YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ARAÇ ROTALAMA PROBLEMLERİ İÇİN POPULASYON ve KOMŞULUK TABANLI METASEZGİSEL BİR ALGORİTMANIN TASARIMI ve UYGULAMASI

Endüstri Müh. Vural EROL

FBE Endüstri Mühendisliği Anabilim Dalı
Sistem Mühendisliği Programında
Hazırlanan

YÜKSEK LİSANS TEZİ

Tez Danışmanı: Yrd. Doç. Dr. Tufan DEMİREL
Yrd. Doç. Dr. Bahadır GÜLSÜN
Yrd. Doç. Dr. Banu DİRİ

İSTANBUL, 2006

İÇİNDEKİLER

Sayfa

SİMGE LİSTESİ

$[a_i, b_i]$	Zaman penceresi
C	Araç kapasite değeri
C_k	k nolu rotanın merkez noktası
d_{ij}	i. ve j. noktalar arası mesafe
f	Problem için amaç fonksiyonu
G	Genetik Algoritmada nesil sayısı
G_{AB}	A ve B çözümleri arasındaki alanın genişlik miktarı
H	Tabu listesi
$\lvert H \rvert$	Tabu listesinin büyüklüğü
L	Araç mesafe kısıtı
m	Araç Rotalama Probleminde araç sayısı
n	Problemde yer alan nokta sayısı
N	Genetik Algoritmada populasyon büyüklüğü
N(x)	x çözümünün komşu çözümlerinin kümesi
$O(n)$	Yöntemin algoritmik karmaşası
ρ	Feramon buharlaşma oranı
q_i	i nolu müşterinin talep miktarı
S	Algoritmada iterasyonlarda ele alınan mevcut çözüm
s_i	Araçların servis süresi
s_{ij}	i ve j noktaları için tasarruf değeri
τ_{ij}	i ve j noktaları arasında yer alan yol üzerindeki feramon miktarı
v_i	Problemdeki i nolu nokta
V	Problemde yer alan tüm noktaların kümesi
(x_i, y_i)	i noktasının Öklidyen düzlemde koordinatları

KISALTMA LİSTESİ

AH	Adaptif Hafıza
ARP	Araç Rotalama Problemi
AUARP	Açık Uçlu Araç Rotalama Problemi
AYARP	Asimetrik Yollu Araç Rotalama Problemi
CMT	Christofides, Mingozzi ve Toth'un 1979'da hazırladıkları 14 problem
DARP	Dinamik Araç Rotalama Problemi
EYK	En Yakın Komşu Yöntemi
GA	Genetik Algoritmalar
GSP	Gezgin Satıcı Problemi
GVR	Genetic Vehicle Routing kodlama tekniği
HYP	Hamilton Yol Problemi
KKARP	Kapasite Kısıtlı Araç Rotalama Problemi
KKO	Karınca Kolonisi Optimizasyon Algoritması
KOP	Kombinatoryal Optimizasyon Problemi
KUARP	Kapalı Uçlu Araç Rotalama Problemi
KYARP	Karma Yüklemeli Araç Rotalama Problemi
LRP	Lokasyon Rotalama Problemi
MKARP	Mesafe Kısıtlı Araç Rotalama Problemi
MTFARP	Müşteri Tipi Farklı Araç Rotalama Problemi
NP	Non Polinomial problemler
PT	Paralel Tasarruf Yöntemi
SARP	Statik Araç Rotalama Problemi
SYARP	Simetrik Yollu Araç Rotalama Problemi
ST	Sıralı Tasarruf Yöntemi
TA	Tabu Arama
TB	Tavlama Benzetimi
ZPARP	Zaman Pencereli Araç Rotalama Problemi

ŞEKİL LİSTESİ

ÇİZELGE LİSTESİ

ÖNSÖZ

Günümüzün küreselleşen dünyasında işletmelerde lojistik ve tedarik zinciri yönetiminde toplam maliyetlerin önemli bir oranını taşıma maliyetleri oluşturmaktadır. Araç filolarının kapasite ve diğer kısıtları dikkate alınarak hazırlanan iyi bir plan, lojistik maliyetlerini önemli ölçüde düşürecek ve firmalara rekabette avantaj sağlayacaktır. Bu çalışmada araç rotalarının planlanmasında bu amaçla geliştirilmiş yöntemlere yer verilmekte ve bir metasezgisel yöntem sunulmaktadır.

Yapmış olduğum çalışmada her konuyu ayrıntısı ile olabildiğince açık ve Türkçe terimlerle ifade etmeye özen gösterdim, görsel anlatımın en iyi anlatım şekli olduğuna inandığımdan şekillerle ve verdiğim örneklerle ifadeleri pekiştirdim. Bu tez çalışmasının, konuyla ilgili daha sonra yapılacak çalışmalar için de faydalı bir referans teşkil etmesini umuyorum.

Tez çalışmamın hazırlanması sırasında bana verdikleri manevi destekten dolayı aileme, yakın dostlarıma, iş arkadaşlarıma ve bana sonuna kadar güvenen ve bilimsel disiplin ve bakış açısı kazanmamı sağlayan danışman hocam Yrd. Doç. Dr. Tufan Demirel'e teşekkürü borç bilirim.

İstanbul, 2006 Vural EROL

ÖZET

Bu tez çalışmasında firmalarda özellikle lojistik planlarının oluşturulması sırasında epeyce sık karşılaşılan kapasite ve mesafe kısıtlı Araç Rotalama Problemi için, populasyon ve lokal arama tabanlı bir metasezgisel algoritma önerilmiştir. Geliştirilen yöntem için populasyon tabanlı metasezgisel tekniklerin çeşitlendirme stratejisi, komşuluk tabanlı tekniklerin ise yoğunlaştırma stratejisi göz önüne alınmıştır. Klasik sezgisel metotlar kullanılarak oluşturulan başlangıç çözümleri çözüm havuzu içerisinden İkili Turnuva mantığı ile seçilmekte ve bu çalışmada tanıtılan "arama uzayında çözümler arası uzaklık" kavramı ile komşu çözümler rasgele bir şekilde taranarak yeni çözümler üretilmektedir. Böylelikle arama işlemi, çözüm uzayının en umut verici alanlarında yapılmakta ve üretilen çözüm kalitesi artırılmaktadır. Ayrıca algoritma süreci tamamen stokastik bir yapıda ilerlemekte ve bu sayede yöntem, oldukça az sayıda parametre içermektedir. Önerilen yöntem literatürde yer alan problemler üzerinde denenmiş ve başarılı sonuçlar alınmıştır.

Anahtar Kelimeler: Araç Rotalama Problemi, Metasezgisel Algoritmalar

ABSTRACT

In this thesis study, a population and neighborhood search based metaheuristic algorithm is proposed for capacity and distance restricted Vehicle Routing Problem, which is usually encountered during logistic planning in companies. When developing this method, population based techniques' diversification and local search based techniques' intensification strategies are taken into account. Initial solutions are produced using classical heuristics and individuals are selected from solution pool with Binary Tournament sense. New generation is created by scanning neighborhood solutions randomly, considering "distance between solutions in search space" concept introduced in this study. In this manner, search can be driven in solution space's most promising regions and solutions quality is enhanced. Furthermore, algorithm process has a stochastic structure providing to set fairly few parameters. Proposed method is put into practice for several problems in literature and produce successful outputs.

Keywords: Vehicle Routing Problem, Metaheuristic Algorithms

1. GİRİŞ

Organizasyonların global pazar koşullarının zorlu rekabet ortamında rakiplerinden önde olabilmesi için müşterilerine ürün ve hizmetlerini en hızlı şekilde ulaştırmaları gerekmektedir. Günümüzde küreselleşme kapsamında firmalar ürünlerini dünyanın dört bir yanında satabilir duruma gelmişler, etkin dağıtım kanalları ile daha geniş bölgelere ulaşım imkanı elde etmişlerdir. Bu noktada tedarik zinciri ve lojistik yönetimi kavramlarının önemi ortaya çıkmaktadır. İşletmeler pazarda lider olabilmek için iyi düzenlenmiş bir lojistik planı ile gecikmeleri tamamen ortadan kaldırarak müşteri memnuniyetini artırmak zorundadır. Müşteri memnuniyeti için çaba sarf edilirken diğer bir yandan, maliyetleri olabildiğince azaltacak yöntemlere başvurulmalıdır.

Özellikle bayii ve kurye sistemiyle çalışan firmalarda değişikliklere karşı esnek, aynı zamanda düşük maliyetli sonuçlar veren rota planların hızlı bir şekilde hazırlanması müşterilerin memnuniyeti ve işletme kârlılığı açısından oldukça önemlidir. Bu noktada maliyet ile kullanılan araç sayısı, araçların gideceği toplam mesafe ve toplam süre gibi ölçütler göz önüne alınmaktadır. Önemli olan bir başka nokta ise, dinamik olarak yeni müşteri taleplerinin ortaya çıkması veya araçlarda oluşabilecek bir arıza durumunda rota planının hızlı bir şekilde yeniden yapılandırılabilmesidir. Bu sebeple esnek araç rotalama sistemleri lojistik planlarının daha önceden hazırlanmasını, hızlı bir şekilde tekrar düzenlenmesini ve bu da depolarda ürünlerin daha az tutulmasını sağlayacaktır.

Araç Rotalama Problemi (ARP) genel olarak bir işletmenin konumları belirli n adet müşterisine (talep noktasına) tam olarak servis sunabilmesi için bazı operasyonel kısıtların göz önüne alındığı ve minimum maliyetin amaçlandığı araç rotalarının belirlenmesi problemidir. İşletmelerde özellikle ürünlerin dağıtımı sırasında karşılaşılan bu problem, bazı sektörlerde oldukça yüksek maliyetlere neden olmaktadır. Bu sebeple ARP'nin etkili bir şekilde çözümü büyük tasarruflar sağlanması açısından önemlidir. Dağıtım sisteminde yer alan kısıtlara göre çeşitli ARP türleri bulunmaktadır. Bu kısıtların en önemlileri araç kapasite kısıtı ve aracın bir defada yol alabileceği maksimum mesafe ve süre kısıtıdır.

ARP çözümü oldukça zor bir kombinatoryal optimizasyon problemidir. Problemin çözüm uzayı, problemde yer alan müşterilerin sayısı ile üstel orantılı olarak genişler. Literatürde ARP'ye çözüm üretmek için geliştirilen birçok yöntem bulunmaktadır. Bu çalışmalar Kesin ve Yaklaşık yöntemler olarak ikiye ayrılmaktadır. Kesin yöntemler küçük problemler için optimum sonuç bulmasına rağmen, problem boyutu büyüdüğünde oldukça fazla hesaplama

gücüne gereksinim duymaktadırlar. Yaklaşık yöntemler ise büyük problemlerde daha az işlemle ve hesaplama süresiyle optimuma yakın, iyi kalitede çözümler üreten sezgisel yöntemlerdir. ARP için en etkili sonuçlar veren algoritmalar metasezgisel algoritmalardır. Metasezgisel yöntemler literatürde son zamanlarda oldukça uygulama alanı bulmuş ve başarılı sonuçlar üretmiş tekniklerdir. Kombinatoryal problemlere iyi kalitede çözümler oluşturan bu tekniklerde çoğunlukla bir veya birden fazla çözüm ele alınıp, bu çözümlerden iteratif olarak daha iyi çözüm üretilmektedir. Bu yöntemlerde süreç, belirlenen komşuluk yapısı ile bir çözümün komşu çözümleri göz önüne alınarak bir arama stratejisi ile yürütülmekte ve belirlenen bir durdurma kriteri ile sonlandırılıp, üretilen çözüm kullanıcıya sunulmaktadır.

Metasezgisel algoritmaların, yerel komşuluk tabanlı ve populasyon tabanlı olmak üzere iki türü bulunmaktadır. Komşuluk tabanlı algoritmalar arama sürecini bir çözüm ile yürütmekte ve belirlenen komşuluk yapısı ile mevcut çözümü iyileştirmeye çalışmaktadırlar. Bu tip yöntemlere, Tabu Arama ve Tavlama Benzetimi teknikleri örnek olarak verilebilir. Populasyon tabanlı algoritmalar ise aynı anda birden fazla çözümü ele alıp, bu çözümlerin özelliklerinden faydalanarak yeni çözümler üretmektedirler. Genetik Algoritma ve Karınca Kolonisi Optimizasyon Algoritması bu tür yöntemlerdendir. Literatürde çoğu metasezgisel yöntemin ARP için uygulaması bulunmaktadır.

Bu çalışmada ilk olarak (ikinci bölümde) Araç Rotalama Problemi irdelenmekte ve literatürde yer alan temel Araç Rotalama Problemi türleri açıklanmaktadır.

Üçüncü bölümde ise ARP ile ilgili günümüze kadar süregelen klasik sezgisel yöntemler belirtilmektedir.

Dördüncü bölümde, literatürde sık sık karşılaşılan metasezgisel yöntemler ve süreç yapıları açıklanmaktadır.

Beşinci bölümde, ARP ile ilgili literatürde yer alan metasezgisel metotlar ayrıntılı bir şekilde incelenmekte ve bu tekniklerle ilgili örnekler verilmektedir.

Altıncı bölümde ise ARP için geliştirilen populasyon ve komşuluk tabanlı bir metasezgisel yöntem detaylarıyla birlikte açıklanmakta ve literatürde yer alan standart problemlere uygulanarak diğer metasezgisel yöntemlerle karşılaştırılması yapılmaktadır.

Son bölümde ise tez çalışması özetlenerek sonuç ve değerlendirme yer almaktadır.

2. ARAÇ ROTALAMA PROBLEMİ

Araç Rotalama Problemi (ARP) ilk olarak 1959 yılında Dantzig ve Ramser tarafından literatüre kazandırılmıştır. Yazarlar bu çalışmalarında benzin istasyonlarına benzin dağıtımı problemi üzerinde durmuşlar ve problemin çözümü için ilk matematiksel programlama modelini kurmuşlardır. Daha sonra 1964 yılında Clarke ve Wright probleme sezgisel bir çözüm önermiş ve bu çalışmadan sonra literatürde ARP'ye ilgi daha da artarak büyümüştür. ARP şu ana kadar üzerinde en fazla yöntem geliştirilen optimizasyon problemlerinden biridir (Toth ve Vigo, 2002).

Klasik ARP'de aynı tip ve kapasiteye sahip olan homojen bir araç filosu, merkezi bir depoda (dağıtım merkezinden) hareket ederek talepleri önceden bilinen bir grup müşteriye hizmet vermektedir. Bu problemde, her müşteri sadece bir araçtan hizmet almakta ve her araç sadece bir rota izlemektedir. Araçlar için kapasite kısıtının yanı sıra, araçların depodan hareket edip rotanın sonunda depoya geri dönmesi zorunluluğu vardır. Söz konusu koşullar altında toplam yolculuk maliyetini minimize eden rotalar kümesinin bulunması amaçlanmaktadır. ARP dağıtım yönetimi ve lojistik alanında önemli bir yol oynamaktadır (Alabaş ve Dengiz, 2004).

Taşıt rotalama konusu 1950'li yıllarda bilim adamlarının ilgi alanına girmeye başlandığı gözlenmiştir. 1970'li yılların başında ortaya çıkan petrol krizine karşın taşımacılıkta hızlı gelişmeler kaydedilmesi bu konuda yapılan çalışmalara yoğunluk kazandırmıştır. 1980'li yıllarda ekonomik sorunların yanı sıra, ulaştırma türleri ve işletmeleri arasında giderek artan ve kırıcı bir rekabet ortaya çıkmıştır. Günümüzde düşük maliyetlerle daha çok ve daha iyi ulaştırma hizmetlerinin sağlanabilmesi işletmecileri yakından ilgilendirmektedir (Erel, 1995).

İşletmelerin toplam lojistik ve dağıtım maliyetlerinin 1/3 – 2/3'ü taşıma maliyetlerinden kaynaklandığından dağıtım ekipmanının ve personelinin etkili ve verimli bir şekilde kullanılması işletme yöneticileri açısından önemli bir ilgi alanı haline gelmiştir. Dağıtım maliyetlerini azaltmak ve müşterilere sunulan servisin kalitesini artırmak için en kısa zamanı yada mesafeyi verecek olan, bir aracın şebeke içerisinde izleyeceği en uygun rotayı bulmak günümüzde en çok tartışılan bir konu haline gelmiştir. Standart bir araç rotalama probleminde depolardan araçlar vasıtasıyla değişik noktalarda bulunan müşteri noktalarının talepleri karşılanmaya çalışılmaktadır. Bunu gerçekleştirirken amaç etkili ve verimli bir şekilde müşteri ihtiyaçlarını mümkün olan en kısa zamanda, en kısa yoldan ve en az maliyetle karşılayan rotayı belirlemektir. İşletmelerde talep noktalarına hizmet vermek için araç rotalama yapılırken aşağıdaki unsurlar dikkate alınmalıdır (Karahan, 2003):

- Şebeke içersinde bulunan müşterilerin talepleri tamamıyla karşılanmalıdır.
- Şebekedeki her varış noktası tek bir araç tarafından sadece bir defa ziyaret edilmelidir.
- Rota depodan başlamalı ve tekrar depoda sonlamalıdır.
- Rota üzerinde bulunan müşterilerin toplam talep miktarı aracın toplam kapasitesinden fazla olmamalıdır.
- Her bir araç sadece bir rota üzerinde faaliyet göstermektedir.
- Araç rotalamanın temel amacı araçların kat edecekleri toplam mesafenin minimize edilmesi olmalıdır.

Bir ARP'de yer alan ana kavramlar aşağıda açıklanmaktadır:

Araçların İlerlediği Yol Şebekesi: ARP'de talep noktaları ve işletme birimleri (depo, dağıtım merkezi, bayi, durak, istasyon, vs.) bir graf üzerindeki noktalar ile gösterilmekte olup, bu noktalar arasındaki doğrular ise müşteriler arası veya müşteri-işletme birimi arası yolları temsil etmektedir. Bu grafta çözümde yer alan her doğru, amaç fonksiyonu değerini (maliyeti) uzunluğu ile orantılı olarak belirli bir ölçüde etkilemektedir.

Müşteriler (Talep Noktaları): Problemde müşteriler belirli bir talep miktarı ve lokasyona sahiptirler. Her müşterinin birbirleri ile ve işletme birimi ile arasındaki mesafesi bilinmektedir. Bir ARP çözümünün uygun bir çözüm olarak kabul edilmesi için tüm müşterilerin talepleri karşılanmak zorundadır. Bazı ARP türlerinde müşterilerin önceliklendirilmesi yapılabilmektedir.

İşletme Birimi: Müşterilere hizmetin götürüldüğü işletmeye ait noktalardır. Araçlar işletme birimlerinden kalkarak müşterilerin taleplerini karşılayıp tekrar işletme birimlerine dönerler. İşletme birimleri gerçek hayatta depolar, dağıtım merkezleri, bayi, vb. üniteleridir. Bazı ARP türlerinde bir tane işletme birimi olduğu halde, bazılarında birden fazla olabilmektedir. Birden fazla işletme biriminin yer aldığı problemlerde birimlerin kapasite kısıtı da (karşılayabileceği maksimum talep miktarı) göz önüne alınabilmektedir.

Araç Filosu: ARP'de araçların hareketleri maliyet oluşturmaktadır ve genellikle rotası planlanacak araç sayısı değişkendir. Araçların yük kapasitesi veya gidebileceği maksimum yol, yükleme boşaltma gibi kısıtları da olabilmektedir. Bazı problemlerde homojen diğer bir deyişle aynı kapasiteli ve kısıtlı araçlarla, bazı problemlerde ise kapasiteleri farklı heterojen araçlarla rotalama yapılmaktadır.

Sürücüler: Bazı ARP türlerinde sürücülere ait çalışma periyodu, fazla mesai, öğle arası gibi kısıtlar da yer alabilmektedir. Araç rotalarının çıkarılmasında bu kısıtlar da dikkate alınmalıdır.

Gerçek hayatta lojistiğin doğası gereği, kalite düzeyini artırmak ve müşteriye daha iyi servis sunabilmek amacıyla ARP'de çeşitli operasyonel kısıtlar bulunabilmektedir. Bu kısıtlara örnek olarak aşağıdakiler verilebilir:

- Her rotaya atanan aracın yükleme ve/veya gidebileceği yol kapasitesi
- Müşteriye servis süresi (Araç yükleme, boşaltma zamanları)
- Müşteriye sadece belirli zaman aralıklarında servis sunulabilmesi (Zaman penceresi kısıtı)
- Rotalama yapılırken sürücünün çalışma saatlerinin göz önüne alınması
- Ziyaret edilecek müşterilerin veya müşteri tiplerinin önceliklendirilmesi
- Araç sayısının belirli sayıda veya belirli bir sayıdan küçük olması

Bir Araç Rotalama probleminde olabilecek tipik amaç fonksiyonları ise aşağıdaki gibidir (Toth ve Vigo, 2002):

- Araçların toplam gideceği mesafeyi veya seyir süresini azaltmaya çalışarak global taşıma maliyetlerinin minimize edilmesi.
- Tüm müşterilerin taleplerini karşılamak koşuluyla sistemdeki araç sayısının minimize edilmesi.
- Araç yüklemeleri veya araç seyir süreleri açısından tüm rotaların dengelenmesi.

