarının hesaplanmasında önemli hatalar bulunduğu tespit edilmiştir. Tablo 4.2'de, Dondo ve Cerda (2007) tarafından, RC101 örnek problemine amaç fonksiyonu değeri (toplam yolculuk mesafesi) 454,19 olarak hesaplanan ve yeni bir en iyi çözüm olduğu belirtilen çözümün detayları yer almaktadır. Çözümde 2 nolu aracın 12 nolu müşteriye varış zamanı 48,35, ayrılış

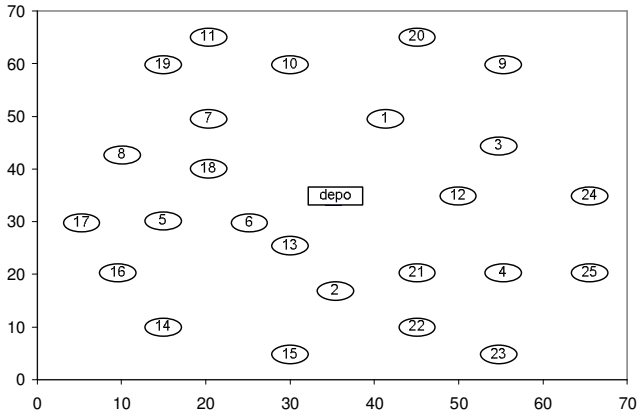
zamanı ise 58,35 olarak hesaplanmıştır. Ancak problem verisi incelendiğinde 12 nolu araç için teslimat zaman aralığının [64, 94] olduğu görülmektedir. Bu durumda, 2 nolu araç 64. dakikaya kadar 12 nolu müşteri önünde bekleme yapmalı, 64. dakikada servise başlamalı ve aracın müşteriden ayrılış zamanı ise 10 dakikalık servis süresi ile 74. dakika olarak hesaplanmalıdır. Bu durumda aracın 11 nolu müşteriye araç varış zamanı 79. dakika, 15 nolu müşteriye varış zamanı ise 95. dakika olarak hesaplanacaktır. 15 nolu müşterinin teslimat zaman aralığı [58, 88] olduğundan kısıtlara uygun olmayan bir çözüm elde edilmiştir.

Tablo 4.2: Dondo ve Cerda (2007) tarafından RC101 problemine bulunan çözüm

Araç	Müşteri	Bekleme süresi	Varış zamanı	Ayrılış Zamanı	Müşteri talebi	Araç yükü	Yolculuk mesafesi	Yolculuk süresi
1	23	19,96	65,00	75,00	30	140	125,72	219,50
	21	0,00	77,00	87,00	10			
	19	0,00	92,38	102,38	40			
	18	0,00	107,77	117,77	20			
	20	0,00	127,97	137,97	10			
	25	3,82	154,00	164,00	20			
	24	0,00	174,44	184,44	10			
2	14	0,00	35,35	43,35	10	190	126,10	216,10
	12	**0,00**	**48,35**	**58,35**	40			
	11	0,00	63,35	73,35	20			
	15	0,00	79,35	89,35	20			
	16	0,00	91,35	101,35	20			
	9	0,00	112,53	122,53	20			
	10	0,00	127,53	137,53	30			
	13	0,00	144,61	154,61	10			
	17	0,00	165,79	175,69	20			
3	22	57,00	92,00	102,00	40	40	70,00	137,00
4	5	0,69	41,00	51,00	20	170	132,39	212,39
	2	0,00	61,20	71,20	30			
	7	0,80	79,00	89,00	20			
	8	0,00	94,00	104,00	10			
	6	0,00	109,83	119,83	20			
	3	0,00	130,60	140,77	10			
	1	0,00	143,77	153,77	40			
	4	0,00	160,14	170,14	20			

Bu tezin hazırlanması sırasında, Dondo ve Cerda (2007) çalışmasında tespit edilen hatalar çalışmanın yazarlarına ve yayımlandığı dergi editörüne bildirilmiştir. İlgili yazarlar tarafından dergiye gönderilecek olan düzeltmenin örneği EK A'da yer almaktadır.

ZAARP-KKS algoritmasının doğruluğunun incelenmesi için çok sayıda örnek problem üzerinde denemeler yapılmıştır.



Şekil 4.9: R110 probleminin depo ve müşteri konumları

R1 örnek grubundaki 25 müşterilik R110 örneğine ZAARP-KKS algoritması ile oluşturulan çözümün detayları aşağıda incelenmiştir. R110 örneğinde her bir müşteri için servis süresi 10 dakika, her bir aracın kapasitesi ise 200 birimdir. Şekil 4.9'da R110 örneğindeki müşterilerin konumları gösterilmiştir.

Tablo 4.3'te R110 örnek problemindeki müşterilerin konumları (x_i, y_i), teslimat zaman aralıkları $([a_i, b_i])$ ve talepleri (q_i) ile birlikte önerilen ZAARP-KKS algoritması ile elde edilen en iyi çözümdeki araç rotaları, araçların müşterilere varış (v_i) ve teslimat başlangıç (h_i) zamanları, her araç rotasının toplam talebi ve rota uzunluğu yer almaktadır.

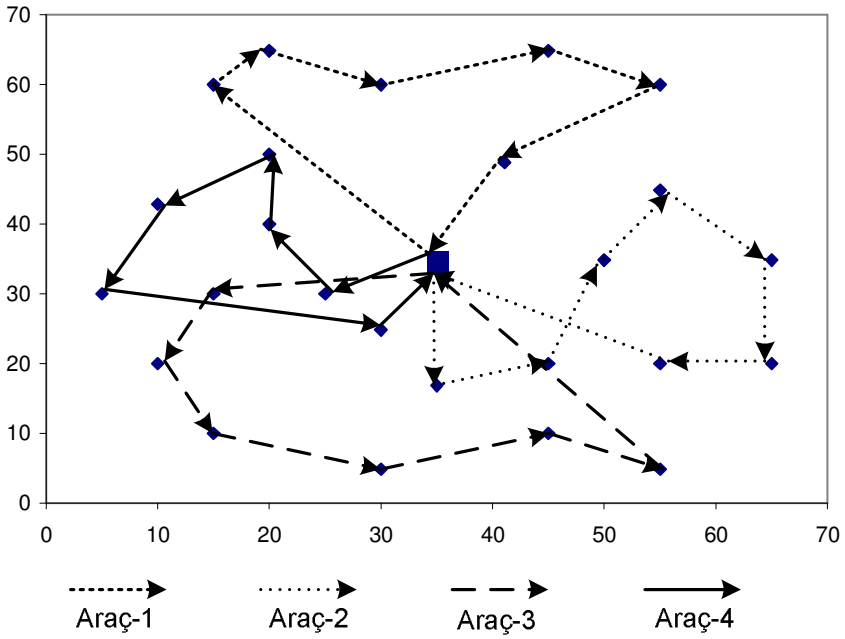
Karmaşıklığa neden olmaması için hesaplanmış değerler Tablo 4.3'e yüzler basamağına yuvarlanarak konulmuştur. Örneğin, toplam mesafenin hesaplanan değeri 446,8748656926319'dir. Probleme, Kohl ve diğ. (1999) tarafından 4 araç kullanımı ile 444,1 toplam mesafeli, Kallehauge ve diğ. (2006) tarafından ise 5 araç kullanımı ile 444,1 toplam mesafeli optimal çözümler elde edilmiştir. Optimal çözümlerin elde edildiği çalışmalarda amaç, toplam mesafenin en küçüklenmesidir. Tablo 4.3'te elde edilen çözüm için yolculuk mesafeleri optimal çözümlerin elde edildiği çalışmalardakine benzer olarak ilk ondalık basamağı tabana yuvarlanan mesafe değerleri için 445.7 olarak hesaplanmıştır.

Tablo 4.3: R110 örneği için bulunan en iyi çözüm

Araç	Rota	(x_i, y_i)	v_i	$[a_i, b_i]$	h_i	q_i	Toplam Talep	Rota Uzunluğu
1	0	(35, 35)	0	[0, 230]	0	-	80	110,29
	19	(15, 60)	32,02	[53, 108]	53	17		
	11	(20, 65)	70,07	[33, 152]	70,07	12		
	10	(30, 60)	91,25	[101, 156]	101	16		
	20	(45, 65)	126,81	[109, 152]	126,81	9		
	9	(55, 60)	147,99	[53, 150]	147,99	16		
	1	(41, 49)	175,8	[130, 201]	175,8	10		
	0	(35, 35)	201,03	[0, 230]	201,03	-		
2	0	(35, 35)	0	[0, 230]		-	78	119,57
	2	(35, 17)	18	[20, 89]	20	7		
	21	(45, 20)	40,44	[37, 96]	40,44	11		
	12	(50, 35)	66,25	[38, 97]	66,25	19		
	3	(55, 45)	87,43	[106, 135]	106	13		
	24	(65, 35)	130,14	[118, 190]	130,14	3		
	25	(65, 20)	155,14	[47, 186]	155,14	6		
	4	(55, 20)	175,14	[71, 195]	175,14	19		
	0	(35, 35)	210,14	[0, 230]	210,14	-		
3	0	(35, 35)	0	[0, 230]		-	120	121,83
	5	(15, 30)	20,62	[20, 107]	20,62	26		
	16	(10, 20)	41,80	[54, 105]	54	19		
	14	(15, 10)	75,18	[32, 137]	75,18	20		
	15	(30, 5)	100,99	[30, 154]	100,99	8		
	22	(45, 10)	126,8	[59, 144]	126,8	18		
	23	(55, 5)	147,98	[36, 155]	147,98	29		
	0	(35, 35)	194,04	[0, 230]	194,04	-		
4	0	(35, 35)	0	[0, 230]		-	54	95,17
	6	(25, 30)	11,18	[54, 153]	54	3		
	18	(20, 40)	75,18	[77, 106]	77	12		
	7	(20, 50)	97	[66, 105]	97	5		
	8	(10, 43)	119,21	[61, 138]	119,21	9		
	17	(5, 30)	143,13	[51, 189]	143,13	2		
	13	(30, 25)	178,63	[70, 208]	178,63	23		
	0	(35, 35)	199,81	[0, 230]	199,81	-		
Toplam:							332	446,87

Şekil 4.10'de ise Tablo 4.3'de detayları verilen çözümün grafiksel gösterimi yer almaktadır.

R110 örnek problemi ve diğer örnek problemler üzerinde gerçekleştirilen çalışmalar neticesinde önerilen algoritmanın ZA-ARP'ne ürettiği çözümlerde hataya rastlanmamıştır.



Şekil 4.10: R110 örneği için elde edilen çözüm

Önerilen ZAARP-KKS algoritmasının performansının değerlendirilebilmesi için örnek problemler üzerinde gerçekleştirilen çalışmalarda elde edilen sonuçlar EK B'de yer alan Tablo B.1-Tablo B.4'te verilmiştir. EK B'de yer alan sonuçlar incelendiğinde, KKS-ZARP algoritmasının küçük boyutlu problemler için oldukça kısa sürelerde çoğunluğu optimal olmak üzere optimale oldukça yakın çözümleri üretebildiği, büyük boyutlu problemler için ise geçmişte önerilen klasik sezgisellere göre daha kaliteli çözümler üretebildiği, metasezgisel yöntemlere yakın bir performans gösterdiği ve çözüm oluşturma sürelerinin uygulama alanı için elverişli olduğu değerlendirildiğinden performansı yeterli görülmüştür.

4.5.2 Arama sürecinin incelenmesi

Bu kısımda KKS-ZARP algoritmasındaki yapay karıncalar arasında gerçekleştirilen dolaylı iletişimin çözüm arama süreci üzerindeki etkisi incelenmiştir. KKS-ZARP algoritmasının başlangıç aşamasında, ağ yapısı üzerindeki tüm yollar üzerine eşit miktarda iz bırakılmaktadır. Dolaylı iletişim, arama sürecinde belirli yollar üzerindeki iz miktarlarının yerel iz güncelleme ve global iz güncelleme mekanizmaları ile azaltılması veya arttırılması ile sağlanmaktadır.

Dolaylı iletişimin etkisinin incelenebilmesi için önerilen KKS-ZARP algoritmasıda değişiklikler yapılarak iki algoritma tasarlanmıştır. Birinci algoritmada iz

güncelleme mekanizmaları kullanılmıştır. İkinci algoritmada ise iz güncelleme mekanizmaları kullanılmayarak tüm yollar üzerindeki iz miktarlarının arama süreci boyunca sabit kalması sağlanmıştır. Dolaylı iletişim etkisinin daha net olarak görülebilmesi için her iki algoritmada da yerdeğiştirme sezgiseli ve rota silme sezgiseli kullanılmamıştır. Kullanılabilecek araç sayısı ise müşteri sayısına eşitlenmiştir. Her iki algoritma da 100 müşterilik Solomon (1987) örnek problemleri üzerinde her örnek grubunun ilk örnek problemi üzerinde denenmiştir. Her problem için 500 döngü içeren üçer koşum gerçekleştirilmiştir. Elde edilen en iyi çözümlerde kullanılan araç sayıları ve çözümlerin toplam yolculuk mesafeleri Tablo 4.4'te yer almaktadır.

Tablo 4.4: Dolaylı iletişimin sonuçlara etkisi

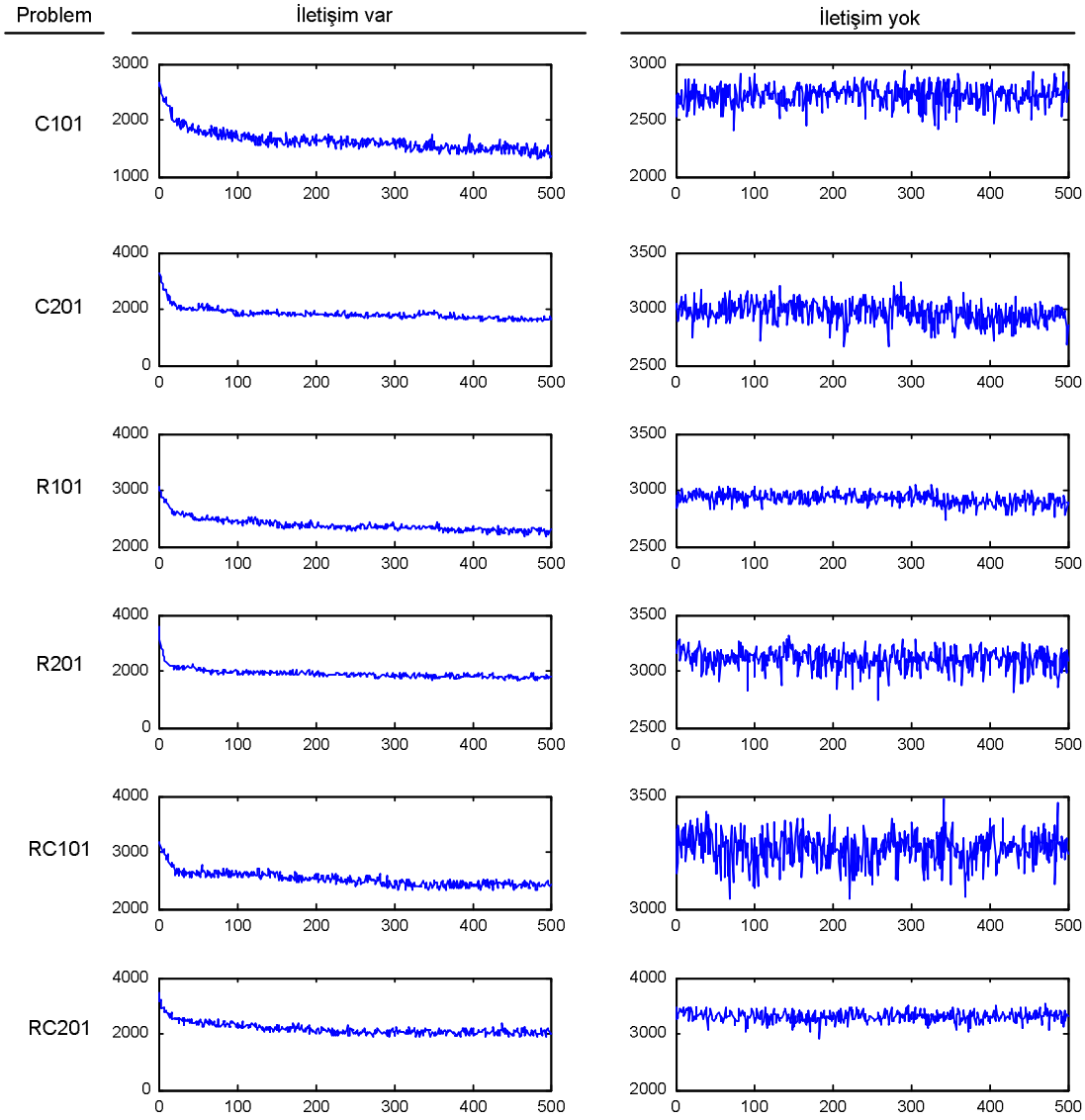
Problem	İletişim var		İletişim yok	
	Araç Sayısı	Toplam Mesafe	Araç Sayısı	Toplam Mesafe
C101	19	1309,7	21	2400,3
C201	17	1540,1	18	2675,3
R101	32	2179,7	32	2886,9
R201	11	1513,9	11	2745,5
RC101	27	2326,2	27	3042,6
RC201	12	1878,2	11	2905,5
Toplam	118	10747,8	120	16656,1

Her iki algoritmada da rota silme sezgiseli kullanılmadığından ve karıncaların kullandığı iz yapısında araç sayısının herhangi bir etkisi olmadığından Tablo 4.4'te görüldüğü gibi dolaylı iletişimin araç sayıları üzerinde anlamlı bir etkisi görülmemektedir. İz güncellemesi aşamasında bırakılacak iz miktarının elde edilen çözümün toplam yolculuk mesafesine göre belirlenmesi nedeniyle dolaylı iletişim ile elde edilen çözümlerin toplam yolculuk mesafelerinde önemli ölçüde iyileşme sağlanmıştır.

KKS-ZARP algoritmasında, oluşturulmakta olan rotaya eklenecek müşteri stokastik bir süreç ile belirlenmektedir. Mevcut rotaya kısıtlara uygun şekilde eklenebilecek her bir müşteri için seçilebilme olasılıkları hesaplanmakta, bir sonraki müşteri oluşturulan bu olasılık dağılımına uygun bir şekilde belirlenmektedir. Söz konusu olasılıklar, iz miktarı ve sezgisel bilgi değerleri kullanılarak belirlenmektedir.

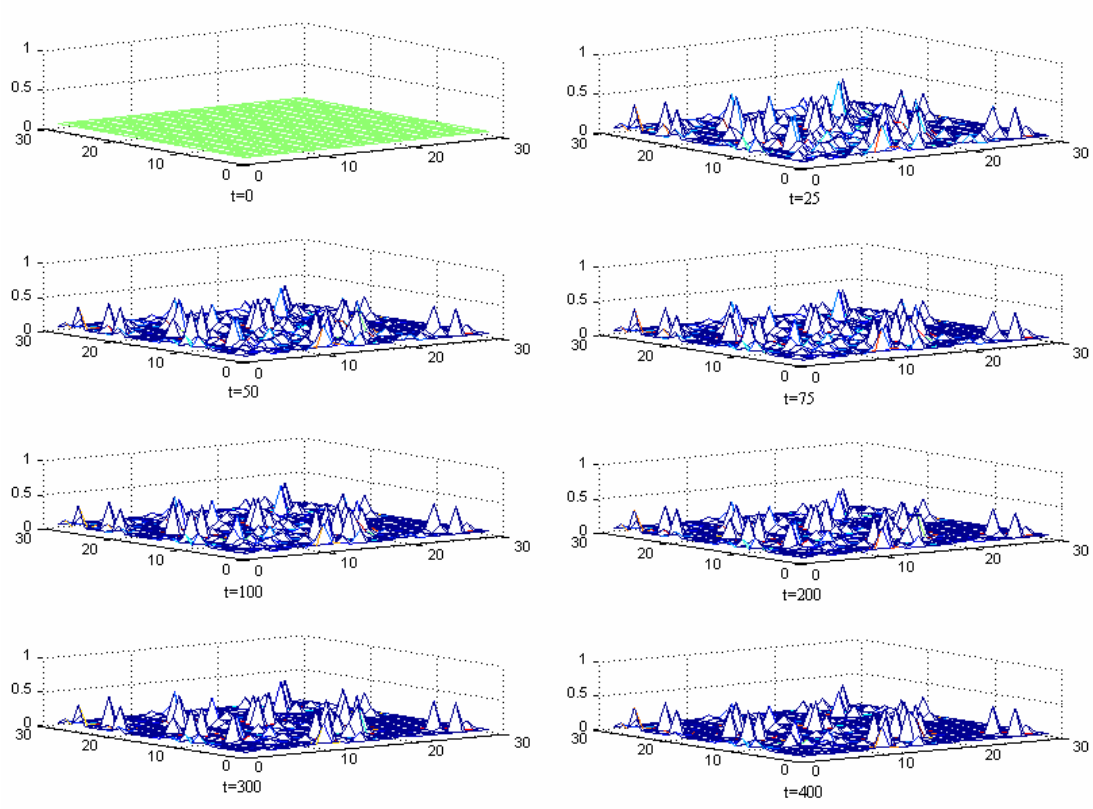
İletişimin kullanılmadığı durumda her bir aday müşteri için belirlenen seçilme olasılıklarında sadece η_{ij} sezgisel bilgisi etkili olmuştur. Bu nedenle arama sürecinin rassal olması ve daha iyi çözümler bölgesine yönlenmemesi beklenir. Dolaylı iletişimin kullanıldığı durumda ise arama sürecini rassal ve daha iyi çözümler bölgesine yönlenmiş olması beklenir.

Tablo 4.4'te elde edilen çözümler için arama süreçlerinin grafiksel gösterimleri Şekil 4.11'de yer almaktadır. Grafikler her döngüde karıncalar tarafından elde edilen en iyi çözümün toplam mesafe değeri için oluşturulmuştur. Görüldüğü gibi arama süreçleri beklentilere uygun özelliktedirler.



Şekil 4.11: Dolaylı iletişimin arama sürecine etkisi

Arama süreci boyunca iz yapısındaki değişimin görülebilmesi için örnek bir problem üzerinde ZA-ARP algoritması ile toplam 400 döngü içeren bir koşum sürecinde, arama sürecinin çeşitli aşamalarında müşteriler arasındaki iz miktarlarının grafiksel gösterimi Şekil 4.12'de yer almaktadır. Grafiklerde farklı döngü sayıları için yatay eksenlerde yer alan müşteriler arasındaki yollarda bulunan iz miktarları ($x10^{-3}$) dikey eksende gösterilmektedir. 25 müşterilik C204 örneği kullanılmıştır.



Şekil 4.12: Arama sürecinde iz yapısındaki değişim

Başlangıç (t=0) anında tüm müşteriler arasında eşit miktarda başlangıç iz miktarı bulunmaktadır. ZA-ARP algoritmasında müşteriler arasındaki iz miktarlarını değiştiren iki mekanizma mevcuttur. Yerel iz güncelleme olarak adlandırılan birinci mekanizma, mevcut çözüme her yeni müşteri eklendiğinde, müşterinin eklendiği rotada bulunan en son müşteri ile yeni eklenen müşteri arasındaki iz miktarını azaltmaktadır. Global iz güncelleme olarak adlandırılna ikinci mekanizma ise her döngü sonunda o ana kadar elde edilmiş en iyi çözümde kullanılan yollar üzerindeki iz miktarlarını arttırmaktadır. Şekil 4.12'de görüldüğü gibi döngü (t) sayısı arttıkça (t=25, t=50, t=75) değişen en iyi çözümler için farklı müşteriler arasındaki iz miktarlarında artışlar gözlenmektedir. 100. döngüden (t=100) itibaren ise en iyi

çözüm değişmemekte ve en iyi çözümdeki ardışık müşteriler arasındaki iz miktarları artarken/korunurken diğer müşteriler arasındaki iz miktarları başlangıç iz miktarına yakınsamaya başlamaktadır. 300. döngüye (t=300) gelindiğinde ise en iyi çözümü oluşturan müşteriler arasındaki iz değerleri dışındakiler başlangıç miktarına oldukça yakınsamıştır. Bu durumda, algoritma, müşteri seçiminde kullanılan rassal süreç nedeniyle farklı çözümler üretmeye devam etmekle birlikte üretilen çözümler çoğunlukla en iyi çözümün yakın komşuluğunda olacaktır. Yani arama süreci durgunluk haline yaklaşmış olacak ve ilerleyen aşamalarda arama süreci en iyi çözümün yakın komşuluğunda yoğunlaşmaya devam edecektir. 400. döngüye (t=400) gelindiğinde ise durgunluk hali nedeniyle iz yapısının 300. döngüden büyük farklılıklar içermediği görülmektedir.

5. BULANIK KARAR ORTAMINDA ARAÇ ROTALAMA

ARP ile ilgili geçmiş çalışmalar incelendiğinde, genellikle problemle ilgili verilerin tam ve kesin olarak bilindiğinin varsayıldığı, belirsizlik durumunda ise istatistiksel yöntemlerin kullanıldığı görülmektedir.

ARP, içerisinde rastgele değişkenler yer aldığında stokastik araç rotalama problemi (SARP) olarak tanımlanmaktadır. Genellikle stokastik talepler ve stokastik yolculuk süreleri ile ilgili çalışmalar yapılmıştır. Bazı çalışmalarda ise ziyaret edilecek müşterilerin de kesin olarak bilinmediği, her müşterinin için bir var olma olasılığının kullanıldığı durumlar ele alınmıştır. Geçmiş çalışmalar incelendiğinde, SARP için ilk çalışmanın Tillman (1969) tarafından Clarke ve Wright algoritması temel alınarak gerçekleştirildiği görülmektedir. Stewart ve Golden (1983) tarafından SARP için şans kısıtlı programlama (chance constrained programming) ve beklenen değer modellerinin (expected value model) ortaya konması önemli bir çalışma olarak değerlendirilmektedir. Sonraki tarihlerde bir çok araştırmacı farklı tipte SARP'leri üzerinde çalışmalar yapmışlardır (Dror ve Trudeau, 1986; Bodin ve diğ., 1983; Warters, 1989). Özellikle Warters (1989) tarafından müşteri talepleri, yolculuk süreleri, ziyaret edilecek müşteriler, müşterilerin konumları, araç kapasiteleri ve araç sayıları gibi değişkenlerin stokastik olabileceğinin belirtilmesinden sonra SARP araştırmacıların daha fazla ilgisini çeken bir problem olmuştur. Laporte ve diğ. (1989) çalışmasında depo yeri de bir karar değişkeni olarak ele alınmıştır. Oluşturulan şans kısıtlı programlama ve beklenen değer modellerinin çözümü için dal sınır algoritmaları kullanılmıştır. Dror ve diğ. (1989) ise SARP için çeşitli operasyon ve servis politikaları içeren modellerin tanıtıldığı bir tarama çalışması ortaya koymuştur. Takip eden yıllarda Bastian ve Rinnooy Kan (1992), Gendreau ve diğ. (1995) ve Teodorovic ve Pavkovic (1992), SARP için farklı modeller ortaya koymuşlardır. Dror (1993) çalışmasında çok aşamalı bir stokastik model ve bir Markov karar modeli önermiştir. Laporte ve diğ. (1992) çalışmasında stokastik yolculuk süreli SARP için şans kısıtlı programlama modeli önerilmiştir. Liu ve Lai (2002), SARP için stokastik programa modelleri ile stokastik benzetim ve genetik

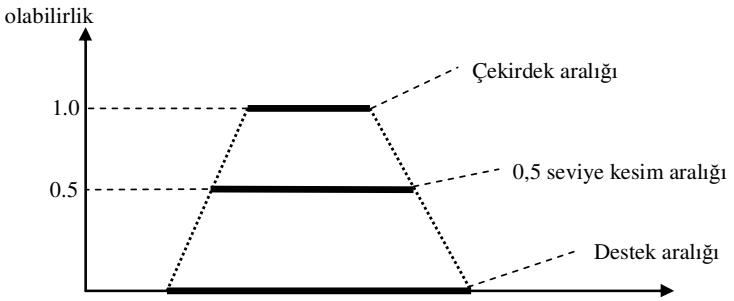
algoritma kullanarak hibrid bir zeki algoritma önermişlerdir. Solomon (1987), Ichoua ve diğ. (2003) ve Fleischman ve diğ. (2004), Donati ve diğ. (2008) zamana göre değişen yolculuk süreli zaman aralıklı SARP ile ilgilenmişlerdir.

Stokastik durumda pek çok akademik çalışma yapılmış olmasına rağmen önerilen modellerin gerçek hayatta kullanımı durumda genellikle yeterli miktarda veri olmaması veya problemin çok karmaşık olması nedeniyle sübjektif yargılara ihtiyaç duyulmaktadır (Peng ve diğ., 2006). Özellikle fazla geçmişi olmayan durumlarda analiz edilecek yeterli miktarda veri olmadığından rastgele değişkenler için parametre değerlerini belirlemek mümkün olmamaktadır. Örneğin, müşteriler arasındaki mesafeler kesin olarak bilinse de kullanılan araçların bu mesafeleri kat etme süreleri değişik faktörler (trafik, hava durumu, araç durumu vb.) nedeniyle belirsizlik içermektedir. Bu durumlarda deneyimlere dayanarak iki müşteri arasındaki yolculuk süresi "yaklaşık 2 saat" veya "35-40 dakika civarında" gibi ifadelerle tanımlanabilmektedir. Bu gibi belirsizliklerle çalışabilmek için bulanık değişkenlerin kullanılabilirliği geçmiş çalışmalarda ortay konmuştur. Teodorovic ve Kikuchi (1991), tüm müşteriler arasındaki yolculuk sürelerinin bulanık sayılarla tanımlandığı bir model önermişler ve araç rotalarını oluşturmak için Clarke ve Wright algoritması prensiplerini kullanmışlardır. Kagaya ve diğ. (1994), araç rotalama ve çizelgeleme problemi için güzergahları benzerliklerine göre gruplandıran bir teknik önermişlerdir. Güzergahlar arasındaki benzerlik bulanık ilişkiler ile tanımlanmıştır. Daha sonra, Teodorovic ve Pavkovic (1996), müşteri taleplerinin bulanık olduğu bir model önermişler ve tarama sezgiseli temelli bir algoritma geliştirmişlerdir. Araç rotalarını oluştururken, belirli bir rotaya müşteri seçimi aşamasında bulanık çıkarım mekanizması kullanmışlardır. En iyi parametre değerlerini belirleyebilmek için bulanık benzetim gerçekleştirmişlerdir. Lai ve diğ. (2003), ARP'yi bulanık programlama ile olabilirlik ölçüsü kullanarak modellemiştir. Zheng ve Liu (2006) ve Peng ve diğ. (2006) çalışmalarında ise ZA-ARP için yolculuk süreleri bulanık değişken olarak ele alınmıştır. Problem için şans kısıtlı programlama modeli oluşturmuşlardır. Bulanık benzetim ve genetik algoritma kullanarak hibrid bir zeki algoritma önermişlerdir.

5.1 Bulanık Yolculuk Süreleri

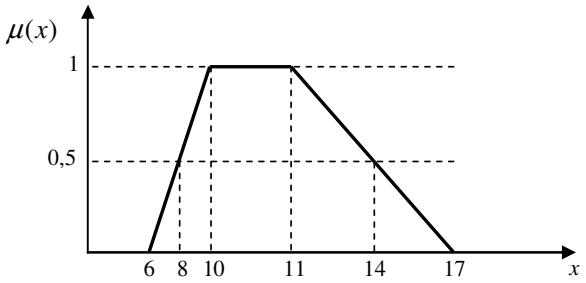
İki müşteri arasındaki yolculuk süresi bazı geçmiş çalışmalarda sadece müşterileri birbirine bağlayan yolun mesafesine bağlı olarak hesaplanabilen kesin bir değer olarak kullanılmıştır. Bu tip modellerde amaç fonksiyonu olarak toplam yolculuk süresinin veya toplam yolculuk mesafesinin kullanımı arasında bir fark olmamaktadır. Gerçek hayatta karşılaşılan ARP'lerinde ise çok özel durumlar haricinde bir aracın iki müşteri arasındaki yolculuk süresi pek çok faktörden etkilenebilen ve planlama aşamasında gerçekleşecek değeri henüz bilinmeyen bir değerdir. Yolculuk süresindeki belirsizlik için, eğer yeteri kadar geçmiş veri varsa olasılık dağılımı oluşturulabilir ama ele alınan problem için yeteri kadar geçmiş veri bulunmaması veya olasılık dağılımlarının belirlenebildiği durumlarda stokastik olarak kurulacak modelin çok karmaşık ve çözüm süresinin uygulanabilir sınırlar dışında olması sıklıkla karşılaşılan bir durumdur. Ancak, gönderici firma tarafından yolculuk süreleri için belirli aralıklar ve bu aralıktaki değerler için olabilirlik dereceleri ortaya konabiliyorsa, yolculuk süreleri için olabilirlik dağılımları elde edilebilir.