Bir ARP çözüldüğünde her müşteri noktasına sadece bir kez uğranıldığı ve her rotada sadece bir araç bulunduğu rotalar kümesi elde edilir. Bu rota kümesi ile tüm operasyonel kısıtlar sağlanmakta ve ayrıca taşıma maliyeti minimize edilmeye çalışılmaktadır. Klasik bir ARP modeli genel olarak şu şekilde tanımlanabilmektedir: G = (V, E) bir grafı, V = $\{v_0 , v_1, ...,v_n\}$ bir nokta kümesini ve E = $\{(v_i , v_j): v_i, v_j \in < V, i \neq j\}$ bir kenar kümesini göstersin. V kümesinde v_0 merkez depoyu, diğer n sayıda nokta ise müşterileri ifade etmektedir. Her müşteri negatif olmayan bir q_i talebine sahiptir ve her biri C kapasiteli m araçtan oluşan bir araç filosu depoda bulunmaktadır. Literatürde bu kapasite kısıtının yer almadığı problemlere Çoklu Gezgin Satıcı Problemi adı verilmektedir. ARP'de temel amaç tüm müşterilere hizmet götürmek için her aracın gideceği güzergahı çizmek, diğer bir deyişle m adet rota

belirlemektir. Kuşkusuz bunu yaparken de kısıtlara uyularak maliyetin minimize edilmesi arzulanmaktadır. Tek depolu klasik bir ARP'nin doğrusal modeli aşağıdaki gibi formüle edilebilir:

M: Araç sayısı

N: Müşteri sayısı

$d_{ij:}$ Nokta i ve nokta j arasındaki mesafe

q_i: Müşteri i'nin talep miktarı

X_{ijk}: $\left\{ \begin{array}{l} 1 \text{ , eğer k nolu araç i noktasından j noktasına hareket ederse} \\ 0 \text{ , aksi takdirde} \end{array} \right\}$

Amaç Fonksiyonu: $\text{Min } Z = \sum_{i=0}^{N} \sum_{j=0,\, j\neq i}^{N} \sum_{k=1}^{M} d_{ij}\, X_{ijk}$ (2.1)

Şu kısıtlara göre:

$i = 0$ için $\sum_{k=1}^{M} \sum_{j=1}^{N} X_{ijk} = M$ (2.2)

$i \in \{1,..., N\}$ için $\sum_{k=1}^{M} \sum_{j=0, j\neq i}^{N} X_{ijk} = 1$ (2.3)

$j \in \{1,..., N\}$ için $\sum_{k=1}^{M} \sum_{i=0, i\neq j}^{N} X_{ijk} = 1$ (2.4)

$k \in \{1,..., M\}$ için $\sum_{i=1}^{N} X_{i0k} \leq 1$ (2.5)

$k \in \{1,..., M\}$ için $\sum_{i=1}^{N} q_i \sum_{j=0, j\neq i}^{N} X_{ijk} \leq C$ (2.6)

Amaç fonksiyonu (2.1) toplam kat edilecek mesafenin yani maliyetin minimize edilmesi gerektiğini ifade etmektedir. 2.2 nolu kısıt denklemi işletme biriminden çıkacak araç sayısının M adet olduğunu, 2.3 ve 2.4 nolu kısıt denklemleri bir müşterinin mutlaka bir araç tarafından ziyaret edilmesi ile müşteriye gelen ve müşteriden çıkan yollardan sadece bir tanesinin kullanılması zorunlu olduğunu, 2.5 nolu kısıt denklemi bir aracın ancak bir defa işletme

biriminden çıkacağını dolayısıyla bir defa rotalamada kullanılacağını ve son olarak 2.6 nolu kısıt denklemi ise araçlara yapılan yüklemelerin araç kapasite değeri C'yi geçmemesini belirtmektedir. Bazı problemlerde araç sayısı kısıtı olarak en fazla M adet aracın kullanılması gerektiği yer almaktadır. Bu durumda 2.2 nolu denklemde eşitlik ifadesi yerine küçük eşit ($\leq$) işareti bulunacaktır. Literatürde karşılaşılan çoğu ARP'de ise araç sayısı ile ilgili bu kısıta yer verilmemektedir. Modelde yer alan temel kısıtlar olan 2.3 ve 2.4 nolu denklemler rotaların sürekliliğini sağlaması açısından önemlidir. Bir ARP'nin çözümünde genel olarak aşağıdaki bilgilere ihtiyaç vardır:

- Her müşteriden diğer müşterilere ulaşım süresi veya aralarındaki mesafe
- İşletme birimlerinden her müşteriye ulaşım süresi veya aralarındaki mesafe
- Talep noktalarındaki talep miktarı
- Araç sayısı ve araç kapasite değeri
- Optimize edilmesi gereken unsur veya unsurlar (amaç fonksiyonu)

ARP'nin çözümünde çok çeşitli yöntemler geliştirilmiştir. Bunlar arasında Kesin yöntemler optimum sonuç vermektedir fakat sadece küçük boyutlu ARP'ye uygulanabilmekte ve çok fazla işlem gereksinimi olduğu için çözüm süreci uzun sürede gerçekleşmektedir. Kesin yaklaşımlar kendi içinde Dal-Sınır, Dal-Kesme, Dinamik Programlama ve Tamsayılı Programlama olmak üzere dört sınıfa ayrılmaktadır. Simülasyon, özellikle taleplerin dinamik olarak değiştiği ARP için kullanılan bir yöntem olup, daha çok gerçek hayatta problemde yer alan darboğazları bulmak için kullanılmaktadır. ARP için geliştirilen en etkili yöntemler ise sezgisel algoritmalardır. Sezgisel algoritmalar oldukça kısa sürelerde optimuma yakın sonuçlar üretebilmektedir (Cordeau vd., 2004).

2.1 Araç Rotalama Problemi Türleri

ARP'ye uygun çözümler bulmak gün geçtikçe zorlaşmaktadır. Bunun nedeni artan rekabet şartları ve değişen çevre koşulları sebebiyle rotalama problemlerine giderek daha fazla kısıtın eklenmesidir. Zaman aralıkları (varış zamanı, servis zamanı, bekleme zamanı ve kalkış zamanı), farklı kapasitelere sahip çok sayıda araç, rota üzerinde müsaade edilebilen seyahat süresi, farklı noktalar arasında farklı hızların olması, araç sürücüleri için dinlenme zamanları araç rotalamada göz önüne alınması gereken kısıtlar olarak karşımıza çıkmaktadır. Kullanılan kısıtların fazlalaşması, ARP'yi giderek daha da karmaşık hale getirmektedir (Erel, 1995).

Araç Rotalama Problemleri çeşitli kriterlere göre sınıflandırılabilmektedir. Bu kriterler çoğunlukla problemin bileşenleri olan kısıtların, araçların, müşterilerin, yolların ve rotaların özellikleri ile ilgilidir.

2.1.1 Dinamik ve Statik Çevre Durumuna Göre Araç Rotalama Problemleri

En çok kullanılan sınıflandırma kriterlerinden biri, olayların statik veya dinamik başka bir deyişle bilgilerin deterministik veya stokastik olarak ele alınması durumudur. Bu kritere göre Araç Rotalama Problemleri, Statik ARP ve Dinamik ARP olarak ikiye ayrılır:

2.1.1.1 Statik Araç Rotalama Problemi

Bu tür problemlerde, problem çözülmeden önce gerekli tüm bilgiler (kısıtlar, talepler, kapasiteler, maliyet bilgileri vb) bilinmektedir ve bu bilgiler problemin çözüm aşamasında da değişkenlik göstermez, sabittir. Literatürde çoğunlukla Statik Araç Rotalama Problemi (SARP) üzerinde çalışmalar yapılmış olup bu problem, deterministik ARP olarak da karşımıza çıkmaktadır. Gerçek hayatta SARP çözüm yöntemleri, önceden miktarı ve zamanı bilinen talepler için rota planları oluşturmada ve servis sistemlerinin genel olarak değerlendirilmesinde kullanılmaktadır.

2.1.1.2 Dinamik Araç Rotalama Problemleri

Dinamik Araç Rotalama Problemi (DARP) lojistik süreci ilerlerken aniden yeni talep noktalarının ortaya çıkması, müşterinin talep miktarının değişmesi, bazı yolların kapanması veya trafik sıkışıklığı sebebiyle ulaşım süresinin uzaması gibi beklenmeyen durumlar karşısında hızlı bir şekilde yeni kararların verilmesini gerektirecek durumlarda ortaya çıkmaktadır. ARP'nin dinamik olabilen elemanları özellikle yeni bir müşteri talebinin ortaya çıkması ve stokastik sapmalara bağlı dinamik seyahat süreleridir. DARP gerçek hayatta müşterilerine kurye, saha satış vb. yoluyla hizmetler sunan işletmelerde önemli problemlerden biridir (Potvin vd., 2004).

2.1.2 Rotaların Durumlarına Göre Araç Rotalama Problemleri

Araç Rotalama Problemleri, bir aracın bir işletme birimi için çalışması diğer bir deyişle rotaların bir işletme biriminde başlayıp aynı işletme biriminde sona ermesi veya aracın işletme birimlerinden bağımsız olup seyir güzergahının en son müşteride bitirilebilmesi durumlarına göre açık ve kapalı uçlu olmak üzere ikiye ayrılır.

2.1.2.1 Kapalı Uçlu Araç Rotalama Problemleri

Kapalı Uçlu Araç Rotalama Problemlerinde (KUARP) her rota bir işletme biriminde başlatılıp, aynı işletme biriminde bitirilmelidir. Tek işletme birimli problemlerde bunun sağlanması için kurulan modele aşağıdaki kısıt eklenmelidir.

$$k \in \{1,..., M\} \text{ için } \sum_{j=1}^{N} X_{0jk} = \sum_{i=1}^{N} X_{i0k} \leq 1 \qquad (2.7)$$

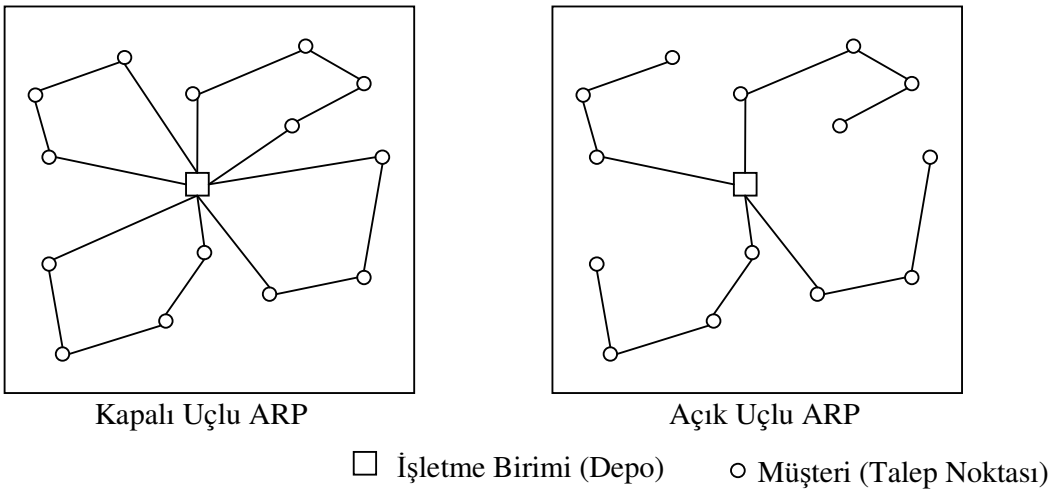
0 nolu nokta işletme birimini (depo) temsil ettiği için bu noktadan çıkan aracın mutlaka bu noktaya dönmesi, 2.7 nolu denklemde aynı araç için bu nokta ile başlayan ve bu nokta ile biten X karar değişkenlerinin değerinin birbirine eşitlenerek sağlanmaktadır. Literatürde çalışmalar çoğunlukla KUARP ile yapılmakta olup, test problemleri KUARP için mevcut olduğundan geliştirilen yöntemler KUARP üzerinden birbirleriyle kıyaslanmaktadır.

2.1.2.2 Açık Uçlu Araç Rotalama Problemleri

Açık Uçlu Araç Rotalama Problemlerinde (AUARP) rotalar merkez depo ile başlamakta, talep noktası ile sona ermektedir. Bunun sağlanması için ek olarak bir kısıtın modele eklenmesine gerek yoktur. Sonuç zaten açık uçlu rotalar doğuracaktır. Fakat rotaların kesin olarak bir müşteride sona ermesini kesinleştirecek kısıt denklemi aşağıda yer almaktadır:

$$k \in \{1,..., M\} \text{için } \sum_{j=1}^{N} X_{0jk} + \sum_{i=1}^{N} X_{i0k} = 1 \qquad (2.8)$$

Bu denklemle bir aracın 0 nolu nokta (işletme birimi) ile başlayan veya biten ilgili X değişkenlerinden ancak biri 1 değerini alabilir böylece aracın sadece işletme biriminden çıkması, oraya tekrar dönmemesi garantilenmiş olmaktadır. Aşağıda şekil 2.1'de KUARP ve AUARP'ye ait örnekler gösterilmektedir. İki problem arasında en önemli fark KUARP'de her bir rotanın Gezgin Satıcı Problemi (GSP) gibi, AUARP'de ise her rotanın Hamilton Yol Problemi (HYP) olarak ele alınmasıdır. Gerçek hayatta AUARP, bir firmanın lojistik taşımacılığını başka bir firmaya verdiği durumlarda görülebilmektedir. Literatürde AUARP için oldukça az sayıda yöntem geliştirilmiştir (Brandao, 2004).



Şekil 2.1 Kapalı ve Açık Uçlu Araç Rotalama Problemleri

2.1.3 Kısıtlarına Göre Araç Rotalama Problemleri

ARP'de gerçek hayatta kurulan lojistik sistemlerin tamamen modellenip çözümlenmesi zor olduğundan dolayı belirli bazı önemli kısıtlar seçilip diğer unsurlar göz ardı edilerek optimum sonuç bulunmaya çalışılmaktadır. Bu kısıtlardan önemlileri araç yükleme, mesafe kısıtı, zaman pencereleri ve yükleme boşaltma durumudur. Bundan dolayı Araç Rotalama Problemleri göz önüne alınan kısıtların isimleri ile beraber anılmakta, kısıtlarla ARP türlere ayrılmaktadır.

2.1.3.1 Kapasite Kısıtlı Araç Rotalama Problemi

Kapasite Kısıtlı Araç Rotalama Problemi (KKARP) bir veya daha fazla sayıda işletme birimi (depo) bulunan bir işletmenin talepleri belli n adet müşterisine ulaşabilmesi için yükleme kapasiteleri kısıtlı araçların rota planlaması problemidir. Literatürde klasik ARP ile KKARP eş olarak tutulup, genellikle tüm ARP uygulamalarında kapasite kısıtı bulunmaktadır. KKARP'de bir rotada yer alan müşterilerin toplam talebi araç kapasitesi C'yi geçmemelidir. ARP modelinde bu kısıt 2.6 nolu denklem ile gösterilmektedir.

KKARP'nin birçok farklı versiyonu bulunmaktadır. Örneğin bazı problemlerde her aracın bir sabit çalıştırma maliyeti bulunup bu değer amaç fonksiyonuna eklenmektedir. Bu durumda sabit maliyet unsurundan dolayı amaç fonksiyonunda kullanılan araç sayısı minimize edilmeye çalışılmakta ve çözümde sonuç olarak bazı araçların rotalaması yapılmamaktadır. Başka bir KKARP çeşidi ise problemde farklı tipte ve dolayısıyla yükleme kapasiteleri farklı olan araçların (C_k , k=1,...,M) söz konusu olmasıdır.

2.1.3.2 Mesafe Kısıtlı Araç Rotalama Problemi

Mesafe Kısıtlı Araç Rotalama Probleminde (MKARP) rotalara atanan her aracın gidebileceği maksimum mesafe kısıtı bulunmaktadır. Daha önce verilen modele bu kısıt aşağıdaki 2.9 nolu denklem eklenerek elde edilmektedir. Denklemde bir aracın gidebileceği maksimum L mesafe değerini ifade etmektedir.

$$k \in \{1,..., M\} \text{ için } \sum_{i=0}^{N}\sum_{j=0}^{N} d_{ij} \sum_{j=0, j\neq i}^{N} X_{ijk} \leq L \qquad (2.9)$$

Literatürde KKARP'de olduğu gibi MKARP'nin de farklı versiyonları bulunabilmektedir. Örneğin farklı tipteki araçlar için farklı mesafe kısıtı (L_k, k=1,...,M) söz konusu olabilir. Bunun yanında mesafe kısıtı yerine mesafeyle orantılı seyir süresi sınırlaması da konulabilir, bu durumda araç her müşteriye uğradığında s_i servis süresi kadar bekleyecektir. Bu tip problemlerde 2.9 nolu kısıt aşağıdaki hali alacaktır (Toth ve Vigo, 2002):

$$k \in \{1,..., M\} \text{ için } \sum_{i=0}^{N}\sum_{j=0}^{N} d_{ij} \sum_{j=0, j\neq i}^{N} p\, X_{ijk} + \sum_{i=0}^{N}\sum_{j=0}^{N} \frac{s_i + s_j}{2} \sum_{j=0, j\neq i}^{N} X_{ijk} \leq T \qquad (2.10)$$

Burada p, aracın seyir süresini kat ettiği toplam mesafe yoluyla hesaplamakta kullanılan bir parametredir. Servis sürelerinin (s_i) ikiye bölünerek hesaplanmasının sebebi ise aracın rotasında bulunan yolları sıralarken, rota içersindeki bir müşterinin bir yolun başlangıcında diğer yolun ise sonunda yer alması ve bu sebeple denklemdeki toplam ifadesinde aynı müşteri ile iki defa karşılaşılmasıdır. Bunların yanında yükleme kapasitesinin ve mesafe kısıtının birlikte kullanıldığı problemler bulunmaktadır. Bu tip problemlerde 2.8 ve 2.9 nolu kısıtlar beraber kullanılmaktadır.

Rota süresi ile birlikte taşıt sayısının da sınırlı olduğu bir problemin çözümünde bazı noktalara uğranılmaması durumu ile karşılaşılabilir. Eğer bütün adreslere mutlaka uğranılması gerekiyorsa ve taşıt sayısını artırmak olanaksızsa, maksimum rota süresi aşılmak zorunda kalınabilir. Bu durumda aşılan her birim süre için ceza maliyeti verilerek, toplam ceza maliyeti minimize edilebilir (Erel, 1995).

2.1.3.3 Zaman Pencereli Araç Rotalama Problemi

Zaman Pencereli Araç Rotalama Probleminde (ZPARP) her müşteriye belirli bir zaman aralığında ($[a_i, b_i]$) ulaşılması gerekmektedir. Burada yer alan $[a_i, b_i]$ ifadesine zaman penceresi denilmektedir. Bu tip problemlerde her araç işletme biriminden 0 zamanında ayrılacak ve bir i müşterisine uğradığında s_i servis süresi kadar müşteriye hizmet edecektir.

Aracın talep noktasına gelme zamanı, belirli bir zaman aralığı içinde olmalıdır. Eğer araç müşteriye daha erken ulaşmış ise a_i başlangıç zamanına kadar beklemesi gerekmektedir. ZPARP'de amaç, zaman pencerelerine uyularak minimum maliyetle rotalamanın yapılmasıdır. Gerçek hayatta hızlı kargo taşımacılığında bu tip problemler görülebilmektedir. ZPARP'ye genellikle malların alınacağı ya da dağıtılacağı adreslere belirli bir zaman aralığında hizmetin verilmesi gereken durumlarda karşılaşılmaktadır ve problem önce müşterinin istediği zaman aralığında hizmet verilmesi sonra rotanın optimize edilmesi stratejisi ile çözülmektedir (Erel, 1995).

2.1.3.4 Müşteri Tipi Farklı Araç Rotalama Problemi

Müşteri Tipi Farklı Araç Rotalama Probleminde (MTFARP) iki grup müşteri bulunmakta, bazı müşterilere (L grubu) ürün teslim edilirken, diğer müşterilerden (B grubu) ürün alımı yapılmaktadır. Bir araç her iki müşteriye de servis yapabilmektedir fakat ilk olarak L grubu müşterileri daha sonra da B grubu müşterileri ziyaret etmesi gerekmektedir (İlk olarak mal teslimatı, daha sonra mal alımı yapmalıdır). Modelde ürün teslimatı yapacak müşterilerin talebi pozitif iken, ürün alımı yapılacak müşterilerin talepleri de negatif belirtilir. Literatürde kapasite veya mesafe kısıtlı MTFARP de yer alabilmektedir (Toth ve Vigo, 2002).

2.1.3.5 Karma Yüklemeli Araç Rotalama Problemi

Karma Yüklemeli Araç Rotalama Probleminde (KYARP) bir müşteriye hem ürün teslimatı hem de müşteriden ürün alımı yapılmaktadır. Bununla ilgili olarak her müşterinin arz o_i ve talep q_i değeri bulunmaktadır. Her talep noktasında müşteriye ilk olarak q_i miktarında ürün teslimatı, daha sonra o_i miktarında ürün alımı yapılmaktadır. Bu sebeple araç müşteriye ulaşmadan önce araçta yeterli miktarda ürün olduğu kontrol edilmektedir.

2.1.4 Yolların Durumuna Göre Araç Rotalama Problemleri

ARP iki nokta arasındaki yol mesafelerinin geliş ve gidiş yönüne göre aynı kalıp kalmamasına göre ikiye ayrılmaktadır:

2.1.4.1 Simetrik Yollu Araç Rotalama Problemleri

Genellikle ARP'de bir noktadan diğerine olan gidiş dönüş mesafesi birbirine eşittir ($d_{ij} = d_{ji}$). Literatürde böyle problemler Simetrik Araç Rotalama Problemleri (SYARP) olarak belirtilmektedir.

2.1.4.2 Asimetrik Yollu Araç Rotalama Problemleri

Bazı durumlarda ARP'de yer alan y ve z noktaları için y noktasından z noktasına gitmek için gerekli olan mesafe, z'den y noktasına olan mesafeye eşit olamayabilir ($d_{yz} \neq d_{zy}$). Bu tip KUARP'de araçların ilk olarak hangi müşteriye gideceği önem kazanmakta, bu da rotanın dönüş yönünü saptayarak rota mesafesinin hesaplanmasını belirlemektedir. Bu tip problemlere Asimetrik Yollu Araç Rotalama Problemleri (AYARP) denmektedir. Simetrik ve Asimetrik Yollu ARP için modele ayrıca yeni bir kısıtın tanımlanmasına gerek yoktur. 2.3 ve 2.4 nolu kısıtlar rotalar için yolların sürekliliğini sağlamaktadır.