Belirsiz yolculuk süresi için olabilirlik dağılımı belirlemek olasılık dağılımlarını belirlemekten oldukça kolay ve basittir. Gönderici firma çalışanı tarafından yolculuk süresinin belirli aralıklar içinde gerçekleşme durumları için güvenilirlik düzeyleri ifade edilebiliyorsa olabilirlik dağılımları elde edilebilir. Rommelfanger (1990), üç güvenilirlik düzeyi için aralıkların belirlenmesiyle olabilirlik dağılımının oluşturulmasını önermiştir. Birinci düzey çekirdek olarak adlandırılan, gerçekleşebilirliğine inancın tam olduğu aralıktır. İkinci düzey, 0,5 seviye kesimi olarak adlandırılan ve gerçekleşmesi sürpriz olmayan değerleri içeren aralıktır. Üçüncü düzey ise destek olarak adlandırılan ve bu aralığın dışındaki değerlerin gerçekleşmesinin fiziki olarak mümkün olmadığı aralıktır. Bu durumda 6 parametre ile doğrusal enterpolasyonla yamuk şekilli bir olabilirlik dağılımı elde edilebilmektedir (Şekil 5.1). Daha basit bir yaklaşım ise bir destek aralığı ve kesin bir sayı ile çekirdek aralığı belirlenmesi durumunda 3 parametre değeri ile oluşturulabilecek üçgen şekilli olabilirlik dağılımıdır.



Şekil 5.1: Güvenilirlik düzeyleri ile olabilirlik dağılımı oluşturulması

Yolculuk sürelerinin taşıyıcı firmadan elde edilen bilgilerin kullanımıyla yamuk şekilli bulanık aralıklar ile modellenebildiği varsayılmıştır. Örnek olarak "*yolculuk süresi yaklaşık 10-11 saattir*", "*normal koşullarda yolculuk süresinin 8-14 saat arasında olması beklenmektedir*" ve "*yolculuk süresi 6 saatten az 17 saatten fazla olamaz*" bilgileri için yolculuk süresi Şekil 5.2'deki bulanık aralık ile modellenerek bulanık yolculuk süresi ($\tilde{t}_{ij}$) olarak tanımlanmıştır.



Şekil 5.2: Bulanık yolculuk süresi ($\tilde{t}_{ij}$)

Yolculuk sürelerindeki belirsizlik, araç varış zamanlarının ve servis başlangıç zamanlarının da belirsiz olmasına neden olacaktır.

5.2 Esnek Kısıtlar

Son yıllarda işletmelerin, tam zamanında üretim, sıfır stok, düşük çevrim zamanı vb. prensiplere önem vererek rekabet güçlerini arttırma çalışmaları, dağıtım firmalarını da etkilemiştir. İstenilen zamanda teslimat yapabilme, dağıtım firması seçiminde önemli bir kriter haline gelmiştir. Zaman aralıkları kısıtı, rotalama problemi için önemli bir kısıttır. Zaman aralıklı bir rotalama problemi için bulunan optimum sonuç,

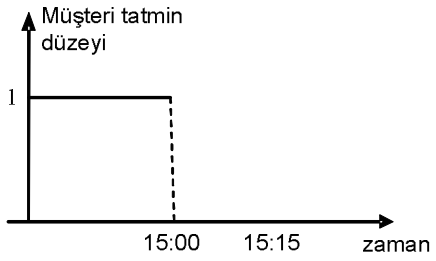
problemde zaman aralıkları kısıtı kaldırıldığında bir üst sınır olarak değerlendirilmektedir. Zaman aralıkları kısıtı, dağıtıcı firmaların veya dağıtım işini kendi araçları ile yapan üreticilerin filolarındaki araç sayılarını arttırmalarına neden olmaktadır.

Günümüzde çizelgeleme ve rotalama problemleri için geliştirilmiş birçok program bulunmasına rağmen genellikle bu programlar ile uygun çözümlerin bulunamadığı ve elle düzeltmeler yapılarak uygulanabilir çözümlerin elde edilebildiği görülmektedir (Fortemps, 2000). Teslim tarihlerinin kesin değerler aldığı bir problemde uygun bir çözüm bulunamıyor ise müşterilerin toleransları göz önünde bulundurularak yapılan düzeltmelerle esnetilen teslim tarihleri için uygulanabilir çözümler bulunabilmektedir. Bu durum aslında modellerin geçerliliği konusundaki bir eksikliği ortaya koymaktadır. Son yıllarda bilgi işlem ve iletişim teknolojilerindeki gelişmelerin yanısıra büyük boyutlu kombinatoryal problemler için iyi sonuçların elde edilebildiği yeni metasezgisel yöntemlerin de ortaya konması, yöneylem araştırması tekniklerinin bu problemler için uygulanabilirliği konusunda klasikleşen çözüm süresinin uzunluğu ve hesaplama yükünün fazlalığı konularındaki dezavantajları geri plana iterek kullanılan modellerin geçerliliği konusundaki eksikliklerin ön plana çıkmasını sağlamıştır.

ARP'de zaman aralıkları kısıtı, bir müşteriye servis yapılabilmesi için sipariş verme aşamasında müşteri tarafından belirtilmiş olan en erken ve en geç zamanları belirten kısıttır. Araç en geç teslimat başlangıç zamanından önce müşteriye ulaşmalıdır. Aracın en erken teslimat başlangıç zamanından önce müşteriye ulaşması durumunda ise araç beklemek durumundadır. Ancak bu tip bir model gerçek hayattaki durumu tam olarak yansıtmamaktadır. Örneğin en geç teslim zamanının 1 saniye bile aşılması bu tip bir modelde uygun olmayan çözüm olarak tanımlanmaktadır. Ancak, gerçek hayatta müşteriler, hizmet tatmin düzeylerinde azalma olsa da makul ölçülerdeki gecikmeleri kabul edebilmekte ve söz konusu araç teslimat yapabilmektedir. Bu tip gecikmelerin, kısmi müşteri tatmini ile veya ceza maliyeti ile ARP modeline dahil edildiği durumlar geçmiş çalışmalarda ele alınmıştır. Koskosidis ve diğ. (1992), Balakrishnan (1993), Fortemps (1997), Taillard ve diğ. (1997) ve Ioannou ve diğ. (2003) bu çalışmaların başlıcalarıdır.

Teslimat zaman aralığındaki esneklik, lojistik ve pazarlama alanında çalışan karar vericilere daha az araç kullanarak rotalama yapabilme konusunda avantaj sağlayabilmektedir. Müşterilerle teslimat zamanları konusunda uygun anlaşmalar yapılarak, en geç teslimat zamanları esnek bir kısıt olarak kullanılabilir ve dağıtım firması maliyetleri azaltılabilir.

Kesin en geç teslimat zamanı içeren modellerde müşteri tatmin düzeyi 0-1 değişkeni ile ölçülmektedir. Araç en geç teslim zamanından önce müşteriye ulaşmış ise tatmin düzeyi 1, en geç teslim zamanı aşılmış ise teslimat yapılamamakta ve müşteri tatmin düzeyi 0 olmaktadır. Gündelik yaşama dair bir örnek verecek olunursa, bir kişinin saat 16:00'da başlayan bir toplantıya katılmak üzere bir taksi durağını aradığı ve saat 15:00 da binmek üzere bir araç istediği varsayılsın. Klasik yaklaşımla kurulan bir modelde, saat 15:00 ve daha öncesinde gelen bir taksi, kişi tarafından kullanılacak ve kişinin tatmin düzeyi tam (1) olacaktır. Ancak saat 15:01'de eğer istenilen taksi henüz gelmemişse taksi kullanılmayacak ve kişinin tatmin düzeyi sıfır olarak hesaplanacaktır. Bu durumda müşteri tatmin düzeyi fonksiyonu Şekil 5.3 ile gösterilebilir.

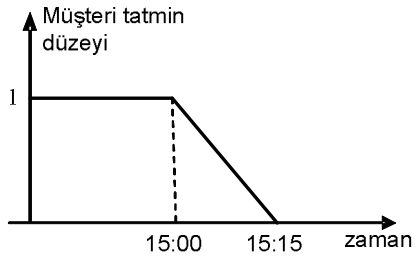


Şekil 5.3: Kesin durum için müşteri tatmin düzeyi

Ancak gerçek hayatta durum daha farklıdır. Saat 15:01'de eğer istenilen taksi henüz gelmemişse kişi gideceği yere ulaşmak üzere kullanabileceği alternatif ulaşım araçlarını da değerlendirerek bir süre daha taksiyi bekleyecektir. Örneğin saat 15:15'e kadar taksiyi beklemeye devam edebilir, eğer 15:15'te söz konusu duraktan taksi gelmemiş ise yoldan geçen herhangi bir taksiye binmeyi veya farklı bir ulaşım aracını kullanmayı tercih edebilir. Bu durumda saat 15:00'a kadar gelen bir taksi kişi tarafından tam bir tatmin düzeyi ile kullanılacak, 15:00 ile 15:15 arasında gelen bir taksi kişi tarafından giderek azalan bir tatmin düzeyi ile kullanılmaya devam edecek, 15:15 ve daha sonra gelen taksi ise kişi tarafından kullanılmayacaktır.

Klasik yaklaşımlarla modellenen geçmiş çalışmalarda esneklik içerebilen kısıtlar iki şekilde ele alınmaktadır. Birincisi Şekil 5.3'de gösterildiği gibi kısıtın esnekliği göz ardı edilmektedir. Bu tip modellerde genellikle zaman aralıklarının darlığı nedeniyle uygun çözümlerin bulunamaması durumu ortaya çıkmaktadır. İkinci bir yaklaşım ise zaman aralıkları kısıtını, müşterinin teslimatı kabul edebileceği son ana kadar genişletmektir. Bu durumda müşteriye genişletilmiş teslimat zaman aralığı içerisinde teslimat yapılmakta ancak müşterinin tatmin düzeyi konusunda bir tercih yapılamamaktadır. Bu tip modellerde, genişletilmiş kısıtlar nedeniyle genellikle çok sayıda çözüm elde edilebilmektedir ancak müşteri tatmin düzeyleri modellenmediğinden karar verici hangi çözümü seçmesi gerektiği konusunda kararsız kalmaktadır.

Yukarıdaki örnekte, saat 15:00 ile 15:15 arasında gelen bir taksi için kişinin tatmin düzeyini Şekil 5.4'deki gibi bir fonksiyonla tanımlanamak, klasik yaklaşıma göre daha gerçekçidir. Şekil 5.4'deki gibi [0,1] değerleri arasında konveks bir fonksiyonla tanımlanan tatmin düzeyi aynı zamanda bulanık bir kümenin üyelik fonksiyonu olarak da görülebilir.



Şekil 5.4: Esnek en geç teslimat zamanı için müşteri tatmin düzeyi

Özetle, en geç teslimat zamanının esnek bir kısıt olarak tanımlanmasıyla müşteri tatmin düzeyini gerçeğe daha yakın olarak ölçebilmek, modelin geçerliliğini ve kullanılabilirliğini arttırmak mümkün olacaktır. Klasik yaklaşımda saat 15:01'de gelen bir taksi kullanılmaz olarak kabul edilmektedir ancak esnek kısıtların kullanıldığı modelde, gerçek hayattaki duruma benzer olarak daha düşük bir müşteri tatmin düzeyi ile taksinin kullanılabileceği modellenebilmektedir. Klasik yaklaşım ile modellenmiş karar destek sistemlerini kullanan karar vericilerin de probleme uygun çözüm bulamadıkları zaman kullandıkları yaklaşım, eğer biliyorlarsa müşterinin toleransları için kısıtları genişleterek tekrar çözüm araması yapmaktır.

5.3 Bulanık karar ortamında ZA-ARP modelleri

Bu bölümde, araç rotalama problemi için gerçek hayatta karşılaşılabilecek belirsizlikler için üç adet bulanık programlama modeli önerilmiştir. İlk modelde sadece yolculuk sürelerindeki belirsizlik ele alınmış, ikinci modelde yolculuk sürelerinin kesin olduğu durumda en geç teslimat zamanındaki esneklik üzerinde çalışılmıştır. Üçüncü modelde ise ilk iki modeldeki belirsizlik ve esnekliğin aynı model içerisinde yer alma durumu ele alınmıştır.

Bulanık kümeler, hem esnek kısıtları hem de belirsiz verileri modelleyebildiğinden, bulanık modellerin oluşturmasına gerekçe olan bu kavramların karıştırılması bir risk oluşturmaktadır. Karar teorisi açısından, bulanık kümeler, fayda fonksiyonları yerine veya olasılık fonksiyonları yerine kullanılabilmektedir. Bulanık kümeler ile üyelik fonksiyonlarındaki muğlaklık modellenebildiğinden önceliklerin veya belirsizliklerin yönetilmesinde kullanılabilmektedir. Çizelgeleme problemleri için belirsizliklerin ve esnekliklerin aynı model içerisinde bulanık kümeler ile ele alındığı çalışmalar yapılmıştır. Bu durumda, stokastik programlamadaki ortalama değerlerin optimize edilmesi yerine, tüm kısıtların yeterli görülen bir güvenilirlik düzeyi için kısmi olarak tatmin edildiği, bulanık tekniklerin de kullanımıyla gürbüz ve hata toleranslı çizelgelerin oluşturulması amaçlanmaktadır. Çizelgeleme konusunda, belirsizliklerin olabilirlik analizi ile ilgili çok sayıda çalışma yapılmış olmasına rağmen, belirsizlik durumunda kritik yol analizini tam olarak ortaya koyabilen bir çalışma henüz yapılamamıştır. Dubois ve diğ. (2003b), bulanık sürelerin kullanıldığı çizelgeleme problemleri için kritik yol analizi konusundaki zorlukları incelemiştir. Örneğin $\tilde{a}$ ve $\tilde{b}$ bulanık sayıları için $\tilde{a} \neq (\tilde{a} + \tilde{b}) - \tilde{b}$ olduğundan kritik yol analizinde geriye doğru hesaplamalar önemli bir sorun olarak görülmektedir. Kritik yol analizi ile araçların müşteri önlerindeki bekleme süreleri azaltılabilir. ZA-ARP konusunda belirlilik ortamında yapılan geçmiş çalışmalarda kullanılan araç sayısının ve toplam yolculuk mesafesinin en küçüklenmesi amaçlanmış, kritik yol analizi ile bekleme sürelerinin azaltılması konusu göz ardı edilmiştir. Belirsizliklerin olabilirlik teorisi temel alınarak modellendiği bu çalışmada, elde edilen sonuçların geçmiş çalışmalarla karşılaştırılabilmesi ve olabilirlik programlama modelleri için kritik yol analizi konusunda henüz aşılamamış önemli zorluklar bulunması nedeniyle araçların müşterilere varış yaptıkları anda teslimata başlayacakları, en erken teslimat

zamanından önce varış yapılması durumunda ise teslimat başlangıç zamanının en erken teslimat zamanına eşit olacağı varsayılmıştır.

Önerilen tüm modellerde amaç fonksiyonu içerisinde yer alan c_{ij} maliyet değeri, $\forall i \in V - \{0\}$ için $c_{ij} = d_{ij}$ ve $i = 0$ için H yeterince büyük bir sayı olmak üzere $c_{0j} = d_{0j} + H$ olarak kullanılmıştır. Böylece, hiyerarşik olarak, önce kullanılan araç sayısını sonra toplam mesafeyi azaltan bir amaç fonksiyonu elde edilmiştir.

5.3.1 Olabilirlik programlama modeli

Olabilirlik programlama modeli, ikinci bölümde belirlilik ortamında tanımlanmış olan ZA-ARP'nde yolculuk sürelerinin kesin olarak bilinmediği ve bulanık aralıklarla tanımlanabildiği durumu ele almaktadır. Yolculuk sürelerinin bulanık aralık olması varış zamanlarının da bulanık aralık olarak hesaplanmasına neden olacaktır.

i müşterisinde bulunan bir g aracının j müşterisine bulanık varış zamanı ($\tilde{v}_j^g$), (5.1) denklemi ile elde edilebilir. ($s_0 = 0, \tilde{v}_0^g = 0, a_0 = 0$)

$$\tilde{v}_j^g = \text{maks}\,(\tilde{v}_i^g, a_i) + s_i + \tilde{t}_{ij} \qquad \textbf{(5.1)}$$

(5.1) denklemindeki $\text{maks}\,(\tilde{v}_i^g, a_i)$ değeri, g aracının i müşterisine teslimat başlangıç zamanını ($\tilde{h}_i^g$) oluşturmaktadır. $\tilde{h}_i^g$ değerine yük boşaltım süresi (s_i) ve yolculuk süresi ($\tilde{t}_{ij}$) eklenerek aracın bir sonraki j müşterisine varış zamanı hesaplanmaktadır.

Araç müşteriye en erken teslimat anından önce varış yaptığında, en erken teslimat anına kadar bekleyecektir. Araç en geç teslimat zamanından sonra müşteriye varış yaptığında ise teslimat yapılamayacağından, oluşturulmakta olan çözüm geçerliliğini kaybedecektir. Belirli karar ortamında, müşterilere teslimat zaman aralığı içerisinde teslimata başlanması (2.9) kısıtı ($a_i \sum_{j \in V} x_{ij}^g \le h_i^g \le b_i \sum_{j \in V} x_{ij}^g \quad \forall i \in V, \forall g \in R$) ile modellenmişti. (2.9) kısıtı aslında iki ayrı kısıtın birleşiminden oluşmaktadır. İlk kısıt ($a_i \sum_{j \in V} x_{ij}^g \le h_i^g$), müşteriye teslimat başlangıç zamanının, müşterinin en erken teslim zamanına eşit veya daha büyük olmasını sağlamaktadır. Araçların müşteriye erken

varış yaptığındaki bekleme sürelerinin uzunluğu ile ilgili bir kısıtlama olmadığından aslında her çözüm için bu kısıt sağlanabilmektedir. İkinci kısıt ($h_i \le b_i \sum_{j \in V} x_{ij}^g$) ise müşteriye teslimat başlangıç zamanının, müşterinin en geç teslimat zamanına eşit veya daha küçük olmasını sağlamaktadır. Belirli karar ortamında bu kısıtın sağlanıp sağlanmadığının kontrolü iki adet kesin sayı arasındaki büyüklük ilişkisinin belirlenmesiyle gerçekleştirilmektedir. Ancak, olabilirlik programlama modelinde bulanık bir $\tilde{h}_i$ aralığı ile kesin bir b_i sayısı arasındaki öncelik ilişkisinin belirlenmesi gerekecektir. Bu aşamada, bulanık değerler arasındaki öncelik ilişkisini durulaştırma ile ortaya koyan yöntemlerin kullanılması önerilebilir. Ancak destek kısımları kesişen bulanık aralıklar arasında belirlenen bir önceliğin her zaman gerçekleşmesi beklenemez. Bazı durumlarda belirlenen önceliğin tersi de gerçekleşebilir. Örneğin (3, 4, 6, 7) ile tanımlanan "5 civarı" bulanık aralığı ve (4, 5, 7, 8) ile tanımlanan "6 civarı" bulanık aralıkları için (3, 4, 6, 7)≤(4, 5, 7, 8) şeklinde oluşan bir kısıt, bulanık değerlerin ağırlık merkezi yöntemiyle durulaştırılmasıyla 5≤6 olarak değerlendirilip kısıtın sağlandığı söylenebilir. Ancak bu söylem uygulama aşamasında her zaman geçerli olmayacaktır. Bazı geçmiş çalışmalarda kullanılan bu tip yaklaşımların ciddi boyutlarda ölçüm hatalarına neden olduğu görülmüştür (Rommelfanger ve Slowinski, 1998). Bu nedenle, bulanık programlama problemlerinde, bulanık kısıtların şans kısıtı olarak modele eklenmesinin daha uygun bir yaklaşım olduğu değerlendirilmektedir.

i müşterisine bulanık varış zamanının, en geç teslimat anından küçük olması olayının kesinliğinin ölçümü için gereklilik ölçütü kullanılmıştır. Problemdeki tüm müşterilere en geç teslimat anından önce teslimata başlanabilmesi için en düşük gereklilik düzeyinin $N_{eşik}$ olması istendiğinde aşağıdaki (5.2) kısıtının kullanılması uygun olacaktır.

$$N\left(b_i > \tilde{v}_i^g\right) \ge N_{\text{eşik}} \qquad \textbf{(5.2)}$$

Belirlilik ortamında ele alınan ZA-ARP'lerinde genellikle yolculuk süreleri ve yolculuk mesafeleri eşdeğer ölçeklerdir. Yani, en küçük toplam yolculuk süreli bir çözüm aynı zamanda en küçük yolculuk mesafeli çözüm olmaktadır. Örneğin, belirlilik durumunda en fazla kullanılan Solomon (1987) örnek problemlerinde

yolculuk süreleri ile müşteriler arasındaki mesafeler birbirine eşittir. Bu nedenle amaç fonksiyonunda yolculuk süresinin veya yolculuk mesafesinin kullanımı fark yaratmamaktadır. Ancak, müşteriler arasındaki yolculuk sürelerinin olabilirlik dağılımları ile temsil edildiği bir karar ortamında, toplam sürenin veya toplam mesafenin minimizasyonu farklı en iyi sonuçlar verebilecektir. Yolculuk süreleri, mesafe dışında faktörlerden de etkilendiğinde ve sürelerin kesin değeri bilinmediğinde iki durum için farklı en iyi çözümler mevcut olabilir. Bu çalışmada ise hiyerarşik olarak, kullanılan araç sayısının ve toplam yolculuk mesafesinin minimizasyonu amaçlanmıştır. Çünkü belirli bir verinin minimizasyonu amacıyla elde edilecek çözümlerin belirsiz bir verinin minimizasyonu amacıyla elde edilecek çözümlere göre daha kararlı ve güvenilir olacağı öngörülmüştür. Ayrıca, bulanık yolculuk süreleri durumunda, toplam yolculuk süresinin minimizasyonu için bulanık bir amaç fonksiyonunun modellenmesi gerekeceğinden modelin karmaşıklığı artacaktır.

Çalışmanın ikinci bölümünde yer alan belirli karar ortamındaki ZA-ARP formülasyonu temel alınarak; bulanık yolculuk süresi, kesin talep ve kesin teslimat zamanı aralığı içeren bir ZA-ARP için aşağıdaki (5.3-5.11) şans kısıtlı olabilirlik programlama modeli önerilmiştir.

$$\min \quad \sum_{g \in R} \sum_{(i,j) \in A} c_{ij} x_{ij}^{g} \tag{5.3}$$

$$N\left(\tilde{v}_i^g < b_i\right) \geq N_{\text{eşik}} \quad \forall i \in V, \forall g \in R \tag{5.4}$$

$$x_{ij}^{g} . \left(\tilde{v}_j^g - \text{maks}\,(\tilde{v}_i^g, a_i) + s_i + \tilde{t}_{ij}\right) \geq 0 \quad \forall (i,j) \in A, \forall g \in R \tag{5.5}$$

$$\sum_{g \in R} \sum_{j \in V} x_{ij}^{g} = 1 \quad \forall i \in V - \{0\} \tag{5.6}$$

$$\sum_{i \in V - \{0\}} q_i \sum_{j \in V} x_{ij}^{g} \leq K_g \quad \forall g \in R \tag{5.7}$$

$$\sum_{j \in V} x_{0j}^{g} = 1 \quad \forall g \in R \tag{5.8}$$

$$\sum_{i \in V} x_{i0}^{g} = 1 \quad \forall g \in R \tag{5.9}$$

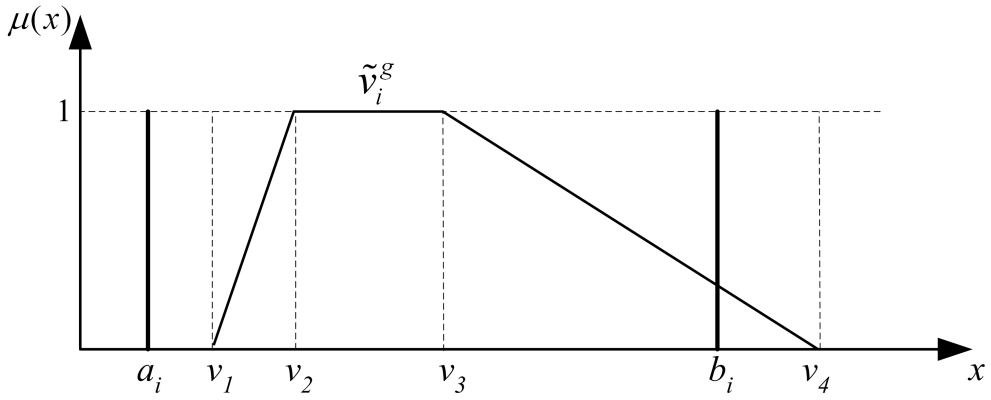
$$\sum_{i \in V} x_{iu}^{g} - \sum_{j \in V} x_{uj}^{g} = 0 \quad \forall u \in V - \{0\}, \forall g \in R \tag{5.10}$$

$$x_{ij}^{g} \in \{0,1\} \quad \forall (i, j) \in A, \forall g \in R \tag{5.11}$$

Olabilirlik programlama modelinde yer alan (5.4) kısıtındaki $N(\tilde{v}_i^g < b_i)$ gereklilik ölçütü değeri; daha önce belirtildiği gibi $(\tilde{v}_i^g < b_i)$ olayının gerçekleşmemesi hakkındaki bilgi, yani $(\tilde{v}_i^g \geq b_i)$ olayının olabilirliği kullanılarak hesaplanmaktadır.

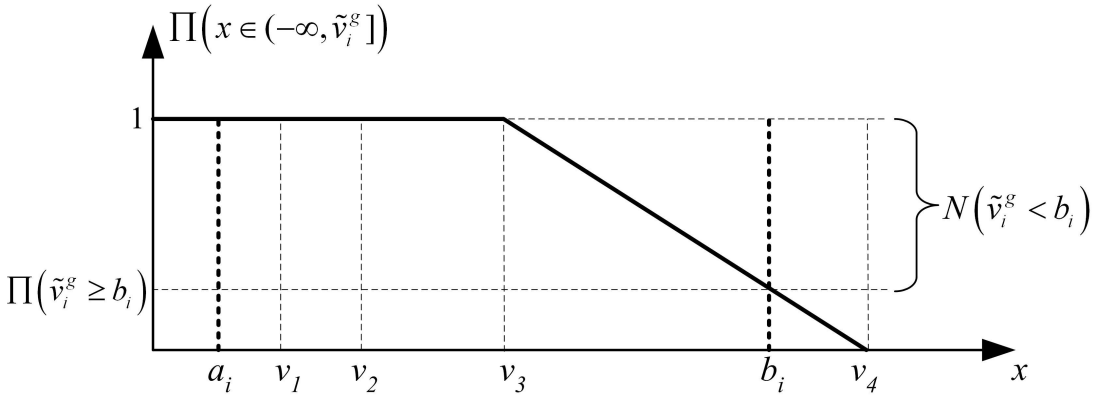
$$N(\tilde{v}_i^g < b_i) = 1 - \prod(\tilde{v}_i^g \geq b_i) = 1 - \prod(b_i \in (-\infty, \tilde{v}_i^g]) \tag{5.12}$$

Şekil 5.5'te, *i* müşterisine teslimat yapılabilecek zaman aralığı [a_i, b_i] ve planlanan rotaya göre *i* müşterisine araç varış zamanının ise $\tilde{v}_i^g$ olduğu örnek bir durum gösterilmektedir.



Şekil 5.5: Olabilirlik programlama modeli için örnek durum

Şekil 5.5'daki örnek durum için (5.12) denklemi ile $N(\tilde{v}_i^g < b_i)$ değerinin hesaplanması Şekil 5.6'da gösterilmiştir.



Şekil 5.6: $N(\tilde{v}_i < b_i)$ ölçütü için örnek durum

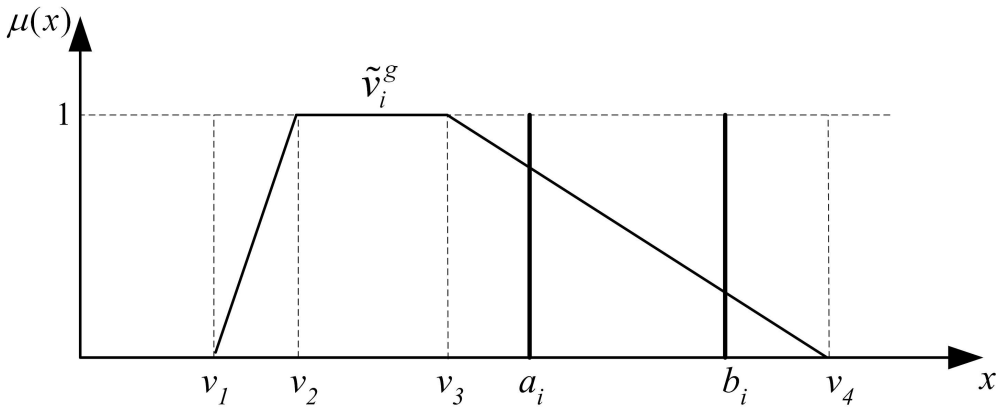
$N(\tilde{v}_i^g < b_i)$ değeri, $\tilde{v}_i^g = (v_1, v_2, v_3, v_4)$ ile tanımlanan yamuk bir bulanık aralık olmak üzere (5.13) ile hesaplanabilir.

$$N(\tilde{v}_i^g < b_i) = \begin{cases} 1, & v_4 \leq b_i \text{ ise,} \\ \dfrac{b_i - v_3}{v_4 - v_3}, & v_3 \leq b_i < v_4 \text{ ise,} \\ 0, & b_i < v_3 \text{ ise.} \end{cases} \quad \textbf{(5.13)}$$

(5.13) yerine kodlama aşamasında kolaylık sağlanması amacıyla (5.14) eşitliği kullanılabilir.

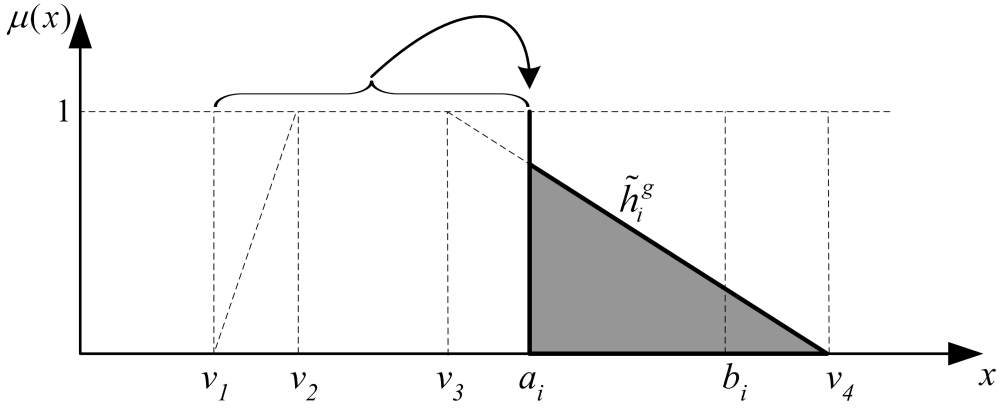
$$N(\tilde{v}_i < b_i) = \text{maks}\left(0, \min\left(1, \frac{b_i - v_3}{v_4 - v_3}\right)\right) \quad \textbf{(5.14)}$$

Yolculuk sürelerinin bulanık aralıklarla tanımlandığı olabilirlik programlama modelinde dikkat edilmesi gereken diğer bir unsur da teslimat başlangıç zamanı ($\tilde{h}_i^g$) değerinin elde edilmesi ile ilgilidir. (5.5) kısıtında yer alan $\text{maks}(\tilde{v}_i^g, a_i)$ ifadesi, $\tilde{h}_i^g = \text{maks}(\tilde{v}_i^g, a_i)$ olmak üzere, g aracının i müşterisine teslimat başlangıç zamanıdır. Şekil 5.7'de, i müşterisine teslimat yapılabilecek zaman aralığının [a_i, b_i] ve planlanan rotaya göre aracın i müşterisine varış zamanın ise $\tilde{v}_i^g$ olduğu örnek bir durum gösterilmektedir.



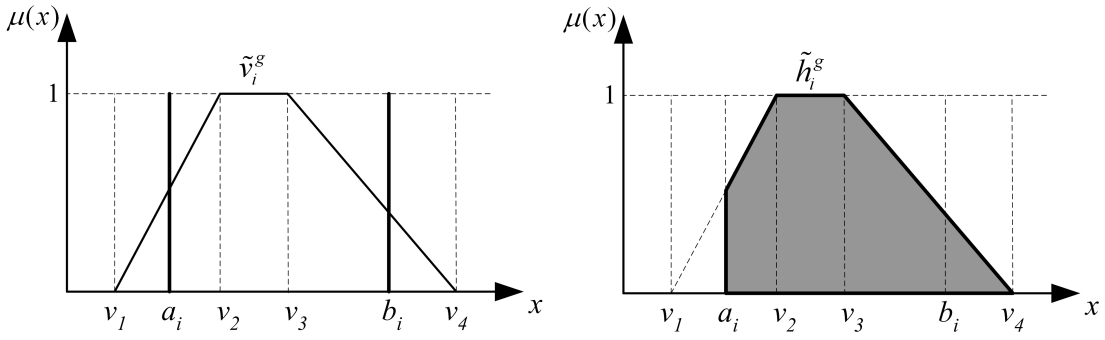
Şekil 5.7: Teslimat başlangıç zamanı için örnek durum

Bu durumda müşteriye teslimat zamanı, $\tilde{h}_i^g = \text{maks}\,(\tilde{v}_i^g, a_i)$ işlemi ile Şekil 5.8'de taralı bölgede gösterildiği gibi yamuk olmayan bir bulanık aralık olacaktır.



Şekil 5.8: Teslimat başlangıç zamanının elde edilmesi

Şekil 5.8'de görüldüğü gibi $\tilde{v}_i^g$ bulanık değerinde a_i kesin değerinin sol tarafında kalan kısım, aracın müşteri önünde bekleme yapacağı durumlar için olabilirlikleri temsil etmektedir. Bu olabilirliklerin gerçekleşmesi durumlarının hepsinde, teslimat başlangıç zamanı a_i olacağından, a_i değerinin $\tilde{h}_i^g$ bulanık aralığındaki olabilirlik değeri (yayılım prensibine göre) söz konusu bölgedeki en büyük olabilirlik değerine eşit olacaktır. Böylece a_i değerinin sol tarafındaki değerler için olabilirlik bilgisi bir anlamda a_i değeri üzerine aktarılmış olacaktır. Şekil 5.9'da farklı bir örnek durum için teslimat başlangıç zamanının elde edilmesi gösterilmiştir.

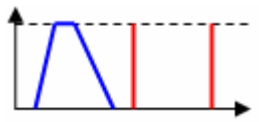
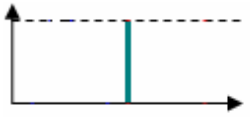
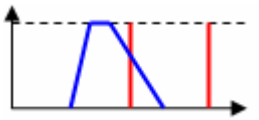
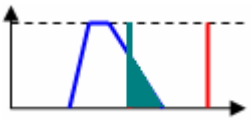
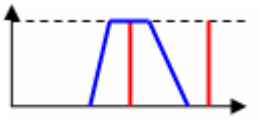
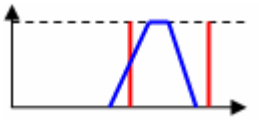
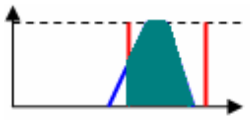
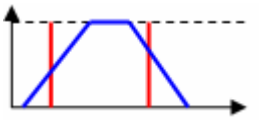
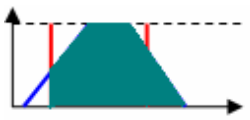
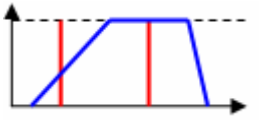
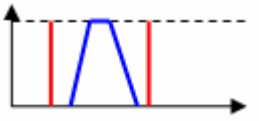
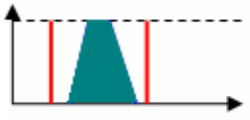
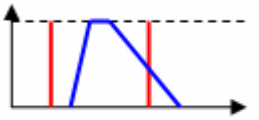
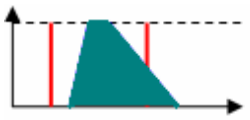
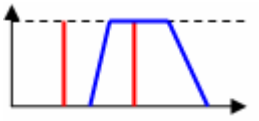
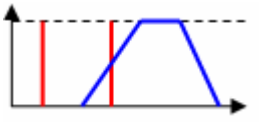
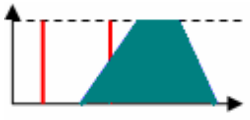
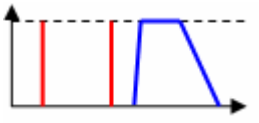
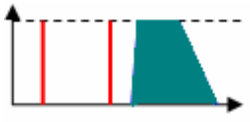


Şekil 5.9: Örnek durum

$\tilde{v}_i^g$ bulanık aralığında, b_i değerinden daha büyük değerlerin olabilirliği söz konusu olduğunda teslimat başlangıç zamanın hesaplanmasında herhangi bir kesinti yapılması gerekmemektedir. Bu tip durumlarda, (5.4) kısıt ile gereklilik ölçütü kullanılarak oluşturulmakta olan rotanın uygun olup olmayacağı değerlendirilmektedir. Ayrıca, b_i değerinden daha büyük değerlerin olabilirlikleri hakkındaki bilgi sonraki müşterilerin varış zamanlarının hesaplanmasında da kullanılacağından korunması önemlidir.

Olabilirlik programlama modelinde karşılaşılabilecek olası durumlar için $\tilde{h}_i^g$ ve $N(\tilde{v}_i^g < b_i)$ değerleri Tablo 5.1'de gösterilmiştir. İlk sütunda $\tilde{v}_i^g$ ve $[a_i, b_i]$ aralıklarının birbirlerine bağlı olarak bulunabilecekleri farklı pozisyonlar gösterilmiştir. İkinci sütunda $\tilde{h}_i^g$ bulanık değeri taralı alan ile gösterilmiştir. Üçüncü sütunda ise $N(\tilde{v}_i^g < b_i)$ ölçütünün alabileceği değer(ler) yer almaktadır.

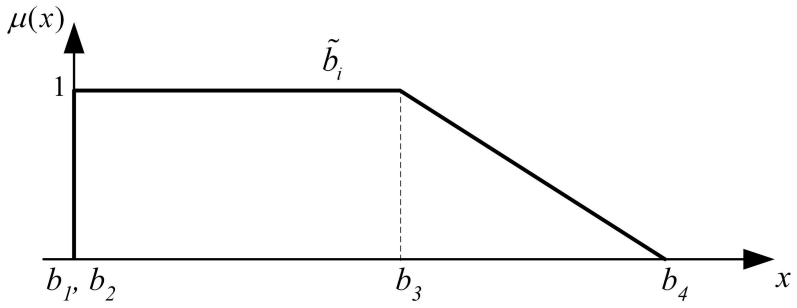
Tablo 5.1: Olabilirlik programlama modeli için olası durumlar

$\tilde{v}_i^g$, $[a_i, b_i]$	$\tilde{h}_i^g = \text{maks}\,(\tilde{v}_i^g, a_i)$	$N\left(\tilde{v}_i^g < b_i\right)$
		1
		1
		1
		1
		(0,1)
		0
		1
		(0,1)
		0
		0
		0

5.3.2 Esnek programlama modeli

Esnek programlama modeli, ikinci bölümde belirlilik ortamında tanımlanmış olan ZA-ARP'nde en geç teslimat zamanının esneklik içerdiği durumu ele almaktadır. Özel uygulama alanları hariç olmak üzere en geç teslimat zamanı için kesin bir değerin kullanılması, modelin gerçek hayattaki problemi tam olarak temsil edememesine neden olmaktadır.

Esnek programlama modelinde, teslimat zamanının alabileceği değerler için müşteri tatmin düzeyini temsil eden bir $\tilde{b}_i$ bulanık kümesi tanımlanmıştır. $\tilde{b}_i$, $b_1 = b_2 = 0$ olmak üzere (b_1, b_2, b_3, b_4) ile tanımlanabilen yamuk şekilli bir bulanık küme olarak modellenmiştir (Şekil 5.10).



Şekil 5.10: Müşteri tatmin düzeyi

Bu durumda, müşterinin en geç teslimat başlangıç zamanı ile ilgili tercihleri ve toleransı modele yansıtılmış olacaktır. En geç teslimat zamanlarının bulanık kümeler ile modellenmesi karar vericiye de esneklik sağlayacaktır. Doğal olarak öncelikle tüm müşterilere tam tatmin düzeyi ile hizmet verilebilecek uygun bir çözüm aranabilir. Eğer uygun bir çözüm bulunamıyorsa, müşterilerin önem derecesine göre bazılarının veya eşit önem dereceli müşteriler söz konusu ise tüm müşterilerin tatmin düzeyini belirli bir eşik değerinin üstünde tutan uygun bir çözüm aranabilir.

Olabilirlik programlama modelinde (5.4) kısıt kümesinde yer alan $N\left(\tilde{v}_i^g < b_i\right)$ ile bulanık bir aralık olarak hesaplanan müşteriye varış zamanının, kesin bir değer olarak bilinen müşteri en geç teslimat başlangıç zamanından küçük olmasının gerekliliği hesaplanmaktadır. Yani bulanık bir aralık ile modellenen bir değişkenin gerçekleşecek değerinin kesin bir sayıya eşit veya daha küçük olması olayının kesinliği konusunda bir ölçüm yapılmaktadır. Esnek programlama modelinde ise

yolculuk sürelerinin kesin olarak bilindiği varsayıldığından müşterilere varış zamanları da kesin birer sayıdır. Bulanık bir küme ile modellenen müşteriye en geç teslimat başlangıç zamanı ise müşteri tatmin düzeyini temsil etmektedir. Bu durumda g aracı i müşterisine v_i^g anında varış yaptığında, i müşterisinin tatmin düzeyi olabilirlik ölçütü kullanılarak (5.15) denklemi ile hesaplanabilir.

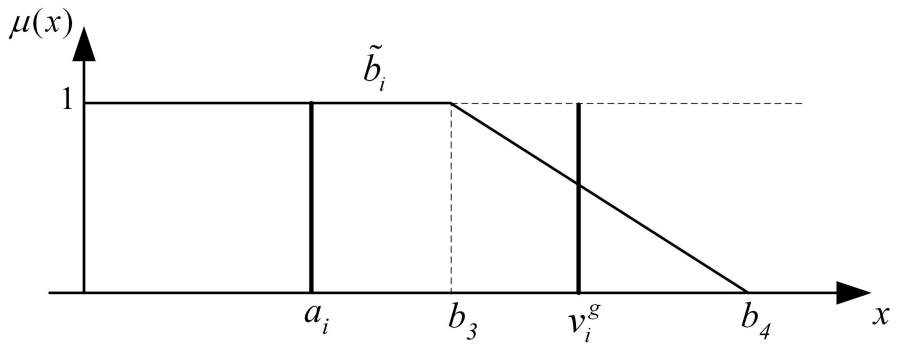
$$\prod(v_i^g \le \tilde{b}_i) = \prod(v_i^g \in (-\infty, \tilde{b}_i]) \qquad \textbf{(5.15)}$$

Tüm müşterilerin tatmin düzeyinin en az belirli bir eşik değer ($\prod_{\text{eşik}}$) kadar olması istendiğinde olabilirlik programlama modelindeki (5.4) ve (5.5) kısıt kümeleri yerine aşağıdaki (5.16) ve (5.17) kısıt kümeleri kullanılarak esnek programlama modeli elde edilebilir.

$$\prod\left(v_i^g \le \tilde{b}_i\right) \ge \prod_{\text{eşik}} \quad \forall i \in V, \forall g \in R \qquad \textbf{(5.16)}$$

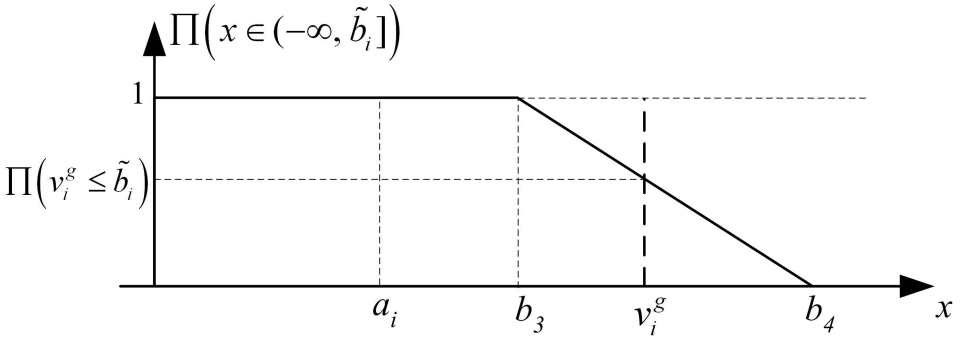
$$x_{ij}^g . \left(v_j^g - \text{maks}\,(v_i^g, a_i) + s_i + t_{ij}\right) \ge 0 \quad \forall (i,j) \in A, \forall g \in R \qquad \textbf{(5.17)}$$

Esnek programlama modelinde v_i^g ile $\tilde{b}_i$ arasındaki ilişki için örnek bir durum Şekil 5.11 üzerinde gösterilmiştir.



Şekil 5.11: Müşteri tatmin düzeyi için örnek durum

Şekil 5.11'deki örnek durum için müşteri tatmin düzeyinin hesaplanması Şekil 5.12'de gösterilmiştir.



Şekil 5.12: $\prod(v_i \le \tilde{b}_i)$ değerinin hesaplanması

$\tilde{b}_i$ kümesinde $b_1 = b_2 = 0$ olduğundan ve negatif değerler ile ilgilenilmediğinden $(-\infty, \tilde{b}_i] = \tilde{b}_i$ olarak kabul edilebilir. Bu durumda $\prod(v_i^g \le \tilde{b}_i) = \mu_{\tilde{b}_i}(v_i^g)$ olarak da gösterilebilir. $\prod(v_i^g \le \tilde{b}_i)$ değeri, $\tilde{b}_i = (b_1, b_2, b_3, b_4)$ ile tanımlanan yamuk bir bulanık küme olmak üzere (5.18) denklemi ile hesaplanabilir.

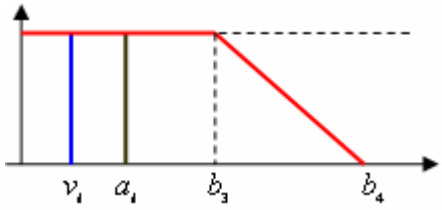
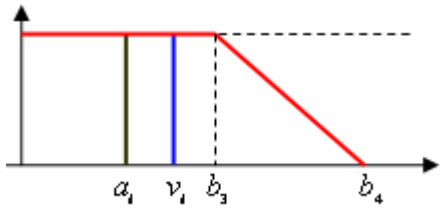
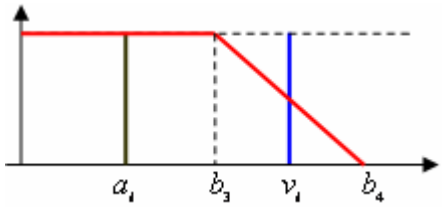
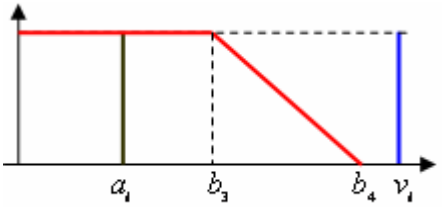
$$\prod(v_i^g \le \tilde{b}_i) = \begin{cases} 1, & v_i^g \le b_3 \text{ ise,} \\ \dfrac{b_4 - v_i^g}{b_4 - b_3}, & b_3 \le v_i^g < b_4 \text{ ise,} \\ 0, & \text{değilse.} \end{cases} \qquad \textbf{(5.18)}$$

(5.18) yerine kodlama aşamasında kolaylık sağlanması amacıyla (5.19) denklemi kullanılabilir.

$$\prod(v_i^g \le \tilde{b}_i) = \text{maks}\left(0, \min\left(1, \frac{b_4 - v_i^g}{b_4 - b_3}\right)\right) \qquad \textbf{(5.19)}$$

Esnek programlama modelinde karşılaşılabilecek olası durumlar için $\prod(v_i^g \le \tilde{b}_i)$ değerleri Tablo 5.2'de özetlenmiştir. İlk sütunda v_i^g ve $[a_i, \tilde{b}_i]$ aralıklarının birbirlerine bağlı olarak bulunabilecekleri farklı pozisyonlar gösterilmiştir. İkinci sütunda h_i^g, teslimat başlangıç zamanını, üçüncü sütunda ise $\prod(v_i^g \le \tilde{b}_i)$ ölçütünün alabileceği değer(ler) yer almaktadır.

Tablo 5.2: Esnek programlama modeli için olası durumlar

v_i^g , $[a_i, \tilde{b}_i]$	$h_i = \text{maks}\,(v_i^g, a_i)$	$\prod(v_i^g \le \tilde{b}_i)$
	a_i	1
	v_i^g	1
	v_i^g	(0,1)
	v_i^g	0

5.3.3 Gürbüz programlama modeli

Olabilirlik ve esnek programlama modellerinde, kısıtların sol taraf katsayılarında yer alan veriler için muğlaklık anlamında ortaya çıkan belirsizlikler ile kısıtların sağ tarafında kesin sınırların belirlenememesi durumunda esneklik anlamında ortaya çıkan belirsizlikler ayrı ayrı ele alınmıştır. Gürbüz programlama modelinde ise her iki tip belirsizliğin aynı model içerisinde bulunması durumu ele alınmıştır. Yolculuk süreleri bulanık aralıklarla, en geç teslimat zamanları ise müşteri tatmin düzeyini temsil eden bulanık kümelerle modellenmiştir.

Belirsizlik yaratan ilk durum müşteriye en geç teslimat başlangıç zamanından önce teslimata başlanabilmesi ile ilgilidir. Bu kısıtın sağlanamadığı her durumda elde edilen çözüm geçersiz olacaktır. g aracının i müşterisine varış zamanı $\tilde{v}_i^g = (v_1, v_2, v_3, v_4)$ yamuk şekilli bulanık aralığı ile, i müşterisinin tatmin düzeyi de

$\tilde{b}_i = (b_1^i, b_2^i, b_3^i, b_4^i)$ yamuk şekilli bulanık kümesi ($b_1^i = b_2^i = 0$) ile tanımlanmıştır. Bir çözümün geçerli olabilmesi için öncelikle araç varış zamanının, teslimat başlangıcı için kabul edilebilir en büyük zaman olan b_4^i kesin değerinden daha küçük olması gereklidir. $\tilde{v}_i^g < b_4^i$ olayının gerekliliği için $N(\tilde{v}_i^g < b_4^i)$ ölçütü kullanılalarak gürbüz programlama modeli için (5.21) kısıt kümesi tanımlanmıştır. Böylece bir çözümün geçerli olması olayı kesinliğinin en az belirli bir $N_{eşik}$ değeri kadar olması sağlanmaktadır.

Belirsizlik yaratan ikinci durum ise teslimat zamanının ne kadar müşteri tatmini sağlayacağı ile ilgilidir. Müşteri tatmin düzeyinin ölçümü için $\prod(\tilde{v}_i^g \le \tilde{b}_i)$ ölçütü kullanılmıştır. (5.22) kısıt kümesi ile elde edilecek çözümlerin en az $\prod_{eşik}$ değeri kadar müşteri tatminine sahip olması sağlanmıştır. ZA-ARP için önerilen gürbüz programlama modeli (5.20)-(5.29) ile oluşturulmuştur.

$$\min \quad \sum_{g \in R} \sum_{(i,j) \in A} c_{ij} x_{ij}^g \tag{5.20}$$

$$N\left(\tilde{v}_i^g < b_4^i\right) \ge N_{eşik} \quad \forall i \in A, \forall g \in R \tag{5.21}$$

$$\prod\left(\tilde{v}_i^g \le \tilde{b}_i\right) \ge \prod_{eşik} \quad \forall i \in A, \forall g \in R \tag{5.22}$$

$$x_{ij}^g . \left(\tilde{v}_j^g - \text{maks}\,(\tilde{v}_i^g, a_i) + s_i + \tilde{t}_{ij}\right) \ge 0 \quad \forall (i,j) \in A, \forall g \in R \tag{5.23}$$

$$\sum_{g \in R} \sum_{j \in V} x_{ij}^g = 1 \qquad \forall i \in V - \{0\} \tag{5.24}$$

$$\sum_{i \in V - \{0\}} q_i \sum_{j \in V} x_{ij}^g \le K_g \quad \forall g \in R \tag{5.25}$$

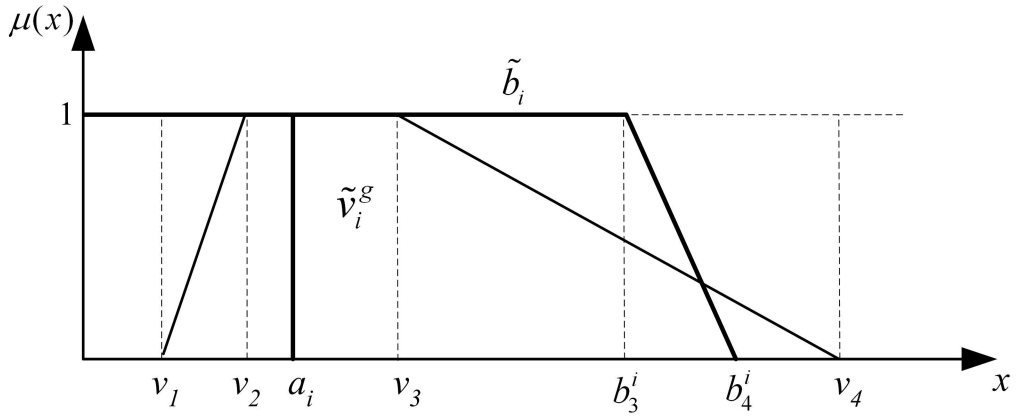
$$\sum_{j \in V} x_{0j}^g = 1 \qquad \forall g \in R \tag{5.26}$$

$$\sum_{i \in V} x_{i0}^g = 1 \qquad \forall g \in R \tag{5.27}$$

$$\sum_{i \in V} x_{iu}^{g} - \sum_{j \in V} x_{uj}^{g} = 0 \quad \forall u \in V - \{0\}, \forall g \in R \tag{5.28}$$

$$x_{ij}^{g} \in \{0,1\} \quad \forall (i, j) \in A, \forall g \in R \tag{5.29}$$

Gürbüz programlama modelinde varış zamanı ve teslimat zaman aralığı arasındaki ilişki için örnek bir durum Şekil 5.13'te gösterilmiştir.

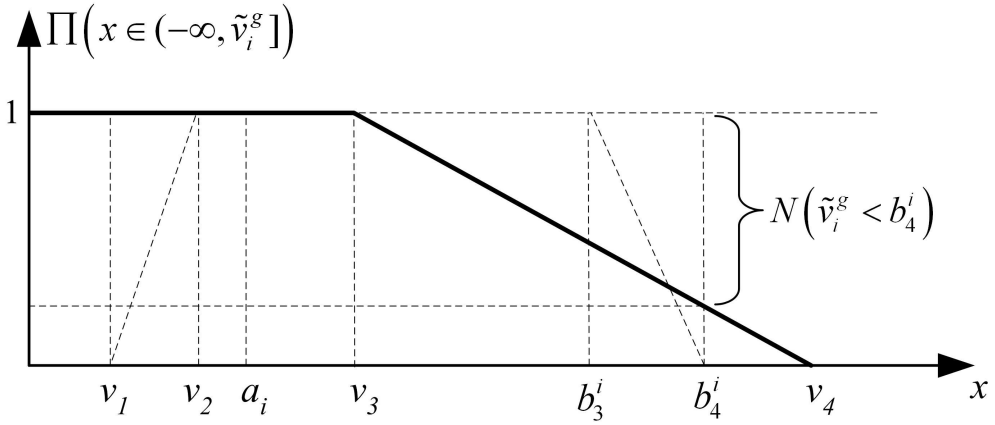


Şekil 5.13: $\tilde{v}_i^g, a_i$ ve $\tilde{b}_i$ için örnek durum

$N(\tilde{v}_i^g < b_4^i)$ değerinin elde edilmesi için (5.30) eşitliği kullanılabilir.

$$N(\tilde{v}_i^g < b_4^i) = 1 - \prod(\tilde{v}_i^g \geq b_4^i) = 1 - \prod(b_4^i \in (-\infty, \tilde{v}_i^g]) \tag{5.30}$$

Şekil 5.13'teki örnek durum için $N(\tilde{v}_i^g < b_4^i)$ değerinin hesaplanması Şekil 5.14'te gösterilmiştir.



Şekil 5.14: $N(\tilde{v}_i^g < b_4^i)$ değerinin hesaplanması

$N(\tilde{v}_i^g < b_4^i)$ değeri (5.31) denklemi ile hesaplanabilir.

$$N(\tilde{v}_i^g < b_4^i) = \begin{cases} 1, & v_4 \le b_4^i \text{ ise,} \\ \dfrac{b_4^i - v_3}{v_4 - v_3}, & v_3 < b_4^i < v_4 \text{ ise,} \\ 0, & b_4^i \le v_3 \text{ ise.} \end{cases} \quad \textbf{(5.31)}$$

(5.31) eşitliği yerine kodlama aşamasında kolaylık sağlanması amacıyla (5.32) eşitliği kullanılabilir.

$$N(\tilde{v}_i^g < b_4^i) = \text{maks}\left(0, \min\left(1, \frac{b_4^i - v_3}{v_4 - v_3}\right)\right) \quad \textbf{(5.32)}$$

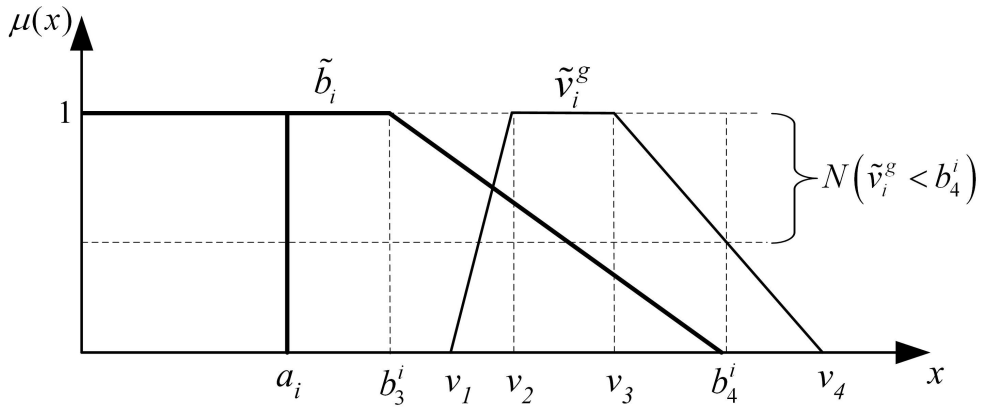
Bu kısımda konuyu daha derin anlamda inceleyebilmek için gürbüz programlama modelinde müşteriye en geç teslimat anından önce teslimat yapabilme olayının kesinlik ölçütünün neden $N(\tilde{v}_i^g < b_4^i)$ yerine $N(\tilde{v}_i^g < \tilde{b}_i)$ ile hesaplanmadığı ele alınmıştır. Aslında ilk bakışta $N(\tilde{v}_i^g < \tilde{b}_i)$ ölçütünün kullanımı daha mantıklı görülebilir. Ancak daha önce de değinildiği gibi bulanık kümeler ile hem esnek kısıtlar hem de belirsiz veriler modellenebildiğinden, bulanık modellerin oluşturmasına gerekçe olan bu kavramların karıştırılması bir risk oluşturmaktadır. $N(\tilde{v}_i^g < b_4^i)$ ölçütü $\tilde{v}_i^g$ bulanık aralığının gerçekleşecek değerinin b_4^i kesin sayısından daha küçük olması olayının kesinliğini tanımlamaktadır. Oysa $N(\tilde{v}_i^g < \tilde{b}_i)$ ölçütü her biri belirsizlik içeren $\tilde{v}_i^g$ ve $\tilde{b}_i$ değerlerinin gerçekleşecek değerleri için varış zamanının teslimat zamanından küçük olması olayının kesinliğini tanımlamaktadır. Ancak önerilen modelde $\tilde{b}_i$ bulanık bir değer olarak tanımlanmış olsa da belirsizlik içermemekte, müşteri tatmin düzeyini temsil etmektedir. Yani $\tilde{b}_i$ bulanık değerinin gerçekleşecek değeri için herhangi bir belirsizlik söz konusu değildir, $\tilde{b}_i$ değeri esnekliği temsil etmektedir. $N(\tilde{v}_i^g < \tilde{b}_i)$ değeri (5.33) denklemi kullanılarak (5.34) denklemi ile hesaplanabilir.

$$N(\tilde{v}_i^g < \tilde{b}_i) = 1 - \prod(\tilde{v}_i^g \ge \tilde{b}_i) = 1 - \left(\prod\left(\tilde{b}_i \in (-\infty, \tilde{v}_i^g]\right) \cup \prod\left(\tilde{v}_i^g \in [\tilde{b}_i, \infty)\right)\right) \quad \textbf{(5.33)}$$

$$N(\tilde{v}_i^g < \tilde{b}_i) = \text{maks}\left(0, \min\left(1, \frac{b_2^i - v_3}{(b_2^i - b_1^i) + (v_4 - v_3)}\right)\right) \quad \textbf{(5.34)}$$

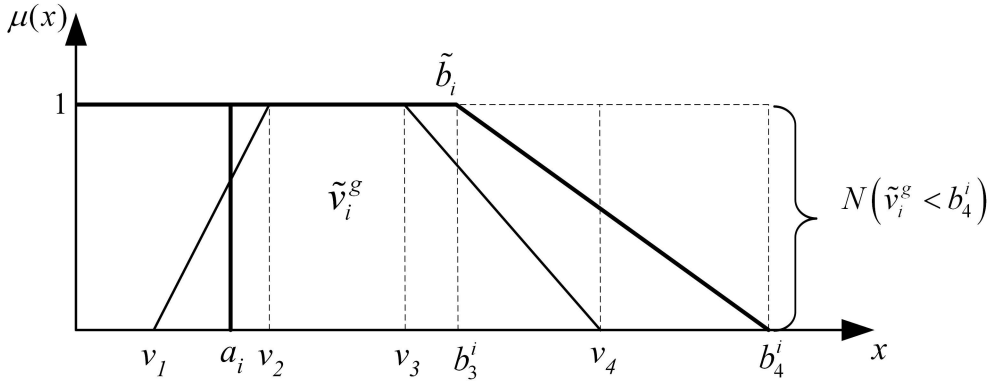
(5.34) denkleminde görüldüğü üzere $(b_2^i - v_3) \leq 0$ olduğu her durumda $N(\tilde{v}_i^g < \tilde{b}_i) = 0$ olacaktır. $b_1^i = b_2^i = 0$ olduğundan $N(\tilde{v}_i^g < \tilde{b}_i)$ ölçütü her zaman sıfır değerini alacaktır. Yani $N(\tilde{v}_i^g < b_4^i)$ yerine $N(\tilde{v}_i^g < \tilde{b}_i)$ ölçütünün kullanılması hesaplama açısından da anlamlı değildir.

Şekil 5.15 ve Şekil 5.16'da $N(\tilde{v}_i^g < \tilde{b}_i)$ ve $N(\tilde{v}_i^g < b_4^i)$ ölçütlerinin hesaplanması için örnek durumlar gösterilmiştir.



Şekil 5.15: $N(\tilde{v}_i^g < \tilde{b}_i)$ *ve* $N(\tilde{v}_i^g < b_4^i)$ için örnek durum-1

Şekil 5.15'de $\prod\left(\tilde{b}_i \in (-\infty, \tilde{v}_i^g]\right)=1$ ve $\prod\left(\tilde{v}_i^g \in [\tilde{b}_i, \infty)\right)=1$ olduğundan $N(\tilde{v}_i^g < \tilde{b}_i)=0$, $N(\tilde{v}_i^g < b_4^i)>0$ olarak hesaplanmaktadır.