2.2 Araç Rotalama Problemlerinin Karşılaşıldığı Süreçler

ARP'ye günümüzün organizasyonlarında sıkça rastlanmaktadır. Bu tip problemler aşağıdaki süreçlerde görülebilmektedir:

- Firmalarda lojistik ve dağıtım planlarının yapılması
- Depolardan bayilere mal sevkıyatı sırasında
- Satış personelinin saha dağıtımı
- Bölge bayi kontrol uygulaması
- Belediyeler için çöp toplama ve sokak temizlik planı
- Benzin istasyonlarına akaryakıt dağıtımı
- Saha satış uygulamaları
- Kargo dağıtımı
- Okul servis araçlarının rotalanması

Bir sıralama ve gruplama problemi olarak ARP için geliştirilen yöntemler aşağıdaki alanlara da bire bir uygulanabilirler (Laporte ve Palekar, 2002), (Bektaş, 2004):

- Üretimde tek makine için ürünler arası hazırlık sürelerinin optimizasyonu
- Bilgisayar hardisklerinde yer alan verilerin düzenlenmesi sırasında manyetik başlığın minimum hareketinin sağlanması
- Okullarda sınav zamanlarının çakışmaların olmadan çizelgelenmesi

- Tekstilde nakış işleme sırasında aynı renk desenlerin belirlenerek sıralı bir şekilde dikimin sağlanması
- Elektronikte bütünleştirilmiş devre testleri için sistemin gruplara bölünerek test noktalarının belirlenmesi
- Hafızada oluşabilecek hata sayısını azaltmak ve bilgisayar çalışma hızını artırmak için sık sık başlatılan bilgisayar programlarının iç yapısının düzenlenmesi
- Matbaalarda baskı makinalarının aynı anda basacağı sayfa sayısının belirlenmesi
- Bankacılıkta şubeler arası para transferinin planlanması
- Bir işi en kısa zamanda yapılabilmesi için robotların görev planlarının çıkartılması

2.3 Kombinatoryal Optimizasyon Problemleri

Karaboğa (2004) optimizasyon problemini şu şekilde tanımlamıştır: Optimizasyon problemi, belirli sınırlamaları sağlayacak şekilde, bilinmeyen parametre değerlerinin bulunmasını içeren bir problem çeşididir. Bir optimizasyon problemi (P), mümkün olabilen çözümler seti ve her çözüm s'e nümerik bir değer (f(s)) atayan bir f amaç fonksiyonu ile tanımlanabilir. Optimizasyon işleminde ilk adım olarak karar parametreleri veya karar değişkenleri olarak da adlandırılan parametreler setinin tanımlanması gerekir. Sonra bu parametrelere bağlı olarak minimize veya maksimize edilecek bir maliyet/kâr fonksiyonu ve problemle ilgili kısıtlar tanımlanmalıdır. Problem için tüm kısıtları sağlayan çözüm bölgesi uygun çözüm bölgesi olarak adlandırılır. Optimizasyon problemleri çeşitli özelliklerine göre sınıflandırılabilir:

- Amaç fonksiyonu üzerinde bir kısıtın olup olmaması durumuna göre sınırlamalı ve sınırlamasız optimizasyon problemi şeklinde bir sınıflandırma yapılabilir.
- Amaç fonksiyonu ve kısıtların doğrusallık durumuna göre lineer programlama ve lineer olmayan programlama problemi olarak iki gruba ayrılabilir. Aynı şekilde kuadratik amaç fonksiyonuna ve lineer kısıtlara sahip bir problem kuadratik optimizasyon problemi olarak adlandırılır.
- Optimizasyon problemlerinde, karar değişkenlerin alabileceği değerlere göre sürekli ve kombinatoryal optimizasyon problemi (KOP) şeklinde iki sınıf oluşturulabilir. Kombinatoryal problemlerde genel olarak ayrık niceliklerin optimal olarak düzenlenmesi, gruplanması, sıralanması ve seçilmesi amaçlanmaktadır.

Bir kombinatoryal optimizasyon probleminin (KOP) modellemesi genellikle şu şekilde yapılmaktadır (Blum ve Roli, 2003):

1. Değişken kümesi: $X = \{x_1, ..., x_n\}$;
2. Değişkenlerin alabileceği değerler kümesi: $D_1, ..., D_n$;
3. Değişkenlerle ilgili kısıtlamalar;
4. Minimize edilecek amaç fonksiyonu: $f = D_1 \times ... \times D_n$;

Tüm uygun çözümlerin kümesi ise aşağıdaki formül ile gösterilmektedir:

$$S = \{s = \{(x_1, v_1), ..., (x_n, v_n)\} \mid v_i \in D_i, \text{ s tüm kısıtları sağlar}\} \quad (2.11)$$

Literatürde S kümesine, çözüm veya arama uzayı denilmekte ve küme içersinde yer alan her çözüm, optimizasyon algoritmaları için arama uzayında göz önüne alınması gereken bir nokta olarak ifade edilmektedir. Bir KOP'yi tam olarak çözmek için en küçük amaç fonksiyonu değerine sahip $s^* \in S$ global optimum çözümünün bulunması gereklidir.

Kombinatoryal optimizasyon problemleri çözümlerin yapısına göre genel olarak üçe ayrılır:

- **Gruplandırma (Sınıflandırma) Problemleri:** Bu tip problemlerde genel amaç bir küme içinde yer alan elemanların gruplandırılmasıdır. Örneğin tesis yeri belirleme probleminde talepleri deterministik müşteriler için en uygun yer saptanmaya çalışılır. Bunun için müşteriler gruplara ayırılarak, her gruba bir tesisin hizmet vermesi sağlanmaktadır.
- **Sıralama Problemleri:** Çözümleri, bir grup elemanın belli bir sıraya göre permutasyon mantığında sıralanması ile oluşan problemlerdir. Sıralama problemlerine örnek olarak iş sıralama ve (paralel olmayan) çizelgeleme problemi, GSP, HYP verilebilir.
- **Karma Problemler:** Hem sıralama, hem de gruplandırmanın birlikte yapıldığı problemlerdir. ARP ve esnek iş çizelgeleme problemi karma problemler kapsamına girmektedir.

ARP, bir NP-Zor kombinatoryal optimizasyon problemidir. Matematiksel açıdan ARP'nin diğer kombinatoryal problemler gibi çözümü oldukça zordur. Problemin çözüm uzayı müşteri ve araç sayısı ile üstel orantılı olarak artar. Bu da gerçek hayattaki bir problemin gelişmiş bilgisayarlarla dahi analitik metotlar kullanılarak çözülmesinin oldukça uzun sürmesine yol açmaktadır. Ayrıca çözüm uzayında yerel minimum noktaların fazla olması ve arama sırasında bu noktalardan kurtulmanın zor olması kombinatoryal optimizasyon problemleri için

bir dezavantaj teşkil eder. Bu sebeple Araç Rotalama Problemlerinin çözümünde, optimuma yakın sonuçlar veren fakat analitik yöntemlere oranla çok daha hızlı olan metasezgisel yöntemler önemli bir konuma gelmişlerdir (Cordeau vd., 2002).

Bir problem için bilgisayarda hazırlanan işlevsel bir algoritmaya verilen girdi iletisinin uzunluğu "L" ve programın çalışma süresi ya da çevrim sayısı "T" ise NP-Zor problemler aşağıdaki kurala uymaktadırlar (Ruelle, 2004).

$$T \geq C\,(L + 1)^n \tag{2.12}$$

Burada C ve n bir tamsayı olup eşitsizliğin polinom haline getirilmesini sağlamaktadırlar.

Kombinatoryal optimizasyon problemleri için geliştirilen yöntemler Kesin ve Yaklaşık algoritmalar olmak üzere ikiye ayrılır. Kesin algoritmalar, her boyuttaki KOP için uygulanabilir olup optimum çözümü kesin olarak bulmaktadırlar. Fakat NP-zor olan KOP için bu algoritmalar hiçbir zaman polinom zamanda çözüm üretemezler. Bu sebepten dolayı Kesin algoritmalarda çözüm uzayındaki her noktanın taranması gerekmektedir. Bunun için de problem boyutuyla üstel olarak orantılı hesaplama süresine ihtiyaç duyulmakta, bu da oldukça fazla beklemelere neden olmaktadır. Yaklaşık algoritmalarda ise optimum sonuç bulunamamasına rağmen hesaplama süresi daha az olmaktadır (Blum ve Roli, 2003).

2.3.1 Kombinatoryal Problemler İçin Optimizasyon Metotları

Problemlerin çözümü için kullanılan algoritmaların, sonuca kısa sürede ulaşması esastır. Bir algoritmanın en yaygın performans ölçüsü, algoritmanın sonucu bulana dek harcadığı süredir. Fakat bu süre, işlem hızı ve bilgi raporlama tekniğine bağlı olarak değişkenlik gösterebilir. Problemin çözümüne yönelik bir algoritma araştırmadan önce, bu problemin sonlu sayıda aşamada çözülüp çözülemeyeceğini bilmek gerekir. Algoritmalar teorisine göre evrensel algoritmik modellerin üç türü ele alınmaktadır (Nabiyev, 2003):

1. **Tür:** Algoritma kavramını klasik olan hesaplama ve sayısal fonksiyonlar gibi matematiksel kavramlarla ilişkilendirmektedir. Bu sınıfın en gelişmiş ve incelenmiş modeli özyinelemeli fonksiyonlardır.

2. **Tür:** Algoritmanın her ayrık zamanda çok basit işlemleri yapan bir deterministik makina ile bağdaştırılmasıdır. Bu modeller yapısal olarak bilgisayara en yakın olanlardır. Bu türün temel teorik modeli 1930'larda oluşturulan Turing makinalarıdır.

3. **Tür:** Bu modeller, herhangi alfabede sözcüklerin değiştirilmesine dayalı kelime işlemcilerdir. Bu sınıfın özelliği onların maksimum derecede sanal olması ve algoritmalar kavramının istenilen nesneye (yalnız sayısal değil) uygulanabilmesidir.

Kullanılan algoritma tipine göre, tüm problemler eşdeğer değildir. Sezgisel çözümler göz önüne alınmadığında, bazılarının çözümü diğerlerinden daha hızlıdır. Bir problemi çözmeye olanak sağlayan algoritmanın, aynı sınıftan olan diğer problemlerin çözümünde de başarılı olması kanıtlanmıştır. Problemin algoritmik çözümlerinin sınıflandırılması, bunların yürütülmesi için gerekli işlemlerin sayısı temel alınarak gerçekleştirilebilir, bu ölçüye algoritmik karmaşıklık adı verilir. Karmaşıklık algoritmanın, uygulanacağı bilgisayarın istemcisinden bağımsız değerlendirilmesi düşünülmektedir. Bir algoritmanın hesaplama karmaşıklığının değerlendirilmesi onun ne kadar hızlı çalışacağı ve bilgisayarın belleğinde ne kadar yer kullanacağına ilişkin bilgiler vermektedir. Algoritmanın giriş değerlerinin boyutu veya problem boyutu, girişi tanımlamak için gerekli sembollerin sayısı ile bağlantılıdır. Anlaşıldığı gibi bir algoritmanın hesaplama karmaşıklığı iki açıdan incelenmektedir: hesaplamayı yapmak için gerekli olan süre ve gerekli bellek sığasının ölçümü. Günümüzde bellek karmaşıklığına zaman karmaşıklığından daha az önem verilmektedir. Bu sebeple bazen zaman karmaşıklığı sadece karmaşıklık olarak da isimlendirilebilmektedir.

Algoritmanın, zaman karmaşıklığını tanımlamadan önce bir fonksiyonun derece gösterimini verelim. Doğal n sayısını, herhangi bir pozitif değere atayan f ve g fonksiyonları olsun. Aşağıdaki önermenin doğruluğunu sağlayan n_0 ve c sayılarının olduğunu varsayalım.

$$f(n) = O(g(n)) \ \forall n \geq n_0: f_n \leq c * g(n) \qquad (2.13)$$

Büyük "*O*" notasyonu bir fonksiyonun üst sınırını belirtir. Algoritma karmaşıklığını incelediğimizde g(n) fonksiyonunun derecelere ve çarpanlara bağımlı olarak dönüşümü yapılmaktadır. Örneğin, $g(n) = (1/4)n^4 + 200\ n + 5$ fonksiyonunda n'nin büyük değerleri ele alındığında, n^4'e nazaran diğer toplananlar önemsenmez ve algoritma karmaşıklığı $c * O(n^4)$'den $c = 0.25$ biçiminde yazılır. Algoritma karmaşıklığı fonksiyonlarına örnek olarak, ikili arama için $O(\log_2(n))$, doğrusal arama için $O(n)$, matris çarpma işlemleri için $O(n^3)$, bir metinde n uzunluklu kelimenin bulunması için $n*O(n)$ fonksiyonları gösterilebilir. Bu problemlerin hepsinde g(n) fonksiyonu polinomiyal karakter taşımaktadır (Nabiyev, 2003).

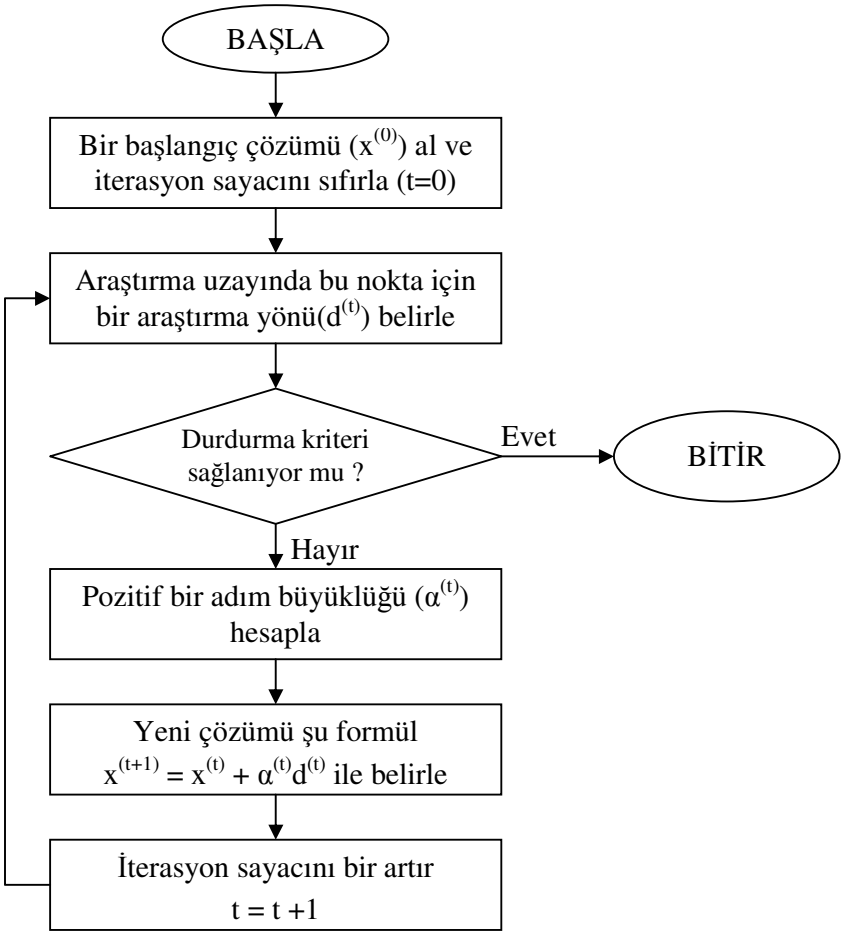
Polinomiyal zamanlı algoritmalar pratikteki problemlerin çözümünde iyi performans göstermektedirler. Büyük algoritmalar genellikle, üstel algoritma olarak adlandırılırlar. Üstel algoritmalarda, 2^n, $n!$, n^2, $n^{\log n}$ gibi polinomiyal olmayan artış hızları geçerlidir. Polinomiyal

algoritmalar, üstel algoritmalara nazaran teknolojik gelişmelerden daha avantajlı olarak faydalanırlar ve kapalılık özelliklerinden dolayı özel problemlerin çözümü için birleştirilebilirler (Eren, 2002).

Genel olarak optimizasyon metotları iki gruba ayrılır (Karaboğa, 2004) :

- Doğrudan Metotlar (Araştırma Metotları)
- Optimalite Kriterlerine Dayalı Dolaylı Metotlar

Dolaylı metotlarda ilk olarak optimallik kriterleri belirlenir, daha sonra bölgesel optimuma aday noktalar için problem çözülmektedir. Araştırma metotlarında ise tahmini bir başlangıç çözümü ile araştırmaya başlanır ve algoritma tarafından başlangıç çözümü iteratif olarak geliştirilir. Diğer bir deyişle optimum çözümleri bulmak amacıyla çözüm uzayı araştırılır. Belirtilen iteratif araştırma işleminin akış şeması aşağıda şekil 2.2'de gösterilmektedir:



Şekil 2.2 İteratif Araştırma işlemi akış şeması

Şekil 2.2'deki algoritmik yapıdan anlaşıldığı gibi araştırma işleminin iki alt işlemden oluştuğu görülmektedir:

- Araştırma yönünün ($d^{(t)}$) tayini
- Adım uzunluğunun ($\alpha^{(t)}$) belirlenmesi

Adım büyüklüğünün belirlenmesi için kullanılan bazı metotlar ise şunlardır:

- Eşit Aralıklı Araştırma
- Altın Bölüm Araştırma
- Polinom İnterpolasyonu

Geçmiş yıllarda araştırmacılar özellikle kombinatoryal optimizasyon problemlerin çözümü için büyük çaba sarf etmişlerdir. Optimal çözümü elde etmek için gerekli olan hesaplama işlemleri açısından değerlendirildiğinde, KOP oldukça zor problemlerdir. Bundan dolayı, büyük boyutlu problemlerin çözümünde uygun hesaplama süresi içerisinde optimuma yakın çözümleri bulabilen yaklaşık algoritmaları geliştirmek ve kullanmak önemli hale gelmiştir. Bu yüzden zeki ve etkili sezgisel yaklaşımların zor problemlerin çözümünde kullanılması üzerinde oldukça fazla araştırma yapılmıştır. Bu yaklaşımlar ağırlıklı olarak biyoloji, zooloji, fizik, bilgisayar ve karar üretme bilimlerinden türetilmiştir (Karaboğa, 2004).

Kombinatoryal optimizasyon problemlerinin büyük çoğunluğu NP-tam polinomiyal zaman sınırı olmayan problemler sınıfına girmektedir. Şu ana kadar NP problemlerine polinomiyal algoritma geliştirilememiştir. NP kapsamında yer alan problemler için asıl optimum çözüm yerine, yakın çözümler tercih edilir. Bu tip problemlerin kesin çözümlerine makul sürelerde ulaşılamadığından, yerel arama ve stokastik arama yöntemleri ile yaklaşık çözümler elde edilir. Stokastik arama yöntemleri, yerel arama yöntemlerinin yerel optimumda takılıp kalma dezavantajlarını ortadan kaldırmak için geliştirilmiş yöntemlerdir.

Kombinatoryal problemlerde, sonlu veya sayılabilir sonsuz bir kümeden, bir alt kümeye, bir nesneye, bir permutasyona ulaşmaya çalışılır. Kombinatoryal optimizasyon problemleri, lineer programlama modellerine dönüştürülerek çözülebilir. Fakat bu işlem oldukça fazla hesaplama süresi gerektirmektedir. Kombinatoryal optimizasyon problemlerinin büyük bir kısmı sezgisel yöntemler yardımı ile çözülebilmektedir. Diğer bir deyişle bu tip problemler için optimum değil, optimuma yakın çözümler elde edilmektedir (Eren, 2002).

3. ARAÇ ROTALAMA PROBLEMİ İÇİN SEZGİSEL YÖNTEMLER

Bu bölümde ilk olarak algoritma kavramı ve sezgisel algoritmalar açıklanmakta, daha sonra literatürde sık sık karşılaşılan ve genellikle 1960'dan 1995'e kadar Kapasite Kısıtlı ARP için geliştirilen sezgisel yöntemler işlenmektedir. Bu yöntemler arama uzayında oldukça kısıtlı bir alanda tarama yapmalarına rağmen kısa sürede iyiye yakın çözümler üretmektedirler. Bunun yanında sezgisel teknikler genelde hızlı bir şekilde uygun bir ilk çözüm bularak bu çözümü iyileştirmeye çalışırlar. ARP için geliştirilen sezgisel yöntemlerin birçoğu simetrik yollu KKARP için geliştirilmiştir.

3.1 Algoritma Kavramı ve Sezgisellik

Algoritma, belirli tek bir problemi çözecek davranışın varolan veya sonradan tanımlanan veri modeline dayandırılarak adım adım ortaya koymak ve bunu bilgisayar ortamında herhangi bir programlama diliyle kodlamaktır. Burada yeralan veri modeli ifadesi ile, verilerin birbirleriyle ilişkisel ve sırasal durumundan bahsedilmiştir. Bir başka açıdan algoritma, belirli bir işin veri yapılarıyla algoritmik ifadesini ortaya koymaktır denebilir (Çölkesen, 2002).

Algoritma, belirli bir işin veya problemin sonucunu elde etmek için art arda uygulanacak adımları ve koşulları kesin olarak ortaya koyar. Buradan da anlaşılacağı gibi, algoritma genel olarak tek bir işin kotarılması üzerine yoğunlaşmıştır. Örneğin, bir kümenin elemanlarını sıralama, bir graf üzerinde en kısa yolun bulunması (GSP), bir matrisin determinantının alınması gibi algoritmalar tek bir amaca yöneliktir. Algoritma oluştururken, adımları belirlemek için aşağıdaki unsurları göz önünde tutmak gerekir:

- Her adım son derece belirleyici olmalıdır. Hiç bir şey şansa bağlı değildir.
- Belirli bir sayıda adım sonunda algoritma sonlanmalıdır.
- Algoritmalar karşılaşılabilecek tüm ihtimalleri ele alabilecek kadar genel olmalıdır.