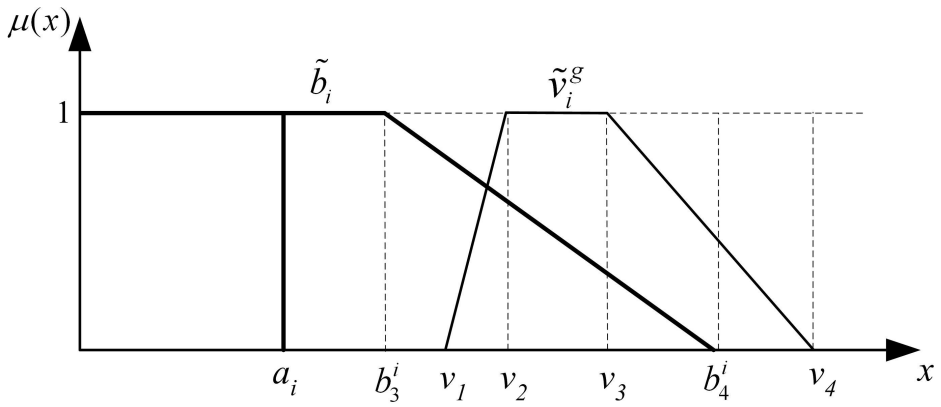
Şekil 5.16: $N(\tilde{v}_i^g < \tilde{b}_i)$ *ve* $N(\tilde{v}_i^g < b_4^i)$ için örnek durum-2

Şekil 5.16'da ise $\prod\left(\tilde{b}_i \in (-\infty, \tilde{v}_i^g]\right)=1$ ve $\prod\left(\tilde{v}_i^g \in [\tilde{b}_i, \infty)\right)=1$ olduğundan $N(\tilde{v}_i^g < \tilde{b}_i)=0,\ N(\tilde{v}_i^g < b_4^i)=1$ olarak hesaplanmaktadır.

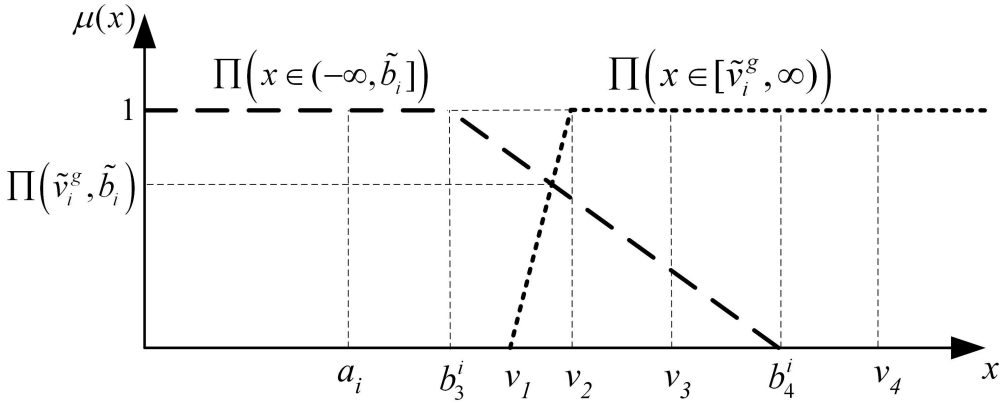
Gürbüz programlama modelinde (5.22) kısıt kümesini oluşturuan $\prod\left(\tilde{v}_i^g \le \tilde{b}_i\right) \ge \prod_{eşik} \quad \forall i \in A, \forall g \in R$ ifadesi her bir müşteri tatmin düzeyinin belirli bir eşik değerin üstünde olmasını sağlanmaktadır. $\prod(\tilde{v}_i^g \le \tilde{b}_i)$ değeri (5.35) denklemi ile elde edilebilir.

$$\prod(\tilde{v}_i^g \le \tilde{b}_i) = \prod\left(\tilde{b} \in [\tilde{v}_i^g, \infty)\right) \cup \prod\left(\tilde{v}_i^g \in (-\infty, \tilde{b}_i]\right) \qquad \textbf{(5.35)}$$

Şekil 5.17'deki örnek durum için $\prod(\tilde{b}_i \ge \tilde{v}_i)$ değerinin hesaplanması Şekil 5.18'de gösterilmiştir.



Şekil 5.17: $\prod(\tilde{b}_i \ge \tilde{v}_i)$ için örnek durum



Şekil 5.18: $\prod(\tilde{b}_i \ge \tilde{v}_i)$ değerinin hesaplanması

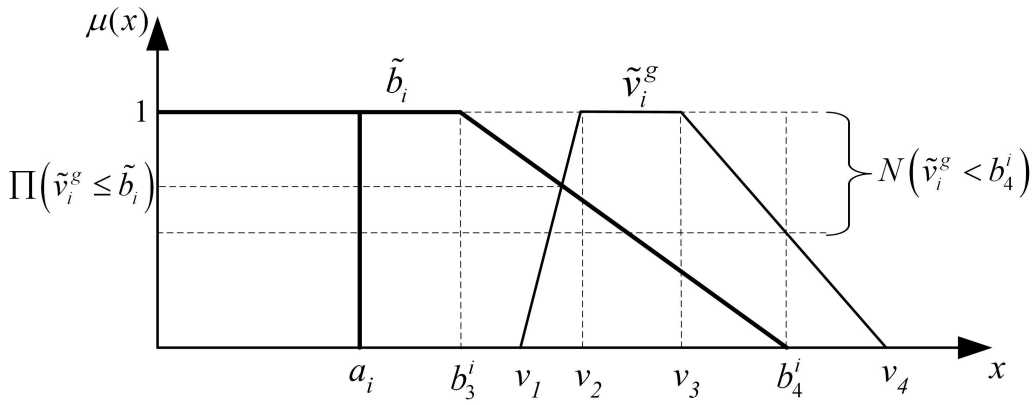
$\tilde{v}_i^g$, (v_1, v_2, v_3, v_4) ile tanımlanan yamuk şekilli bir bulanık aralık, $\tilde{b}_i$ ise $b_1^i = b_2^i = 0$ olmak üzere $(b_1^i, b_2^i, b_3^i, b_4^i)$ ile tanımlanan yamuk şekilli bir bulanık küme ise, $\prod(\tilde{v}_i^g \le \tilde{b}_i)$ değeri (5.36) denklemi ile hesaplanabilir.

$$\prod(\tilde{v}_i^g \le \tilde{b}_i) = \begin{cases} 1 & v_2 \le b_3^i \text{ ise} \\ 1 + \dfrac{b_3^i - v_2}{(b_4^i - b_3^i) + (v_2 - v_1)} & b_3^i < v_2 \text{ ve } v_1 < b_4^i \text{ ise} \\ 0 & v_1 \ge b_4^i \text{ ise} \end{cases} \quad \textbf{(5.36)}$$

(5.36) denklemi yerine kodlama aşamasında (5.37) denklemi kullanılabilir.

$$\prod(\tilde{v}_i^g \le \tilde{b}_i) = \text{maks}\left(0, \min\left(1, 1 + \frac{b_3^i - v_2}{(b_4^i - b_3^i) + (v_2 - v_1)}\right)\right) \quad \textbf{(5.37)}$$

$N(\tilde{v}_i^g < b_4^i)$ değeri (v_3, v_4, b_4^i) değerlerine bağlı olarak hesaplanırken $\prod(\tilde{v}_i^g \le \tilde{b}_i)$ değeri (v_1, v_2, b_3^i, b_4^i) bağlı olarak hesaplanmaktadır. Şekil 5.19'da aynı örnek durum üzerinde $N(\tilde{v}_i^g < b_4^i)$ ve $\prod(\tilde{v}_i^g \le \tilde{b}_i)$ değerleri gösterilmiştir.

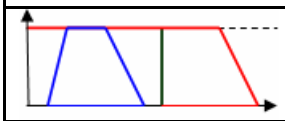
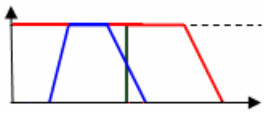
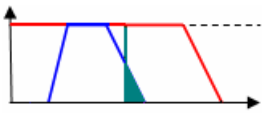
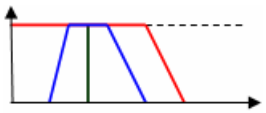
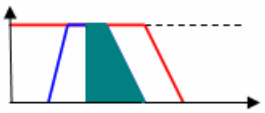
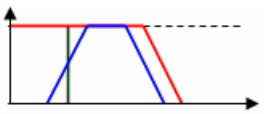
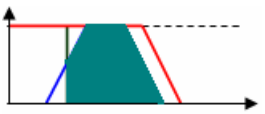
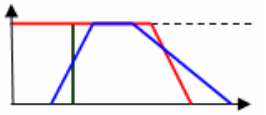
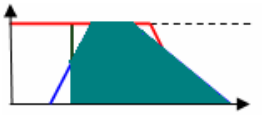
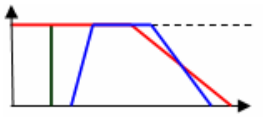
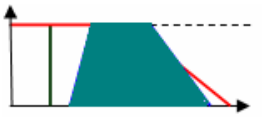
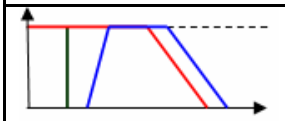
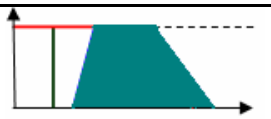
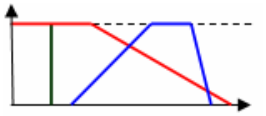
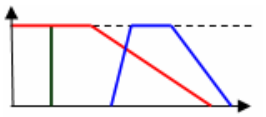
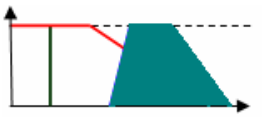
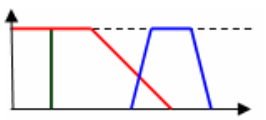


Şekil 5.19: $N(\tilde{v}_i^g < b_4^i)$ *ve* $\prod(\tilde{v}_i^g \le \tilde{b}_i)$ hesaplanması

Gürbüz programlama modelinde teslimat başlangıç zamanı ($\tilde{h}_i^g$), olabilirlik programlama modelinde olduğu gibi hesaplanmaktadır. Varış zamanını temsil eden bulanık değerin en erken teslimat zamanından küçük olan kısmı kesilerek olabilirlikleri en erken teslimat zamanı ile temsil edilmektedir. Gürbüz programlama

modelinde karşılaşılabilecek olası durumlar için $\tilde{h}_i^g$, $\prod(\tilde{v}_i^g \le \tilde{b}_i)$ ve $N(\tilde{v}_i^g < b_i)$ değerleri Tablo 5.3'de gösterilmiştir.

Tablo 5.3: Gürbüz programlama modeli için olası durumlar

$\tilde{v}_i^g, a_i, \tilde{b}_i$	$\tilde{h}_i^g = \text{maks}\,(\tilde{v}_i^g, a_i)$	$\prod(\tilde{v}_i^g \le \tilde{b}_i)$	$N(\tilde{v}_i^g < b_i)$
		1	1
		1	1
		1	1
		1	1
		1	(0,1)
		1	1
		1	(0,1)
		(0,1)	1
		(0,1)	(0,1)
		(0,1)	0

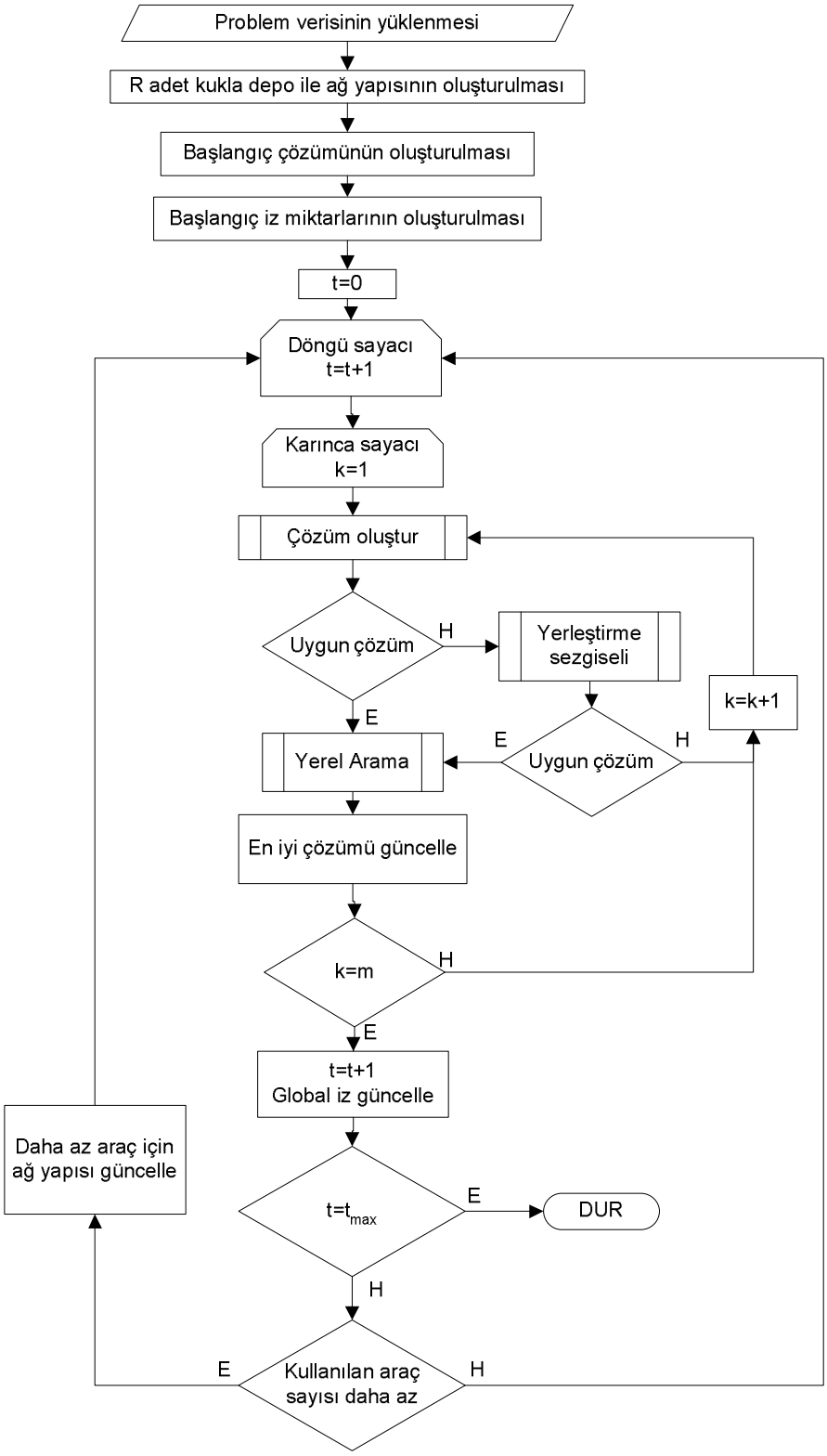
Tablo 5.3'ün ilk sütununda $\tilde{v}_i^g$, a_i ve $\tilde{b}_i$ değerlerinin birbirlerine bağlı olarak bulunabilecekleri farklı pozisyonlar gösterilmiştir. İkinci sütunda $\tilde{h}_i^g$ bulanık değeri taralı alan ile gösterilmiştir. Üçüncü sütunda, $\prod(\tilde{v}_i^g \le \tilde{b}_i)$ ölçütünün, dördüncü sütunda ise $N(\tilde{v}_i^g < b_i)$ ölçütünün alabileceği değer(ler) yer almaktadır. Tablo 5.3'de görüldüğü gibi $N(\tilde{v}_i^g < b_i)$ değeri oluşturulan çizelgede *i* müşterisine en geç teslimat anından önce teslimat yapılamaması olayının riskini temsil ederken $\prod(\tilde{v}_i^g \le \tilde{b}_i)$ değeri *i* müşterisinin tatmin düzeyini temsil etmektedir.

5.4 Önerilen Modellerin Çözümü ve Deneyler

Bu bölümde, bulanık karar ortamında ZA-ARP için önerilen modellere çözüm oluşturabilmek için dördüncü bölümde belirlilik ortamı için tasarlanan ZAARP-KKS algoritması gerekli değişiklikler yapılarak kullanılmıştır. Şekil 5.20'de ZAARP-KKS algoritmasının temel adımlarını gösteren akış şeması yer almaktadır. Algoritmanın "Çözüm Oluştur" alt sürecinde değişiklikler yapılarak önerilen modeller için kullanımı sağlanmıştır.

Gürbüz programlama modeli, olabilirlik ve esnek programlama modellerini de kapsadığından öncelikle ZAARP-KKS algoritmasının gürbüz programlama modeli için kullanımı ele alınmıştır. Kesin sayılar bulanık kümelerin özel bir durumu gibi ele alınabileceğinden, gürbüz programlama modeli için önerilen algoritma olabilirlik programlama, esnek programlama ve kesin durum modelleri için de kullanılabilir yapıdadır. Önerilen algoritma, bulanık karar ortamında kullanılabilirliğini belirtmek amacıyla BZAARP-KKS olarak adlandırılmıştır. Tüm deneylerde sabit olarak $\alpha = 1,\ \beta = 2,\ \rho = 0,1,\ q_0 = 0,9,\ m = 10$ parametre değerleri kullanılmıştır.

Şekil 5.20'deki akış şemasının ilk dört adımı algoritmanın başlangıç bölümünü oluşturmaktadır. Bu bölümde çözülecek problem ile ilgili verinin matris yapısında yüklenmesi, karıncaların arama yapacağı ağ yapısının oluşturulması, başlangıç çözümünün oluşturulması ve başlangıç iz miktarının belirlenerek tüm yollar üzerine bırakılması gerçekleştirilmektedir.

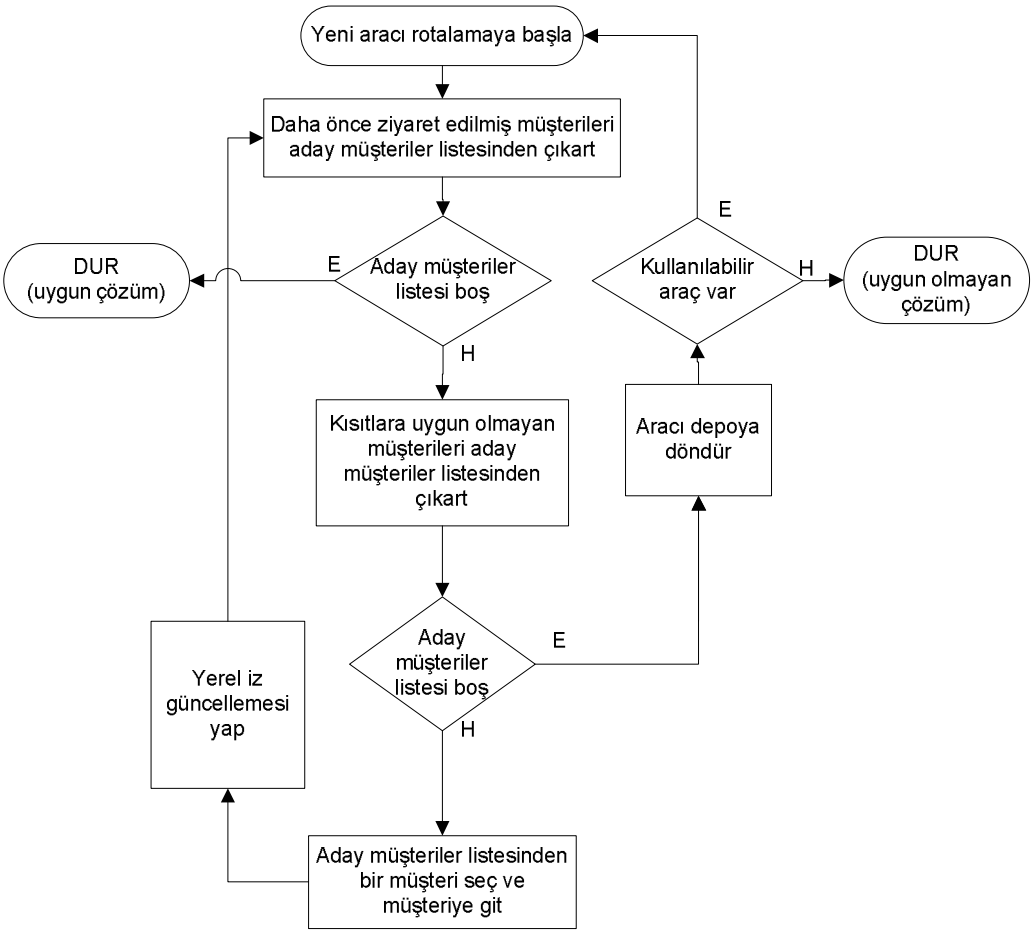


Şekil 5.20: ZAARP-KKS algoritması akış şeması

Başlangıç çözümü en yakın komşu sezgiseli ile oluşturulmuştur. Rasgele bir depodan başlanarak, modeldeki kısıtları ihlal etmeyen ve en yakın mesafede bulunan müşterin oluşturulmakta olan tura eklenmesiyle devam eden bir süreçtir. Kısıtlar nedeniyle

tura eklenecek bir müşteri bulunamaması durumunda daha önce ziyaret edilmemiş bir depoya dönülür, zaman ve elverişli kapasite değerleri güncellenerek turun tamamlanması amaçlanır.

Çözüm oluşturma aşamasında toplam *m* adet karıncanın her biri tam bir çözüm elde etmek amacıyla ağ yapısı üzerinde arama yapacaktır. Bir karınca için çözüm arama sürecinin akış şeması Şekil 5.21'de gösterilmiştir.



Şekil 5.21: Bir karınca için çözüm oluşturma süreci akış şeması

BZAARP-KKS algoritması, belirlilik durumunda ve bulanık karar ortamı için önerilen modellerin çözümü için kullanılabilir yapıdadır. Farklı modelller için karıncaların çözüm oluşturma süreçlerindeki tek farklılık kısıtlara uygun olmayan müşterilerin aday müşteriler listesinden çıkartılması aşamasında oluşmaktadır. Araç kapasiteleri kısıtı tüm modellerde aynı şekilde yer almakla beraber; belirlik durumunda, kapasite ve en geç teslimat zamanı kısıtına uymayan müşteriler; olabilirlik programlama modelinde, $N_{eşik}$ düzeyinde gereklikle en geç teslimat

zamanına kadar teslimata başlanamayacak müşteriler; esnek programlama modelinde, $\prod_{eşik}$ düzeyinde tatmini gerçekleştirilemeyecek müşteriler; gürbüz programlama modelinde ise hem $N_{eşik}$ hem de $\prod_{eşik}$ düzeylerinde ilgili kısıtları sağlayamayan müşteriler, aday müşteriler listesinden çıkarılmaktadır.

Gürbüz programlama modelinde $N(\tilde{v}_i^g < b_4^i)$ gereklilik değeri ve $\prod(\tilde{v}_i^g \leq \tilde{b}_i)$ müşteri tatmin düzeyi değeri ölçümü için kullanılan (5.32) ve (5.37) denklemleri olabilirlik programlama, esnek programlama ve kesin durum modelleri için de kullanılabilir yapıdadır.

Örneğin, gürbüz programlama modelinde $\prod(\tilde{v}_i^g \leq \tilde{b}_i)$ değeri (5.37) denklemi ile aşağıdaki gibi hesaplanmaktadır.

$$\prod(\tilde{v}_i^g \leq \tilde{b}_i) = \text{maks}\left(0, \min\left(1, 1+\frac{b_3^i - v_2}{(b_4^i - b_3^i)+(v_2 - v_1)}\right)\right)$$

Esnek programlama modelinde ise $\prod(v_i^g \leq \tilde{b}_i)$ değeri (5.19) denklemi ile aşağıdaki gibi hesaplanmaktadır.

$$\prod(v_i^g \leq \tilde{b}_i) = \text{maks}\left(0, \min\left(1, \frac{b_4^i - v_i^g}{b_4^i - b_3^i}\right)\right)$$

Esnek programlama modelinde yolculuk süreleri kesin olduğundan ($v_1 = v_2 = v_3 = v_4 = v_i^g$); (5.37) denklemi aşağıda görüldüğü gibi (5.19)'a dönüşmektedir.

$$1+\frac{b_3^i - v_2}{(b_4^i - b_3^i)+(v_2 - v_1)} = 1+\frac{b_3^i - v_i^g}{b_4^i - b_3^i} = \frac{b_4^i - v_i^g}{b_4^i - b_3^i}$$

Böylece, gürbüz programlama modelinin çözümüne yönelik olarak kodlanan BKKS-ZARP algoritması ile bu çalışmada yer alan tüm ZA-ARP modellerinin çözümünü gerçekleştirebilmektedir. Tüm bulanık işlemler $\lambda = \{e \to 0,1\}$ olmak üzere 2 düzeyde kesimle gerçekleştirilmiştir.

25 müşterilik bir problemde BZAARP-KKS algoritmasının 1 döngü süresi 0.30 saniye civarında hesaplanmıştır. 25 müşterilik Solomon (1987) problemlerine 500 döngülük bir koşum (yaklaşık 2.5 dakika) sonunda iyi kalitede çözümler oluşturulabilmektedir.

5.4.1 Olabilirlik Programlama Modeli için Deneyler

Geçmiş çalışmalar incelendiğinde zaman aralıklı araç rotalama problemini olabilirlik programlama modeli ile ele alan sadece iki adet çalışmaya rastlanmıştır. Zheng ve Liu (2006), ZA-ARP'ni bulanık yolculuk süreleri ile modellemiştir. Modelin çözümü için hibrid bir algoritma önermişlerdir. Önerilen algoritma, genetik algoritma ve simülasyon kullanarak arama yapmaktadır. Çaprazlama ve mutasyon operatörlerinin kullanımıyla elde edilen çözümlerin kısıtlara uygunluğu simülasyonla belirlenmektedir. Önerilen sürecin iyi çözümlere ulaşma süresinin, pratik alanda uygulanabilir sınırların dışında olduğu görülmektedir. Önerdikleri çözüm yöntemini 18 adet müşteri içeren bir adet örnek problem üzerinde denemişlerdir. Peng ve diğ. (2006) çalışmasında ise Zheng ve Liu (2006) çalışmasında önerilen model ve çözüm yönteminin aynen kullanıldığı ayrıca Zheng ve Liu (2006) çalışmasındaki örnek problemin ilk 18 müşteri için aynı olmak üzere 20 müşterilik bir probleme dönüştürüldüğü ve kapasiteleri farklı araçların kullanıldığı görülmektedir.

Aşağıda, BZAARP-KKS algoritması ile Zheng ve Liu (2006) çalışmasında yer alan örnek problem çözülerek elde edilen sonuçlar karşılaştırılmıştır. Ayrıca Solomon (1987) örnek problemlerindeki yolculuk süreleri bulanık sayılara dönüştürülerek elde edilen uyarlanmış Solomon örnekleri üzerinde çalışılmıştır.

5.4.1.1 Zheng ve Liu (2006) Örneği Sonuçları

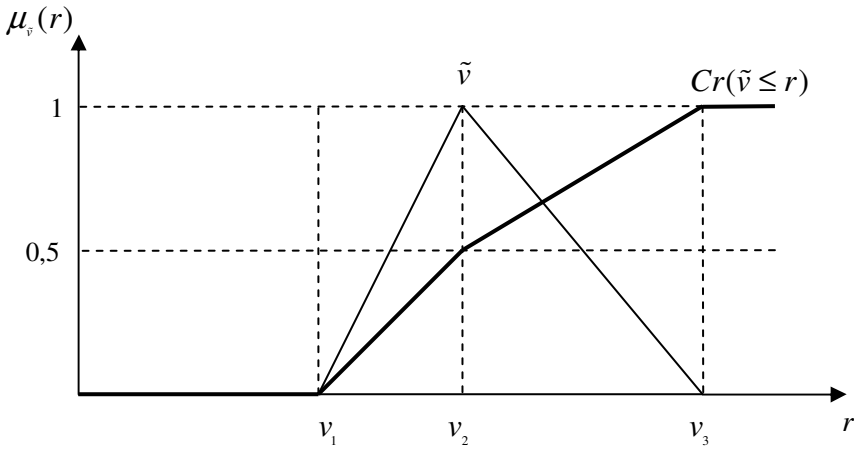
Zheng ve Liu (2006) çalışmasında teslimat zamanlarının müşteri tarafından belirlenen en erken ve en geç teslimat zamanları arasında olma olayının güvenilirliği için Liu ve Liu (2002) tarafından önerilmiş olan kredibilite ölçütü kullanılmıştır. Bir U olayının kredibilitesi olabilirlik ve gereklilik fonksiyonları kullanılarak (5.38) denklemi ile belirlenmektedir.

$$Cr(U) = \frac{1}{2}(\Pi\{U\} + N\{U\}) \qquad \textbf{(5.38)}$$

$\tilde{v} = (v_1, v_2, v_3)$ üçgen bulanık sayısının gerçekleşecek değerinin r gerçel sayısına eşit veya daha küçük olması olayı için kredibilite değeri (5.39) denklemi ile hesaplanmaktadır;

$$Cr(\tilde{v} \le r) = \begin{cases} 0, & r \le v_1 \text{ ise,} \\ \dfrac{a_1 - r}{2(a_1 - a_2)}, & v_1 \le r \le v_2 \text{ ise,} \\ \dfrac{a_3 - 2a_2 + r}{2(a_3 - a_2)}, & v_2 \le r \le v_3 \text{ ise,} \\ 1, & r \ge v_3 \text{ ise.} \end{cases} \quad \textbf{(5.39)}$$

Şekil 5.22'de $Cr(\tilde{v} \le r)$ fonksiyonu gösterilmiştir.



Şekil 5.22: Kredibilite fonksiyonu

Zheng ve Liu (2006) örnek probleminde tüm müşteriler için servis süresi 15 dakika ve kullanılabilir 4 aracın herbirinin kapasitesi 1000 birimdir. Kabul edilebilir en düşük kredibilite değeri 0.90'dır. Amaç en kısa yolculuk mesafeli çözümü bulmaktır. Örnek problemin verileri Tablo 5.4 -Tablo 5.8'de yer almaktadır. Bulanık yolculuk süreleri üçgen bulanık sayılarla tanımlanmıştır. Bu nedenle $\tilde{t}_{ij} = (t_1, t_2, t_3, t_4)$ dörtlü gösterimi yerine $t_2 = t_3$ olmak üzere $\tilde{t}_{ij} = (t_1, t_2 = t_3, t_4)$ üçlü gösterimi ile bulanık sayılar tanımlanmıştır.