Algoritma, mekanik davranan kişiye veya bir makineye, birtakım verilerden yola çıkarak ve sonlu sayıda aşamalardan geçerek, belli bir problemi çözme imkanı veren, çok kesin komutlar bütününden oluşmaktadır. Bir algoritmanın çalışmasındaki mutlak zorunluluk, her türlü belirsizlikten arınmış olmasıdır. Bir algoritmanın yürütülmesi, her biri bir komutla belirlenen bir etkiler dizisi oluşturur ve bu dizi önceki komutun yürütülmesinin sona ermesiyle birlikte yürütülmeye başlar. Genel olarak algoritmaların aşağıdaki özelikleri ele alınmaktadır (Nabiyev, 2003):

- Genellik
- Kesin sıralılık
- Sırayı belirleyen kumanda yapısı
- Sonluluk ve neticelik

Kombinatoryal optimizasyon problemlerinin çözümünde genellikle sezgisel yöntemler kullanılmaktadır. Literatürde bu açıdan bir boşluk bulunmakta ve sezgisel algoritmaların her zaman çalışmadığı savunulmaktadır. Gerçekten de bir problem için geçersiz olan sezgisel yaklaşım, diğerinde başarılı sonuçlar verebilir. Fagenbaum ve Fieldman tarafından sezgiselliğin tanımı şu şekildedir: Sezgisellik (sezgisel kurallar, sezgisel yöntem) problemin durum uzayı çok büyük olduğunda çözümün aranmasını kesin biçimde sınırlayan herhangi kural, strateji, hile, sadeleştirme ve diğer etmenlerin kullanımıdır. Dolayısıyla sezgisellik, problem karmaşıklık içerdiğinde, çözüm için yolun bulunmasındaki yardımcı anahtardır. İyi seçilmiş anahtarla tek bir kapıyı açıp amaca ulaşmak mümkün olduğu gibi, kötü seçilmiş anahtarlarla bu yolu zora sokmak da mümkündür. Sezgisel yaklaşımın temel adımları aşağıdaki gibi sıralanabilir (Nabiyev, 2003):

1) Mümkün olabilecek durumlar içerisinde herhangi birisinin ele alınması
2) Ele alınmış duruma mümkün gidişler uygulayarak durumun değiştirilmesi
3) Durumun değerlendirilmesi
4) Gereksiz durumların atılması
5) Eğer sonuca ulaşılmışsa çözümün tamamlanması, aksi halde yeni değer ele alınarak işlemlerin tekrarlanması

3.2 Sezgisel Algoritmalar

Sezgisel algoritmalar, herhangi bir amacı gerçekleştirmek veya hedefe varmak için çeşitli alternatif hareketlerden etkili olanlara karar vermek amacıyla tanımlanan kriterler veya bilgisayar metotlarıdır. Bu tür algoritmalar yakınsama özelliğine sahiptir, ama kesin çözümü garanti edemezler ve sadece kesin çözüm yakınındaki bir çözümü garanti edebilirler. Bu algoritmalar çözüm uzayında optimum çözüme yakınsaması ispat edilemeyen algoritmalar olarak da adlandırılırlar. Sezgisel algoritmalara gerek duyulmasının sebepleri aşağıdaki gibidir (Karaboğa, 2004):

- Optimizasyon problemi kesin çözümü bulma işleminin tanımlanamadığı bir yapıya sahip olabilir.
- Anlaşılırlık açısından sezgisel algoritmalar karar verici açısından çok daha basit olabilir.
- Sezgisel algoritmalar, öğrenme amaçlı ve kesin çözümü bulma işleminin bir parçası olarak kullanılabilir.
- Matematik formülleriyle yapılan tanımlamalarda genellikle gerçek dünya problemlerinin en zor tarafları (hangi amaçlar ve hangi sınırlamalar kullanılmalı, hangi alternatifler test edilmeli, problem verisi nasıl toplanmalı) ihmal edilir. Model parametrelerini belirleme aşamasında kullanılan verinin hatalı olması, sezgisel yaklaşımın üretebileceği alt optimal çözümden daha büyük hatalara sebep olabilir.

Bir problem için geliştirilmiş bir sezgisel algoritma, aşağıdaki faktörler göz önüne alınarak değerlendirilebilmektedir:

- **Çözüm Kalitesi ve Hesaplama Zamanı:** Çözüm kalitesi ve hesaplama zamanı bir algoritmanın etkinliğinin değerlendirilmesi için önemli kriterlerdir. Bundan dolayı bir algoritma, ayarlanabilir parametreler setine sahip olmalı ve bu parametreler kullanıcıya önemlilik açısında hesaplama maliyeti ile çözüm kalitesi arasında bir vurgulamanın yapılabilmesine imkan vermelidir. Diğer bir deyişle, çözüm kalitesi ile hesap zamanı arasındaki ilişki kontrol edilebilmelidir.
- **Kod Basitliği ve Gerçeklenebilirlik:** Algoritma prensipleri basit olmalı ve genel olarak uygulanabilir olmalıdır. Bu durum problem yapısı ile ilgili başlangıçta çok az bilgiye sahip olunması halinde bile algoritmanın yeni alanlara kolaylıkla uygulanabilmesini sağlar.
- **Esneklik:** Algoritmalar modelde, sınırlamalarda ve amaç fonksiyonlarında yapılacak değişiklikleri kolayca karşılayabilmelidir.
- **Dinçlik:** Yöntem, başlangıç çözümünün seçimine bağlı olmaksızın her zaman yüksek kaliteli, kabul edilebilir çözümleri üretebilme kabiliyetine sahip olmalıdır.
- **Basitlik ve Analiz Edilebilirlik:** Karmaşık algoritmalar, esneklik ve çözüm kalitesi açısından basit algoritmalardan daha zor analiz edilebilmektedir. Algoritma kolayca analiz edilebilir olmalıdır.

- **Etkileşimli Hesaplama ve Teknoloji Değişimleri:** Algoritma içinde insan-makina etkileşimini kullanma fikri çoğu sistemde yaygın olarak gerçekleştirilmektedir. Herkesçe bilindiği gibi iyi bir kullanıcı arayüzü herhangi bir bilgisayar sistemini veya algoritmayı daha çekici yapmaktadır. Bunun en önemli avantajı çözümlerin grafiksel olarak sergilenebilmesidir.

Bazı sezgisel-bölgesel araştırma algoritmaları başlangıç çözümüne bağlı olarak bölgesel optimum çözümler üretirler. Bu, genellikle iteratif gelişme metotlarında karşılaşılan bir durumdur. Bölgesel optimal çözüm, global optimum çözümden çok uzak olabilir ve yine çoğu ayrık optimizasyon problemlerinde uygun bir başlangıç çözümünün seçimi için genel bilgiler mevcut olmayabilir. Sezgisel-bölgesel araştırma metotlarının dezavantajlarının bazılarını ortadan kaldırmak için basitlik ve genellik muhafaza edilmek kaydıyla aşağıda sıralanan fikirler değerlendirilebilir (Karaboğa, 2004):

- Çok sayıda başlangıç çözümleri ile algoritmanın tekrar tekrar yürütülmesi sağlanabilir. Bu durumda rasgele farklı başlangıç çözümleriyle hesaplama işlemi oldukça maliyetlidir ve hiçbir zaman optimal çözümün bulunması garanti edilemez.

- Çok daha karmaşık bir komşuluk yapısının tanımlanması suretiyle çok daha iyi komşu çözümlerin üretilmesinin sağlanması.

- Algoritmanın koşulması esnasında bilgi toplanmasını sağlayan karmaşık öğrenme stratejilerinin kullanılması ve bu bilginin her bir koşma sonunda belirli bölgeleri veya çözümleri cezalandırmak amacıyla kullanılması.

- Bölgesel araştırma metotlarında amaç, sadece fonksiyon değerini azaltan çözümlerin kabul edilmesi olmasına rağmen, bölgesel optimallikten kaçmak için gelişmeleri önleyen veya kısıtları aşan hareketlerin kabullenilmesi.

3.3 Araç Rotalama Problemi İçin İyi Bir Sezgisel Algoritmanın Özellikleri

Cordeau ve arkadaşları ARP için iyi sonuçlar veren bir sezgisel algoritmanın dört temel özelliğe sahip olması gerektiğini vurgulamışlardır. Bu özellikler ve açıklamaları aşağıda yer almaktadır (Cordeau vd., 2002):

1. Kesinlik: Literatürde kesinlik kavramı, bir algoritmanın ürettiği değerin problemin optimum çözümüne oranı olarak bilinir. ARP literatüründe bazı büyük boyutlu problemler için arama uzayı tam olarak taranmadığı ve dolayısıyla optimum çözüm kesin olarak

bilinmediği için çoğu karşılaştırmalar bilinen en iyi çözüm ile yapılmaktadır. Bundan dolayı karşılaştırma için İnternet'te [1] araştırmacıların ortak kullandığı soru kütüphaneleri bulunmaktadır. Böylece yapılan çalışmalarda önerilen yöntemin bu problemler için verdiği çözüm değerleri verilmekte ve bu da araştırmacılara iyi bir kıyaslama imkanı sunmaktadır. Yazarlar kesinlik kavramının ARP için çok önemli olduğunu belirtmekte, iki nokta arasındaki mesafe değerlerindeki yuvarlamalara dikkat çekmektedirler. Örneğin, bazı büyük boyutlu problemlerde mesafe değerinin alt veya üst tamsayıya yuvarlanması amaç fonksiyonu değerinde %3'lük bir değişime sebebiyet verebilmektedir. Bu durum aynı şekilde Zaman Kısıtlı ARP için de geçerlidir. Bunun yanında KKARP kütüphanesinde yer alan sorular değişik özelliklerde olduğu için, geliştirilen sezgisel yöntem sadece bazı sorularda iyi sonuç vermemeli, tüm sorularda optimuma yakın sonuçlar verebilmelidir. Son olarak yazarlar iyi bir sezgisel algoritma işlerken, zaman geçtikçe sürekli daha iyi sonuçlar üretmesi gerektiğini vurgulamışlar ve iyi bir sezgiselin genellikle işleyiş sürecinin erken zamanlarında iyiye yakın bir çözüm bulduğunu ve eğer optimum sonuç bulunamamışsa algoritmanın sürekli aynı çözümde takılıp kalmadığını, sürekli bulduğu çözümleri geliştirdiğini belirtmişlerdir.

2. Hız: Bilgisayar hesaplama hızının günümüzde son derece artmış olması, ARP gibi kombinatoryal problemlerin iteratif süreçlerle çözümlerinin kısa zamanda gerçekleşmesini mümkün kılmıştır. Özellikle gerçek sistemlerde yapılan uygulamalarda çözüm süresi önemli bir faktör teşkil etmekte, uygulanabilir bir çözüm bulmak için problemin iyi bir şekilde analiz edilip bellek-işlemci kaynak kullanımını doğru planlanarak yazılım diline dökülmesi gerekmektedir. Bir lojistik firmasının günlük araç rotalarının çıkartılması gibi büyük boyutlu problemlerde hesaplama süresinin 10 - 20 dakikayı bulması, toplam maliyetin azaltılması açısından mantıklı gözükse de, özellikle gerçek zamanlı interaktif uygulamalarda örneğin saha satış planlarının yapılmasında anlık talep değişiklikleri olabileceği için, son derece hızlı sonuçlar veren algoritmaların geliştirilmesi daha uygundur. Bu tip algoritmalarda kayıp, üretilen sonucun kesinliğinin düşük olması başka bir deyişle optimum sonuca uzak çözümlerin oluşturulmasıdır. Bunların yanında yazarlar, paralel programlamanın doğru bir şekilde kullanılabilirse ARP için sezgisel yöntemlerle hızlı çözümler üretilebileceğinin üzerinde durmuşlardır.

3. Basitlik: Çoğu sezgisel yöntemin gerçek zamanlı sistemlere uygulanmayışının sebeplerinden biri anlaşılmasının ve kodlanmasının zor olmasıdır. Algoritma geliştiriciler yayınladıkları çalışmalarında algoritmanın her adımını detaylı bir şekilde açıklamaları gerekmektedir. Bilhassa problem bazında geliştirilmiş ve çok fazla unsuru bir arada

barındıran bir yöntemin başka tip bir probleme uygulanmasında zorluklar yaşanabilmektedir. Bunun yanında çok fazla sayıda parametre değerinin belirlenmesi gereken algoritmaların kullanımı da ayrı bir zorluk teşkil etmekte, hangi durumlarda parametrelerin hangi değerleri alacağı muallakta kalmaktadır. Bu sorun özellikle son 10 yıldır geliştirilmekte olan metasezgisel yöntemlerde sıkça görülmektedir. Yazarlar buna çözüm olarak geliştiricilere, işleyiş süresince sabit tutulan veya parametrelerine kendi kendine değerler atayabilen mekanizmalar kurulmasını önermişlerdir. Bir sezgisel algoritmanın uygulanabilirliğinin fazla olması için gereken karakteristiklerinden biri de üretilen çözüm kalitesinin başlangıç koşullarına çok fazla bağlı olmamasıdır. Yazarlara göre Clark ve Wright algoritmasının çok popüler olmasının sebeplerinden en önemlisi anlaşılırlığının kolay olması, kısa sürede kodlanabilmesi ve bir problem için aynı sonucu üretebilmesidir. Yöntemin basit olması uygulanabilirliğini artırırken, bilgisayar kaynaklarının daha verimli bir şekilde kullanımını da sağlayacağından ARP'ye hızlı bir şekilde çözüm üretmek de mümkün olmaktadır.

4. Esneklik: Yazarlara göre iyi bir ARP sezgisel algoritması çeşitli kısıtlar ile entegre olarak gerçek sistemlere uygulanabilir esneklikte olmalıdır. Genellikle ARP literatüründe kapasite ve mesafe kısıtlı problemler yer alıyorsa da yeni bir yan kısıt eklendiğinde algoritmada ne tür bir değişiklik yapılması gerektiğinden çoğunlukla bahsedilmemektedir. Modele bu tür değişikliğin yapılması genellikle algoritma performansında büyük ölçüde düşüşe sebebiyet vermektedir. Yazarlar bir ARP sezgiseline yeni kısıtlar eklemek için en iyi yolun, amaç fonksiyonuna her yeni kısıtla ilgili olarak ağırlıklandırılmış ceza terimlerinin eklenmesi olduğunu belirtmişlerdir. Eğer Q(x) ve D(x) fonksiyonları sıra ile kapasite ve süre kısıtları ile ilgili ceza terimlerini ifade ederse yeni amaç fonksiyonu $F'(x) = F(x) + \alpha Q(x) + \beta D(x)$ halini almaktadır. Bu formülasyonda F(x), toplam maliyetin (toplam gidilen mesafe) minimum hale getirilmesi ile ilgili fonksiyon olurken, α ve β ise algoritmanın kendi içinde değerlerini ayarlayabildiği ceza ağırlıklardır. Yazarlara göre ilk olarak 1 değeri atanan bu ağırlıkların değerleri, iterasyonlarda elde edilen çözümlerin uygun olup olmamasına göre periyodik olarak artırılabilir veya azaltılabilir. Bu sayede arama uzayı daha etkili bir şekilde taranırken, algoritmanın lokal minimum noktalarına takılma olasılığı da azaltılmış olur. Bu şekilde oluşturulan bir modelin diğer bir avantajı da iterasyonlarda yeni çözümlerin, örneğin noktaların rotalar arasında yer değiştirmesi gibi basit bir şekilde üretilebilmesidir. Böylece algoritmik esneklik, tasarımın basitleştirilmesi ile elde edilebilmektedir.

Sezgisel yöntemler genel olarak Yapısal, Çok Aşamalı ve İyileştirmeli olmak üzere üç kategoriye ayrılmaktadır. Yapısal sezgisel algoritmalarda amaç fonksiyonu göz önüne

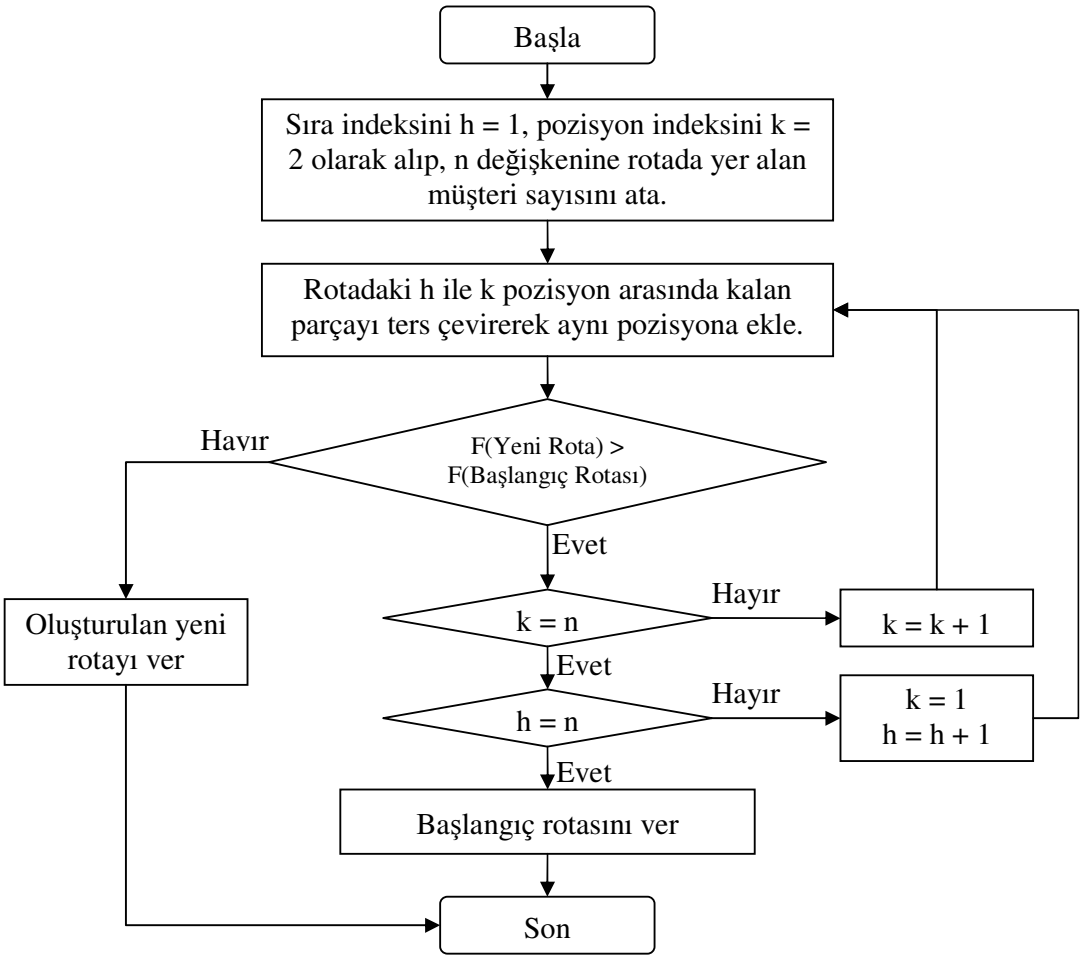
alınarak adım adım uygun (fizıbıl) bir çözüm oluşturmaya çalışılır, fakat çoğunlukla bulunan bu çözüm üzerinde iyileştirilme yapılmaz. Çok Aşamalı sezgisellerde problem birden fazla parçaya bölünerek ve her parça için ayrı süreçler izlenerek çözüm bulunmaya çalışılır. İyileştirmeli sezgisel algoritmalarda ise uygun bir çözüm ele alınır, bu çözümün komşu çözümleri taranarak daha iyi bir çözüm elde edilmeye çalışılır. Yapısal ve İyileştirmeli sezgisel metotlar birleştirilerek yeni yöntemler oluşturulabilir. Diğer kategorilerde de kullanılabildiği için ilk olarak İyileştirmeli sezgisel algoritmalar konusuna değinilecektir.

3.4 İyileştirmeli Sezgisel Algoritmalar

İyileştirmeli Sezgisel Algoritmalarda mevcut çözümde yer alan araç rotaları arasında nokta (müşteri) veya yol değiş tokuşu yapılarak uygun çözüm geliştirilmeye çalışılır. Aynı anda tek rota ya da çok rotanın iyileştirilebilmesine göre bu algoritmalar, tek rota ve çok rota olmak üzere ikiye ayrılır. GSP'ye uygulanabilen herhangi bir sezgisel yöntem Tek Rota İyileştirmeli Algoritmalar içinde sayılabilir (Cordeau vd., 2004).

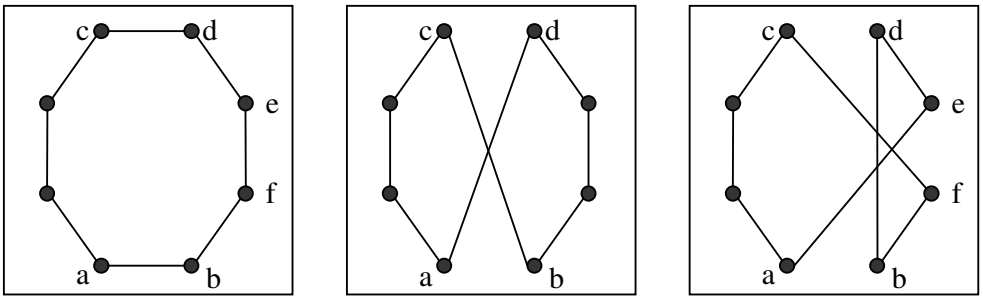
3.4.1 Tek Rota İyileştirmeli Sezgisel Algoritması

ARP içinde yer alan tek bir rotanın iyileştirilmesi amacıyla kullanılan ve 1965 yılında Lin tarafından Gezgin Satıcı Problemi için geliştirilen λ-opt metodu literatürde en fazla kullanılan yöntemlerden biridir. Bu yöntemde λ adet doğru (yol) bir rotadan çıkartılarak mümkün olan tüm permutasyonlarda rotanın çeşitli noktalarına eklenmektedir. Mevcut çözümden daha iyi bir çözüm bulunması halinde yöntem, bu yeni rotayı çıktı olarak vermektedir. λ-opt yöntemi ile genellikle birbiri ile kesişmeyen doğruların bulunduğu bir rota oluşturulmaya çalışılır (Laporte vd., 2000). Şekil 3.1'de 2-opt yönteminin akış şeması yer almaktadır. Akış şemasında yer alan F fonksiyonu o rota için toplam mesafeyi vermektedir. Bu adımları bir yazılım dilinde "opt" adındaki bir fonksiyon içine kodlanıldığı takdirde 3-opt algoritması opt(opt(rota)) ile, 4-opt algoritması ise opt(opt(opt(rota))) ile tekrarlamalı bir şekilde elde edilebilir. Bu prosedür bir iyileştirmenin yapılamadığı lokal bir minimum noktada durmaktadır. λ-opt yönteminin bazı versiyonlarında başlangıç rotasından daha iyi sonucun elde edildiği durumda çözüm saklanarak algoritmadan çıkılmayıp, arama süreci devam ettirilmektedir. Genel olarak bir ARP'de λ-opt algoritması $O(n^{\lambda})$ sürede sonuç bulması beklenmektedir. λ-opt yöntemi üzerinde değişiklikler yapılarak örneğin ardarda gelen belli sayıda nokta yer değiştirilerek bu süre kısaltılabilmektedir.



Şekil 3.1 2-opt Algoritması akış şeması

Şekil 3.2'de ise 2-opt ve 3-opt algoritmasının bir rotaya uygulanması gösterilmektedir. Yöntemler uygulandığında mevcut rotanın uzunluğunda 2-opt ile $d_{ad} + d_{bc} - d_{ab} - d_{cd}$, 3-opt ile ise $d_{ae} + d_{bd} + d_{cf} - d_{ab} - d_{cd} - d_{ef}$ kadar değişiklik olmaktadır.