Tablo 5.4: Müşteriler arası uzaklıklar (depo:0)

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*
1	19														
2	17,5	6													
3	28	11	10,5												
4	24	21	15	20											
5	24,5	32	26	34	15,5										
6	31,2	44,5	39,5	49	31	16									
7	31	48,5	45	55,5	41,5	28,5	16								
8	21	37,5	33,5	44	30	18,5	13	11,5							
9	18	36	33	44	38	24	20	13,5	7						
10	21,5	40	39	49,5	43	36,5	32,5	21	20	13					
11	36,5	55	54	65	56	46	37	21	28	23	15,5				
12	31,5	46,5	48	57	55,5	51	48,5	36	36	29	16	21,5			
13	23	38,5	39	48,5	47	44	43	32,5	30	23	12	23	9		
14	28	38,5	41,5	49	52	51	52,5	43,5	40	33	22,5	33	13	11	
15	34,5	40	44	50	56,5	58,5	62	54	50	43	34	44,5	24	22	11,5
16	30	29,5	34,5	38	48,5	54	60,5	56	49	43,5	38	51,5	33	28	20,5
17	18,5	16,5	21,3	26,5	35	41	50	48	39	35	33	48	34	27	24
18	24	14	20	22	35	44	54,5	55	45	41,5	41	56,5	43	35	32

Tablo 5.5: Müşteriler arası uzaklıklar (devamı)

	15	*16*	*17*
16	14		
17	23,5	13,5	
18	36	17	8,5

Tablo 5.6: Müşteriler arası yolculuk süreleri

	0	*1*	*2*	*3*	*4*	*5*	*6*
1	(25, 50, 75)						
2	(5, 10, 15)	(20, 40, 60)					
3	(25, 50, 75)	(5, 10, 15)	(20, 40, 60)				
4	(7, 15, 23)	(25, 50, 75)	(7, 15, 23)	(22, 45, 68)			
5	(25, 50, 75)	(17, 35, 53)	(17, 35, 53)	(15, 30, 45)	(17, 35, 53)		
6	(25, 50, 75)	(7, 15, 23)	(20, 40, 60)	(2, 5, 8)	(22, 45, 68)	(15, 30, 45)	
7	(12, 25, 38)	(20, 40, 60)	(15, 30, 45)	(17, 35, 53)	(7, 15, 23)	(12, 25, 38)	(17, 35, 53)
8	(7, 15, 23)	(20, 40, 60)	(5, 10, 15)	(22, 45, 68)	(10, 20, 30)	(17, 35, 53)	(20, 40, 60)
9	(25, 50, 75)	(7, 15, 23)	(22, 45, 68)	(5, 10, 15)	(22, 45, 68)	(15, 30, 45)	(5, 10, 15)
10	(10, 20, 30)	(22, 45, 68)	(12, 25, 38)	(22, 45, 68)	(7, 15, 23)	(15, 30, 45)	(20, 40, 60)
11	(25, 50, 75)	(5, 10, 15)	(17, 35, 53)	(15, 30, 45)	(17, 35, 53)	(5, 10, 15)	(15, 30, 45)
12	(27, 55, 83)	(17, 35, 53)	(17, 35, 53)	(15, 30, 45)	(17, 35, 53)	(2, 5, 8)	(15, 30, 45)
13	(5, 10, 15)	(20, 40, 60)	(5, 10, 15)	(20, 40, 60)	(7, 15, 23)	(15, 30, 45)	(17, 35, 53)
14	(25, 50, 75)	(5, 10, 15)	(20, 40, 60)	(2, 5, 8)	(22, 45, 68)	(15, 30, 45)	(2, 5, 8)
15	(22, 45, 68)	(5, 10, 15)	(20, 40, 60)	(5, 10, 15)	(22, 45, 68)	(15, 30, 45)	(5, 10, 15)
16	(7, 15, 23)	(22, 45, 68)	(7, 15, 23)	(22, 45, 68)	(10, 20, 30)	(15, 30, 45)	(22, 45, 68)
17	(15, 30, 45)	(20, 40, 60)	(12, 25, 38)	(20, 40, 60)	(10, 20, 30)	(12, 25, 38)	(17, 35, 53)
18	(25, 50, 75)	(5, 10, 15)	(22, 45, 68)	(5, 10, 15)	(25, 50, 75)	(15, 30, 45)	(7, 15, 23)

Tablo 5.7: Müşteriler arası yolculuk süreleri (devamı-1)

	7	*8*	*9*	*10*	*11*	*12*	*13*
8	(17, 35, 53)						
9	(20, 40, 60)	(20, 40, 60)					
10	(5, 10, 15)	(12, 25, 38)	(22, 45, 68)				
11	(5, 10, 15)	(17, 35, 53)	(17, 35, 53)	(15, 30, 45)			
12	(7, 15, 23)	(17, 35, 53)	(17, 35, 53)	(12, 25, 38)	(7, 15, 23)		
13	(17, 35, 53)	(5, 10, 15)	(20, 40, 60)	(20, 40, 60)	(17, 35, 53)	(17, 35, 53)	
14	(17, 35, 53)	(17, 35, 53)	(5, 10, 15)	(20, 40, 60)	(15, 30, 45)	(15, 30, 45)	(20, 40, 60)
15	(17, 35, 53)	(17, 35, 53)	(2, 5, 8)	(20, 40, 60)	(15, 30, 45)	(20, 40, 60)	(20, 40, 60)
16	(17, 35, 53)	(10, 20, 30)	(22, 45, 68)	(15, 30, 45)	(17, 35, 53)	(7, 15, 23)	(7, 15, 23)
17	(2, 5, 8)	(12, 25, 38)	(20, 40, 60)	(5, 10, 15)	(12, 25, 38)	(12, 25, 38)	(12, 25, 38)
18	(17, 35, 53)	(20, 40, 60)	(7, 15, 23)	(20, 40, 60)	(15, 30, 45)	(20, 40, 60)	(20, 40, 60)

Tablo 5.8: Müşteriler arası yolculuk süreleri (devamı-2)

	14	*15*	*16*	*17*
15	(2, 5, 8)			
16	(22, 45, 68)	(22, 45, 68)		
17	(17, 35, 53)	(17, 35, 53)	(12, 25, 38)	
18	(7, 15, 23)	(7, 15, 23)	(20, 40, 60)	(20, 40, 60)

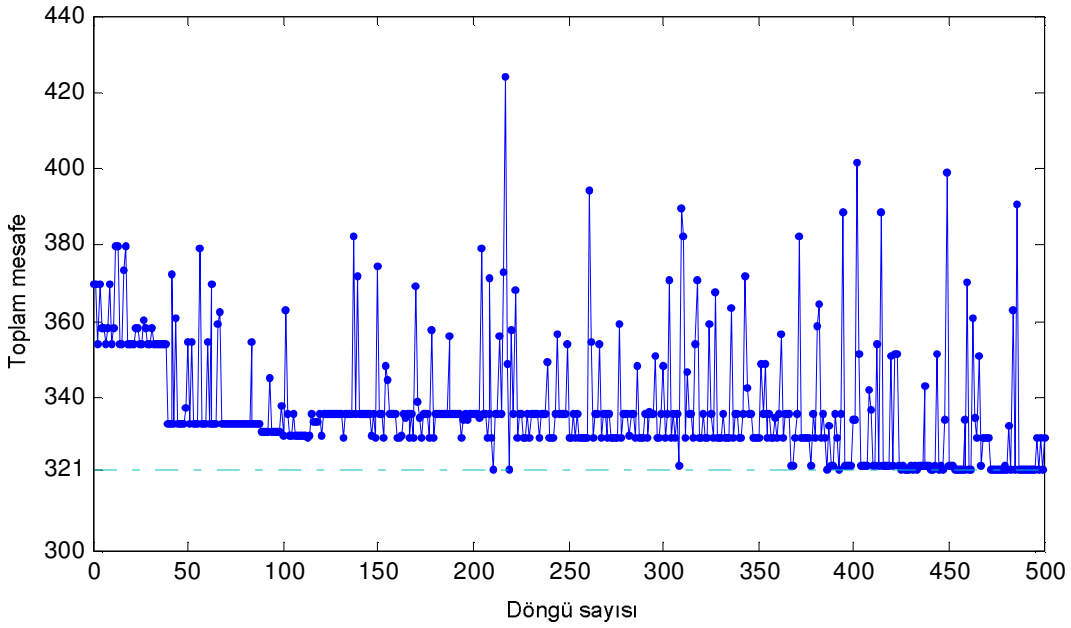
Zheng ve Liu (2006) örnek problemine BZAARP-KKS algoritması ile $N_{eşik} = 0.90$ durumunda elde edilen en iyi çözüm Tablo 5.9'da yer almaktadır. Yolculuk süreleri üçgen bulanık sayılarla tanımlandığından, varış zamanları ve teslimat başlangıç zamanları da üçgen bulanık sayı olarak elde edilmiştir. Son üç sütunda elde edilen çözümdeki her bir araç rotasının toplam müşteri talebi, toplam tur uzunluğu ve tur bitiş zamanı yer almaktadır. Elde edilen çözümün gereklilik ve kredibilite değerleri 1'dir.

BZAARP-KKS algoritması ile 18 müşterilik problem için bir döngü süresi 0.18 saniyedir. Bu durumda BZAARP-KKS algoritmasının Zheng ve Liu (2006) örnek problemine çözüm bulma süresi yaklaşık 90 saniye olarak hesaplanabilir. 500 döngü içeren 5 deneme yapılmış tüm denemelerde Tablo 5.9'da verilen çözüm elde edilmiştir.

Tablo 5.9: Zheng ve Liu (2006) örneği için bulunan en iyi çözüm

g	i	$\tilde{v}_i^g$	$[a_i, b_i]$	$\tilde{h}_i^g$	q_i	Toplam Talep	Tur Uz.	Tur Zamanı
1	2	(5, 10, 15)	[20, 390]	(20, 20, 20)	100	800	94,5	(184, 275, 366)
	1	(55, 75, 95)	[0, 410]	(55, 75, 95)	200			
	3	(75, 100, 125)	[40, 340]	(75, 100, 125)	140			
	4	(112, 160, 208)	[20, 330]	(112, 160, 208)	160			
	5	(144, 210, 276)	[0, 380]	(144, 210, 276)	200			
2	10	(10, 20, 30)	[40, 260]	(40, 40, 40)	140	755	112	(232, 335, 435)
	11	(70, 85, 100)	[10, 320]	(70, 85, 100)	100			
	7	(90, 110, 130)	[30, 310]	(90, 110, 130)	200			
	6	(122, 160, 198)	[0, 320]	(122, 160, 198)	60			
	8	(157, 215, 273)	[0, 390]	(157, 215, 273)	135			
	9	(192, 270, 348)	[10, 410]	(192, 270, 348)	120			
3	17	(15, 30, 45)	[0, 440]	(15, 30, 45)	200	930	114,5	(221, 340, 459)
	18	(50, 85, 120)	[0, 440]	(50, 85, 120)	100			
	16	(85, 140, 195)	[0, 370]	(85, 140, 195)	90			
	15	(122, 200, 278)	[20, 330]	(122, 200, 278)	200			
	14	(139, 220, 301)	[20, 420]	(139, 220, 301)	60			
	12	(169, 265, 361)	[0, 380]	(169, 265, 361)	200			
	13	(201, 315, 429)	[20, 450]	(201, 315, 429)	80			
					Toplam	2485	321	

500 döngülük bir arama sürecinde, her döngüde karıncalar tarafından bulunan en kısa yolculuk mesafesini için oluşturulan grafik Şekil 5.23'te gösterilmiştir.



Şekil 5.23: Arama süreci

Zheng ve Liu (2006) çalışmasında örnek problem için bulunan en iyi çözüm aşağıdaki gibi verilmiştir:

Araç 1 : depo → 16 → 17 → 18 → 5 → depo, yük: 590, mesafe: 100,5

Araç 2 : depo → 10 → 12 → 13 → 14 → 15 → 8 → depo, yük: 815, mesafe: 124,5

Araç 3 : depo → 1 → 2 → 3 → depo, yük: 440, mesafe: 63,5

Araç 4 : depo → 9 → 6 → 7 → 11 → 4 → depo, yük: 640, mesafe: 133,5

Toplam mesafe : 422 , çözümün kredibilitesi : 0,94

Yukarıdaki çözüm Zheng ve Liu (2006) tarafından 10.000 simülasyon döngüsü ve 5000 genetik algoritma jenerasyonu ile 2000MHz Pentium 4 işlemcili ve 256 MB RAM'li bir bilgisayar ile yaklaşık 10 saat sürede elde edilmiştir. Toplam çözüm süresinin büyük bir kısmının simülasyon sürecinde geçtiği belirtilmiştir. Simülasyonun bir en iyileme aracı olarak kullanılması önerilen yöntemin çözüm süresini oldukça uzatmakta, daha büyük boyutlu problemlere uygulamaya elverişli sürelerde çözüm bulunamamaktadır.

Zheng ve Liu (2006) tarafından oluşturulan çözüm için araçların yolculuk mesafeleri hesaplandığında aşağıdaki değerler elde edilmektedir.

Araç 1 rotası : 30 + 13,5 + 8,5 + 44 + 24,5 = 120,5

Araç 2 rotası : 21,5 + 16 + 9 + 11 + 11,5 + 50 + 21 = 140

Araç 3 rotası : 19 + 6 + 10,5 + 28 = 63,5

Araç 4 rotası : 18 + 20 + 16 + 21 + 56 + 24 = 155

Toplam mesafe : 479

Rota mesafelerinin hesaplanmasındaki tutarsızlıklar ve BZAARP-KKS algoritması ile bulunan daha iyi çözüm, Zheng ve Liu (2006) çalışmasının yazarlarından Boading Liu ile yazışma yapılarak doğrulanmıştır. Zheng ve Liu (2006) çalışmasında elde edilen sonuç ile bu çalışmada önerilen karınca algoritması ile elde edilen sonuçlar karşılaştırıldığında, önerilen karınca algoritmasının hem elde edilen çözümün kalitesi (araç sayısı ve toplam mesafe açısından) hem de çözüm elde etme süresi yönünden oldukça üstün bir performansa sahip olduğu açıktır.

5.4.1.2 Uyarlanmış Solomon Örnekleri

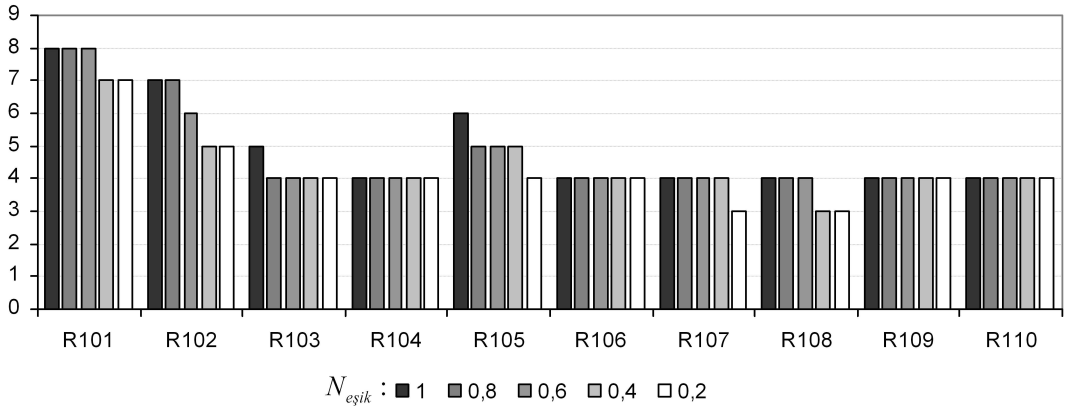
Bu kısımda Solomon (1987) örnek problemlerindeki kesin yolculuk sürelerinin üçgen bulanık sayılara dönüştürülmesiyle olabilirlik programlama modeli için oluşturulan örnek problemler üzerinde çalışılmıştır. Solomon örneklerindeki kesin t_{ij} yolculuk sürelerinin $0{,}30\times t_{ij}$ ile t_{ij} değerleri arasında değişebileceği ve gerçekleşmesi en makul değerin $0{,}50\times t_{ij}$ olduğu varsayılarak t_{ij} kesin değerleri $\tilde{t}_{ij} = (0{,}3\times t_{ij},\ 0{,}5\times t_{ij},\ t_{ij})$ üçgen bulanık sayılarına dönüştürülerek bulanık yolculuk süreli Solomon örnekleri oluşturulmuştur. Tablo 5.10 ve Tablo 5.11'de R1 ve RC1 grubundaki 25 müşterilik problemler için farklı gereklilik eşik değerleri, $N_{\text{eşik}} = \{1,\ 0{,}8,\ 0{,}6,\ 0{,}4,\ 0{,}2\}$ için BZAARP-KKS algoritması ile elde edilen çözümler yer almaktadır. $N_{\text{eşik}} = 1$ durumunda elde edilen bir çözümde, bulanık varış zamanının gerçekleşecek değeri, müşteriye en geç teslimat başlangıç zamanından hiçbir zaman büyük olamayacağından problem belirlilik durumu için de çözülmüş olmaktadır. Her durum için 500 döngülük üçer koşum gerçekleştirilmiş, elde edilen en iyi çözümler kullanılmıştır. Örnek olarak, R101 probleminde $N_{eşik} = 0{,}6$ durumunda 8 araç kullanan, toplam yolculuk mesafesi 618,33 birim olan ve çözümdeki en düşük gereklilik ($N(\tilde{v}_i^g < b_i)$) değeri 1 olan bir çözüm elde edilmiş, $N_{eşik} = 0{,}4$ durumunda ise 7 araç kullanan, toplam yolculuk mesafesi 609,66 birim olan ve en düşük gereklilik değeri 0,43 olan bir çözüm elde edilmiştir. Tablo 5.10 ve Tablo 5.11'in son

satırları elde edilen çözümlerdeki toplam araç sayıları ve toplam yolculuk mesafelerinden oluşmaktadır.

Tablo 5.10: R1 grubu örnekler için elde edilen sonuçlar

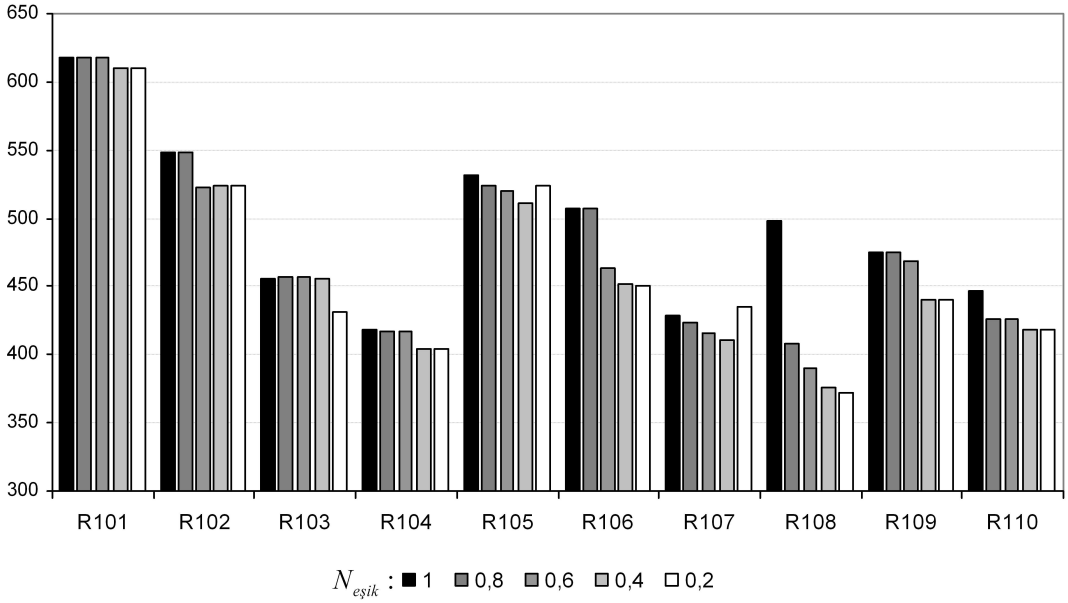
Problem	$N_{eşik}$	1	0,8	0,6	0,4	0,2
R101	Araç sayısı	8	8	8	7	7
	Mesafe	618,33	618,33	618,33	609,66	609,66
	Gereklilik	1	1	1	0,43	0,43
R102	Araç sayısı	7	7	6	5	5
	Mesafe	548,11	548,11	522,2	523,75	523,75
	Gereklilik	1	1	0,73	0,42	0,42
R103	Araç sayısı	5	4	4	4	4
	Mesafe	455,70	457,21	457,21	456,05	431,77
	Gereklilik	1	0,83	0,83	0,45	0,26
R104	Araç sayısı	4	4	4	4	4
	Mesafe	417,96	417,27	417,27	404,8	404,8
	Gereklilik	1	0,81	0,81	0,45	0,45
R105	Araç sayısı	6	5	5	5	4
	Mesafe	531,54	524,12	519,41	510,77	523,54
	Gereklilik	1	0,94	0,62	0,47	0,22
R106	Araç sayısı	4	4	4	4	4
	Mesafe	507,12	507,12	462,8	452,23	450,86
	Gereklilik	1	1	0,63	0,42	0,21
R107	Araç sayısı	4	4	4	4	3
	Mesafe	429,2	424,07	415,27	410,66	434,73
	Gereklilik	1	0,88	0,63	0,53	0,39
R108	Araç sayısı	4	4	4	3	3
	Mesafe	398,29	398,29	390,28	376,09	371,84
	Gereklilik	1	1	0,77	0,51	0,26
R109	Araç sayısı	4	4	4	4	4
	Mesafe	474,5	474,5	468,64	440,22	440,22
	Gereklilik	1	1	0,78	0,40	0,40
R110	Araç sayısı	4	4	4	4	4
	Mesafe	446,87	426,06	426,06	418,28	418,28
	Gereklilik	1	0,84	0,84	0,45	0,45
Toplam	Araç sayısı	50	48	47	44	42
	Mesafe	4827,62	4795,08	4697,47	4602,51	4609,45

R1 grubu problemlerde farklı $N_{eşik}$ değerleri için kullanılan araç sayıları ve yolculuk mesafesindeki değişimler Şekil 5.24 ve Şekil 5.25'te gösterilmiştir. Görüldüğü üzere $N_{eşik}$ değeri azaldıkça yani müşterilere en geç zamanında teslimat yapabilme konusundaki katlanılabilir risk seviyesi arttıkça daha düşük maliyetli çözümler elde edilebilmektedir.



Şekil 5.24: Farklı $N_{\text{eşik}}$ değerleri için elde edilen çözümlerdeki araç sayıları

Şekil 5.25'de daha düşük $N_{\text{eşik}}$ seviyesinde daha uzun yolculuk mesafeli çözümlerin bulunduğu görülmektedir. Bu tip durumların azalan araç sayısından kaynaklandığı Tablo 5.10'dan kontrol edilebilir.

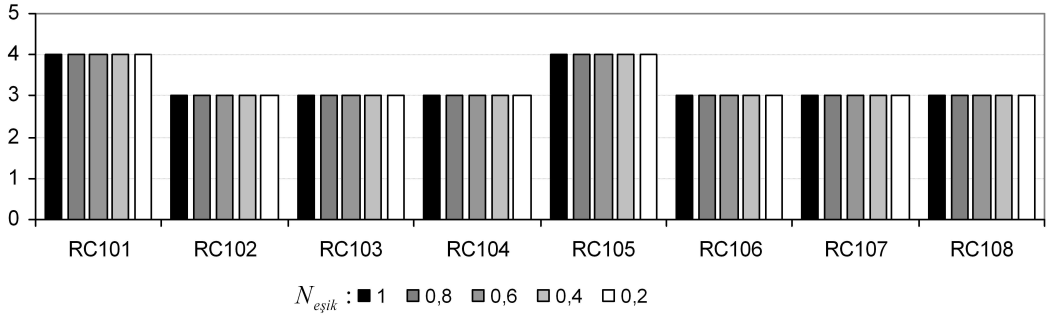


Şekil 5.25: Farklı $N_{\text{eşik}}$ değerleri için elde edilen çözümlerin yolculuk mesafeleri

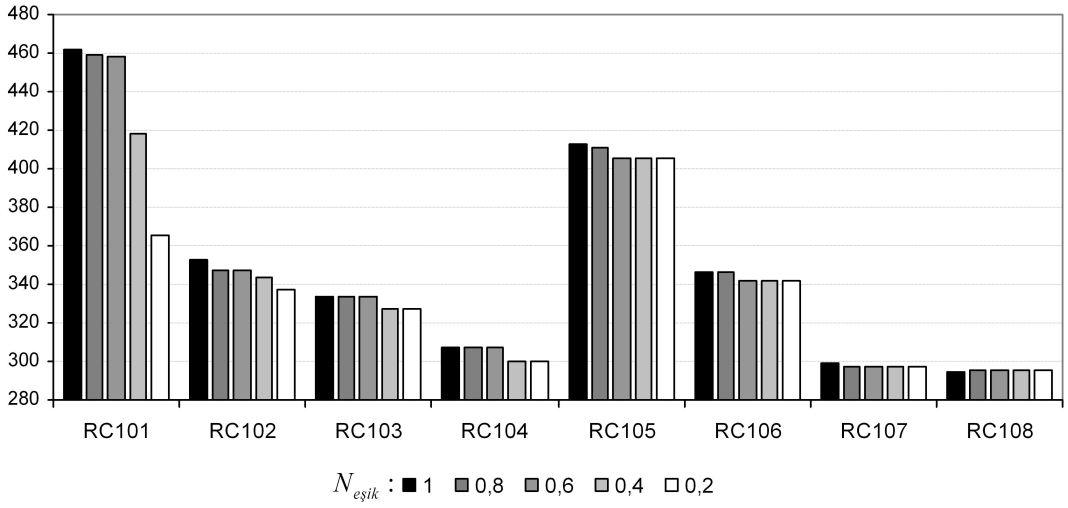
Tablo 5.11: RC1 grubu örnekler için elde edilen sonuçlar

Problem	$N_{eşik}$	1	0,8	0,6	0.4	0.2
RC101	Araç sayısı	4	4	4	4	3
	Mesafe	462,16	459,38	457,76	418,10	365,23
	Gereklilik	1	0,92	0,67	0,44	0,23
RC102	Araç sayısı	3	3	3	3	3
	Mesafe	352,74	346,96	346,96	343,37	337,57
	Gereklilik	1	0,85	0,85	0,58	0,23
RC103	Araç sayısı	3	3	3	3	3
	Mesafe	333,92	333,92	333,92	327,69	327,69
	Gereklilik	1	1	1	0,50	0,50
RC104	Araç sayısı	3	3	3	3	3
	Mesafe	307,14	307,14	307,14	300,23	300,23
	Gereklilik	1	1	1	0,46	0,46
RC105	Araç sayısı	4	4	4	4	4
	Mesafe	412,38	411,07	405,42	405,42	405,42
	Gereklilik	1	0,82	0,67	0,67	0,67
RC106	Araç sayısı	3	3	3	3	3
	Mesafe	346,51	346,51	341,98	341,98	341,98
	Gereklilik	1	1	0,66	0,66	0,66
RC107	Araç sayısı	3	3	3	3	3
	Mesafe	298,95	296,83	296,83	296,83	296,83
	Gereklilik	1	0,87	0,87	0,87	0,87
RC108	Araç sayısı	3	3	3	3	3
	Mesafe	294,99	295,74	295,74	295,74	295,74
	Gereklilik	1	1	1	1	1
Toplam	Araç sayısı	26	26	26	26	25
	Mesafe	2808,79	2797,55	2785,75	2729,36	2670,69

RC1 grubu problemler için farklı $N_{eşik}$ değerleri için kullanılan araç sayıları ve yolculuk mesafelerindeki ve değişimler Şekil 5.26 ve Şekil 5.27'da gösterilmiştir.

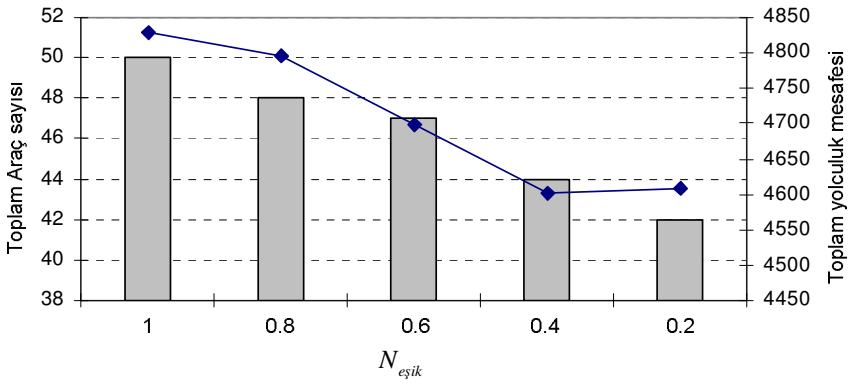


Şekil 5.26: Farklı $N_{eşik}$ değerleri için elde edilen çözümlerdeki araç sayıları

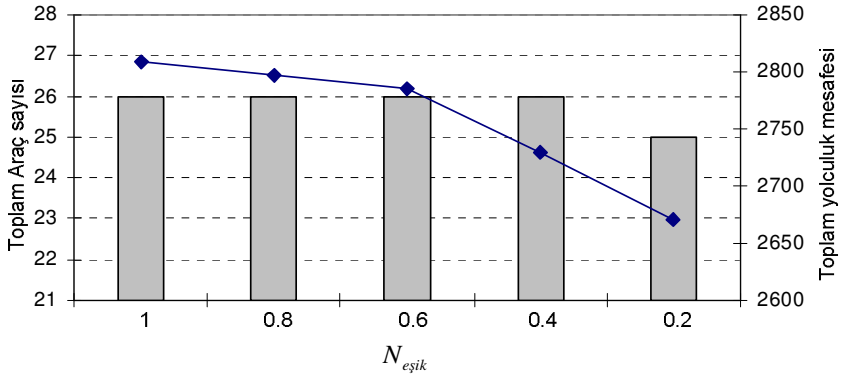


Şekil 5.27: Farklı $N_{\text{eşik}}$ değerleri için elde edilen çözümlerin yolculuk mesafeleri

Şekil 5.28 ve Şekil 5.29'da R1 ve RC1 gruplarında yer alan problemlere farklı $N_{eşik}$ değerlerinde BZAARP-KKS algoritması ile elde edilen çözümlerin toplam araç sayıları ve toplam yolculuk mesafelerini içeren grafikler yer almaktadır. Karşılaştırma yapıldığında, R1 grubu problemler için elde edilen çözümlerin $N_{eşik}$ değerindeki değişimlerden daha fazla etkilendiği görülmektedir. R1 grubunda müşteriler coğrafi alana rassal bir şekilde yerleştirilmiştir. RC1 grubunda ise müşteriler coğrafi alanda kümeler halinde bulunmaktadır. R1 grubundaki problemlerde zaman aralıkları kısıtı uygun çözümler uzayını daha çok daralttığından, $N_{eşik}$ değerinin azaltılması ile elde edilen maliyet kazanımları daha fazla olmaktadır.

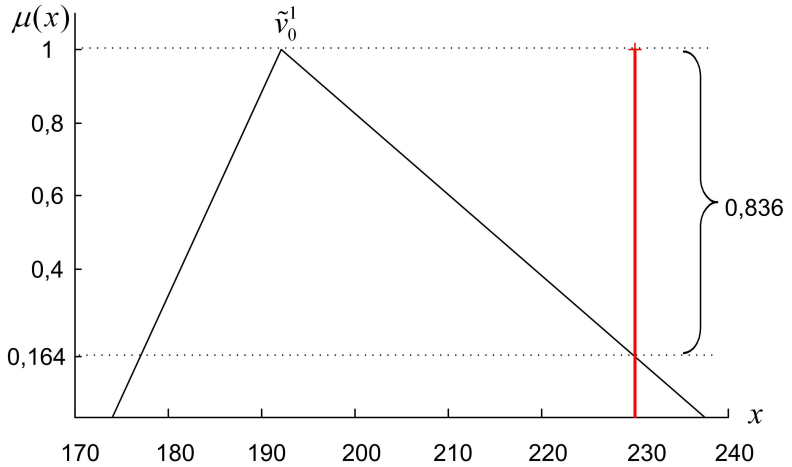


Şekil 5.28: R1 grubu problemler için toplam araç sayısı ve toplam yolculuk mesafeleri



Şekil 5.29: RC1 grubu problemler için toplam araç sayısı ve toplam yolculuk mesafeleri

Tablo 5.12'de, daha önce belirlilik durumundaki detaylı çözümü dördüncü bölümde Tablo 4.3'te verilmiş olan R110 problemine, $N_{eşik} = 0,80$ durumunda elde edilen en iyi çözümün detayları yer almaktadır. BZAARP-KKS algoritması ile 4 araç kullanan ve toplam mesafesi 426,06 olan bir çözüm bulunmuştur. Tablo 5.12'de görüldüğü gibi elde edilen çözümün gereklilik değerini, 1 nolu aracın depo kapanış zamanından önce depoya dönmüş olması olayının gerekliliği belirlemektedir. Şekil 5.30'da 1 nolu araç için $N(\tilde{v}_0^1 < b_0)$ değerinin bulunması gösterilmiştir.



Şekil 5.30: $N(\tilde{v}_0^1 < b_0)$ değerinin gösterimi

Depo kapanış saatinin dağıtım firması tarafından yaklaşık olarak 8 dakika geciktirilmesi durumunda 1 nolu araç için gereklilik değeri 1 olacaktır. Bu durumda 2 nolu aracın 20 nolu müşteriye 152. dakikadan önce teslimata başlayabilme olayının

gerekliliği olan 0,986 değeri, çözümün gereklilik değerini belirleyecektir. Ancak 1 nolu aracın 20 nolu müşteriye 152. dakikadan sonra varış yapması olayının gerçekleşebilirliği ancak en uç düzeydeki kötümserlik şartlarında mümkündür. Yani, kullanıcı bu riski rahatlıkla göz ardı edebilir veya 20 nolu müşteri ile görüşerek 0.986 gereklilik (kesinlik) düzeyinde en geç 152. dakikada teslimata başlanmış olacağını ve en kötü şartlarda en fazla 11 saniyelik bir gecikme yaşanabileceği konusunda bilgi verilip onayı alınarak oluşturulan çözüm uygulayabilir. Bu durumda R110 problemine kesin çözüm yöntemleri ile bulunan optimal sonuç olan 444,1 toplam mesafe değerine göre yaklaşık olarak %4'lük bir iyileşme sağlanabilir.