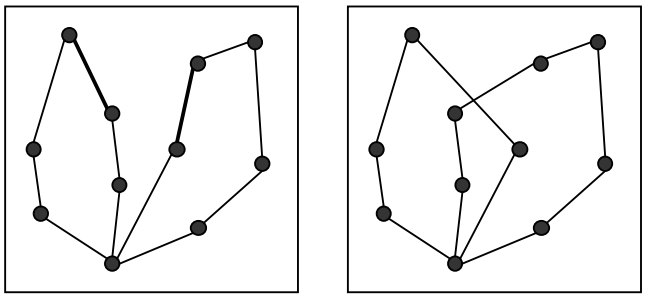
Şekil 3.2 2-opt ve 3-opt algoritmasının bir rotaya uygulanması

Literatürde ARP, GSP ve LRP gibi rotalama problemleri için geliştirilen sezgisel ve metasezgisel yöntemlerde sıkça kullanılan λ-opt algoritması, sürecin ilk aşamalarında ele alınan çözüm için %60, son iterasyonlarda ise bu oran giderek düşerek %2 civarında iyileştirmeler yapmaktadır (Laporte vd., 2000).

3.4.2 Çok Rota İyileştirmeli Sezgisel Algoritmalar

Çok Rota İyileştirmeli Sezgisel Algoritmalarda bir ARP içinde yer alan rotalar arasında nokta alış verişi yaptırılmaktadır. Thompson ve Psaraftis'in 1993 yılında geliştirdikleri b-devirsel k-transfer yönteminde dairesel permutasyon ile sıralanan b rota ele alınıp her rotadan k adet müşteri diğer rotaya atanarak yeni çözümler elde edilmektedir. Literatürde bu konuda yapılan birçok çalışmayı Breedam toplayarak 2-devirsel yer değiştirme yöntemleri için aşağıdaki sınıflandırmayı yapmıştır (Breedam, 2001):

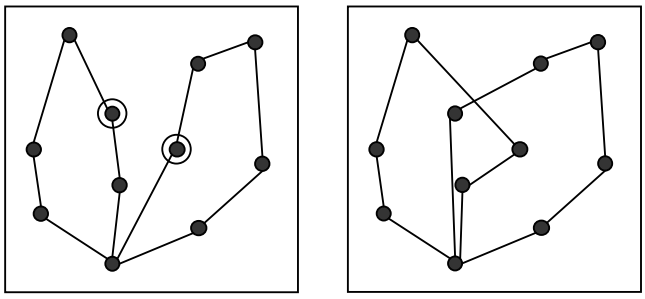
a) Yol Değişimi Yöntemi: Bu yöntemde iki rota arasında, rotalarda yer alan ardışık iki noktayı bağlayan bir yol (kenar) yer değiştirilerek yeni rotalar oluşturulur. Şekil 3.3'de bu yöntem ile ilgili bir örnek verilmektedir.



Önceki Durum Sonraki Durum

Şekil 3.3 Yol Değişimi yöntemi

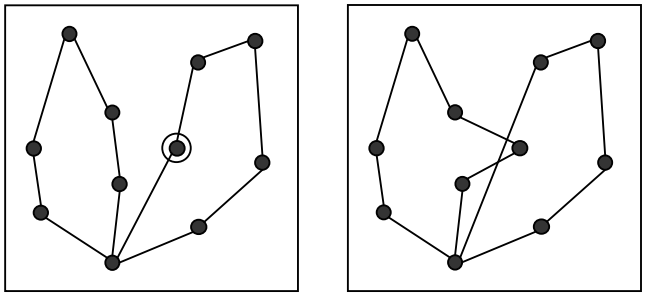
b) Nokta Değişimi Yöntemi: Mevcut iki rota arasında birer nokta yer değiştirilip, rota içinde uygun bir sıraya eklenerek yeni rotalar oluşturulmaktadır. Şekil 3.4'de yer değişimi yapılacak noktalar daire içinde gösterilmektedir.



Önceki Durum Sonraki Durum

Şekil 3.4 Nokta Değişimi yöntemi

c) Nokta Atama Yöntemi: Bu teknikte bir rota içersinden bir nokta seçilip diğer rotada en az maliyet artışını sağlayacak lokasyona atanmaktadır. Bu yöntemle ilgili bir örnek Şekil 3.5'de gösterilmektedir.



Önceki Durum Sonraki Durum

Şekil 3.5 Nokta Atama yöntemi

d) Karma Yöntemler: Probleme yukarıda anlatılan Nokta Değişimi ve Nokta Atama yöntemleri sıra ile uygulanarak en iyi sonuç veren rotalar seçilir. Böylelikle daha etkin bir komşuluk yapısı kurulmuş olur.

Bu dört metot Van Breedam tarafından yaptığı çalışmada test edilmiş ve Nokta Değişimi yönteminin hesaplama süresi ve çözüm kalitesi bakımından en iyi çözümler ürettiği gözlemlenmiştir (Toth ve Vigo, 2002).

3.5 Yapısal Sezgisel Algoritmalar

Yapısal tekniklerde belirli adımlarla problem için araç rotaları oluşturulmaya çalışılmaktadır. KKARP'de Yapısal Sezgisel Algoritmalarla çözüm üreten genel olarak iki yaklaşım bulunmaktadır. Bunlar bir tasarruf kriteri kullanılarak varolan rotaları birleştirme ve bir ekleme maliyeti ile müşterileri araç rotalarına adım adım atama teknikleridir.

3.5.1 Clarke ve Wright Tasarruf Algoritması

ARP için en çok bilinen yöntemlerden biri Clarke ve Wright'ın 1964'de geliştirdikleri Tasarruf algoritmasıdır. Algoritma araç sayısının değişken olarak ele alındığı problemlerde uygulanmaktadır. Bu metot ile rotaların hazırlanmasının yanısıra, kaç adet aracın da gerekeceğini bulunmaktadır. Tasarruf algoritmasının Paralel ve Sıralı olmak üzere iki versiyonu bulunmaktadır. Bu yöntemin adımları aşağıda belirtilmiştir (Laporte vd., 2000):

Adım 1: Her müşteri çifti için tasarruflar $s_{ij} = d_{i0} + d_{0j} - d_{ij}$ formülü ile hesaplanmaktadır. s_{ij} değerleri büyükten küçüğe sıralanır.

Adım 2: Paralel Tasarruf algoritmasında s_{ij} değerlerine göre büyükten küçüğe sıralanan müşteri ikilileri (i-j) şu kurala göre birleştirilir: Eğer sıralamada xy ve yz müşteri çiftleri ardı ardına geliyorsa araç kapasite kısıtı göz önüne alınarak 0-x-y-z-0 güzergahında bir rota oluşturulur. Aksi takdirde xy ve tz formatında müşteri çifti aradarda geliyorsa, bu müşteri çiftleri için ayrı ayrı rota oluşturulur. Sıralamada ele alınan müşteri çiftleri daha önce oluşturulmuş rotalar ile mukayese edilir, eğer uygunsa rotaya dahil edilir. Burada birden fazla rota oluşturulup bu rotaları paralel işleyerek müşteri çiftlerini bu rotalardan birine atamak mümkündür. Tasarruf algoritmasının Sıralı versiyonunda ise ele alınan xy ve tz formatındaki müşteri çiftleri kapasite kısıtı dikkate alınıp birleştirilerek 0-x-y-t-z-0 güzergahında rota oluşturur. Aynı rota için, sıralamada ele alınan müşteri çiftleri mukayese edilir, eğer kapasite kısıtına uygunsa rotaya dahil edilir. Bu yöntemde bir rota tamamıyla oluşuncaya kadar (kapasite kısıtı dikkate alınarak) müşteri çiftleri taranmaktadır. Her iki yöntemde de müşteri çiftleri rotalara toplam maliyet minimum tutulacak şekilde yerleştirilir.

Tasarruf algoritmasında Paralel yöntem, Sıralı yönteme göre daha iyi sonuçlar vermektedir. Bunun sebebi Sıralı Tasarruf algoritmasında sadece bir rota ele alınıp bu rotayı oluşturmak için s_{ij} tasarruf sıralamasının dikkate alınması, Paralel versiyonda ise s_{ij} sıralamasına bakılarak müşteri çiftinin birden fazla rota ile uyumluluğu kontrol edilmesi ve bu sayede daha fazla alternatifin gözden geçirilebilmesidir (Breedam, 2002).

Çizelge 3.1'de mesafe ve s_{ij} tasarruf değerleri verilen ve 5 talep noktasından oluşan bir ARP probleminin Tasarruf algoritması ile çözüm süreci yer almaktadır. Problemde "0" merkez depoyu göstermektedir.

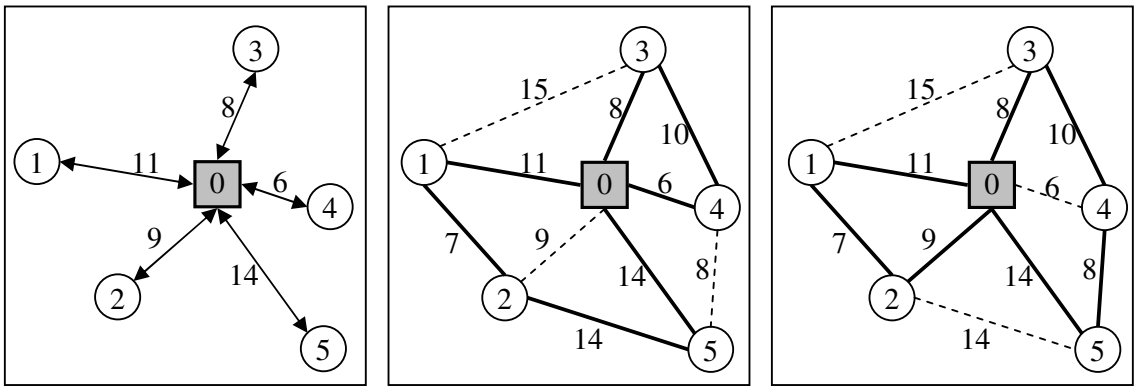
Çizelge 3.1 Noktalar arası mesafe ve s_{ij} değerleri

v_i	v_j	d_{ij}	s_{ij}	v_i	v_j	d_{ij}	s_{ij}	v_i	v_j	d_{ij}	s_{ij}
0	1	11	-	1	2	7	13	2	4	13	2
0	2	9	-	1	3	15	4	2	5	14	9
0	3	8	-	1	4	17	0	3	4	10	4
0	4	6	-	1	5	19	6	3	5	16	8
0	5	14	-	2	3	16	1	4	5	8	12

Problemde araç kapasitesi 10 birim olarak, her müşteri noktası için de talep miktarı 3 birim olarak verilmiştir. Çizelge 3.1'e göre s_{ij} değerleri büyükten küçüğe sıralanırsa, şu müşteri çiftleri elde edilir: 1-2, 4-5, 2-5, 3-5, 1-5, 1-3, 3-4, 2-4, 2-3 ve 1-4.

Sıralı Tasarruf algoritmasına göre ilk olarak 0-1-2-0 başlangıç rotası oluşturulur ve bu rotaya uygun olan diğer müşteri çiftlerine sıra ile bakılır. 0-1-2-5-4-0 rotası kapasite kısıtı olan 10 birimi aştığı için 4-5 nolu müşteri çifti atlanır ve sıradaki 2-5 müşteri çifti rotaya eklenerek 0-1-2-5-0 rotası kontrol edilir. Bu rota kapasite kısıtına uymaktadır, buna göre 5 müşterisi rotaya atanmıştır. Bu rotaya başka müşteri atanamayacağı için diğer rotaya geçilir. Bu rota 4-5 nolu müşteri çifti ile kurulamaz çünkü 5 müşterisi daha önceki rotada yer almaktadır. Diğer rota mecburen 0-3-4-0 güzergahında olacaktır. Böylece Sıralı Tasarruf yöntemi ile 0-1-2-5-0 ve 0-3-4-0 rotası oluşturulmuş olup toplam gidilen mesafe yani amaç fonksiyonu değeri (11+7+14+14) + (8+10+6) = 70 olarak bulunmuştur.

Paralel Tasarruf algoritmasına göre problem çözülürse ilk olarak 1-2 müşteri çifti ele alınarak 0-1-2-0 rotası oluşturulur. Sıradaki müşteri çifti 4-5 için ise 0-4-5-0 güzergahında yeni rota oluşturulmak durumundadır, çünkü 0-1-2-0 rotasında araya atanabilecek müşteri içermemektedir. 2-5 nolu müşteri çifti geçilir, çünkü hem 2 ve hem de 5 nolu müşteri de daha önce rotalarda kullanılmıştır. Sıradaki 3-5 nolu müşteri çifti ise 0-4-5-0 rotasına atanarak 0-3-4-5-0 rotası elde edilir. Bu rota kapasite kısıtına uymaktadır. Tüm müşteriler rotalara atandığı için algoritma sonlandırılır. Bu teknik ile çözüm olarak iki araç için 0-1-2-0 ve 0-3-4-5-0 rotaları bulunmuş, amaç fonksiyonu değeri de (11+7+9) + (8+10+8+14) = 67 olarak hesaplanmıştır. Bu örneğe göre Paralel Tasarruf algoritması, Sıralı Tasarruf algoritmasından daha az maliyetli rotalar hazırlamıştır. Şekil 3.6'da bu probleme ait çözümler yer almaktadır.



Şekil 3.6 Tasarruf Algoritmasına ait örnek

Tasarruf algoritmasının mantığında, ilk olarak her müşteri için bir rota hazırlanılıp bu rotaların bir sıraya göre uygun bir şekilde (kısıtlar dikkate alınarak) birleştirilmesi yatmaktadır. Bu sıra ise her müşteri çifti için tasarruf miktarı hesaplanarak bulunmaktadır. Clarke ve Wright'ın Tasarruf Algoritmaları asimetrik KKARP'ye de uygulanabilmektedir.

Tasarruf algoritması ile çok hızlı bir şekilde çözümler alınabilmesine rağmen, çözüm kalitesi çoğu araştırmacı tarafından vasat kabul edilmektedir. Ayrıca bu yöntemin esnekliği düşük olup yeni kısıtların probleme eklenmesi halinde çözümler oldukça kötüleşmektedir. Bunun sebebi daha önce oluşturulmuş rotalar üzerinde değişikliklerin yapılamamasıdır (Cordeau vd., 2002).

3.5.2 Tasarruf Algoritmasının Geliştirilmiş Türleri

Tasarruf algoritmasının daha iyi sonuçlar bulması amacıyla Gaskell (1967) ve Yellow (1970) tarafından tasarruf hesaplama fonksiyonu $s_{ij} = d_{i0} + d_{0j} - \lambda d_{ij}$ olarak alınmış ve bu sayede müşteriler arasındaki mesafe λ terimi ile ağırlandırılarak daha etkin bir tasarruf değeri elde edilmiştir (Cordeau vd., 2002). Ayrıca 1970 ve 1980'lerde bilgisayarlar çok hızlı olmadığı için bu algoritma kullanılarak büyük ölçekli problemler yavaş çözülmekteydi. Bundan dolayı birbirlerine görece uzak talep noktaları için tasarruf değerlerinin tümü hesaplanmadan, KKARP'yi çözmeye yönelik yöntemler geliştirilmiştir (Laporte vd., 2000).

3.5.3 Eşleme Tabanlı Tasarruf Algoritması

Desrochers ve Verhoog'un 1989 yılında literatüre kazandırdıkları bu yöntem özellikle çok talep noktası olan KKARP problemleri için uygundur. Bu yönteme göre tasarruf değeri s_{pq} bir

kerede hesaplanmayıp, her iterasyonda, p ve q rotasının birleştirilmesinden elde edilmektedir. S_k, k rotasına ait nokta kümesi ve $t(S_k)$ bu noktalara ait Gezgin Satıcı Probleminin (GSP) optimum çözümü ise tasarruf miktarı $s_{pq} = t(S_p) + t(S_q) - t(S_p \cup S_q)$ ile bulunmaktadır. Eşleme Tabanlı Tasarruf Algoritmasına göre rotalar s_{pq} değerlerine göre büyükten küçüğe eşlenerek kapasite kısıtı da dikkate alınıp uygun bir şekilde birleştirilmektedir (Breedam, 2002).

Wark ve Holt'un 1994'de geliştirdikleri başka bir yöntemde ise eşleme ağırlıkları olarak s tasarruf miktarı değeri alınmakta fakat bazı durumlarda belirli bir olasılığa göre rotalar parçalara da bölünebilmekte veya birleştirilebilmektedir. Rotaların her birleştirilme işleminden sonra eşleme ağırlıkları hesaplanmakta ve eğer tüm gruplar birbirleri ile eşlenirse rasgele olarak bazı gruplar seçilerek ikiye bölünmektedir. Bu yöntem iterasyonlar boyunca gruplardan oluşan bir ağaç gibi gelişerek uygun bir çözüm üretir (Toth ve Vigo, 2002).

3.5.4 Sıralı Ekleme Sezgisel Algoritması

Sıralı Ekleme Sezgisel Algoritması, araç sayısının değişken olarak alındığı fakat yükleme kapasitesi bilinen KKARP için geliştirilmiş bir yöntemdir. Bu algoritmada süreç genel olarak başlangıç rotalarının oluşturulması ve bu rotalara ilgili talep noktalarının rotaya en az maliyet artışına sebebiyet verecek şekilde eklenmesine dayanmaktadır. Bu bölümde konuyla ilgili iki çalışma yer alacaktır. Bunlardan biri olan ve 1976'da Mole ve Jameson tarafından geliştirilen modelde bir iterasyonda sadece bir rota ele alınmaktadır. Christofides, Mingozzi ve Toth ise 1979'da bu yönteme sıralı ve paralel rota oluşturan yordamlar uygulamışlardır.

3.5.4.1 Mole ve Jameson Sıralı Eklemeli Sezgisel Algoritması

Mole ve Jameson Sıralı Eklemeli Sezgisel Algoritmasında ele alınan rotaya atanacak yeni nokta için belirli kurallar λ ve μ parametreleri ile belirlenmekte olup, yöntemin akış süreci aşağıda yer almaktadır (Toth ve Vigo, 2002):

Adım 1. Herhangi bir rotaya atanmayan bir talep noktası k için (0, k, 0) başlangıç rotası oluştur.

Adım 2. Rotalanmamış her talep noktası m için kapasite kısıtını dikkate alarak mevcut rotaya ekleme maliyetini $\alpha^*(i_k, m, j_k) = \min\{\alpha(r, m, s)\}$ formülü ile hesapla (r ve s mevcut rotada ardı ardına gelen noktalardır. Eşitlik 3.1'de yer alan α, tasarruf hesaplamaya benzer bir fonksiyon olup, içerdiği λ parametresi ile müşteriler arasındaki mesafe ağırlıklandırılır).

$$\alpha(i, k, j) = d_{ik} + d_{kj} - \lambda d_{ij} \qquad (3.1)$$

Eğer yükleme kısıtından dolayı rotaya herhangi bir müşteri eklemek mümkün değilse adım 1'e dön. Aksi takdirde rotaya eklenecek talep noktası k^*, yükleme kısıtı da dikkate alınarak $\beta(i_{k*}, k^*, j_{k*}) = \max\{\beta(i_k, k, j_k)\}$ formülü ile belirlenir (Burada yer β fonksiyonu aşağıda 3.2'de verilmiş olup, μ parametresi ile müşteri ile depo arasındaki mesafe ağırlıklandırılır).

$$\beta(i, k, j) = \mu d_{0k} - \alpha(i, k, j) \qquad (3.2)$$

Adım 3. Mevcut rotayı 3-opt prosedürü kullanarak optimize et ve adım 2'ye dön (3-opt prosedürü GSP'de kullanılan ve rota içerisinde yer alan noktaların güzergah sırasının yerlerinin değiştirilip, oluşan yeni rotalarda da bu işlemin tekrarlanması ile sürdürülen bir iyileştirme sezgisel algoritmasıdır).

Mole ve Jameson algoritmasında kullanılan μ ve λ parametreleri değiştirilerek farklı ekleme kuralları belirlenebilir. Örneğin $\lambda = 1$ ve $\mu = 0$ alınarak, rotalara en az mesafe artışına sebep olacak talep noktaları eklenebilir. $\lambda = \mu = 0$ alındığında ise rotada yer alan en yakın nokta ile arasındaki mesafenin en az olduğu müşteri rotaya eklenecektir. Eğer $\lambda = \infty$ ve $\mu > 0$ olursa, depoya en uzak nokta rotaya eklenecektir.

3.5.4.2 Christofides, Mingozzi ve Toth Sıralı Eklemeli Sezgisel Algoritması

Bu metotta rotalar ilk olarak sıralı ve daha sonra paralel şekilde ele alınarak problem çözülmeye çalışılmaktadır. Bu yöntemin sıralı versiyonu 4 adımdan oluşmakta olup bu adımlar aşağıda açıklanmaktadır (Toth ve Vigo, 2002):

Adım 1. İlk rotanın indeksini k = 1 olarak ata.

Adım 2. Herhangi bir rotaya atanmamış bir h noktasını k nolu rotaya ekle. Rotalanmamış her i noktası için $\delta_i = d_{0i} + \lambda d_{ih}$ ile λ parametresine bağlı k rotasına ekleme maliyeti hesapla.

Adım 3. $\delta_{i*} = \min\{ \delta_i \}$ sağlayan i^* noktasını kapasite kısıtı gözetilerek k rotasına ata. 3-opt algoritması ile rotayı iyileştir. Bu adımı, k rotasına müşteri atanamayıncaya kadar devam ettir.

Adım 4. Eğer tüm talep noktalarının herhangi bir rotaya atanması tamamlanmışsa dur, aksi takdirde k'yı 1 artır ve adım 2'ye git.

Yöntemin paralel versiyonunda ise kaç adet rota oluşturulacağı önceden biliniyor kabul edilmektedir. Rota sayısı problemin sıralı çözümünde bulunan k değeri de alınabilir.

Adım 1. İlk olarak $R_t = (0 , i_t, 0)$ ($t = 1,...,k$) olmak üzere k adet rota oluştur (Oluşturulan rotalar bir $J = \{ R_1, ..., R_k \}$ kümesi içinde yer almaktadır).

Adım 2. Herhangi bir rotaya atanmamış her i talep noktası için mevcut t rotasına bağlı olarak $\varepsilon_{ti} = d_{0i} + \mu d_{ij}$ ($t = 1,...,k$) işbirliği maliyetini hesapla (μ, noktalar arasındaki uzaklığın ağırlıklandırılmasına yarayan bir parametredir). $\varepsilon_{t*i} = \min_t \{ \varepsilon_{ti} \}$ sağlayan i noktasını t rotası ilişkilendir. Bu adımı açıkta müşteri kalmayıncaya kadar devam ettir.