Tablo 5.12: R110 örneğine $N_{eşik} = 0,80$ durumu için elde edilen en iyi çözüm

g	Rota	$\tilde{v}_i^g$	$[a_i, b_i]$	$\tilde{h}_i$	$N(\tilde{v}_i^g < b_i)$
1	0	0	[0, 230]	0	
	21	(5,41, 9,01, 18,03)	[37, 96]	(37, 37, 37)	1
	2	(50,13, 52,22, 57,44)	[20, 89]	(50,13, 52,22, 57,44)	1
	15	(64,03, 68,72, 80,44)	[30, 154]	(64,03, 68,72, 80,44)	1
	22	(78,78, 86,63, 106,25)	[59, 144]	(78,78, 86,63, 106,25)	1
	23	(92,13, 102,22, 127,43)	[36, 155]	(92,13, 102,22, 127,43)	1
	4	(106,63, 119,72, 152,43)	[71, 195]	(106,63, 119,72, 152,43)	1
	25	(119,63, 134,72, 172,43)	[47, 186]	(119,63, 134,72, 172,43)	1
	24	(134,13, 152,22, 197,43)	[118, 190]	(134,13, 152,22, 197,43)	1
	0	**(174.13, 192,22, 237,43)**	**[0, 230]**	**(174,13, 192,22, 237,43)**	**0,836**
2	0	0	[0, 230]	0	
	12	(4,5, 7,5, 15)	[38, 97]	(38, 38, 38)	1
	3	(51,35, 53,59, 59,18)	[106, 135]	(106, 106, 106)	1
	9	(120,5, 123,5, 131)	[53, 150]	(120,5, 123,5, 131)	1
	20	**(133,85, 139,09, 152,18)**	**[109, 152]**	**(133,85, 139,09, 152,18)**	**0,986**
	1	(148,8, 157,34, 178,67)	[130, 201]	(148,8, 157,34, 178,67)	1
	0	(174,03, 182,57, 203,90)	[0, 230]	(174,03, 182,57, 203,90)	1
3	0	0	[0, 230]	0	
	5	(6,19, 10,31, 20,62)	[20, 107]	(20, 20, 20,62)	1
	8	(34,18, 36,96, 44,54)	[61, 138]	(61, 61, 61)	1
	7	(74,66, 77,10, 83,21)	[66, 105]	(74,66, 77,10, 83,21)	1
	19	(88,02, 92,69, 104,39)	[53, 108]	(88,02, 92,69, 104.39)	1
	11	(100,14, 106,23, 121,46)	[33, 152]	(100,14, 106,23, 121.46)	1
	10	(113,49, 121,82, 142,64)	[101, 156]	(113,49, 121,82, 142.64)	1
	0	(148,99, 157,31, 178,13)	[0, 230]	(148,99, 157,31, 178.13)	1
4	0	0	[0, 230]	0	
	14	(9,61, 16,01, 32,02)	[32, 137]	(32, 32, 32,02)	1
	16	(45,35, 47,59, 53,20)	[54, 105]	(54, 54, 54)	1
	17	(67,35, 69,59, 75,18)	[51, 189]	(67,35, 69,59, 75,18)	1
	18	(82,76, 88,60, 103,21)	[77, 106]	(82,76, 88,60, 103,21)	1
	6	(96,12, 104,19, 124,39)	[54, 153]	(96,12, 104,19, 124,39)	1
	13	(108,24, 117,73, 141,46)	[70, 208]	(108,24, 117,73, 141,46)	1
	0	(121,59, 133,32, 162,64)	[0, 230]	(121,59, 133,32, 162,64)	1

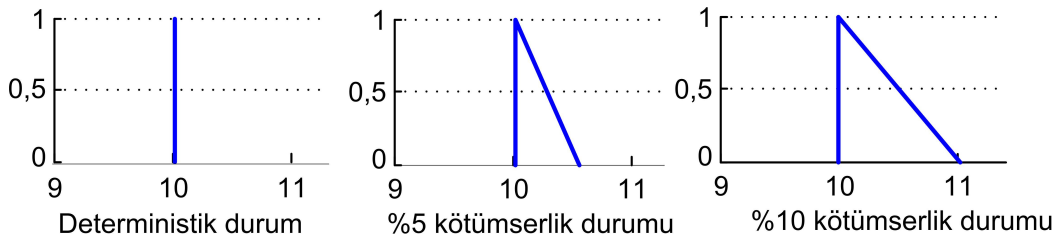
Müşteriler arasındaki yolculuk sürelerini kesin olarak bilinen değerler ile modelleyen yöntemler ile elde edilen çözümler, uygulama öncesinde planlayıcılara çok fazla güven vermemektedir. Çünkü, tüm yolculuk sürelerinin çözüm oluşturma esnasında belirlenen kesin değerlerde gerçekleşmesi (çok özel uygulama alanları haricinde) mümkün değildir. Aşağıda, yolculuk sürelerinin kesin olarak modellendiği geçmiş çalışmalarda elde edilen çözümlerin uygulama aşamasında zaman aralıkları kısıtlarına uygun olup olmayacağının gereklilik düzeyleri incelenmiştir.

100 müşterilik R103 (Solomon, 1987) örneği için bilinen en iyi sonuç Li ve Lim (2003) tarafından oluşturulmuştur. Çözümde 13 araç kullanılmakta ve toplam mesafe 1292,68'dir (Tablo 5.13).

Tablo 5.13: R103 örneği için sezgisel yöntemlerle bulunmuş en iyi çözüm

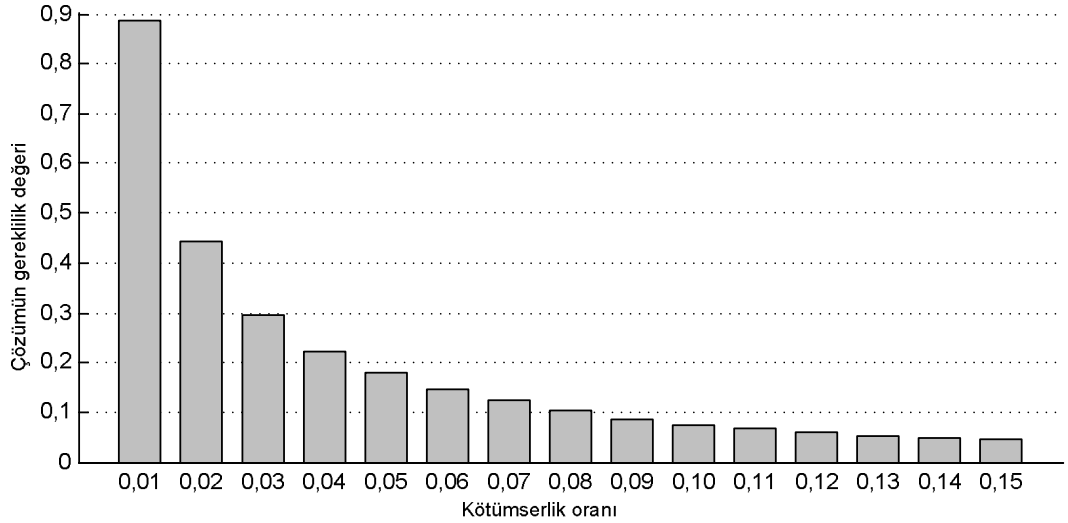
Araç	Rota													
1	0	40	53	12	68	80	0							
2	0	92	98	14	44	38	86	16	61	85	91	100	37	0
3	0	36	64	49	63	90	32	70	0					
4	0	71	65	78	34	35	81	77	28	0				
5	0	50	33	76	79	10	31	0						
6	0	94	96	95	97	87	13	0						
7	0	7	19	11	8	46	47	48	82	18	89	0		
8	0	52	62	88	84	17	93	59	0					
9	0	60	45	83	5	99	6	0						
10	0	2	22	75	56	4	25	54	0					
11	0	26	39	23	67	55	24	29	3	0				
12	0	27	69	30	9	66	20	51	1	0				
13	0	42	43	15	57	41	74	72	73	21	58	0		

Li ve Lim (2003) tarafından oluşturulan çözümde tüm yolculuk süreleri kesin sayılar ile modellenmiştir. Örneğin Tablo 5.13'deki çözümde 10 nolu araç rotasında yer alan 4 ve 25 nolu müşteriler arasındaki yolculuk süresi 10 birimdir. Bu çözümün uygulanması durumunda 4 ve 25 nolu müşteriler arasındaki yolculuk süresinin daha fazla olabileceği konusunda planlayıcının şüpheleri varsa çözümün en geç teslimat zamanı kısıtına uygun olup olmayacağı konusunda kullanıcıya bilgi sağlanabilir. Şöyle ki, planlayıcı 4 ve 25 nolu müşteriler arasındaki yolculuk süresinin en kötü şartlarda kullanılan değerden %5 daha fazla olabileceğinden şüpheli ise 10 kesin sayısı destek aralığı [10 10x(1,05)] olan dik üçgen şekilli bir bulanık sayıya dönüştürülebilir. %5 ve %10'luk kötümserlik oranları için 10 kesin sayısının dönüştürüldüğü bulanık sayılar Şekil 5.31'de gösterilmiştir.



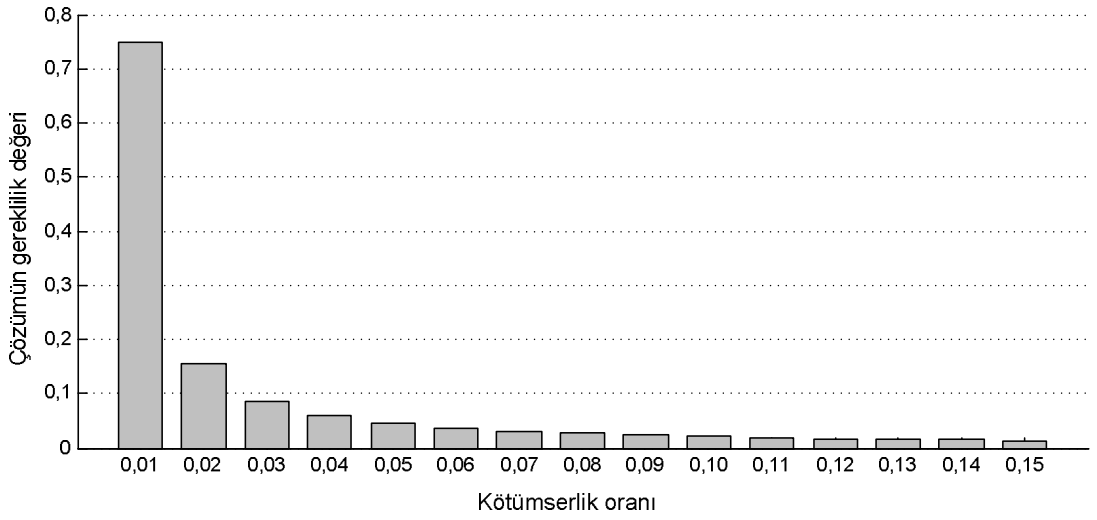
Şekil 5.31: Kesin yolculuk süresinin bulanıklaştırılması.

Şekil 5.32'te {0,01, 0,02, …, 0,15} kötümserlik oranları için R103 problemindeki kesin yolculuk sürelerinin bulanık yolculuk sürelerine dönüştürülmesi durumunda Li ve Lim (2003) tarafından oluşturulan araç rotalarını içeren çözümler için hesaplanan gereklilik ölçütü değerleri grafiği yer almaktadır. Görüldüğü üzere, yolculuk sürelerinde yüzde birlik bir artış durumu, çözümün gereklilik ölçütü değerini 0,89'a düşürmekte, yüzde ikilik bir şüphe durumunda ise gereklilik ölçütü değeri 0,42'ye düşmektedir. Önerilen model ile deterministik durumda üretilen çözümler hakkında kullanıcıya daha fazla bilgi sunulabilmektedir.



Şekil 5.32: R103 problemi bilinen en iyi çözümü gereklilik değerinin bulanık yolculuk süreleri durumundaki değişimi

R207 örnek problemi için bilinen en iyi çözümün (Psinger ve Ropke, 2007), farklı kötümserlik oranlarındaki gereklilik ölçütü değerleri için oluşturulan grafik Şekil 5.33'te yer almaktadır.



Şekil 5.33: R207 problemi bilinen en iyi çözümü gereklilik değerinin bulanık yolculuk süreleri durumundaki değişimi

Şekil 5.32 ve Şekil 5.33'te yer alan grafikler, yolculuk sürelerinin ait oldukları çözümlerde belirsizlik içermesi durumunda çözümlerin geçerliliklerini koruyup koruyamayacakları hakkında kullanıcıya bilgi sağlamaktadırlar.

Yukarıda üretilen bilgiler, aynı yolculuk mesafesi değerine sahip farklı çözümlerden hangisinin yolculuk sürelerindeki olası değişimlere daha dayanıklı olabileceğinin belirlenmesinde kullanılabilir.

5.4.2 Esnek Programlama Modeli için Deneyler

Yolculuk sürelerinin kesin olduğu, müşteriler tarafından belirlenen en geç teslimat zamanlarının ise esneklik içerdiği esnek programlama modelinin kullanımı ile elde edilebilecek kazanımları görebilmek için, Solomon (1987) problemlerindeki en geç teslimat zamanları belirli düzeylerde esnek hale getirilerek elde edilen uyarlanmış problemler üzerinde çalışılmıştır. Problemlerdeki en geç teslimat zamanları %30 oranında arttırılarak tüm müşteriler için müşteri tatmin düzeyini temsil eden bulanık kümeler oluşturulmuştur. Örneğin, orijinal durumda R101 örnek probleminde 1 nolu müşteri için en geç teslimat zamanı 171'dir. Problemin esnek programlama modeli için uyarlanması durumunda 1 nolu müşteri için tatmin düzeyini temsil eden bulanık kümenin üyelik fonksiyonu aşağıdaki Şekil 5.34'te gösterilmiştir.



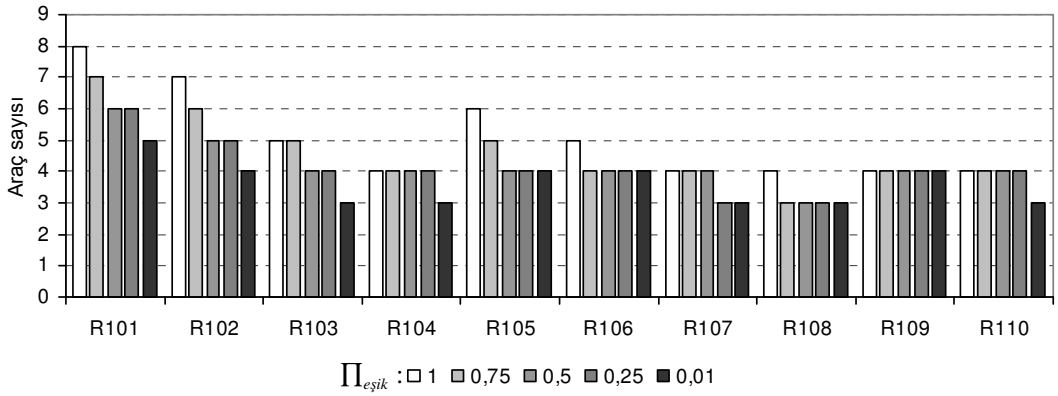
Şekil 5.34: Müşteri tatmin düzeyini temsil eden bulanık küme

Solomon (1987) problemlerinden R1 ve RC1 grubunda bulunan problemler için deneyler yapılıştır. Bu gruplarda bulunan problemlerin araç kapasiteleri düşük, teslimat zaman aralıkları ise dardır. Tablo 5.14 ve Tablo 5.15'te R1 ve RC1 grubundaki 25 müşterilik problemler için farklı müşteri tatmin düzeyi eşik değerleri, $\prod_{eşik} = \{0{,}01,\ 0{,}25,\ 0{,}50,\ 0{,}75,\ 1\}$ için BZAARP-KKS algoritması ile elde edilen çözümler yer almaktadır. Her durum için 500 döngülük üçer koşum gerçekleştirilmiş, elde edilen en iyi çözümler kullanılmıştır. Örnek olarak R101 probleminde $\prod_{eşik} = 0{,}75$ durumunda 7 araç kullanan, yolculuk mesafesi 601,52 birim olan ve çözümdeki en düşük müşteri tatmin düzeyi ($\prod(v_i^g \le \tilde{b}_i)$) 0,77 olan bir çözüm elde edilmiş, $\prod_{eşik} = 0{,}50$ durumunda ise 6 araç kullanan, yolculuk mesafesi 575,57 birim olan ve en düşük müşteri tatmin düzeyi 0,54 olan bir çözüm elde edilmiştir. Tablo 5.14 ve Tablo 5.15'in son satırları elde edilen çözümlerdeki toplam araç sayıları ve toplam yolculuk mesafelerinden oluşmaktadır.

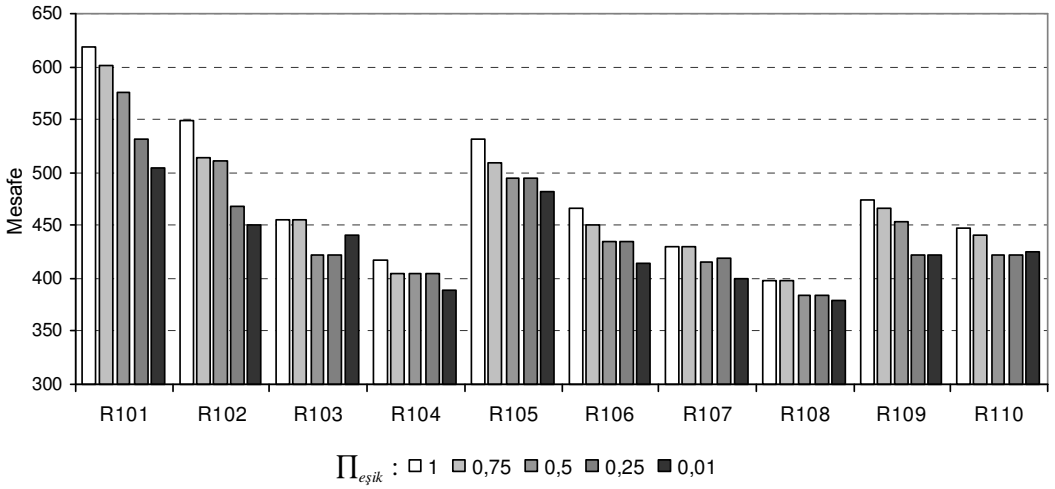
Tablo 5.14: R1 grubu örnekler için elde edilen sonuçlar

Problem	$\Pi_{eşik}$	1,00	0,75	0,50	0,25	0,01
R101	Araç sayısı	8	7	6	6	5
	Mesafe	618,33	601,52	575,57	531,22	504,92
	Müşteri tatmin düzeyi	1	0,77	0,54	0,30	0,13
R102	Araç sayısı	7	6	5	5	4
	Mesafe	548,11	513,97	511,34	468,36	450,71
	Müşteri tatmin düzeyi	1	0,81	0,54	0,30	0,05
R103	Araç sayısı	5	5	4	4	3
	Mesafe	455,70	455,70	421,45	421,45	440,75
	Müşteri tatmin düzeyi	1	1	0,69	0,69	0,19
R104	Araç sayısı	4	4	4	4	3
	Mesafe	417,96	404,11	404,11	404,11	388,99
	Müşteri tatmin düzeyi	1	0,78	0,78	0,78	0,07
R105	Araç sayısı	6	5	4	4	4
	Mesafe	531,54	509,09	495	501,87	481,82
	Müşteri tatmin düzeyi	1	0,79	0,56	0,26	0,06
R106	Araç sayısı	5	4	4	4	4
	Mesafe	466,48	451,23	434,98	434,98	414,41
	Müşteri tatmin düzeyi	1	0,79	0,63	0,63	0,20
R107	Araç sayısı	4	4	4	3	3
	Mesafe	429,2	429,2	415,24	418,44	400,31
	Müşteri tatmin düzeyi	1	1	0,74	0,47	0,08
R108	Araç sayısı	4	3	3	3	3
	Mesafe	398,29	398,95	384,48	384,48	379,71
	Müşteri tatmin düzeyi	1	0,75	0,60	0,60	0,20
R109	Araç sayısı	4	4	4	4	4
	Mesafe	474,5	465,94	453,66	422,49	422,49
	Müşteri tatmin düzeyi	1	0,81	0,77	0,28	0,28
R110	Araç sayısı	4	4	4	4	3
	Mesafe	446,87	441,21	422,68	422,68	424,89
	Müşteri tatmin düzeyi	1	0,81	0,50	0,50	0,04
Toplam	Araç Sayısı	51	46	42	41	36
	Mesafe	4786,98	4670,92	4518,51	4410,08	4309

R1 grubu problemlerde farklı $\Pi_{eşik}$ değerlerinde kullanılan araç sayıları ve yolculuk mesafelerindeki değişimler Şekil 5.35 ve Şekil 5.36'da gösterilmiştir.



Şekil 5.35: Farklı $\Pi_{eşik}$ değerleri için araç sayıları

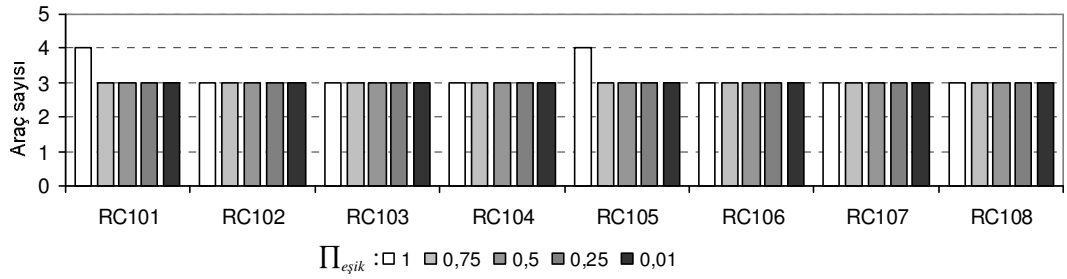


Şekil 5.36: Farklı $\prod_{eşik}$ değerleri için yolculuk mesafeleri

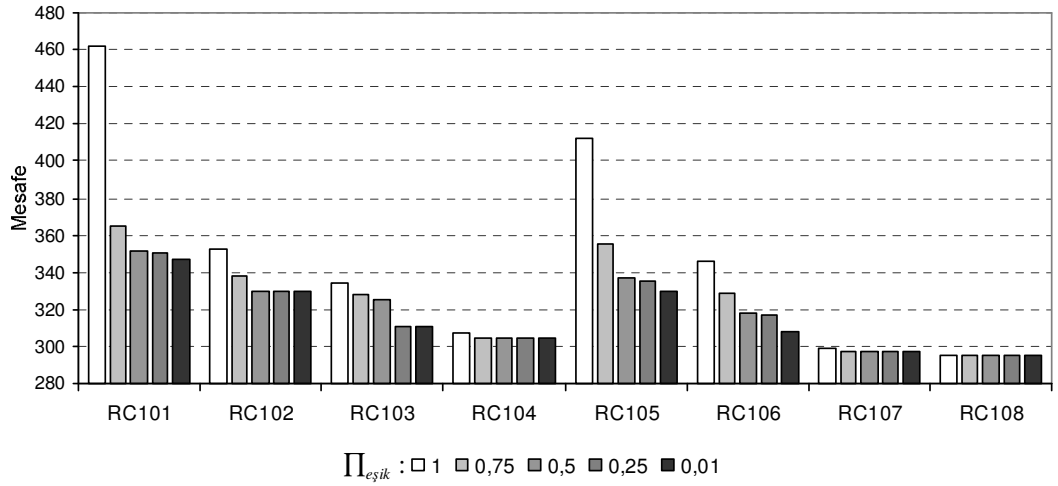
Tablo 5.15: RC1 grubu örnekler için elde edilen sonuçlar

Problem	$\prod_{eşik}$	1,00	0,75	0,50	0,25	0,01
RC101	Araç sayısı	4	3	3	3	3
	Mesafe	462,16	365,23	351,09	350,33	347,31
	Müşteri tatmin düzeyi	1	0,77	0,58	0,26	0,16
RC102	Araç sayısı	3	3	3	3	3
	Mesafe	352,74	337,57	329,94	329,94	329,65
	Müşteri tatmin düzeyi	1	0,77	0,59	0,59	0,14
RC103	Araç sayısı	3	3	3	3	3
	Mesafe	333,92	327,69	324,91	310,34	310,34
	Müşteri tatmin düzeyi	1	0,91	0,64	0,40	0,40
RC104	Araç sayısı	3	3	3	3	3
	Mesafe	307,14	304,28	304,28	304,28	304,28
	Müşteri tatmin düzeyi	1	0,91	0,91	0,91	0,91
RC105	Araç sayısı	4	3	3	3	3
	Mesafe	412,38	354,99	336,81	335,4	329,94
	Müşteri tatmin düzeyi	1	0,76	0,51	0,32	0,19
RC106	Araç sayısı	3	3	3	3	3
	Mesafe	346,51	328,55	317,82	317,18	308,23
	Müşteri tatmin düzeyi	1	0,80	0,53	0,28	0,02
RC107	Araç sayısı	3	3	3	3	3
	Mesafe	298,95	296,83	296,83	296,83	296,83
	Müşteri tatmin düzeyi	1	0,97	0,97	0,97	0,97
RC108	Araç sayısı	3	3	3	3	3
	Mesafe	294,99	294,99	294,99	294,99	294,99
	Müşteri tatmin düzeyi	1	1	1	1	1
Toplam	Araç Sayısı	26	24	24	24	24
	Mesafe	2808,79	2610,13	2556,67	2539,29	2521,57

RC1 grubu problemlerde farklı $\prod_{eşik}$ değerleri için ve kullanılan araç sayıları ve yolculuk mesafelerindeki değişimler Şekil 5.37 ve Şekil 5.38'de gösterilmiştir.

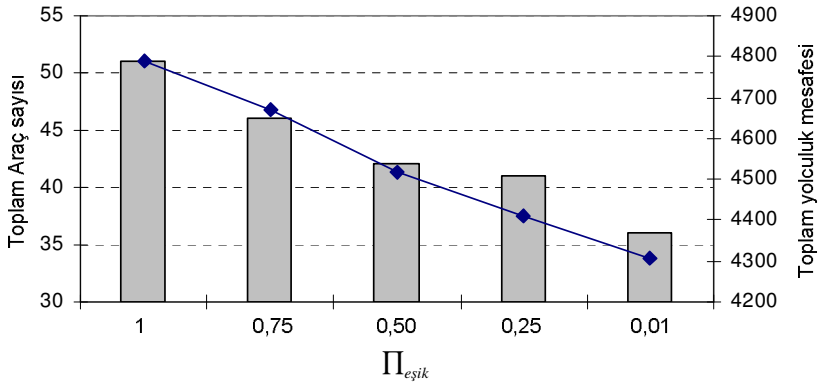


Şekil 5.37: Farklı $\Pi_{eşik}$ değerleri için araç sayıları

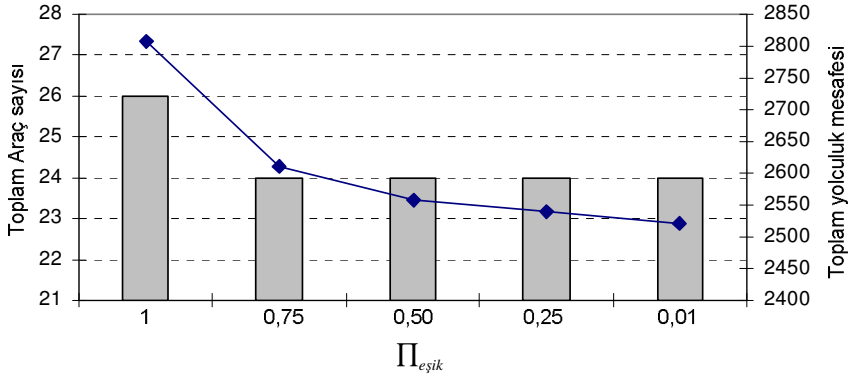


Şekil 5.38: Farklı $\Pi_{eşik}$ değerleri için yolculuk mesafeleri

Şekil 5.39 ve Şekil 5.40'da R1 ve RC1 gruplarında yer alan problemler için farklı $\Pi_{eşik}$ değerlerinde BZAARP-KKS algoritması ile elde edilen çözümlerin toplam araç sayıları ve toplam yolculuk mesafelerini içeren grafikler yer almaktadır.



Şekil 5.39: R1 grubu problemler için toplam araç sayısı ve toplam yolculuk mesafeleri



Şekil 5.40: RC1 grubu problemler için toplam araç sayısı ve toplam yolculuk mesafeleri

Esnek programlama modeli için gerçekleştirilen deneylerde $\Pi_{eşik}$ değeri azaldıkça daha düşük maliyetli çözümler elde edilebildiği görülmüştür. Ayrıca, en geç teslimat zamanın esnek bir kısıt olarak modellenmesiyle, müşteri tatmin düzeylerinde önemli bir azalma yaratmadan daha düşük maliyetli çözümler elde edilebileceği tespit edilmiştir.

5.4.3 Gürbüz Programlama Modeli için Deneyler

Solomon (1987) örnek problemlerindeki kesin yolculuk süreleri üçgen bulanık sayılara, en geç teslimat zamanları ise müşteri tatmin düzeyini temsil eden bulanık kümelere dönüştürülerek gürbüz programlama modeli için örnek problemler elde edilmiştir. Kesin değerlerin bulanık değerlere dönüştürülmesinde olabilirlik programlama ve esnek programlama modelleri için oluşturulan örnek problemlerdeki yöntem ve katsayılar kullanılmıştır.

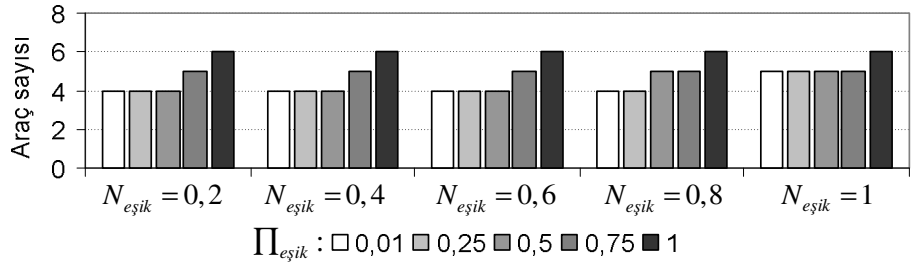
Tablo 5.16'da gürbüz programlama modeli için uyarlanmış 25 müşterilik R101 problemine BZARP-KKS algoritması ile $N_{eşik}$={0,2, 0,40, 0,6, 0,8, 1} ve $\Pi_{eşik}$={0,01, 0,25, 0,50, 0,75, 1} düzeylerinde oluşturulan çözümler yer almaktadır. Örneğin $N_{eşik}$=0,2 ve $\Pi_{eşik}$=0,01 için 4 araç kullanan, yolculuk mesafesi 469,36 birim olan bir çözüm oluşturulmuştur. Oluşturulan çözümdeki en düşük gereklilik değeri 0,34, en düşük müşteri tatmin düzeyi ise 0,25'tir. Daha önce de belirtildiği üzere $N_{eşik}$ değeri oluşturulan çözümde herhangi bir müşteriye teslimat yapılabilirliği için en düşük gereklilik düzeyidir. Elde edilen çözümün uygulama aşamasında en geç teslimat zamanı

kısıtlarına uygun olarak gerçekleşip gerçekleşmeyeceği olayının kesinliği konusunda bir ölçüttür. $\prod_{eşik}$ değeri ise oluşturulan çözümde herhangi bir müşteri için en düşük tatmin düzeyidir. BZARP-KKS algoritması $N_{eşik}$ ve $\prod_{eşik}$ değerlerinden daha düşük seviyelerde gereklilik veya müşteri tatmini sağlayan çözümler üretmemektedir.

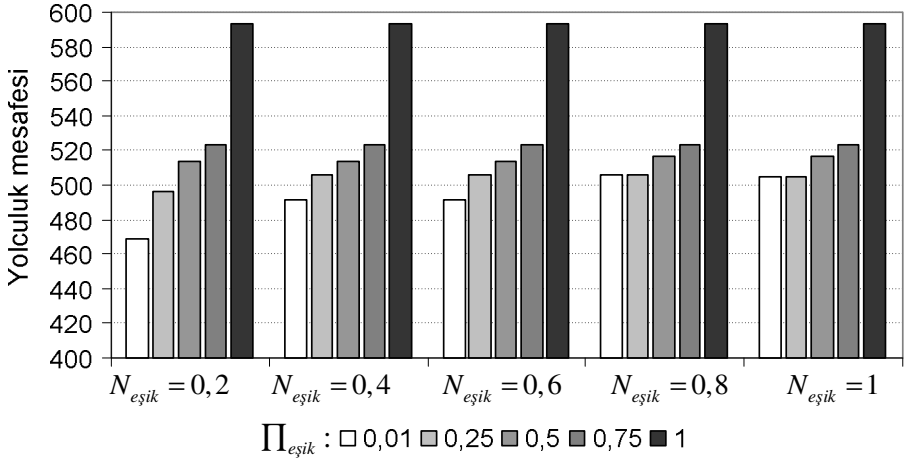
Tablo 5.16: R101 problemine elde edilen sonuçlar

Deney no	$N_{eşik}$	$\prod_{eşik}$	BZARP-KKS algoritması ile oluşturulan çözüm			
			Araç sayısı	Mesafe	Gereklilik	Müşteri tatmin düzeyi
1	0,2	0,01	4	469,36	0,343	0,245
2		0,25	4	495,92	0,431	0,437
3		0,5	4	513,2	0,76	0,564
4		0,75	5	523,31	1	0,786
5		1	6	593,54	1	1
6	0,4	0,01	4	491,01	0,627	0,245
7		0,25	4	505,59	0,507	0,437
8		0,5	4	513,2	0,76	0,564
9		0,75	5	523,31	1	0,786
10		1	6	593,54	1	1
11	0,6	0,01	4	491,01	0,627	0,245
12		0,25	4	505,76	0,806	0,439
13		0,5	4	513,2	0,76	0,564
14		0,75	5	523,31	1	0,786
15		1	6	593,54	1	1
16	0,8	0,01	4	505,76	0,806	0,439
17		0,25	4	505,76	0,806	0,439
18		0,5	5	516,47	1	0,546
19		0,75	5	523,31	1	0,786
20		1	6	593,54	1	1
21	1	0,01	5	504,92	1	0,437
22		0,25	5	504,92	1	0,437
23		0,5	5	516,47	1	0,546
24		0,75	5	523,31	1	0,786
25		1	6	593,54	1	1

R101 problemine farklı $N_{eşik}$ ve $\prod_{eşik}$ değerleri için oluşturulan çözümlerde kullanılan araç sayıları ve yolculuk mesafelerindeki değişimler Şekil 5.41 ve Şekil 5.42'te gösterilmiştir.