Adım 3. R_t ile ilişkili olan her i müşterisini için, R_t rotası dışındaki rotalar ile $\varepsilon'_{ti} = \min \{ \varepsilon_{ti} \}$ değerini hesapla ve $\tau_i = \varepsilon'_{ti} - \varepsilon_{ti}$ değerini bul.

Adım 4. R_t rotasına ilişkili olduğu, $\max\{ \tau_i \}$ değerini sağlayan ve kapasite kıtsına uyan i^* müşterisini ata. R_t rotasını 3-opt yöntemi ile iyileştir. R_t rotasına yeni bir talep noktası atanamayıncaya kadar bu adımı devam ettir.

Adım 5. Eğer tüm müşteriler rotalara atanmışsa algoritmayı durdur. Aksi takdirde oluşturulacak rota sayısı k'yı 1 artırarak adım 1'e dön.

3.5.5 En Kısa Yol Yöntemi

Tek işletme biriminin bulunduğu simetrik KKARP'ye uygulanabilen bu yöntemin temel mantığı müşteri noktaları arasında yakınlık uzaklık durumuna ve araç yükleme kapasitelerine göre araçlara müşterilerin atanmasıdır. Hamilton Yol Probleminde de (HYP) kullanılabilen En Kısa Yol Yönteminde süreç aşağıdaki gibi işlemektedir (Breedam, 2002):

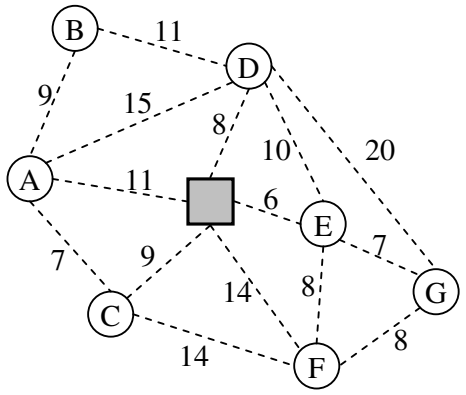
1. Merkezi birimden rotalamaya başlanır. İlk araca merkezi noktaya en yakın müşteri atanır.
2. Rotaya atanan müşteriye en yakın, daha önce rotaya eklenmemiş noktalar incelenir. Eğer müşteriye en yakın iki nokta varsa her biri için süreç oluşturularak ayrı çözüm dalları yaratılır.
3. Eğer müşteri direkt olarak merkezi birim ile bağlantılı değilse (Oluşturulan rotada müşteri merkezi birimden sonra gelmiyorsa) ve müşteriye en yakın başka müşteri ile merkezi birim aynı mesafede ise süreç yine ikiye ayrılarak yeni çözümler oluşturulur. Rotalamada ilk önce müşteri başka müşteriye araç kapasite kısıtı sağlıyorsa bağlanır. Daha sonra çözüm ağacında yeni bir dal oluşturularak müşteri direkt olarak merkezi birime bağlanır.
4. Çözümler ayrı ayrı hesaplanarak en uygun çözüm seçilir.

En Kısa Yol Yöntemini, yükleme kapasitesi 15 birim olan 2 adet araçtan oluşan ve çizelge 3.2'de müşteri talepleri verilmiş KKARP'de inceleyelim. Şekil 3.7'de Öklidyen düzlem üzerinde müşteriler, işletme birimi ve düğümler arası mümkün olan yollar gösterilmektedir.

Çizelge 3.2 Örnek problem için müşteriler ve talep miktarları

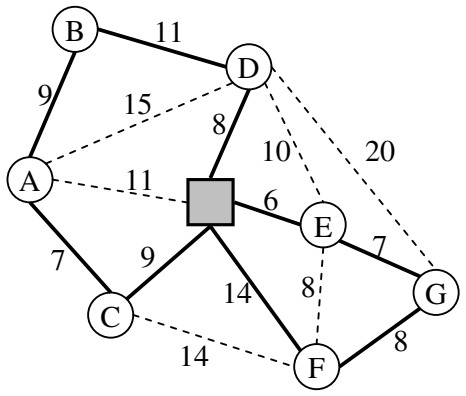
Müşteri	Talep Miktarı	Müşteri	Talep Miktarı
A	3	E	4
B	5	F	5
C	3	G	3
D	4		

Buna göre ilk araç merkezi birimden (kare) ilk olarak en yakın nokta olan E müşterisine, daha sonra G müşterisine ve oradan da F müşterisine gidecektir. Bu üç müşteriyi dolaşması için gerekli yükleme miktarı 4 + 3 + 5 = 12'dir. F müşterisine en yakın ve daha önce uğranmamış olan müşteri C müşterisidir, bununla beraber merkezi birim de aynı mesafededir (14). Bu durumda süreç ikiye ayrılarak iki koldan devam edecektir. İlk çözüm dalına göre araç rotasını E-G-F müşteri sırasıyla tamamlayacaktır. Buna göre de ikinci aracın rotası D-B-A-C sırasıyla ve 15 birim yükleme yaparak olacaktır. Bu durumda ilk çözüm dalının amaç fonksiyon değeri 35 + 44 = 79 olacaktır. İkinci çözüm dalına göre ise ilk aracın rotası E-G-F-C şeklinde olacak ve yükleme 15 birim yapılacaktır. İkinci araç ise D-B-A güzergahını izleyecek ve yükleme kapasitesinin12 birimi kullanılacaktır. Böylece ikinci çözüm dalının amaç fonksiyonu değeri de 44 + 39 = 83 olacaktır.



Şekil 3.7 En Kısa Yol Yöntemi örnek KKARP

En Kısa Yol Yöntemine göre en uygun çözüm ilk aracın E-G-F rotasını, ikinci aracın ise D-B-A-C rotasını izlemesi olup, toplam mesafe olarak da 79 birim gidilecektir. Şekil 3.8'de bu çözüm kalın çizgilerle çizilmektedir.



Şekil 3.8 En Kısa Yol Yöntemi örnek KKARP Çözümü

3.6 Çok Aşamalı Sezgisel Algoritmalar

KKARP için geliştirilen Çok Aşamalı Sezgisel Algoritmalar genellikle iki adımda yapılmaktadır. Bunlar, müşterilerin kapasite kısıtına göre uygun gruplara ayrılması ve grup içinde rotalama işleminin yapılmasıdır. Çok Aşamalı Sezgisel Algoritmalar hangi adımın önce ve sonra yapılacağına göre İlk Grupla Sonra Rotala ve İlk Rotala Sonra Grupla olmak üzere ikiye ayrılmaktadır. Genellikle ikinci adımda yapılan işlem, ilk adımda yapılan işleme geri beslemeler ile sürdürülmektedir.

3.6.1 İlk Grupla Sonra Rotala Yöntemleri

İlk Grupla Sonra Rotala yöntemleri genel olarak üç kategoriye ayrılmaktadır. İlk kategoride talep noktaları bir kez temel gruplara bölünüp her grup için rotalama yapılmaktadır. Bu çalışmada bu kategoriye örnek olarak Süpürme Algoritması, Fisher ve Jaikumar Atama Tabanlı Algoritması ve Bramel ve Simchi-Levi Lokasyon Tabanlı Algoritması yer alacaktır. İkinci kategori tam sonuç veren bir yöntemin sadeleştirilmiş şekli olan ve diğer yöntemlere nazaran daha iyi sonuçlar veren Kısaltılmış Dal Sınır Yöntemidir. Üçüncü kategoride ise Taç Yaprağı Algoritmaları yer almakta ve bu teknikte çok sayıda nokta grubu oluşturulup uygun olanları seçilerek çözüm bulunmaya çalışılmaktadır.

3.6.1.1 Süpürme Algoritması

1974 yılında Gillet ve Miller tarafından geliştirilen Süpürme Algoritmasında rotalarda yer alacak müşteriler, depo merkezli bir doğrunun döndürülmesi ile elde edilmektedir. Döndürme esnasında doğrunun üzerinden geçtiği müşteriler bir gruba ayrılır ve kapasite veya mesafe kısıtı aşıldığı zaman grup kapatılarak yeni bir grup ile devam edilir. Oluşturulan nokta

gruplarına merkez depo da eklenip, genel olarak GSP gibi çözülerek rotalar belirlenir (Cordeau vd., 2002). Bu yöntemde müşterilerin koordinatları Öklidyen formatta (x, y) değil, θ açı ve ρ doğru uzunluğu olmak üzere polar formatta (θ_i, ρ_i) tutulur. Böylece talep noktaları θ açısına göre küçükten büyüğe sıralanıp, kapasite ve mesafe kısıtları dikkate alınarak gruplanır. Genel olarak Süpürme Algoritmasının çözüm adımları aşağıdaki açıklanmaktadır (Laporte vd., 2000):

Adım 1. Bir harita üzerinde depo ve müşteri noktalarının yeri tespit edilir ve koordinatlar polar formata (θ_i, ρ_i) çevrilir. Rotaya atanmamış herhangi bir araç belirlenir.

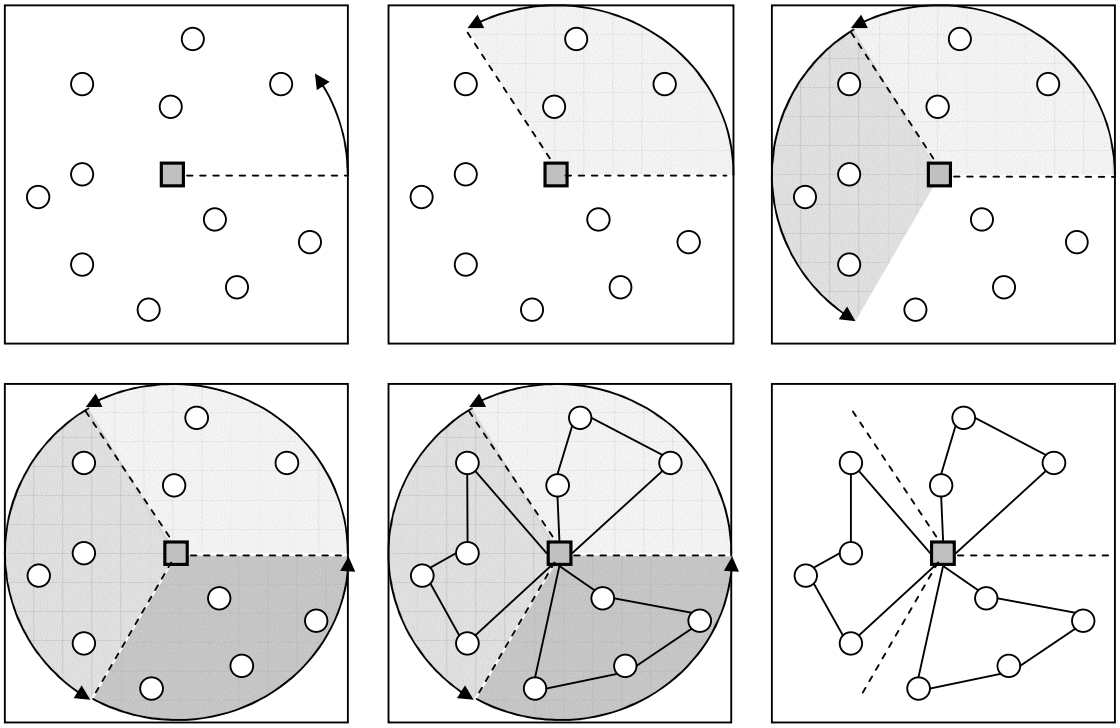
Adım 2. Depodan yatayla 0^o açı ile başlanarak saat yönünün tersine doğru taranmaya başlanır. Eğer bir müşteri ile karşılaşılırsa ve eğer müşterinin talep miktarı aracın kapasitesini geçmiyorsa müşteri araca atanır. Aksi takdirde saat yönünün tersi yönde hareket edilir. Eğer araca her iki yönde de müşteri atanamıyorsa, diğer araca geçilir.

Adım 3. Eski aracın kaldığı yerden taramaya devam edilir. Eğer daha önce rotalanmayan müşteri ile karşılaşılırsa ve müşterinin araca atanması kapasite kısıtının aşılmasına sebep olmuyorsa, müşteri araca atanır. Bu süreç açıkta bir talep noktası kalmayıncaya kadar (tüm noktalar rotalanıncaya kadar) sürer.

Adım 4. Tüm noktalar araçlara atandıktan sonra gruplar uygun bir şekilde optimize edilerek rotalar belirlenir.

Süpürme Algoritmasının adımları aşağıda yer alan 11 müşterili KKARP üzerinde Şekil 3.9'da gösterilmektedir. Şekilde 0^o açısından başlanarak saat yönünün tersi yönünde dönülmektedir. Kapasite veya mesafe kısıtı ihlal edildiğinde ise diğer gruba geçilmektedir. Daha sonra noktalar birleştirilerek rotalar oluşturulmaktadır.

Süpürme yöntemi Tasarruf algoritmasına nazaran uygulanması daha basit olmasına rağmen, çoğu ARP için kesinlik ve hız bakımından Tasarruf algoritmasından geridedir. Bunun sebebi gruplama yapılırken noktalar arasındaki uzaklıkların dikkate alınmaması ve rotaların belirli bir alandan başlayarak oluşturulmasıdır. Ayrıca bu tekniğin yeni kısıtlarla entegre olması zor olmakla birlikte sadece iki boyutlu Öklidyen sistemde çözümler üretmesi uygulanabilirliğini kısıtlamaktadır (Cordeau vd., 2002).



Şekil 3.9 Süpürme Algoritması adımları

3.6.1.2 Fisher ve Jaikumar Atama Tabanlı Algoritması

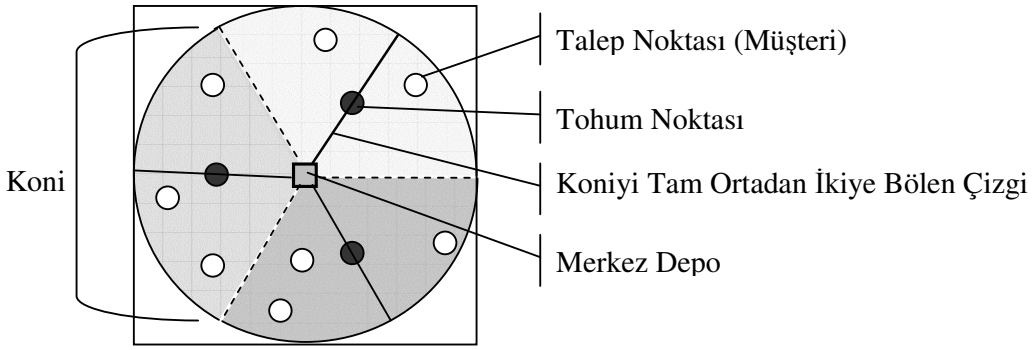
Araç sayısı (rota sayısının) sabit kabul edildiği KKARP'ye uygulanan bu yöntem 1981 yılında Fisher ve Jaikumar tarafından geliştirilmiştir. Bu teknikte geometrik olarak gruplama yerine, Genel Atama Problemi çözümü ile gruplama yapılmakta ve oluşturulan gruplara GSP çözüm teknikleri uygulanarak rotalar belirlenmektedir (Cordeau vd., 2002). Bu yöntemin işleyiş adımları aşağıda belirtilmiştir (Laporte vd., 2000):

Adım 1. Problemin yer aldığı düzlemi, tepe noktaları depo olacak şekilde müşteri talep miktarları ve kapasite kısıtları dikkate alarak K adet koniye böl. Her bir koni için koniyi tam ortadan bölen bir doğru parçasının ortasında yer alan sanal j_k tohum noktası belirle (k = 1,...,K) (j_k tohum noktası k konisi için maliyet hesaplamada kullanılacaktır. Şekil 3.10'da 8 noktalı bir KKARP için tohum noktalarının yerlerinin nasıl belirlendiği gösterilmektedir).

Adım 2. Her i. müşterisinin k. gruba atanma maliyetini (c_{ik}) aşağıdaki formüle göre hesapla:

Asimetrik KKARP için $c_{ik} = \min \{ d_{0i} + d_{ij_k} + d_{j_k 0} , d_{0j_k} + d_{j_k i} + d_{i0} \} - (d_{0j_k} + d_{j_k 0})$ (3.3)

Simetrik KKARP için $c_{ik} = d_{0i} + d_{ij_k} - d_{j_k 0}$ (3.4)



Şekil 3.10 Fisher ve Jaikumar algoritması için kurulan yapı

Adım 3. Genel Atama Problemini c_{ij} maliyeti, q_i talep miktarı ve C araç kapasite değerine göre çöz ve grupları tamamla.

Adım 4. Oluşturulan her grup için rotaları bir iyileştirme sezgiseli ile optimize et.

3.6.1.3 Bramel ve Simichi-Levi Lokasyon Tabanlı Algoritması

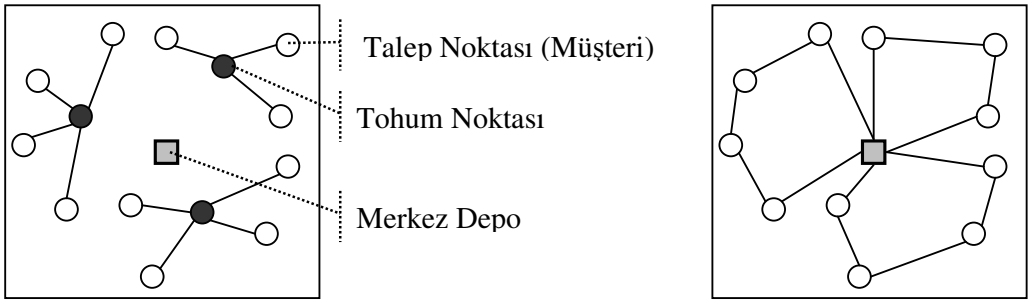
Fisher ve Jaikumar'ın geliştirdikleri atama tabanlı yönteminden farklı olarak, Bramel ve Simichi-Levi Lokasyon Tabanlı Algoritmasında tohum noktalarının yeri, Kapasite Kısıtlı Çoklu Tesis Lokasyonu Belirleme Problemi çözülerek bulunmaktadır. Kapasite Kısıtlı Çoklu Tesis Lokasyonu Belirleme Problemi, müşteri koordinatları ve talep değerleri bilinen bir düzeneğe, n adet ve C kapasiteli tesisin bağlı olduğu müşterilere en yakın konumda olması için hangi koordinatlara kurulacağının bulunduğu problemdir. Algoritmanın işleyiş adımları aşağıda belirtilmektedir (Toth ve Vigo, 2002):

Adım 1. Toplam talepleri C araç kapasitesini geçmeyecek şekilde müşteriler gruplanarak n adet sanal tesise (tohum noktasına) bağlanırlar. Tohum noktalarının koordinatları, oluşturdukları grup içinde yer alan müşterilere toplam mesafesinin minimum olması hedefiyle belirlenir. Aynı tohum noktasına bağlı talep noktaları bir grup oluştururlar. Şekil 3.11'de tohum noktalarının yerlerinin gösterildiği bir örnek yer almaktadır. Örneğe göre bir tohum noktası ile ilişkili tüm noktalar aynı rotanın içinde tutulmaktadır.

Adım 2. Her grupta yer alan müşterilere merkez depo da eklenip bir rota oluşturulur. Aracın müşterilere ziyaret sırası, ekleme mantığı ile belirlenir. Bir k grubunda yer alan tüm noktalar $T_k = \{0, i_1,...,i_L\}$ ise rasgele bir nokta seçilerek başlangıç rotası oluşturulur. Daha sonra, rotaya eklenecek talep noktası i'nin sırası daha önce rotalanmış noktalar ile aşağıdaki maliyet fonksiyonlarından biri seçilerek belirlenir:

Direkt maliyet: $c_{ik} = \min \{2d_{ih}\}$ (h → daha önce rotaya atanmış noktalar) (3.5)

En yakın nokta maliyeti: $c_{ik} = \min \{d_{ih} + d_{ih-1} + d_{ih+1}\}$ (3.6)



Şekil 3.11 Bramel ve Simichi-Levi algoritması örneği

3.6.1.4 Kısaltılmış Dal-Sınır Yöntemi

Chritofides, Mingozzi ve Toth tarafından 1979'da geliştirilen bu algoritmada basit bir arama ağacı yapısı kullanılmaktadır. Arama ağacında her seviyede tek bir dal bulunmakta, her iterasyonda rotalanmamış bir müşteri seçilerek onun içinde yer alması mümkün rotalar üzerinden sonuç bulunmaya çalışılmaktadır. Kısaltılmış Dal-Sınır yönteminin ilerleyiş süreci aşağıda verilmektedir:

Adım 1. Rota indeksi değişkenine h = 1 ata. Rotalanmamış müşteri kümesini $F_h = V / \{0\}$ olarak düzenle.

Adım 2. Eğer $F_h = Ø$ ise algoritmayı durdur. Aksi takdirde F_h kümesinden rotalanmamış bir i müşterisi seç ve kapasite kısıtını dikkate alarak i müşterisinin içinde yer aldığı R_i rota kümesini oluştur (Bu küme oluşturulurken tasarruf ve ekleme maliyeti fonksiyonlarının lineer kombinasyonu kullanılır).

Adım 3. R_i'de yer alan tüm r rotaları için $f(r) = t(S_r \cup \{0\}) + u(F_h \setminus S_r)$ değerleri hesaplanır. Burada S_r, r rotasında yer alan noktalar kümesi, $t(S_r \cup \{0\})$ ifadesi $S_r \cup \{0\}$ nokta grubundan oluşan GSP'nin iyi bir çözüm değerini ve $u(F_h \setminus S_r)$ ise rotalanmamış müşteriler için en kısa arama ağacı uzunluğunu vermektedir.

Adım 4. R_i rota kümesi içersinden $\min_{r \in R_i} = \{ f(r) \}$ sağlayan r^* rotasını seç. Rota indeksi h değerini bir artır ve rotalanmamış müşteri kümesini $F_h = F_{h-1} \setminus S_{r^*}$ olarak güncelle ve adım 2'ye git.