Şekil 5.41: Farklı $N_{eşik}$ ve $\prod_{eşik}$ değerleri için araç sayıları



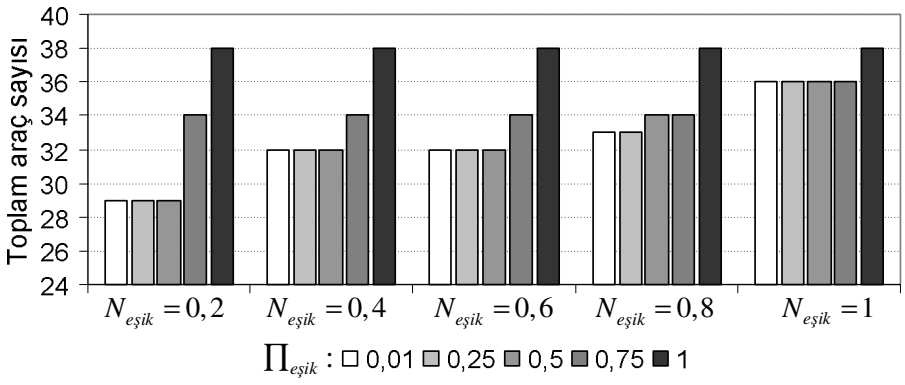
Şekil 5.42: Farklı $N_{eşik}$ ve $\prod_{eşik}$ değerleri için yolculuk mesafeleri

Tablo 5.17'de gürbüz programlama modeli için uyarlanmış R1 grubu problemlerine farklı $N_{eşik}$ *ve* $\prod_{eşik}$ düzeylerinde BZARP-KKS algoritması ile elde edilen çözümlerin kümülatif değerleri yer almaktadır.

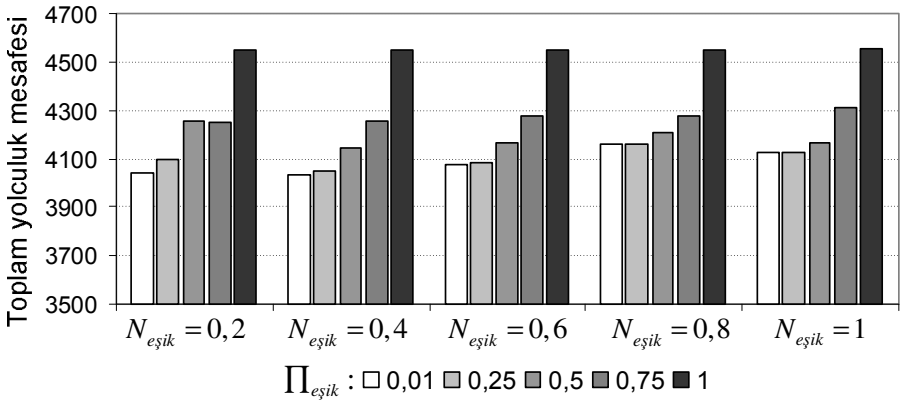
Tablo 5.17: R1 grubu problemlere oluşturulan çözümlerin kümülatif değerleri

Deney no	$N_{eşik}$	$\Pi_{eşik}$	Toplam araç sayısı	Toplam mesafe
1	0,2	0,01	29	4041,58
2		0,25	29	4099,48
3		0,5	29	4255,92
4		0,75	34	4251,67
5		1	38	4546,17
6	0,4	0,01	32	4036,25
7		0,25	32	4050,48
8		0,5	32	4145,31
9		0,75	34	4255,3
10		1	38	4546,61
11	0,6	0,01	32	4074,88
12		0,25	32	4080,49
13		0,5	32	4166,5
14		0,75	34	4278,7
15		1	38	4545,57
16	0,8	0,01	33	4158,54
17		0,25	33	4158,54
18		0,5	34	4210,91
19		0,75	34	4278,7
20		1	38	4546,17
21	1	0,01	36	4122,76
22		0,25	36	4122,76
23		0,5	36	4164,63
24		0,75	36	4309,73
25		1	38	4555,54

Tablo 5.17'de yer alan değerler Şekil 5.43 ve Şekil 5.44'te grafiksel olarak gösterilmiştir.



Şekil 5.43: R1 grubu problemler için toplam araç sayıları



Şekil 5.44: R1 grubu problemler için toplam yolculuk mesafeleri

Gürbüz programlama modeli ile farklı risk seviyelerinde ve farklı müşteri tatmin düzeylerinde çözümler oluşturulabilmektedir. Böylece kullanıcıya bir çok alternatif çözüm içerisinden seçim yapabilme imkanı ve seçilen çözümün uygulama aşamasında hangi seviyede risk taşıdığı konusunda bilgi sağlanmaktadır.

6. TARTIŞMA ve SONUÇ

Bu tez çalışmasında, KKO metasezgiseli yaklaşımı altında KKS algoritması temel alınarak, bulanık karar ortamında ZA-ARP için önerilen modellere çözüm oluşturabilecek bir algoritma geliştirilmiştir. Çalışmanın dördüncü bölümünde açıklandığı üzere, KKO metasezgiseli yaklaşımı altında önerilmiş olan farklı isimlerde çok sayıda algoritma vardır. Kullanılan iz yapısı, sezgisel bilgi ve yerel arama süreçleri aralarındaki temel farklılıkları oluşturmaktadır. Sezgisel algoritmaların performans değerlendirmesi, koşum süresi, çözüm kalitesi, uygulama kolaylığı, gürbüzlük ve esneklik gibi birden çok kriterin karşılaştırılmasını gerektirmektedir. Sezgisel yöntemler, pratik alandaki problemleri çözmeye yönelik olarak geliştirildiklerinden esneklik önemli bir kriterdir. Sezgisel algoritma, modeldeki, kısıtlardaki veya amaç fonksiyonundaki değişiklere kolaylıkla uyum sağlayabilmelidir. Gürbüzlük anlamında, sezgisel bir algoritma, problemin karakteristik özelliklerindeki farklılıklara karşı aşırı duyarlı olmamalıdır. Bu çalışmada geliştirilen algoritmanın, gürbüz, esnek ve kolay uygulanabilir olduğu ayrıca uygulamaya elverişli sürelerde iyi kalitede çözümler üretebildiği söylenebilir. Ancak, bu çalışmanın amaçlarından biri bulanık karar ortamında kullanılabilecek bir algoritmanın geliştirilmesi ve iyi bir performans gösterdiğinin ortaya konabilmesi olarak belirlenmiştir. Bu nedenle, bulanık karar ortamında kullanılabilecek en iyi KKO algoritmasının hangisi olduğu üzerine kapsamlı bir araştırma, konu kapsamına dahil edilmemiştir. Özellikle farklı yerel arama süreçlerinin kullanımıyla, önerilen algoritmanın koşum süresi ve çözüm kalitesi yönünden iyileştirilmesi ilerideki çalışmalarda ele alınabilir.

Bulanık karar ortamında araç rotalama problemi konusunda geçmiş çalışmalar incelendiğinde var olan az sayıdaki çalışmada GA ve simülasyon ile benzer süreçlerin kullanıldığı görülmüştür. Bu çalışmalardaki arama süreci, var olan çözümler üzerinde iyileştirmeler yapılması için GA kullanımı ve çözümlerin uygulanabilirliklerinin değerlendirilmesi için ise simülasyon kullanımı içermektedir. Bu kapsamda değerlendirildiğinde, bu çalışmada bulanık karar ortamı için ilk defa

çözüm oluşturucu bir yöntemin kullanımı önerilmiş ve kullanılabilirliği gösterilmiştir. Sonraki çalışmalarda, bulanık karar ortamında ARP çözümüne yönelik farklı çözüm oluşturucu sezgisel veya metasezgisel yöntemler kullanılmak istendiğinde, bu çalışmadaki temel süreçlerden yararlanılabilir.

Zaman aralıkları kısıtının esnek bir kısıt olarak ele alındığı geçmiş çalışmalarda, aracın gecikmesi durumunda teslimata izin verilmekte ancak gecikme süresi için ceza uygulanmaktadır. Bu çalışmadaki esnek kısıt durumu ise daha farklıdır. Esnek kısıt durumunda müşteri en geç teslimat anı için iki zaman belirlemektedir. Birincisi, kendisi için tam tatmin düzeyi sağlayacak en geç teslimat anı, ikincisi ise aşılması durumunda teslimatı kabul edemeyeceği teslimat anıdır. Bu iki değer arasındaki zamanlarda, müşteri tatmin düzeyi, müşteri tarafından belirlenebilecek bir fonksiyonla 1'den 0'a düşmektedir. Bu tip bir modelleme ile müşteri tatmin düzeyinin belirli bir değer üzerinde olmasını sağlayabilecek en iyi çözümün aranması sağlanabilmektedir.

Önerilen modellerde, bulanık kısıtlar, olabilirlik ve gereklilik ölçütleri ile ele alınmıştır. Modellerde minimum eşik ölçüt değerlerini [0,1] aralığında değiştirilebilir olması, ilgili kısıtın model içerisinden kolayca çıkarılması (0 değerinde) veya kesin olarak tatmin edilebilmesinin sağlanması (1 değeri ile) veya gevşek olarak sağlanabilirliğinin kontrol edilmesi ((0,1) değerleri ile) mümkündür. Böylece, önerilen modeller farklı tipte problemlerin çözümünde kolaylıkla kullanılabilir durumdadır.

Belirsizliklerin bulanık kümelerle ele alındığı geçmiş çalışmalar incelendiğinde, elde edilen çözümlerin karşılaştırılmasında genellikle, bulanık bir değer olarak elde edilen amaç fonksiyonu değerlerinin, farklı durulaştırma yöntemleri ile kesin değerlere dönüştürülmesi yaklaşımının kullanıldığı görülmektedir. Böylece, bulanık değerleri temsil eden kesin sayıların birbirlerine göre öncelik ilişkileri, çözümlerin birbirlerine göre önceliklerinin belirlenmesini sağlamıştır. Bu çalışmada ise amaç fonksiyonu değeri kesin bir sayı olarak elde edilmektedir. Bulanık değerler ise kısıtlar içerisinde yer almakta ve birbirlerine göre öncelik (büyüklük/küçüklük) ilişkileri olabilirlik ve gereklilik ölçütleri ile belirlenmektedir. Böylece, genel yaklaşıma göre daha tutarlı bir yöntem önerildiği söylenebilir. Çünkü iki bulanık değer arasında (eğer bulanık değerlerin destek kısımlarının kesişimi bir boş küme değilse) durulaştırma ile

belirlenen öncelik ilişkisi de belirsizlik içermektedir. Yani, planlama aşamasında henüz gerçek değeri bilinmeyen bu değerler kesinleştiğinde ortaya çıkacak olan öncelik ilişkisi, durulaştırma ile elde edilen ilişkiden farklı olabilir. Önerilen yaklaşım ile ilgilenilen değerlerin birbirleri arasındaki öncelik ilişkilerininin ne olacağı konusunda olabilirlik ve gereklilik değerleri ortaya konarak karar vericiye daha tutarlı ve çözüm hakkında daha fazla bilgi sağlayan bir yöntem geliştirilmiştir.

Önerilen modellerde, tüm müşteriler için en düşük tatmin düzeyi ve en düşük gereklilik değerleri eşit kullanılmıştır ancak yapılacak küçük değişikliklerle her müşteri için farklı değerler kullanılabilir. Örneğin, önemli kabul edilen sürekli müşteriler için ve yapılacak teslimattan elde edilecek kazancın yüksek olduğu durumlarda bu eşik değerleri yüksek tutulurken, önemsiz kabul edilen ve yapılacak teslimatın çok küçük boyutlu olacağı durumlarda bu eşik değerleri azaltılabilir. Bu durum, önerilen çözüm yöntemine herhangi bir ek hesaplama yükü getirmeyecektir. Böylece, karar vericiye önemli bir esneklik sağlanırken önemli müşterilere sunulmakta olan hizmetin kalitesi de arttırılabilir ve dağıtım firmaları müşteri odaklı çalışabilirler. Müşterileri farklı kategorilere ayırarak (farklı hizmet bedeli isteyerek) rotalama yapabilirler. Günümüzde dağıtım firmalarının bu tip hizmetlerinin olduğu bilinmektedir ancak bu tip özel müşteriler için farklı dağıtım araçlarının kullanılarak hizmet verildiği görülmektedir. Önerilen yöntem ile aynı rotada farklı kategoride müşteriler yer alabilmektedir.

Önerilen modellerin ve çözüm sürecinin bir karar destek sistemi içerisinde kullanımı ile karar verici tarafından belirlenecek en düşük kesinlik ve/veya müşteri tatmin düzeylerinde en az maliyetli çözümler elde edilebilir. Karar verici tarafından farklı müşteri tatmin düzeyleri ve risk seviyeleri için denemeler yapılmasıyla katlanılacak maliyetin müşteri tatmin düzeyindeki ve çözümlerin güvenilirliğindeki artışa etkisi görülebilir. Kullanıcıya alternatif çözümler arasında tercih yapma imkanı sağlanabilir. Gereklilik ölçütü ile oluşturulan çözümlerin geçerli olup olamayacağı konusundaki risk düzeyini belirlenebilmektedir. Olabilirlik ölçütü ise teslimat zamanlarının esnek olarak modellendiği durumlarda oluşturulan çözümlerdeki müşteri tatmin düzeyinin belirlenmesi amacıyla kullanılmıştır. Kullanıcı farklı risk düzeylerinde çözümler oluşturarak, bazı riskleri göze alarak sağlayabileceği maliyet avantajını görebilir veya yeni bir siparişi kabul edip etmeme konusunda karar verebilmek için modeli kullanabilir. Önerilen yöntem ile sipariş veren müşterinin

siparişinin istenen teslimat zamanı için kabul edilip edilmemesi, müşteriye teslimat zamanında esneklik yapmasının önerilmesi ile siparişin kabul edilip edilmemesine karar verilmesi veya müşterinin sağlayacağı esneklik ile maliyetlerde sağlanacak düşüşlerin müşteriye de yansıtılması konularında bilgi sağlanabilir.

Kısaca özetlemek gerekirse, bu çalışmada ZA-ARP problemi bulanık karar ortamında ele alınmış, bulanık küme teorisi ve olabilirlik teorisi kullanılarak belirsiz veri ve/veya esnek kısıt durumlarında kullanılabilecek modeller önerilmiştir. Önerilen en kapsamlı model olan gürbüz programlama modeli, kısıtların sol taraf katsayılarının ve/veya sağ taraf sabitlerinin belirli veya belirsiz olma durumlarında kullanılabilmektedir. Önerilen modeller ile ZA-ARP problemi gerçek hayata daha benzer olarak modellenebilmiş ve kullanıcılara oluşturulan çözümler hakkında daha fazla bilgi sahibi olma imkanı sağlanmıştır. Önerilen modelin çözümü için KKO metasezgisel yaklaşımı altında bir algoritma geliştirilerek kullanılmıştır. Yapılan deneylerde, geliştirilen algoritmanın doğru çalıştığı ve iyi bir performansa sahip olduğu görülmüştür. Bu aşamada, tez çalışmasının başlangıç aşamasında belirlenen hedeflere ulaşıldığı söylenebilir. Önerilen model ve çözüm yönteminin bir karar destek sistemi içerisinde kullanımının, kullanıcıların bir çok tercih ve önceliklerine göre alternatif çözümlerin oluşturulmasını sağlayarak uygulama alanında oldukça faydalı olabileceği değerlendirilmektedir.

KAYNAKLAR

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

Antes, J., Derigs, U., 1995. A new parallel tou construction algorithm for the vehicle routing problem with time windows, Working Paper, Department of Economics and Computer Science, University of Köln, Germany.

Baker, E., Schaffer, J., 1986. Solution improvement for the vehicle routing and scheduling problem with time windows, *American Journal of Mathematical and Management Sciences,* **6,** 261-300.

Balakrishnan, N., 1993. Simple heuristics for the vehicle routing problem with soft time windows. *Journal of the Operational Research Society,* **44(3)**, 279–87.

Barr, R.S., Golden, B.L., Kelly, J.P.,Resende, M.G.C., Stewart, J.R., 1995. Designing and reporting on computational experiments with heuristic methods, *Journal of Heuristics,* **1,** 9-32.

Bastian, C., Rinnooy Kan, A.H.G., 1992. The stochostic vehicle routing problem revisited, *European Journal of Operational Research,* **56**, 407-412.

Bell, J.E., McMullen, P.R., 2004. Ant colony optimization techniques fort he vehicle routing problem, *Advanced Engineering Informatics,* **18,** 41-48.

Bellman, R.E., Zadeh, L.A., 1970. Decision-making in a fuzzy environment, *Management Science*, **17B,** 141–164.

Bent, R, Van Hentenryck, P., 2004. A two-stage hybrid local search for the vehicle routing problem with time windows, *Transportation Science,* **38,** 515-530.

Berger, J., Salois, B., and Begin, R., 1998. A hybrid genetic algorithm for the vehicle routing problem with time windows, *Lecture Notes in Artificial Intelligence,* **vol. 1418,** 114-127.

Berger, J., and Barkaoui, M., 2000. An improved hybrid genetic algorithm for the vehicle routing problem with time windows, *International ICSC*

Congress on Intelligent Systems and Applications (ISA2000), University of Wollongong, Sydney, Australia, December 12-15.

Bodin, L.D., Golden, B.L., Assad, A.A., Ball, M.O., 1983. Routing and scheduling of vehicle crews: The state of the art, *Computers & Operations Research,* **10**, 69-211.

Blanton, J., Wainwright, R., 1993. Multiple vehicle routing with time and capacity constraints using genetic algorithms, *Proceedings of the Fifth International Conference on Genetic Algorithms(ICGA93),* Illinois, USA, June 17-22, 452-459.

Blum, C., Roli, A., 2003. Metaheuristics in combinatorial optimization: Overview and and conceptual comparison, *ACM Computing Surveys*, **35(3)**, 268-308.

Blum, C., 2005. Beam-ACO – Hybridizing ant colony optimization with beam search: An application to open shop scheduling, *Computers and Operations Research,* **32(6),** 1565-1591.

Bräysy, O., 2002. Fast local searches for the vehicle routing problem with time windows, *INFOR,* **40(4),** 319-330.

Bräysy, O., 2003. A reactive variable neighborhood search for the vehicle routing problem with time windows. *INFORMS Journal on Computing,* **15(4),** 347-368.

Bräysy, O., Hasle, G., Dullaert, W., 2004. A multi-start local search algorithm for the vehicle routing problem with time windows, *European Journal of Operational Research,* **159,** 586-605.

Bräysy, O., Gendreau, M., 2005a. Vehicle routing problem with time windows, part I: route construction and local search algorithms, *Transportation Science,* **39,** 104-118.

Bräysy, O., Gendreau, M., 2005b. Vehicle routing problem with time windows, part II: metaheuristics, *Transportation Science,* **39,** 119-139.

Bullnheimer, B., Hartl, R.F. and Strauss, C., 1999a. A new rank-based version of the ant system: A computational study, *Central European Journal for Operations Research and Economics,* **7(1)**, 25-38.

Bullnheimer, B., Hartl, R.F. and Strauss, C., 1999b. An improved ant system algorithm for the vehicle routing problem, *Annals of Operations Research,* **89**, 319-328.

Caseau, Y., Laburthe, F., 1999. Heuristics for large constrained vehicle routing problems, *Journal of Heuristics*, **5**, 281-303.

Chabrier, A., 2006. Vehicle routing problem with elementary shortest path based column generation, *Computers and Operations Research,* **33(10),** 2972-2990.

Chanas, S., Kasperski, A., 2004. Possible and necessary optimality of solutions in the single machine scheduling problem with fuzzy parameters, *Fuzzy Sets and Systems,* **142,** 359-371.

Chanas, S., Kasperski, A., 2003. On two single machine scheduling problem with fuzzy processing times and fuzzy due dates, *European Journal of Operational Research,* **147,** 281-296.

Chanas, S., Kasperski, A., 2001. Minimizing maximum lateness in a single machine scheduling problem with fuzzy processing times and fuzzy due dates, *Engineering Applications of Artificial Intelligence,* **14,** 377-386.

Cheng, C.B., Mao, C.B., 2007. A modified ant colony system for solving the travelling salesman problem with time windows, *Mathematical and Computer Modelling,* **46,** 1225-1235.

Chiang, W., Russel, R., 1996. Simulated annealing metaheuristics for the vehicle routing problem with time windows, *Annals of Operations Research,* **63,** 3-27.

Chiang, W., Russel, R., 1997. A reactive tabu search metaheuristics for the vehicle routing problem with time windows, *INFORMS Journal of Computing,* **9,** 417-430.

Christofides, N., Mingozzi, A. and Toth, P., 1979. The vehicle routing problem, in *Combinatorial Optimization,* pp. 315-338, Eds. Christofides, N., Mingozzi, A., Toth, P., Sandi, C., Wiley, Chichester, UK.

Christofides, N., Mingozzi, A. and Toth, P., 1981. An algorithm for the time constrained traveling salesman problem, *Technical Report*, Imperial College, London.

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

Colorni, A., Dorigo, M., Maniezzo, V., Trubian, M., 1994. Ant system for job-shop scheduling, *Belgian Journal of Operations Research,* **34,** 39-53.

Cook, W., Rich, J.L., 1999. A parallel cutting plane algorithm for the vehicle routing problem with time windows, *Working Paper,* Computational and Applied Mathematics, Rice University, Houston, TX.

Cook, T., Russel, R., 1978. A simulation and statistical analysis of stochastic vehicle routing with timing constraints, *Decision Science,* **9,** 673-687.

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

Cordeau J-F., Desaulniers, G., Desrosiers, J., Solomon, M.M., Sournis, F., 2002. VRP with time windows, in *The vehicle routing problem, SIAM monographs on discrete mathematics and applications,* vol.9., pp. 157-193, Eds. Toth, P., Vigo, D, SIAM, Philedelphia.

Cordone, R., Wolfler-Calvo, R., 2001. A heuristic for the vehicle routing problem with time windows, *Journal of Heuristics*, **7,** 107-109.

Costa, D., Hertz, A., 1997. Ants can colour graphs, *Journal of Operations REsearch Society,* **48,** 295-305.

Dantzig, G.B., Wolfe, P.A., 1960. A decomposition principle for linear programs, *Operations Research*, **8,** 101-111.

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

De Backer, B., **Furnon, V., Shaw, P., Kilby, P., Prosser, P.,** 2000. Solving vehicle routing problems using constraint programming and metaheuristics, *Journal of Heuristics,* **6,** 501-523.

Deneubourg, J.L., Aron, S., Goss, S., and Pasteels, J.M., 1990. The self-organizing exploratory pattern of the Argentine ant, *Insect Behavior,* **3,** 159-168.

Desrochers, M., Desrosier, J., Solomon, M., 1992. A new optimization algortihm for the vehicle routing problem with time windows, *Operations Research,* **40,** 342-354.

Desrosiers, J., Soumins, F., Sauve, M., 1988. Lagrangean relaxation methods for solving the minimum fleet size multiple traveling salesman problem with time windows, *Management Science,* **34,** 1005-1022.

Di Caro, G., Dorigo, M., 1998. AntNet: Distributed stigmercy control for communications networks, *Journal of Artificial Intelligence Research,* **9,** 317-365.

Donati, A.V., Montemanni, R., Casagrande, N., Rizzoli, A.E., Gambardella, L.M., 2008. Time dependent vehicle routing problem with a multi ant colony system, *European Journal of Operational Research,* **185,** 1174-1191.

Dondo, R., Cerda, J., 2007. A cluster-based optimization approach for the multi-depot heteregeneous fleet vehicle routing problem with time windows, *European Journal of Operational Research,* **176,** 1478-1507.

Dongarra, J., 1998. Performance of various computers using standart linear equations software, *Report CS-89-85*, Department of Computer Science, University of Tennesse, Knoxville, TN.

Dorigo, M., Maniezzo, V., Colorni, A., 1991. The Ant System: An auto catalytic optimizing process, *Technical Report 91-016*, Dipartimento di Elettronica, Politecnico di Milano, Milan.

Dorigo, M., 1992. Optimization, Learning and Natural Algorithms, *PhD thesis*, Politecnico di Milano, Italy.

Dorigo, M., Maniezzo, V., Colorni, A., 1996. Ant System: Optimization by a Colony of Cooperating Agents, *IEEE Transactions on Systems, Man and Cybernetics- Part B,* **26(1),** 29-41.

Dorigo, M., and Gambardella, L., 1997. Ant Colony System: A cooperative learning approach to the traveling salesman problem, *IEEE Transactions on Evolutionary Computation,* **1,** 53-66.

Dorigo, M., Di Caro, G., 1999a. Ant Colony Optimization: A new meta-heuristic, *Proceedings of the 1999 Congress on Evolutionary Computation (CEC'99),* Piscataway, NJ, 1470-1477.

Dorigo, M., Di Caro, G., 1999b. The ant colony optimization meta-heuristic, in *New Ideas in Optimization*, pp. 11-32, Eds. D. Corne, M. Dorigo, F. Glover, McGraw Hill, London.

Dorigo, M., and Stützle, T., 2004. Ant Colony Optimization, The MIT press, London.

Dror, M., Trudreau, P., 1986. Stochastic vehicle routing with modified savings algorithm, *European Journal of Operational Research,* **23**, 228-235.

Dror, M., Laporte, G., Trudreau, P., 1989. Vehicle routing with stochastic demands: properties and solution frameworks, *Transportation Science,* **23**, 166-176.

Dror, M., 1992. Vehicle routing with uncertain demands: stochastic programming and its correponding TSP solution, *European Journal of Operational Research,* **64**, 432-441.

Dror, M., 1993. Modeling vehicle routing with uncertain demands as a stochastic programming: properties of the corresponding solution, *European Journal of Operational Research,* **64**, 432-441.

Dror, M., 1994. Note on the comlexity of the shortest path models for column generation in VRPTW, *Operations Research,* **42(5),** 977-978.

Dullaert, W., 2000. Impact of relative route length on the choice of time insertion criteria for insertion heuristics for the vehicle routing problem with

time windows, *Proceedings of the Rome Jubilee 2000 Conference*, University of Rome, Italy, September 11-14, 153-156.

Dullaert, W., Bräysy, O., 2003. Routing with relatively few customers per route, *TOP*, **11,** 325-336.

Dubois D., Fargier, H., Prade, H., 2003a. Fuzzy scheduling: Modelling felxible constraints vs. coping with incomplete knowledge, *European Journal of Operational Research,* **147,** 231-252.

Dubois D., Fargier, Galvagnon, V., 2003b. On latest starting times and floats in activity networks with ill-known durations, *European Journal of Operational Research,* **147,** 266-280.

Dubois D., Prade, H., 2000. Fundamentals of Fuzzy Sets, Kluwer Academic Publishers, Boston.

Dubois D., Fargier, H., Prade, H., 1996. Possibility theory in constraint satisfaction problems: Handling priority, preference and uncertainty, *Applied Intelligence,* **6,** 287-309.

Dubois D., Fargier, H., Prade, H., 1995. Fuzzy constraints in job-shop scheduling, *Journal of Intelligent Manufacturing,* **6**, 215-234.

Dubois, D., Prade, H., 1988. Possibility theory: an approach to computerized processing of uncertainty, Plenum Press, New York.

Dubois, D., 1987. Linear programming with fuzzy data, in *Analysis of Fuzzy Information*, *vol. 3, Applications in Engineering and Science*, pp. 241-263, Ed. Bezdek, J.C., CRC press, Boca Raton, FL.

Dubois, D., Prade, H. 1980a. Fuzzy Sets and Systems – Theory and Applications, Academic Press, New York.

Dubois, D., Prade, H., 1980b. Systems of linear fuzzy constraints, *Fuzzy Sets and Systems,* **3**, 37-48.

Flood, M.M., 1956. The traveling salesman problem, *Operations Research,* **4,** 61-75.

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

Fisher, M., 1995. Vehicle Routing, in: *Network Routing Handbooks in Operations Research and Management Science, Vol. 8,* pp. 1-33, Eds. Ball, M. O., Magnanti, T. L., Monma, C. L., Nemhauser, G. L.,North-Holland, Amsterdam, Netherlands.

Fisher, M., Jornsten, K., and Madsen, O., 1997. Vehicle routing with time windows: two optimization algorithms, *Operations Research,* **45,** 488-492.

Fleischmann, B., Gietz, M., Gnutzmann, S., 2004. Time-varying travel times in vehicle routing, *Transportaiton Science,* **38(2)**, 160-173.

Fortemps, P., 2000. Introducing flexibility in scheduling: the preference approach, in *Scheduling Under Fuzziness,* pp. 61-79, Eds. Slowinski, R., Hapke, M., Physica Verlag, Heidelberg, New York.

Fortemps, P., 1997. Fuzzy sets for modelling and Handling Imprecision and Flexibility, *doctoral dissertion*, Faculte Polytechnique de Mons.

Fortemps, P., Roubens, M., 1996. Ranking and defuzzification methods based on area compensation. *Fuzzy sets and Systems,* **82,** 319-330.

Gambardella, L.M., Dorigo, M., 1995. Ant-Q: A reinforcement learning approach to the travelling salesman problem, in *Proceedings of the 12th International Conference on Machine Learning,* Tahoe City, California, USA, July 9-12, 252-260.

Gambardella, L.M., Taillard, E.D. and Agazzi, G., 1999a. MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows, in *New Ideas in Optimization*, pp. 63-76, Eds. Corne, D., Dorigo, M., Glover, F., McGraw Hill, London.

Gambardella, L.M., Taillard, E.D., Dorigo, M., 1999b. Ant colonies for the quadratic assignment problem. *Journal of the Operational Research Society,* **50,** 167-176.

Gambardella, L.M., Dorigo, M., 2000. Ant colony system hybridized with a new local search for the sequential ordering problem, *INFORMS Journal on Computing,* **13(3),** 237-255.

Garcia, B., Potvin, J., Rousseau, J., 1994. A parallel implementation of the tabu search heuristic for vehicle routing problem with time windows, *Computers and Operations Research,* **21,** 1025-1033.

Gehring, H., Homberger, J., 2001. A parallel two-phase metaheuristic for routing problems with time windows, *Asia-Pasific Journal of Operations Research,* **18,** 35-47.

Gendreau, M., Hertz, A., & Laporte, G., 1994. A tabu search heuristic for the vehicle routing problem, *Management Science,* **40,** 1276-1290.

Gendreau, M., Laporte, Seguin, R., 1995. An exact algorithm for the vehicle routing problem with stochastic customers and demands, *Transportaion Science,* **29**, 143-155.

Gendreau, M., Laporte, G., Potvin, J-Y., 2002. Metaheuristics for the VRP, in *The Vehicle Routing Problem, SIAM monographs on Discrete*

Mathematics and Applications, pp. 129-154, Eds. Toth, P., Vigo, D., Philedelphia.

Gillet, B.E., Miller, L.R., 1974. A heuristic algorithm for the vehicle dispatch problem, *Operations Research,* **22,** 240-349.

Glover, F., 1986. Future paths for integer programming and links to artificial intelligence, *Computers & Operations Research*, **13**, 533–549.

Glover, F., 1992. New ejection chain and alternating path methods for traveling salesman problems, in *Computer Science and Operations Research: New Developments in their interface,* pp. 449-509, Eds. Balci, O., Sharda, R., and Zenios, S., Pergamon Press, Oxford.

Glover, F., Laguna, M., 1997. Tabu Search, Kluwer Academic Publishers, Boston.

Goldberg, B., 1989. Genetic algorithms in search, optimization, and machine learning, Addison-Wesley, New York.

Gravel, M., Price, W.L., Gagne, C., 2002. Scheduling continuous casting of aluminum using a multiple objective ant colony optimization problem, *European Journal of Operational Research,* **143,** 218-229.

Han, S., Ishii, H., Fujii, S., 1994. One machine scheduling problem with fuzzy duedates, *European Journal of Operations Research,* **79,** 1-12.

Hapke, M., Slowinski, R., 1996. Fuzzy scheduling under resource constraints, *Proceedings of the European Workshop on Fuzzy Decision Analysis for Management, Planning and Optimization,* Dortmund, Germany, May 21-22, 121-126.

Hapke, M., Wesolek, P., 2003. Handling imprecision and flexible constraints in vehicle routing problems: fuzzy approach, *Report RA-005/2003*, Polytechnika Poznanska.

Ho, S.C., Hougland, D., 2004. A tabu search heuristic for the vehicle routing problem with time windows and split deliveries, *Computers and Operations Research,* **31,** 1947-1964.

Holland, J., 1975. Adaptation in Natural and Artificial Systems: An introductory Analysis with Applications to Biology, Control and Artificial Intelligence, University of Michigan Press, Ann Arbor.

Homberger, J., Gehring, H.A., 1999. Two evolutionary metaheuristics for the vehicle routing problem with time windows, *INFOR,* **37,**297-318.

Homberger, J., Gehring, H.A., 2005. A two-phase hybrid metaheuristic for the vehicle routing problem with time windows, *European Journal of Operational Research,* **162,** 220-238.

Horner, P., 1999. Planning under uncertainty. Questions & answers with George Dantzig, *OR/MS Today,* **26**, 26-30.

Ibaraki, T., Imahori, S., Kubo, M., Masuda, T., Uno, T., Yagiura, M., E, 2005. Effective local search algorithms for routing and scheduling problems with general time window constraint, *Transportation Science,* **39(2),** 206-232.

Ichoua, S., Gendreau, M., Potvin, J.Y., 2003. Vehicle dispatching with time-dependent travel times, *European Journal of Operational Research,* **144(2)**, 379-396.

Inuiguchi, M., Ramik, J., 2000. Possibilistic linear programming: a brief review of fuzzy mathematical programming and a comparison with stochastic programming in portofilo selection problem, *Fuzzy Sets and Systems,* **111,** 3-28.

Ioannou, G., Kritikos, M., Prastacos, G., 2003. A greedy look-ahead heuristic for the vehicle routing with soft time windows, *Journal of Operations Research Society,* **52,** 523-537.

Ioannou, G., Kritikos, M., Prastacos, G., 2001. A problem generator-solver heuristic for vehicle routing with soft time windows, *Omega,* **31,** 41-53.

Irnich, S., Funke, B., Grünert, T., 2006. Sequential search and its application to Vehicle-routing problems, *Computers & Operations Research,* **33,** 2405-2429.

Irnich, S., Villeneuve, D., 2006. The shortest path problem with resource constraints and k-cycle elemination ($k \geq 3$): Improving a branch-and-price algorithm for the VRPTW, *INFORMS journal of computing,* **18(3)**, 391-406.

Ishibuchi, H., Yamamoto, N., Misaki, S., Tanaka, H., 1994. Local search algorithms for flowshop scheduling with fuzzy due dates, *International Journal of Production Economics,* **33**, 53-66.

Ishibuchi, H., Yamamoto, N., Murata, T., Tanaka, H., 1994. Genetic algorithms and neighborhood search algorithms for fuzzy flowshop scheduling problems, *Fuzzy Sets and Systems,* **67**, 81-100.

Ishii, H., Tanaka, M., Masuda, T., 1992. Two scheduling problems with fuzzy due-dates, *Fuzzy Sets and Systems,* **46,** 339-347.

Itoh, T., Ishii, H., 1999. Fuzzy due-date scheduling problem with fuzzy processing time, *International Transactions on Operations Research,* **6,** 639-647.

Kagaya, S., Kikuchi, S., Donnelly, R.A., 1994. Use of fuzzy theory technique for grouping trips in the vehicle routing and scheduling problem. *European Journal of Operational Research,* **76**, 143-154.

Kallehauge, B., 2008. Formulations and exact algorithms for the vehicle routing problem with time windows, *Computers and Operations Research,* **35,** 2307-2330.

Kallehauge, B., Larsen, J., Madsen, O.B.G, 2006. Lagrangian duality applied to the vehicle routing with time windows, *Computers and Operations Research,* **33,** 1464-1487.

Kasperski, A., 2005. A possibilistic approach to sequencing problems with fuzzy parameters, *Fuzzy Sets and Systems,* **150,** 77-86.

Kaufmann, A., Gupta, M.M, 1985. Introduction to fuzzy arithmetic: theory and applications. New York: Van Nostrand Reinhold.

Kılıç, S., Kahraman, C., 2008. Solution of a fuzzy flowshop scheduling problem using a necessity measure, *Journal of Multiple Valued Logic and Soft Computing,* **15(1)**, 51-64.

Kılıç, S., Kahraman, C., 2006a. Scheduling a fuzzy flowshop problem with fuzzy processing times using ant colony optimization, *Proceedings of the 7th International FLINS Conference,* Genova, Italy, August 29-31, 449-456.

Kılıç, S., Kahraman, C., 2006b. Metaheuristic techniques for job shop scheduling problem and a fuzzy ant colony optimization algorithm, *Fuzzy Applications in Industrial Engineering,* pp. 401-425, Ed. Kahraman, C., Springer-Verlag, Netherlands.

Kılıç, S., Kaylan, A.R., 2003. Scheduling aircraft landings with ant colony optimization, *EURO/INFORMS Joint International Meeting,* İstanbul, July 6-10.

Kilby, P., Prosser, P., Shaw, P., 1999. Guided local search for the vehicle routing problem with time windows, in *META-HEURISTICS Advanced Trends Local Search Paradigms for Optimization,* pp. 473-486, Eds. Voss, S., Martello, S., Osman, I.H., Rocairol, C., Kluwer Academic Publishers, Boston, MA.

Kirkpatrick, S., Gellat, C.D., Vecchi, P.M., 1983. Optimization by simulated annealing, *Science,* **220,** 671-680.

Knight, K., Hofer, J., 1968. Vehicle Scheduling with timed and connected calls: A case study, *Operational Research Quarterly*, **19,** 299-310.

Kohl, N., Madsen, O., 1997. An optimization algorithm for the vehicle routing problem with time windows based on Lagrange relaxation, *Operations Research,* **45,** 395-406.

Kohl, N., Desrosiers, J., Madsen, O., Solomon, M., and Soumis, F., 1999. 2-path cuts for the vehicle routing problem with time windows, *Transportation Science,* **33,** 101-116.

Kolen, A., Rinnooy Kan, A., Trienkens, H., 1987. Vehicle routing with time windows, *Operations Research,* **35,** 266-273.

Kontoravdis, G.,A., Bard, J.F., 1995. A GRASP for the vehicle routing problem with time windows, *INFORMS Journal of Computing,* **7,**10-23.

Koskosidis, Y.,A., Powell, W.,B., Solomon, M.M., 1992. An optimisation-based heuristic for vehicle routing and scheduling with soft time window constraints, *Transportation Science*, **26(2),** 69–85.

Kuo, R.J., Wang, H.S., Hu, T.L., Chou, S.H., 2005. Application of ant k-means clustering analysis, *Computers and Mathematics with Applications,* **50,** 1709-1724.

Laguna, M. 2002. Scatter search, in *Handbook of applied optimization*, pp. 183-193, Eds. Pardalos, P.M., Resende, M.G.C., Oxford University Press, New York.

Lai, K.K., Liu, B., Peng, J., 2003. Vehicle routing problem with fuzzy travel times and its genetic algorithm, *Technical Report.*

Laporte, G., Semet, F., 2002. Classical heuristics for the capacitated vrp, in *The Vehicle Routing Problem*, pp. 109-128, Eds. Toth, P. and Vigo, D., SIAM, Philadelphia, 2002.

Laporte, G., 1992. The vehicle routing problem: an overview of exact and approximate algorithms, *European Journal of Operational Research*, **59**, 345-358.

Laporte, G., Louveaux, F.V., Mercure, H., 1992. The vehicle routing problem with stochastic travel times, *Transportation Science*, **26**, 161-170.

Laporte, G., Louveaux, F.V., Mercure, H., 1989. Models and exact solutions for a class of stochastic location-routing problem, *European Journal of Operational Research*, **39**, 71-78.

Larsen, J., 1999. Parallelization of the vehicle routing problem with time windows, *Ph.D. Thesis*, Department of Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark.

Lau, H.C., Sim, M., Teo, K.M., 2003. Vehicle routing problem with time windows and a limited number of vehicles, *European Journal of Operational Research*, **148**, 559-569.

Lenstra, J., Rinooy Kan, A., 1981. Complexity of vehicle routing ans scheduling problems, *Networks,* **11,** 221-227.

Li, H., Lim, A., 2003. Local search with annealing-like restarts to solve the VRPTW, *European Journal of Operational Research***, 150**, 115-127.

Lin, S., 1965. Computer solutions of the travelling salesman problem, *Bell System Technical Journal,* **44,** 2245-2269.

Liu, B., 2002. Theory and Practice of Uncertain Programming, Physica-Verlag, Heidelberg.

Liu, B, Liu, Y.K., 2002. Expected value of fuzzy variable and fuzzy expected value models, *IEEE Transactions on Fuzzy Systems,* **10(4)**, 445-450.

Liu, B., Lai, K.K., 2002. Stochastic programming models for vehicle routing problems, *Asian Information Science Life,* **1**, 13-28,

Liu, F.F., Shen, S., 1999. A route-neighborhood-based metaheuristic for vehicle routing problem with time windows, *European Journal of Operational Research,* **118,** 485-504.

Madsen, O.B.G., 1990. Lagrangean relaxation and vehicle routing, *IMSOR Working Paper,* The Technical University of Denmark.

Maniezzo, V., Colorni, A., 1999. The ant system applied to the quadratic assignment problem, *IEEE Transactions on Knowledge and Data Engineering,* **11,** 769-784.

McMullen, P.R., 2001. An ant colony optimization approach to addressing a JIT sequencing problem with multiple objectives, *Artificial Intelligence in Engineering,* **15,** 309-317.

Mester, D., Bräysy, O., 2005. Active guided evolution strategies for large-scale vehicle routing problems with time windows, *Computers and Operations Research,* **32,** 1593-1614.

Mladenovic, N., Hansen, P., 1997. Variable neighborhood search, *Computers and Operations Research*, **24,** 1097-1100.

Moscato, P., Cotta, C., 2002. A gentle introduction to memetic algorithms, in *Handbook of metaheuristics,* pp. 1-56, Eds. Glover, F., Kochenberger, G., Kluwer Academic Publishers, Boston.

Negoita, C.V., Minoiu, S., Stan, E., 1976. On considering imprecision in dynamic linear programming, *Economic Computation and Economic Cybernetics Studies and Research Journal*, **3,** 83–95.

Or, I., 1976. Traveling Salesman-Type Combinatorial Problems and their Relation to the Logistics of Regional Blood Banking, *PhD thesis*, Northwestern University, Evanston, Illionis.

Osman, I.H., 1993. Metastrategy simulated annealing and tabu search algortihms for combinatorial optimisation problems, *Annals of Operations Research,* **41,** 421-451.

Peng, J., Shang, G., Liu, H., 2006. A hybrid intelligent algorithm for vehicle routing models with fuzzy travel times, *Lecture Notes in Artificial Intelligence*, **4114**, 965-976.

Potvin, J., Rousseau, J., 1993. A parallel route building algorithm for the vehicle routing and scheduling problem with time windows, *Journal of Operational Research Society,* **46,** 1433-1446.

Potvin, J., Rousseau, J., 1995. An exchange heuristic for routing problems with time windows, *European Journal of Operational,* **66,** 331-340.

Potvin, J., Robillard, C., 1995. Clustering for vehicle routing with a competitive neural network, *Neurocomputing,* **8,** 125-139.

Potvin, J., Dube, D., Robillard, C., 1996. A hybrid approach to vehicle routing using neural networks and genetic algorithms, *Applied Intelligence,* **6,** 241-252.

Potvin, J., Begio, S., 1996. The vehicle routing problem with time windows – Part II: Genetic search, *INFORMS Journal of Computing,* **8,** 165-172.

Psinger, D, Ropke, S., 2007. A general heuristic for vehicle routing problems, *Computers and Operations Research,* **34**, 2403-2435.

Pullen, H., Webb, M., 1967. A computer application to transport scheduling problem*, Computers Journal,* **10,** 10-13.

Ralphs, T., Kopman, L., Pulleyblank, W., Trotter, Jr.L., 2003. On the capacitated vehicle routing problem, *Mathematical Programming,* **94,** 343-359.

Reimann, M., Stummer, M. and Doerner, K., 2002. A savings based Ant System for the vehicle routing problem, *Genetic and Evolutionary Computation Conference (GECCO-2002),* New York, USA, July 9-13.

Reimann, M., Doerner, K., Hartl, R.F., 2004. D-Ants: Savings based ants divide and conquer the vehicle routing problem, *Computers and Operations Research,* **31(4),** 563-591.

Rochat, Y., Taillard E.D., 1995. Probabilistic diversification and intensification in local search for vehicle routing, *Journal of Heuristics,* **1,** 147-167.

Rommelfanger, H.J., Slowinski, R., 1998. Fuzzy Linear Programming with Single or Multiple Objective Functions, in *Fuzzy Sets in Decision Analysis, Operations Research ans Statistics,* Ed. Slowinski, R., Kluwer Academic Publishers, Boston.

Rommelfanger, H.J., 1995. Reduction costs as a vital argument in favor of fuzzy models, *BUSEFAL,* **51,** 30-38.

Rommelfanger, H.J., 1990. FULPAL: An interactive method for solving (multiobjective) fuzzy linear programming problems, in *Stochastic versus Fuzzy Approaches to Multiobjective Mathematical Programming under Uncertainty,* pp. 321-330, Eds. Slowinski, R., Teghem, J., Kluwer.

Russel, R., 1995. Hybrid heuristics for the vehicle routing problem with time windows, *Transportation Science,* **29(2),** 156-166.

Sahinidis, N.V., 2004. Optimization under uncertainty: state-of-the-art and opportunities, *Computers and Chemical Engineering*, **28**, 971-983.

Sakawa, M., Kubota, R., 2000. Fuzzy programming for multiobjective job shop scheduling with fuzzy processing time and fuzzy duedate through genetic algorithms, *European Journal of Operational Research,* **120**, 393-407.

Sakawa, M., Kato, K., Sunada, H., Shibano, T., 1997. Fuzzy programming for multiobjective 0-1 programming problems through genetic algorithms, *European Journal of Operational Research,* **97**, 149-158.

Savelsbergh, M., 1985. Local search in routing problems with time windows, *Annals of Operations Research,* **4,** 285-305.

Schoonderwoerd, R., Holland, O., Bruten, J., Rothkrantz, L., 1997. Ant-based load balancing in telecommunications networks, *Adaptive Behavior,* 169-207.

Schulze, J., Fahle, T., 1999. Parallel algorithm for the vehicle routing problem with time window constraint, *Annals of Operations Research,* **86,** 585-607.

Schrimpf, G., Schneider, J., Stamm-Wibrandt, H., Dueck, G., 2000. Record breaking results using the ruin recreate principle, *Journal of Computational Physics*, **159**, 139-171.

Shaw, P., 1997. A new local search algorithm providing high quality solutions to vehicle routing problems, Working paper, Department of Computer Science, Universtiy of Strathclyde, Glasgow, Scotland.

Shaw, P., 1998. Using constraint programming and local search methods to solve vehicle routing problems, in *Principles and Practise of Constraint Programming – CP98, Lecture Notes in Computer Science*, pp. 417-431, Eds. Maher, M., Puget, J.F., Springer-Verlag, New York.

Shelokar, P.S., Jayaraman, V.K., Kulkarni, B.D., 2004a. An ant colony classifier system: Application to some process engineering problems, *Computers and Chemical Engineering,* **28,** 1577-1584.

Shelokar, P.S, Jayaraman, V.K., Kulkarni, B.D., 2004b. An ant colony approach for clustering, *Analytica Chimica ACTA,* **509,** 187-195.

Shyu, S.J., Lin, B.M.T., Lin, P.Y., 2004. Application of ant colony optimization for no-wait flowshop scheduling problem to minimize the total completion time, *Computers and Industrial Engineering,* **47,** 181-193.

Slowinski, R., Hapke, M., 2000. Scheduling under fuzziness, Physica-Verlag.

Solimanpur, M., Vrat, P., Shankar, R., 2004. Ant colony optimization algorithm to the inter-cell layout problem in cellular manufacturing, *European Journal of Operational Research,* **157,** 592-606.

Solnon, C., 2000. Solving permutation constraint satisfaction problems with artificial ants, *Proceedings of the 14th European Conference on Artificial Intelligence,* Berlin, Germany, August 20-25, 118-122.

Solomon, M, 1986. On the worst-case performance of some heuristics for the vehicle routing and scheduling problem with time window constraints, *Networks,* **16,** 161-174.

Solomon, M., 1987. Algorithms for the vehicle routing and scheduling problem with time windows constraints, *Operations Research,* **33**, 254-267.

Solomon, M., Desrosiers, J. 1988. Time window constrained routing and scheduling problems, *Transportation Science,* **22,** 1-13.

Stewart Jr., W.R., Golden B.L., 1983. Stochastic vehicle routing: a comprehensive approach, *European Journal of Opeerational Research*, **14**, 371-385.

Stützle, T., 1998a. Local Search Algorithms for Combinatorial Problems - Analysis, Algorithms and New Applications, *PhD thesis*, Technische Universitat Darmstadt, Darmstadt, Germany.

Stützle, T., 1998b. An ant approach to the flow shop problem, *Proceedings of the 6th European congress on Intelligent Techniques & Soft Computing, EUFIT'98,* Aachen, Germany, 1560-1564.

Stützle, T., Hoos, H.H., 2000. MAX-MIN ant system, *Future Generation Computer Systems,* **16(8),** 889-914.

Sung, S.C., Vlach, M., 2003. Single machine scheduling to minimize the number of late jobs under uncertainty, *Fuzzy Sets and Systems,* **139,** 421-430.

T'Kindt, Monmarche, V., Tercinet, F., Laugt, D., 2002. An ant colony optimization algotirhm to solve a 2-machine bicriteria flowshop scheduling problem, *European Journal of Operational Research,* **142,** 250-257.

Taillard, E.D., 1993. Parallel iterative search methods for the vehicle routing problem, *Networks,* **23,** 661-672.

Taillard, E.D., 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(2),** 170-186.

Taillard, E., Gambardella, L.M., Gendreau, M., Potvin, J-Y., 2001. Adaptive memory programming: a unified view of metaheuristics, *European Journal of Metaheuristics,* **115(1),** 227-241.

Talbi, E.G., Roux, O., Fonlupt, C., Robillard, D., 2001. Parallel ant colonies for the quadratic assignment problem, *Future Generation Computer Systems,* **17,** 441-449.

Tan, K.C., Lee, T.H., Ou, K., and Lee, H., 2001. A messy genetic algorithm for the vehicle routing problem with time windows, *Proceedings of the 2001 Congress on Evolutionary Computation*, Seoul, Korea, May 27-30, 679-686.

Tanaka, H., Okuda, T., Asai K., 1974. On fuzzy mathematical programming, *Journal of Cybernetics*, **3**, 37–46.

Tanaka, K., Vlach, M., 1997. Single machine scheduling with fuzzy due dates, *Seventh IFSA world Congress,* Prague, June 25-29.

Tarantilis, C.D., & Kiranoudis, C.T., 2002c. BoneRoute: an adaptive memory-based method for effective fleet management, *Annals of Operations Research,* **115(1),** 227-241.

Tarantilis, C.D., Ioannou, G., Prastacos, G., 2005a. Advanced vehicle routing algorithms for complex operations management problems, *Journal of Food Engineering*, **70**, 455-471.

Tarantilis, C.D., 2005b. Solving the vehicle routing problem with adaptive memory programming methodology, *Computers & Operations Research,* **32,** 2309-2327.

Teodorovic, D., Kikuchi, S., 1991. Application of fuzzy sets theory to the saving based vehicle routing algortihm, *Civil Engineering Systems,* **8**, 87-93.

Teodorovic, D., Pavkovic, G., 1992. A simulated annealing technique approach to the vehicle routing problem in case of stochastic, *Transportation Planning and Technology,* **16**, 261-273.

Teodorovic, D., Pavkovic, G., 1996. A fuzzy set theory approach to the vehicle routing problem when demand at nodes is uncertain, *Fuzzy Sets and Systems,* **82**, 307-317.

Thangiah, S., Nygard, K.E., Juell, P.L., 1991. GIDEON: A genetic algorithm system for vehicle routing with time windows, *Proceedings of the 7th IEEE Conference on Artificial Intelligence Applications*, Florida, February 1991, 322-328.

Thangiah, S., Osman, I, Sun, T., 1994. Hybrid genetic algorithm, simulated annealing and tabu search methods for vhicle routing problems with time windows, *Working Paper*, **UKC/IMS/OR94/4**, Institute of Mathematics and Statisticsi University of Kent, Canterbury, UK.

Thangiah, S., 1995. Vehicle routing problem with time windows using genetic algortihms, in *Application handbook of Genetic Algorithms: New Frontiers,* pp. 253-277, Ed. Chambers, L., CRC Press.

Tillman, F., 1969. The multiple terminal delivery problem with stochastic demands, *Transportation Science,* **3**, 192-204.

Toth, P., Vigo, D., 2002. The vehicle routing problem, SIAM Monographs on Discrete Mathematics an Applications, Philedelphia.

Toth, P., Vigo, D., 2003. The granular tabu search and its applicaiton to the vehicle-routing problem, *INFORMS Journal on Computing,* **15(4),** 333-346.

Tsujimura, Y., Gen, M., Kubota, E., 1995. Solving job-shop scheduling problem with fuzzy processing time using genetic algorithm, *Journal of Japan Society Fuzzy Theory Systems,* **7(5)**, 1073-1083.

Voudouris, C., Tsang, E., 1998. Guided local search, *European Journal of Operations Research*, **113,** 80-119.

Wang, C., Wang, D., Ip, W.H., Yuen, D.W., 2002. The single machine ready time scheduling problem with fuzzy processing times, *Fuzzy sets and Systems,* **127,** 117-129.

Warters, C.D.J., 1989. Vehicle-scheduling problems with uncertainty and omitted customers, *Journal of the Operational Research Society,* **40**, 1099-1108.

Yang, Y. Kamel, M.S., 2006. An aggregated clustering approach using multi-ant colonies algorithms, *Pattern Recognition,* **39,** 1278-1289.

Ying, K.C., Liao, C.J., 2004. Ant colony system for permutation flow-shop sequencing, *Computers and Operations Research,* **31,** 791-801.

Zadeh, L.A., 1965. Fuzzy sets, *Information and Control*, **8**, 338-353.

Zadeh, L.A., 1975. The concept of a linguistic variable and its application to approximate reasoning Part I, *Information Science*, **8,** 199-249.

Zadeh, L.A., 1978. Fuzzy sets as the basis for a theory of possibility, *Fuzzy Sets and Systems,* **1**, 3-28.

Zheng, Y., Liu, B., 2006. Fuzzy vehicle routing model with credibility measure and its hybrid intelligent algorithm, *Applied Mathematics and Computation*, **176**, 673-683.

Zimmermann, H., 1976. Description and optimization of fuzzy systems, *International Journal of General Systems,* **2**, 209-216.

Zimmermann, H., 1978. Fuzzy programming and linear programming with several objective functions, *Fuzzy Sets and Systems,* **1**, 45-55.

EK A

Dondo ve Cerda (2007) çalışması için yayımlanacak düzeltme metninin örneği aşağıdaki gibi olup Rodolfo Dondo tarafından tarihinde bilgi amaçlı olarak Sezgin KILIÇ'a gönderilmiştir.

Errata

We have been informed on two mistakes found in the paper "A Cluster-based Optimization Approach for the Muti-Depot Heterogeneous Fleet Vehicle Routing Problem with Time Windows" by Rodolfo Dondo and Jaime Cerdá. As the errors exist we must correct them and explain their causes.

First, in the schedule of the solution to problem C-102 (25 nodes), there are two time windows missed. Those are the time windows for customers T8 and T10. Neverthless, the clustering procedure correctly generated five clusters leading to a problem partition that leads to the optimal solution to the problem. In fact, if we correctly sequence the customers of cluster C^3, via a TSPTW, we obtain the same route reported for the vehicle V2. That is:

Depot – T7 - T8 - T10 - T11- T9 - T6 - T4 - T2 - T1- T3- T5 – Depot

Neverthless, arrival times are different because some errors are found on visiting times written in Table 8. So, although the route is optimal, the time windows of nodes T8 and T10 are missed just because of ill-specified arrival times. The correct visiting times to nodes are summarized on Table 8bis. Clearly, as the visits sequence is the same, the route presents the same travelled distance. The objective function value of the solution coincides with the one of the optimal solution found by solving the proposed model whitout the use of the clustering procedure. Consequently, it can be said that the hybrid algorithm was able to find the optimal solution.

It is worth noting that our reported objective function value of 190.7 units differs from the objective function value of 190.3 units reported on some papers like i.e. Kallehauge et al. (2001) because the authors rounded down to the nearest first decimal the inter-node distances while we computed these distances via the Euclidean norm.

Second, the reported solution to problem RC-101 (25 nodes) presents an error that arise from a different source. As mentioned in the paper, the combination of tight and randomly distributed time-windows together with approximated intra-cluster travel-times forces using more than one vehicle to service some clusters or to allow small time-windows violations. In our example, the clustering stage generated four clusters. One of them comprises customers T9, T10, T11, T12, T13, T14, T15, T16 and T17. If we try to sequence nodes of this cluster considering time-windows as hard constraints, no solution is feasible. On the contrary, if we use the objective function that considers time windows as penalized soft constraints, we found the solution reported on the Table 14 of the paper. This solution presents a violation to the time window of node T12 that Rodolfo Dondo was unable to detect when presenting the results. So, the travelled distance is exactly the same but there is a time-window violation that must be informed if we want to preserve the solution. In the end, we want to replace Table 14 by Table 14bis on which the time-window violation is included. Off-course the claim that we have found a new optimal solution is no longer valid.

Conclusion: We corrected two examples of the article "A Cluster-based Optimization Approach for the Muti-Depot Heterogeneous Fleet Vehicle Routing Problem with Time Windows" by Rodolfo Dondo and Jaime Cerdá. In the first example we changed nodes visiting times for the tour of vehicle V2 in order to correctly specify the found optimal solution. In the second example, we reported an un-informed time windows violation to the node T12.

In the end, the conclusions of the paper remain unaltered since our hybrid procedure is able to provide optimal solutions or near optimal solutions for C-class problems of different sizes and to provide sub-optimal solutions that remain within acceptable limits for RC and R-class problems.

Acknowledgments: We want to thank to Sezgin Kilic from the Industrial Engineering Department of the Turkish Air Force Academy for their observations.

References:

Kallehauge, B., J. Larsen, O. Madsen. Lagrangean duality applied on vehicle routing with time windows. Experimental results. Technical Report IMM-TR-2001-9, IMM (2001)

EK B

Tablo B.1 ve Tablo B.2'de 25 müşterilik Solomon (1987) örnek problemlerine KKS-ZAARP algoritması ile 500 döngü içeren 5 koşum sonucunda elde edilen en iyi sonuçlar ile kesin çözüm yöntemleri ile bulunmuş olan optimal sonuçlar karşılaştırılmıştır. Kesin çözüm yöntemlerinde tek amaç yolculuk mesafesinin azaltılması olduğundan altıncı sütundaki fark değerleri sadece aynı araç sayısında çözüm elde edilmesi durumunda hesaplanmıştır. Ayrıca, KKS-ZARP algoritması ile elde edilen çözümlerin yolculuk mesafeleri optimal çözümlerin elde edildiği çalışmalardakine benzer olarak ilk ondalık basamağı tabana yuvarlanan mesafe değerleri ile hesaplanmıştır. KKS-ZARP algoritması ile 25 müşterilik problemler için 500 döngü içeren bir koşum süresi yaklaşık olarak 2.5 dakikadır.

Tablo B.1: 25 müşterilik C problem grubu için elde edilen sonuçlar

Problem	Optimal		ZAARP-KKS		% fark mesafe
	Araç sayısı	Mesafe	Araç sayısı	Mesafe	
C101	3	191,3	3	191,3	0
C102	3	190,3	3	190,3	0
C103	3	190,3	3	190,3	0
C104	3	186,9	3	186,9	0
C105	3	191,3	3	191,3	0
C106	3	191,3	3	191,3	0
C107	3	191,3	3	191,3	0
C108	3	191,3	3	191,3	0
C109	3	191,3	3	191,3	0
C201	2	214,7	2	214,7	0
C202	2	214,7	1	222,5	-
C203	2	214,7	1	222,5	-
C204	2	213,1	1	243,4	-
C205	2	214,7	1	296,7	-
C206	2	214,7	1	284,5	-
C207	2	214,5	1	274,0	-
C208	2	214,5	1	230,4	-

Tablo B.2: 25 müşterilik R problem grubu için elde edilen sonuçlar

Problem	Optimal		ZAARP-KKS		% fark mesafe
	Araç sayısı	Mesafe	Araç sayısı	Mesafe	
R101	8	617,1	8	617,1	0
R102	7	547,1	7	547,1	0
R103	5	454,6	5	454,6	0
R104	4	416,9	4	416,9	0
R105	6	530,5	6	530,7	0,04
R106	5	465,4	5	465,4	0
R107	4	424,3	4	428,4	0,97
R108	4	397,3	4	397,3	0
R109	5	441,3	5	441,3	0
R110	4	444,1	4	445,7	0,36
R201	4	463,3	2	523,7	-
R202	4	410,5	2	455,5	-
R203	3	391,4	2	399,5	-
R204	2	355,5	1	416,1	-
R205	3	393,0	1	416,2	-
R206	3	374,4	1	444,9	-
R207	3	361,6	1	423,2	-
R208	1	328,2	1	394,7	20,26
R209	2	370,7	1	433,8	-
R210	3	404,6	2	495,5	-

Tablo B.3 ve Tablo B.4'te C1 ve C2 grubu 100 müşterilik Solomon (1987) örnek problemlerine KKS-ZAARP algoritması ile 1000 döngü içeren 5 koşum sonucunda elde edilen en iyi sonuçlar ile geçmiş çalışmalarda sezgisel çözüm yöntemleri ile bulunmuş olan en iyi sonuçlar karşılaştırılmıştır. KKS-ZARP algoritması ile 100 müşterilik problemler için 1000 döngü içeren bir koşum süresi yaklaşık olarak 53 dakikadır.

Tablo B.3: 100 müşterilik C1 problem grubu için elde edilen sonuçlar

Problem	Sezgisel en iyi		ZAARP-KKS		% fark araç	% fark mesafe
	Araç sayısı	Mesafe	Araç sayısı	Mesafe		
C101	10	828,94	10	828,94	0	0
C102	10	828,94	10	842,70	0	1,66
C103	10	828,06	10	978,59	0	18,18
C104	10	824,78	10	927,00	0	12,39
C105	10	828,94	10	824,94	0	0
C106	10	828,94	10	828,94	0	0
C107	10	828,94	10	951,80	0	14,82
C108	10	828,94	10	856,32	0	3,30
C109	10	828,94	10	995,63	0	20,11
		Ortalama	10	892,76		

Tablo B.4: 100 müşterilik C2 problem grubu için elde edilen sonuçlar

Problem	Sezgisel en iyi		ZAARP-KKS		% fark araç	% fark mesafe
	Araç sayısı	Mesafe	Araç sayısı	Mesafe		
C201	3	591,56	3	591,56	0	0
C202	3	591,56	3	591,56	0	0
C203	3	591,17	3	669,52	0	13,25
C204	3	590,60	3	593,16	0	0,43
C205	3	588,88	3	588,88	0	0
C206	3	588,49	3	588,49	0	0
C207	3	588,29	3	590,24	0	0,33
C208	3	588,32	3	756,05	0	23,41
		Ortalama	3	621,18		

ÖZGEÇMİŞ

Sezgin Kılıç, 08.11.1976 tarihinde Turhal'da doğdu. 1987'de Turhal Kahraman Cengiz Topel İlkokulu'ndan mezun oldu. Orta ve lise öğrenimini birer yılı Amasya Anadolu Lisesi ve Tokat Anadolu Lisesi'nde olmak üzere 1994 yılında Ankara Atatürk Anadolu Lisesi'nde tamamladı. 1998 yılında Hava Harp Okulu, Endüstri Mühendisliği Bölümü'nde lisans eğitimini tamamladı. 1999-2001 yılları arasında ikmal subayı olarak görev yaptı. 2003 yılında Hava Harp Okulu, Havacılık ve Uzay Teknolojileri Enstitüsü, Endüstri Mühendisliği Bölümü'nde yüksek lisansını tamamladı. 2003-2007 yıllları arasında öğretim elemanı olarak Hava Harp Okulu, Endüstri Mühendisliği Bölümü'nde görev yaptı. 2003 yılı Güz döneminde İstanbul Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Endüstri Mühendisliği Bölümü'nde doktora eğitimine başladı. Halen Genelkurmay Başkanlığı'nda ikmal subayı olarak görev yapmaktadır.