3.6.1.5 Taç Yaprağı Algoritmaları

Süpürme Algoritmasının doğal olarak geliştirilmiş hali olan Taç Yaprağı algoritmasında oluşturulan rotalara taç yaprağı adı verilmektedir. Bu algoritmada birbiri ile ortak noktaları kullanılabilen bir rota havuzu oluşturulmakta ve aşağıda yer alan modele göre rotaların arasından seçim yapılarak KKARP'ye çözüm üretilmektedir (Laporte vd., 2000):

$$\min \sum_{k \in S} d_k x_k \tag{3.7}$$

Şu kısıtlar ile:

$$\sum_{k \in S} a_{ik} x_k = 1 \quad i = 1,...,n \tag{3.8}$$

$$x_k \in \{ 0, 1 \} \quad k \in S \tag{3.9}$$

$$a_{ik} \in \{ 0, 1 \} \quad k \in S \text{ ve } i = 1,...,n \tag{3.10}$$

Burada S ifadesi rota setlerini, n müşteri sayısını, k oluşturulan rota sayısını ve d_k ise k rotasının toplam mesafe değerini ifade etmektedir. Eğer $a_{ik} = 1$ ise i nolu müşteri k rotasına aittir. Eğer $x_k = 1$ ise de k rotası çözüme aittir. Bu model ilk olarak Balinski ve Quandt tarafından 1964 yılında kurulmuş olup, büyük boyutlu problemlerde oluşturulabilecek rota sayısı ($|S|$) fazla olduğundan dolayı bu problemlere uygulanması zorlaşmaktadır. Foster ve Ryan tarafından 1976'da geliştirilen, sezgisel kurallar ile bir araca atanan müşterilerin daha uygun bir şekilde belirlendiği Taç Yaprağı Algoritmasına, Tekli Taç Yaprağı Algoritması denilmektedir. Renaud vd. tarafından bu kurallar 1996 senesinde bir adım daha ileri götürülerek aynı anda iç içe veya kesişen iki rota ele alınmaktadır. Bu yönteme, Çiftli Taç Yaprağı Algoritması adı verilmiştir. Yazarlara göre Çiftli Taç Yaprağı Algoritması, KKARP için diğer klasik sezgisel yöntemlere göre daha iyi sonuçlar vermektedir (Toth ve Vigo, 2002).

Esneklik göz önüne alındığında Taç Yaprağı algoritmalarına birçok kısıt etkili olarak eklenebilmektedir. Fakat algoritmanın kodlanması zor olmakla beraber, diğer klasik sezgisel tekniklere göre en karmaşık yöntemdir (Cordeau vd., 2002).

3.6.2 İlk Rotala Sonra Grupla Yöntemleri

Bu tip algoritmalarda ARP ilk olarak, GSP gibi ele alınıp kapasite kısıtları göz ardı edilerek her müşterinin içinde yer aldığı büyük bir tur oluşturulur. Daha sonra ikinci aşamada bu tur

uygun araç rotalarına bölünerek probleme çözüm bulunmaya çalışılır, dolayısıyla bu yöntem değişken sayılı KKARP için uygulanabilirdir. Bu algoritmayla ilgili ilk çalışmayı 1983 yılında Beasley yapmış ve ikinci aşamanın standart bir en kısa yol problemi olduğunu ileri sürmüştür. En kısa yol probleminde, i. ve j. düğüm noktaları arasındaki c_{ij} maliyeti $d_{0i} + d_{0j} - L_{ij}$ ile hesaplanmaktadır. Burada L_{ij} ifadesi i. ve j. düğüm noktaları arasındaki en kısa mesafedeki GSP turunu vermektedir (Toth ve Vigo, 2002). Literatürde genel olarak İlk Rotala Sonra Grupla Yöntemleri ile ilgili çalışma az sayıda olmakla beraber bu çalışmalar da şimdiye kadar diğer yaklaşımlarla kıyaslanmamıştır.

3.7 ARP İçin Klasik Sezgisel Metotların Karşılaştırılması

Çizelge 3.3'de yer alan ARP için Cordeau ve arkadaşları tarafından yapılan değerlendirmede klasik sezgisel yöntemlerin karakteristikleri belirtilmiştir. Buna göre genellikle basit olan bu algoritmalar hızlı sonuçlar vermektedir. Fakat üretilen çözümlerin kalitesi iyi değildir ve gerçek hayatta bu algoritmaları yeni kısıtlarla uygulamak zordur (Cordeau vd., 2002).

Çizelge 3.3 ARP için klasik sezgisel metotların değerlendirilmesi

Yöntem	Kesinlik	Hız	Basitlik	Esneklik
Clarke ve Wright Tasarruf Algoritması	Düşük	Çok Yüksek	Çok Yüksek	Düşük
Eşleme Tabanlı Metotlar	Yüksek	Çok Düşük	Düşük	Düşük
Süpürme Yöntemi	Düşük	Orta-Yüksek	Yüksek	Düşük
1-Taç Yaprağı Yöntemi	Düşük	Yüksek	Orta	Orta
2-Taç Yaprağı Yöntemi	Orta	Orta	Orta	Orta
Fisher ve Jaikumar Atama Tabanlı Yöntemi	-	Orta	Düşük	Düşük
Bramel ve Simichi-Levi Lokasyon Tabanlı Algoritması	Orta	Düşük	Düşük	Düşük

4. METASEZGİSEL ALGORİTMALAR

Literatürde ilk defa 1986 senesinde Glover tarafından kullanılan "metasezgisel" terimi Yunanca'da "daha ilerisi, üst derecede olanı, daha moderni" anlamına gelen "meta" öneki ile İngilizce'de "sezgisel, buluşsal" anlamına gelen "heuristic" kelimesinin birleşiminden oluşmuştur. "Heuristic" kelimesi ise Arşimet'in ünlü "Eureka!" (Buldum!) sözünden türetilmiştir. Literatürde "Metasezgisel" kavramı, "Modern Sergisel" veya "Yapay Zeka Yaklaşımı" olarak da geçmektedir (Nabiyev, 2003).

Metasezgisel yöntemler çözüm uzayında etkili bir şekilde arama yapmak için, farklı yapılardaki alt kademe sezgisel algoritmaların zekice birleştirilmesi ile oluşturulmuş iteratif problem çözme prosedürleridir. Bu yöntemler her iterasyonda bir çözümden veya çözüm koleksiyonunda yola çıkarak yeni çözümler üretirler. Çoğu metasezgisel yaklaşım çözüm uzayında stokastik fakat bilinçli bir şekilde arama yapar (Blum ve Roli, 2003).

Metasezgisel algoritmalar, kombinatoryal problemler için geliştirilmiş genel optimizasyon tekniklerdir. Bu yöntemlerin en önemli özelliği sadece belli bir tip problem için değil, tüm kombinatoryal problemlere uygulanabilir esneklikte olmasıdır. Bu sebeple literatürde bu tekniklerin birçok başarı hikayesine rastlanmaktadır. Bir metasezgisel algoritmanın iyi sonuçlar üretmesi için yapılması gereken ilk işlem, yöntemin temel kavramlarının probleme iyi bir şekilde adapte edilmesidir (Hertz ve Widmer, 2003).

Metasezgisel algoritmalarda amaç, arama uzayının en umut verici noktalarında arama yapıp yerel optimum noktalardan kurutularak, optimum çözüme yakın sonuçlar elde etmektir. Bu yöntemler karmaşık komşuluk arama kuralları, bellek yapıları ve çözümlerin kombinasyonundan yeni çözümler elde ederek sonuç üretmeye çalışırlar. Bu metotlarla üretilen sonuçlar, klasik sezgisel algoritmalarla üretilen sonuçlardan daha iyidir fakat hesaplama işlemi klasik algoritmalara kıyasla daha uzun sürmektedir. Metasezgisel yöntemler problemin özelliklerine bağımlıdırlar, diğer bir deyişle problem tipine göre farklı parametreler gerekebilmektedir. Bu da diğer problemlere uygulanabilirliğini zor kılar. Metasezgisel algoritmalar, klasik sezgisel algoritmaların doğadan esinlenerek geliştirilmiş hali olarak görülebilir (Laporte vd., 2000).

Çoğu sezgisel algoritmalar probleme bağımlı algoritmalardır. Diğer bir deyişle, bir problem için en iyi performans gösterirken diğer bir problem için aynı şekilde başarılı çözüm üretmeyebilir. Bunda dolayı çok daha genel olarak uygulanabilen algoritmaların geliştirilmesine yönelik çalışmaların sayısı her geçen gün hızlı bir şekilde artmaktadır. Geçen

30-40 yılda bu özelliğe sahip çeşitli yaklaşımlar geliştirilmiştir. Bu yaklaşımlar sosyal, biyoloji, zooloji, fizik, bilgisayar ve karar verme gibi bilimler temel alınarak türetilmiştir. Bundan dolayı bu tür yaklaşımlara modern sezgisel yaklaşımlar ya da yapay zeka yaklaşımları adı verilmektedir (Karaboğa, 2004).

Blum ve Roli, metasezgisel yöntemler hakkında aşağıdaki genellemeleri yapmışlardır (Blum ve Roli, 2003):

- Metasezgisel yöntemler arama prosesi sırasında kılavuzluk yapan stratejilerdir.
- Genel amaç, arama uzayını etkili bir şekilde araştırıp optimum veya optimuma yakın sonuçlar elde etmektir.
- Metasezgisel yöntemler lokal arama tekniklerinden, karmaşık öğrenme prosedürlerine kadar yaygınlık gösterir.
- Metasezgisel algoritmalar Yaklaşık yöntemlerdir ve genellikle deterministik değildir.
- Arama uzayında lokal optimum noktalardan takılıp kalmayı engelleyecek mekanizmaları içinde barındırırlar.
- Metasezgisel teknikler probleme özel yöntemler değildirler. Genellikle tüm kombinatoryal problemlere uygulanabilirler.
- Günümüzde gelişmiş metasezgisel algoritmalarda arama sırasında kılavuzluk etmesi için hafıza tabanlı süreçler bulunmaktadır.

Metasezgisel algoritmalar zor kombinatoryal optimizasyon problemlerinde iyi sonuçlar üretmek için ileri seviye sezgisel prosedürlerdir. Her metasezgisel yöntemin birden fazla parametreye sahip olması esneklik sağlar fakat bu durum problem tiplerine göre de değişiklik göstermektedir. Çoğu metasezgisel algoritma paralel işlem için uyumludur. Metasezgisel algoritmaların sınıflandırılması aşağıdaki şekilde yapılabilir (Tarantilis vd., 2004a):

- Doğadan esinlenerek geliştirilenler – Doğadan esinlenmeden geliştirilenler
- Populasyon tabanlı algoritmalar – Tek nokta (lokal arama) algoritmalar
- Dinamik amaç fonksiyonlu – Statik amaç fonksiyonlu algoritmalar
- Tek komşu yapılı – Çok komşu yapılı algoritmalar
- Hafıza kullanan – Hafıza kullanmayan algoritmalar

Metasezgisel algoritmaların lokal komşuluk ve populasyon arama tabanlı iki türü vardır. Lokal komşuluk arama yöntemlerinde her iterasyonda bir çözüm ele alınıp uygun komşu çözümleri incelenerek arama uzayında belirli bir noktada yoğun bir tarama yapılır. Lokal arama yöntemine örnek olarak Tavlama Benzetimi ve Tabu Arama algoritması verilebilir. Populasyon tabanlı arama yöntemlerinde ise her iterasyonda bir çözüm havuzu içersinde bazı çözümler seçilip uygun bir şekilde birleştirilerek yeni çözümler elde edilir. Genetik Algoritma, iki çözümün rasgele seçilen parçaları birleştirilerek yeni çözüm elde edilen bu tip yöntemlere bir örnektir. Adaptif Hafıza prosedürlerinde ise birçok çözümden birden fazla yeni çözüm elde edilir (Cordeau vd., 2002).

4.1 Tavlama Benzetimi Algoritması

Tavlama metal malzemelerde katı halde sıcaklık değişmeleri ile bir yada birbirine bağlı birkaç işlemle amaca uygun özellik değişmelerinin sağlanması olayı şeklinde tanımlanır. Bu tanım gereği malzemenin belirli bir sıcaklığa kadar ısıtılması, bu sıcaklıkta uygun bir süre tutulması ve belirli bir stratejiye göre sıcaklığın oda sıcaklığına kadar azaltılması sayesinde üç aşamada malzemede özellik değişmeleri sağlanır. Metal malzemelerde ısıl işlemle özellik değişmeleri sağlanırken, malzemenin kimyasal bileşiminde değişiklik yapılmadan kristal ya da kafes yapısında düzenlemeler yapılabilir (Karaboğa, 2004).

Tavlama Benzetimi (TB), fiziksel tavlama prosesi ile kombinatoryal optimizasyon problemlerinin benzeşiminden ortaya çıkarılmış bir yöntemdir. Fiziksel tavlama prosesi, ısı banyosu içerisindeki katı bir cismin düşük enerjilerini elde etmek için uygulanmaktadır. Eğer katı bir cisim erime noktasına kadar ısıtılır ve sonra hızla soğutulmaya başlanırsa, katı cismin moleküler yapısı (kristal yapı) soğutma oranına bağlı olarak değişir. TB, ilk olarak 1953 yılında Metropolis, Rosenbluth ve Teller tarafından bir ısı banyosu içindeki taneler kümesinin denge dağılımını hesaplamak amacıyla uygulanmıştır. Metropolis simülasyon yardımı ile enerjideki değişimi hesaplamıştır. Metropolis algoritmasının temel prensibi, soğutma prosesindeki enerji değişimine göre belirlenir. Termodinamik kanuna göre, t. Durumdaki sıcaklığa bağlı olarak enerjideki azalma olasılığı aşağıdaki formül ile elde edilir (Eren, 2002):

$$P(\delta E) = \exp(\delta E/k.t) \qquad (4.1)$$

4.1 nolu formüldeki k, fiziksel sabit olup Boltzman sabiti olarak da bilinir. Eğer enerji azalır ise sistem yeni bir duruma geçer, enerji artar ise yeni durum kabul edilir. Bu işlemler donma noktasına kadar devam eder. Termodinamik simülasyonundaki Metropolis algoritması ve

TB'nin optimizasyon problemlerine uygulanması ile ilgili çizelge 4.1'de gösterilen benzerlikler bulunmaktadır (Eren, 2002):

Çizelge 4.1 Termodinamik Simülasyonu ve Kombinatoryal Optimizasyon

Termodinamik Simulasyonu	**Kombinatoryal Optimizasyon**
Sistem Durumu	Uygun Çözüm
Enerji	Maliyet
Durum Değişmesi	Komşuluk Aralığı Çözümü
Sıcaklık	Kontrol Parametreleri
Donma Noktası	Yaklaşık Çözüm

Tavlama Benzetimi (TB) orijinini doğal tavlama işleminden almaktadır. Bu algoritmanın optimizasyon problemlerine uygulanması ile ilgili çalışmalar 1983 yılında Kirkpatrick ve arkadaşları tarafından yapılan bir çalışma ile başlamıştır. Algoritma metallerin tavlama işlemi ile bir optimizasyon problemine çözüm araştırma olayları arasındaki benzerlikten ilham alınarak ortaya konulmuştur. Burada metaldeki atomların durumları optimizasyon probleminin muhtemel çözümlerine ve durumların enerjileri de çözümlere ait amaç fonksiyon değerlerine karşılık gelmektedir. Hızlı soğutma işlemi ise bölgesel optimizasyon işlemine tekabül etmektedir. Dış sıcaklık sıfır olduğunda daha yüksek enerjili bir duruma geçiş mümkün olmaz. Böylece bölgesel optimizasyondaki gibi yukarı doğru hareketler yasaklanır ve araştırma bir bölgesel minimaya takılı kalır. Bu işlemde sıcaklık (T) çeşitli seviyeler boyunca yavaşça düşürülür. Enerji seviyesinden uzaklaşmamayı sağlamak için mevcut sıcaklığı belirli bir süre muhafaza etmek suretiyle ve sıfır dereceye yaklaşıncaya kadar bu işlemlerin tekrarlanması bölgesel optimallikten kaçışı sağlayabilmektedir (Karaboğa, 2004).

TB, bilinen en eski metasezgisel algoritma olup, lokal optimum noktalara takılmayı önleyici, belirgin bir stratejiye sahiptir. Algoritmanın ana mantığında, arama uzayında lokal noktalardan kurtulmak için mevcut çözümden daha kötü çözümlerin belirli bir olasılıkla kabul edilmesi yatar. Bu olasılık değeri iki çözümün amaç fonksiyon değerleri arasındaki farkla ve sıcaklık değeri ile hesaplanır. Kötü çözümlerin kabul edilme olasılığı arama süresince düşürülür ve bu düşürme işlemi de metallerin ve camların tavlama prosesine benzemektedir (Blum ve Roli, 2003).

4.1.1 Tavlama Benzetimi İşlem Adımları

Güvenilir sezgisel bir araştırma algoritması başlangıç noktasına bağımlılığı az olan algoritmadır. TB'de çözüm uzayı taranırken yukarı doğru hareketler yani amaç değerinin aleyhine hareketler, değişen bir olasılık tabanlı fonksiyon ile kontrol edilir. TB genel olarak iteratif bir geliştirme algoritmasıdır. Standart bir TB algoritmasının temel adımları aşağıda verilmektedir (Breedam, 2001):

Adım 1. Rasgele olarak bir başlangıç çözümü üret ve en iyi çözüm S olarak ata. Ayrıca t iterasyon indeksini 0 olarak ata.

Adım 2. Bir başlangıç sıcaklık değeri T_B belirle ve mevcut sıcaklık değeri $T_0 = T_B$ olarak ata.

Adım 3. En iyi çözümden hareketle rasgele komşu bir çözüm $S^1 \in N(S)$ oluştur.

Adım 4. Üretilen S^1 çözümüyle S çözümünün amaç fonksiyonu değerleri arasındaki farkı hesapla ($\delta = C(S^1) - C(S)$)

Adım 5. Eğer S^1, S'den daha iyi ($\delta < 0$) ise S çözümüne S^1 çözümünü ata. S^1, S'den daha kötü fakat mevcut T_t sıcaklığında ($e^{\frac{-\delta}{T}}$) > θ sağlanıyorsa (θ, 0 ile 1 arasında rasgele üretilmiş bir sayıdır) S çözümü ile S^1 çözümünü yer değiştir. Yoksa S'i mevcut çözüm olarak muhafaza et.

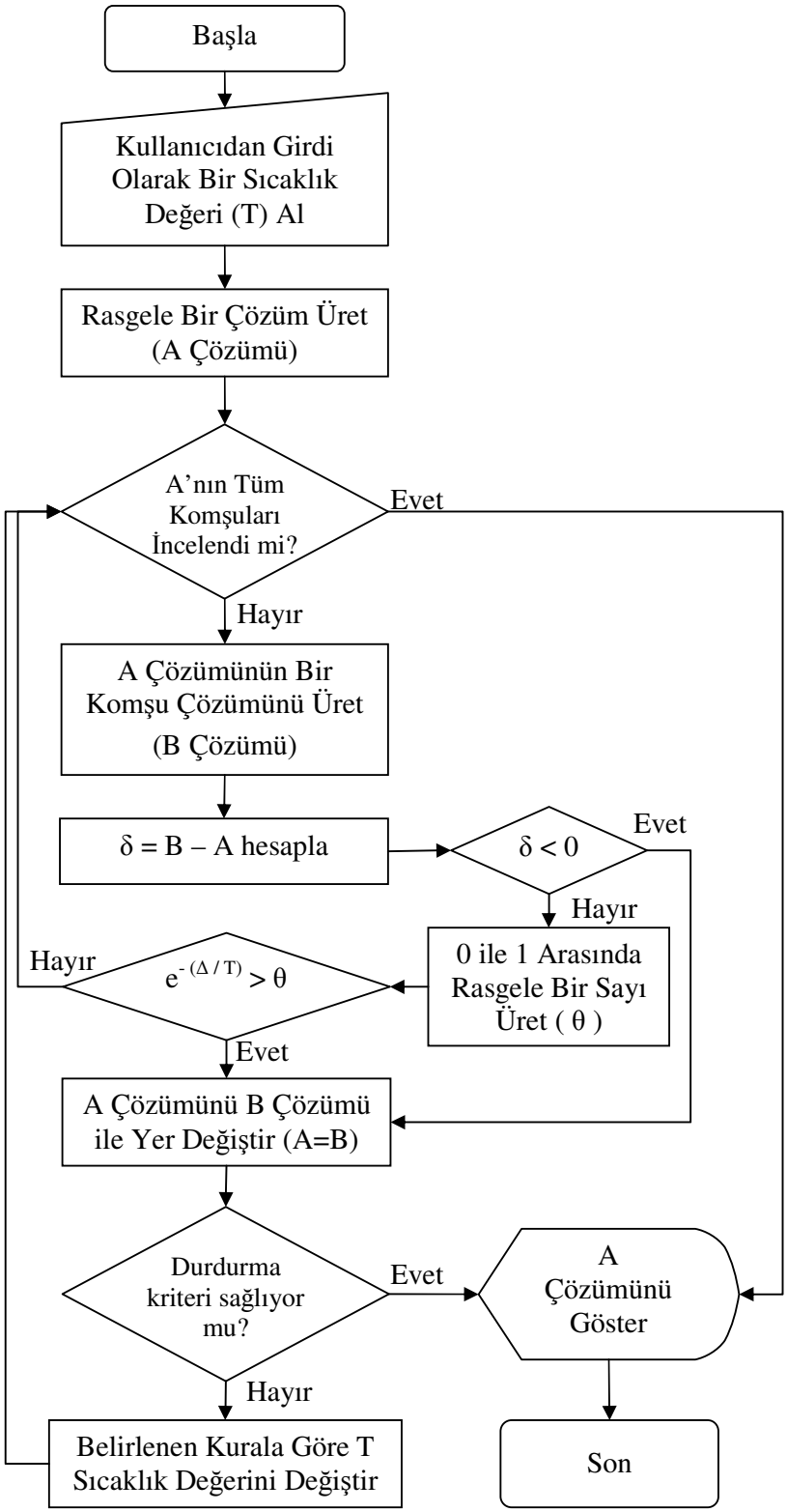
Adım 6. T sıcaklığını 4.2 veya 4.3'deki formüle göre değiştir.

$$T_t = R \,.\, T_{t-1} \quad (0 < R < 1) \tag{4.2}$$

$$T_t = t / (1 + \beta t) \quad (\beta \text{ uygun küçük bir değerdir}) \tag{4.3}$$

Adım 7. Durdurma kriteri sağlanıyorsa araştırmayı durdur, aksi halde iterasyon indeksi t'yi bir artırarak üçüncü adıma git.

TB, bölgesel minimalardan uzaklaşmak için rasgele çıkış hareketleri kullanan modifiye edilmiş bir iteratif iniş algoritmasıdır. Yukarı doğru hareketlerin kabul seviyesi maliyet fonksiyonundaki artışın genliğine ve kontrol parametresi olarak adlandırılan sıcaklık T'ye bağlıdır. Algoritmanın hesaplama zamanı, yukarıda verilen genel adımlara değişik fikirler ilave etmek suretiyle hızlandırılabilir. Örneğin, bunlar amaç fonksiyonundaki değişimin yaklaşık olarak hesaplanması alternatif metotların kullanılması, araştırma yapmak için iyi çözümlerin bulunduğu ümit verici araştırma bölgelerinin tanımlanması ve komşuluk büyüklüğünün belirlenmesi gibi olabilir (Karaboğa, 2004). Şekil 4.1'de standard bir TB algoritmasının akış şeması gösterilmektedir.



Şekil 4.1 Tavlama Benzetimi akış şeması

TB algoritmasının gerçekleştirilmesinde aşağıdaki adımların yerine getirilmesi gerekir:

- Başlangıç çözümünün üretilmesi için bir metodun bulunması.
- Komşu üretme mekanizmasının tanımlanması.
- Komşuluğun nasıl araştırılacağının tanımlanması.
- Soğutma tarifesinin tanımlanması.

Sıcaklık parametresinin azaltılması çözüme ulaşılmada önemli bir parametredir. TB yönteminin performansı önemli derecede seçilen soğutma stratejisine bağlıdır ve bunun gerçekleştirilmesi için aşağıdaki parametrelerin tanımlanması gerekir:

- Kontrol parametresi T'nin başlangıç değeri olan T_B için bir değer atanması.
- Sıcaklık azaltılmasını sağlayan fonksiyonun tanımlanması.
- Her sıcaklık kademesinde icra edilecek olan iterasyon sayısının belirlenmesi.
- Araştırmanın sonlandırılması için bir kriterin tanımlanması.

4.2 Tabu Arama Yöntemi

Literatürde en fazla kullanılan metasezgisel yöntem olan Tabu Arama (TA) yöntemi, Glover tarafından 1989'da geliştirilmiştir. Bu algoritmada arama sırasında lokal optimum noktalara takılmaktan kurtulmak için daha önce arama sırasında elde edilen bilgiler kullanılmaktadır. Kısa dönemli hafıza denilen bu yapıda arama sırasında henüz ziyaret edilen çözümler bir tabu listesinde tutulmakta ve algoritma süresince bu çözümlerin tekrar ele alınması engellenmektedir. TA'da TB'den farklı olarak bir iterasyonda bir çözümün tabu listesinde yer almayan tüm komşu çözümleri taranmaktadır ve en iyi komşu çözüm seçilip başlangıçtaki çözüm tabu listesine atılarak diğer iterasyona geçilmektedir. Tabu listesine çözüm ekleme ve çıkarma işlemleri ise ilk giren son çıkar (FIFO) prensibi ile yapılmaktadır. Bu sebeple TA yöntemi literatürde "dinamik komşuluk arama tekniği" olarak da telaffuz edilmektedir. TA algoritması herhangi bir durdurma kriteri sağlanınca veya bir S çözümünün tüm komşu çözümleri N(S), tabu listesinde yer alıyorsa durdurulmaktadır (Blum ve Roli, 2003).

TA algoritması bölgesel optimali aşmak amacıyla kullandığı temel prensip, değerlendirme fonksiyonu denilen bir fonksiyon tarafından her iterasyonda en yüksek değerlendirme değerine sahip hareketin bir sonraki çözümü oluşturmak amacıyla seçilmesine dayanmaktadır.

TA'nın altında yatan temel fikir araştırma işlemini kontrol etmek amacıyla esnek bir hafıza yapısının kullanılmasıdır. Böyle bir hafıza yapısının kullanım etkisi şu şekilde açıklanabilir: Araştırma boyunca karşılaşılan durumların seçici bir kaydı (H) tutulur ve $N(x^{mevcut})$, $N(H,x^{mevcut})$ ile tanımlanan modifiye edilmiş bir komşuyla yer değiştirir. Bundan dolayı H, mevcut bir çözümden hareketle hangi çözümlere erişilebileceğini ve $N(H, x^{mevcut})$ komşuluk setinden $x^{birsonraki}$ çözümün seçimini belirler. TA'nın temel işlem basamakları aşağıda verilmektedir (Karaboğa, 2004):

Adım 1. Bir başlangıç çözümü (S) al. Başlangıçta değer atanması gereken parametreler için (tabu listesi uzunluğu, durdurma kriteri, vs) değerlerini ata.

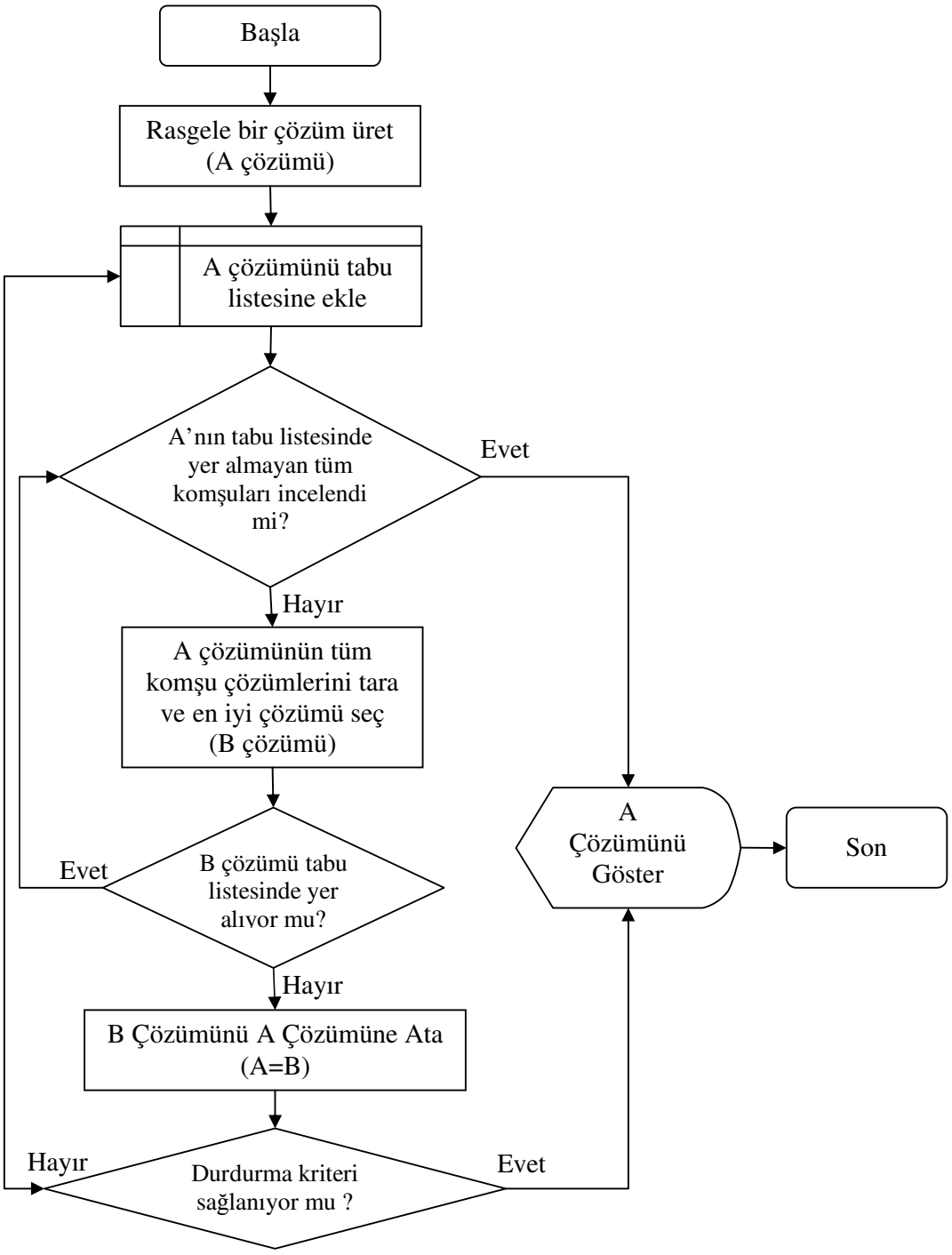
Adım 2. Belirlenen bir komşuluk yapısı ile S'ye ait komşu çözümler üret ve bu çözümler arasından tabu listesinde olmayan tüm $S^1 \in N(S)$ en iyi kabul edilebilir olan (S_{eniyi}) seç.

Adım 3. Mevcut çözümü (S), S_{eniyi} ile yer değiştir ve tabu listesini yenile

Adım 4. Durdurma kriteri sağlanıncaya kadar Adım 2 ve 3'ü tekrar et.

Bu araştırma işlemine göre en iyi kabul edilebilir hareket, mevcut çözümün komşuluğunda bulunan amaç değeri ve tabu sınırlamaları açısından en yüksek değerlendirme değerine sahip olan harekettir. Değerlendirme fonksiyonu, amaç fonksiyonunda en az kötüleşme veya en çok iyileşme sağlayan hareketi seçer. Tabu listesinde kabul edilen hareketlerin karakteristikleri depolanmakta ve bu karakteristikler sonraki iterasyonlarda tabu olacak belirli hareketleri sınıflandırmak için kullanılmaktadır. Kötüleşme sağlayan hareketlerin kabul edilebilir olması nedeniyle daha önceden denenmiş olan diğer bir deyişle ziyaret edilmiş çözümlere tekrar dönmek mümkün olabilmekte ve böylece bir döngü oluşabilmektedir. Tabu listesini kullanmanın amacı böyle bir olayım oluşmasını önlemektir. Bu yüzden görevi sadece tabu listesini kontrol etmek ve yenilemek olan bir yasaklama stratejisini kullanarak araştırmayı sınırlamak zorunludur. Yasaklama stratejisinin amacı, önceden takip edilmiş araştırma yollarının tekrarlanmamasını sağlamak ve böylece yeni bölgelerin araştırılmasını mümkün kılmaktır. Şekil 4.2'de standart bir TA algoritmasının akış şeması gösterilmektedir.

TA, lokal optimalliğin tıkanıklığından kaçınmak ve diğer yöntemlere yol göstermek için geliştirilmiş, optimuma yakın çözüm veren sezgisel bir arama metodudur. Tabu araştırmaları hedefe, kısıt ve tabu kullanarak belirli bir olasılıkla ulaşır. Hedefe ulaşma olasılığının yükselmesi için kısıt ve tabu sayılarını artırmak gerekir (Eren, 2002).



Şekil 4.2 Standart Tabu Arama algoritması akış şeması

4.2.1 Tabu Araştırma Stratejileri

TA algoritması, katı ve özel bir algoritma olmaktan ziyade genel bir yaklaşımdır. TA algoritması genellikle bir olası başlangıç çözümü ile araştırmaya başlar ve bu çözümü daha iyi çözüme dönüştürmek için ardışık hareketler gerçekleştirir. Araştırma esnasında aşağıdaki

stratejileri kullanmak suretiyle bölgesel optimallikten kurtulmaya ve diğer bölgeleri araştırmaya çalışır:

- Tabu listesine neyin girip neyin girmeyeceğine karar veren yani kontrol eden yasaklama stratejisi
- Tabu listesinden neyin ne zaman çıkacağını kontrol eden serbest bırakma stratejisi.
- Denenecek hareketlerin seçimi için yukarıdaki stratejiler arasında ilişkiyi kontrol eden kısa dönem stratejisi.

Yukarıdaki stratejilere ek olarak TA algoritması, çalışması esnasında bilgileri toplayan ve bu bilgilerin sonraki adımlarda TA'yı yönlendirmek amacıyla kullanılmasını sağlayan öğrenme stratejisine de sahip olabilir. Bu strateji orta ve uzun dönem hafıza fonksiyonlarının kullanımından oluşmaktadır. Netice olarak TA algoritmasının gerçekleştirilmesinde ve herhangi bir probleme uygulamasında aşağıdaki elemanları tanımlamak gerekmektedir (Karaboğa, 2004):

- **Tabu Listesi ve Tabu Listesi Büyüklüğü:** Tabu durumundaki şartlar, hareket özelliklerine dayalıdır. Bundan dolayı bu özelliklerin neler olduğu tayin edilmelidir. Bir hareketin tabu durumunu belirlemek için tabu listesi kurulmalıdır. Tabu listesi için veri yapısı bir matris formunu almakta ve bu veri yapısı otomatik olarak iterasyon sayısı arttıkça yenilenmektedir. Tabu listesinin büyüklüğü deneysel olarak bulunabilir. Şayet tabu listesi büyüklüğü gereğinden küçükse çevrim olayı meydana gelecektir. Bununla birlikte gereğinden büyükse araştırma işlemi küresel optimumum bulunduğu bölgeden uzağa sürülecektir. Optimum tabu listesi büyüklüğü, çevrimi önlemek için yeterli uzunlukta ve sürekli çözüm uzayını araştırmaya müsaade edecek kadar da yeterli kısalıkta olmalıdır. Tabu liste büyüklüklerinin dinamik olarak değişmesini sağlayan stratejilerin mevcut olmasına rağmen, daha basit stratejilerle $|H|$ sabit bir değerde tutularak önemli başarılar elde edilmiştir. $|H|$'nin değeri problemin büyüklüğüne bağlıdır. Bu değer istatistiksel olarak türetilebilir. Tabu listesi, TA işleminin birincil kısa dönem hafıza fonksiyonlarından birini ihtiva eder ve sadece en son $|H|$ hareketlerin veya bunların kısmi özelliklerini kaydedilmesi suretiyle gerçekleştirilir. Liste dolduğunda her yeni hareket, listedeki en eski hareket üzerine yazılır. Tabu listesi çoğunlukla önce giren önce çıkar (FIFO) işlem formunda dairesel bir dizil olarak çalıştırılır. Bu veri yapısı şekil 4.3'de gösterilmektedir. Burada her sütun her bir durumun (k) özelliklerini (a(k)) temsil etmektedir.

$a_1(1)$	$a_1(2)$	...	$a_1(k)$	...	$a_1(\lvert T_s \rvert)$
$a_2(1)$	$a_2(2)$	...	$a_2(k)$	...	$a_2(\lvert T_s \rvert)$

Tabu Listesi Büyüklüğü $\lvert H \rvert$

Şekil 4.3 Tabu Listesinin yapısı

- **Aspirasyon Seviyesi:** Aspirasyon kriterine göre herhangi bir tabu hareket mevcut çözümün amaç fonksiyon değerini, o ana kadar bulunan en iyi çözümün amaç fonksiyon değerinin altına düşürürse tabuluktan çıkartılmaktadır. Aspirasyon kriterleri, şayet bir hareket yeteri kadar iyi performansa sahipse ve çevrimi önlemek için de gerekli yeterliliği sağlıyorsa bu hareketin tabu durumunu ortadan kaldırmak amacıyla kullanılan ölçütlerdir. Tabu sınırlamalar ve aspirasyon kriteri araştırma işleminin sınırlandırılmasında ve araştırmaya kılavuzluk edilmesinde rol oynarlar. Tabu sınırlama ancak şartların veya kriterlerin sağlanmamaları durumunda hareketlerin kabul edilebilir olarak değerlendirilmesine olanak tanır. Bu strateji, aday hareket için seçme stratejisine de aynı zamanda durdurma kriterine de karar vermektedir. Bir hareket tabu değilse veya tabu durumu aspirasyon kriteri vasıtasıyla ortadan kaldırılmışsa kabul edilebilir olarak değerlendirilir. Aspirasyon kriteri, amaç fonksiyonuna C(S) uygulanan bir aspirasyon fonksiyonu A(C(S)) ile kontrol edilir. Aspirasyon fonksiyonları, ya zamandan bağımsız ya da zamana bağımlıdır. İlk durumda mevcut çözüme (S) uygulanan tabu bir hareket, şayet şu ana kadar bulunan ve hafızada tutulan en iyi çözümden daha iyi amaç fonksiyonu değerine sahip çözüm üretirse çevrime sebep olmaz. Yani, $C(S)<A(C(S))$ ise aspirasyon verilmesi gerekmektedir.

- **En İyi Kabul Edilebilir (EK):** Bu strateji tüm koşulları değerlendirir ve S^1 çözümünü elde etmek için bir hareket seçer. S^1, en yüksek değerlendirmeye sahip harekettir ve kabul edilebilirlik şartlarını sağlamaktadır (S^1, tabu olmayan veya tabu ya da aspirasyon kriterini geçmiş bir harekettir). Bu durumda S^1 bir sonraki çözüm olarak kabul edilir. EK stratejisi en iyi gelişimi seçme stratejisinin genelleştirilmiş hali olarak görülebilir.

- **İlk En İyi Kabul Edilebilir (İEK):** Bu strateji ilk gelişim ile en iyi gelişim stratejisini kabul edilebilirlik şartları ile birleştiren bir seçme stratejisidir. İEK, ilk gelişim gösteren hareketi kabul eder. Ancak mevcut çözümü geliştiren hiçbir hareket yoksa o zaman en iyi kabul edilebilir hareket bir sonraki çözüm olarak seçilir.

- **Durdurma kriteri:** Durdurma kriteri, TA işlemini daha önceden belirlenmiş olan toplam iterasyon sayısı kadar işlem yapıldıktan sonra veya bulunan çözümün daha önceden tayin edilen belirli bir hata toleransını sağlamasına göre belirlenmektedir. Örneğin, en iyi çözüm üzerinde belirli bir iterasyon boyunca gelişim kaydedilmemişse algoritma durabilir.

4.3 Genetik Algoritmalar

Genetik Algoritmalar (GA), en iyinin korunumu ve doğal seçilim ilkesinin benzetim yolu ile bilgisayarlara uygulanması ile elde edilen bir arama yöntemidir. Evrimsel programlamanın bir parçası olan GA, Darwin'in evrim teorisinden esinlenerek ortaya çıkmıştır. Evrimsel programlama ise 1960'lı yıllarda Rechenberg'in "Evrim Stratejileri" adlı çalışmasıyla gündeme getirilmiştir. GA'nın bugünkü biçimi ilk olarak Holland tarafından 1975 yılında belirtilmiş ve Goldberg'in 1989'da yazdığı kitap ile kombinatoryal problemlerdeki başarısı ortaya konulmuştur. Standard bir GA'da aday sonuçlar eşit boyutlu vektörler olarak ifade edilir. Başlangıçta bu vektörlerden bir grup rastlantısal olarak seçilerek belirli büyüklükte bir populasyon oluşturulur. Kromozom adı verilen bu vektörler, yeni populasyonlar oluşturarak nesiller boyunca değişikliklere uğrar. Bir kromozomun üzerindeki genler, n boyutlu vektörlerin bir boyutuna karşılık gelmektedir. Her yeni nesilde kromozomların iyiliği ölçülür, başka bir deyişle her vektör (kromozom), amaç fonksiyonuna yerleştirilerek vermiş olduğu sonuç hesaplanır. Bir sonraki nesil oluşturulurken, bazı kromozomlar yeniden üretilir, çaprazlanır ve mutasyona uğratılır. Bu işlemler için özel tip operatörler geliştirilmiştir (Nabiyev, 2003).

GA, doğada canlıların değişen çevreye adapte olma yeteneğinden ilham alınarak geliştirilmiş evrimsel bir programlama tekniğidir. Her iterasyonda (nesilde) seçilen bazı çözümlere, bir sonraki iterasyondaki yeni çözümleri oluşturmak için bazı genetik operatörler uygulanır. Doğal seleksiyon kavramıyla oldukça ilişkili olarak GA'nın yönlendirici özelliği, çözümlerin uygunluk değeri başka bir deyişle amaç fonksiyon değerine göre olasılıksal bir şekilde seçilmesine dayanır. Buna göre diğer çözümlere göre daha iyi amaç fonksiyon değerine sahip çözümlerin seçilme olasılığı daha fazladır (Blum ve Roli, 2003).

GA, doğal işlemleri simule ederek optimum çözüm bulmaya çalışan bir yöntemdir ve problemin tipinden, kodlama tekniğinden ve kullanılan operatörlerden bağımsız, dinamik bir yaklaşımdır. GA'nın en temel özelliği, algoritma süresince çözüm populasyonun giderek daha uniform hale gelmesi, diğer bir deyişle çözümlerin birbirlerine yakınsamasıdır. Yönteme

rasgele oluşturulmuş bir populasyondan başlanıldığı takdirde bir müddet sonra populasyondaki bireylerin birbirine benzediği ve populasyonun ortalama uygunluk değerinin dengede kaldığı görülmektedir (Preux ve Talbi, 1999).

GA, rassal arama tekniklerini kullanarak çözüm bulmaya çalışan parametre kodlama esasına dayanan bir arama tekniğidir. GA doğadaki canlıların geçirdiği evrim sürecini dikkate alır. Amaç doğal sistemlerin uyum sağlama özelliğini dikkate alarak yapay sistemler tasarlamaktır. GA'yı diğer arama yöntemlerinden farklı kılan özellikler aşağıda sunulmuştur (Eren, 2002).

- GA, parametre setlerinin kodları ile ilgilenir, parametrelerin kendileri ile direkt olarak ilgilenmez.
- GA'nın arama alanı, yığının veya populasyonun tamamıdır; tek nokta veya noktalarda (çözüm kümesinin daraltılmış bölgelerinde) arama yapılmaz.
- GA'da amaç fonksiyonu kullanılır, sapma değerleri veya diğer hata faktörleri kullanılmaz.
- GA uygulanmasında kullanılan operatörler, stokastik yöntemlere dayanır, deterministik yöntemler kullanılmaz.

4.3.1 Genetik Algoritmaların Çalışma Prensibi

Genetik Algoritmalar doğadaki evrimsel süreçlerini model olarak kullanan bilgisayara dayalı problem çözme teknikleridir. Geleneksel programlama teknikleriyle çözülmesi güç olan, özellikle sınıflandırma ve çok boyutlu optimizasyon problemleri, GA'nın yardımıyla daha kolay ve hızlı olarak çözülebilmektedir. Genel anlamıyla GA bir araştırma konusu olup model haline getirilmiş neden-sonuç işleminin tersine, rasgele örnekleme olgusu altında modellenmiştir. Kontrol edilerek onaylanan kod bilgisi organizma olarak adlandırılan aday çözümler içerisinde saklanmıştır. Organizmalar ise populasyon olarak adlandırılıp grup halinde yer almışlardır. Genetik Algoritmalar ile ilgili temel kavramlar aşağıda kısaca açıklanmaktadır (Eren, 2002):

Gen: Kalıtsal molekülde bulunan ve organizmanın karakterlerinin tayininde rol oynayan kalıtsal birimlere denir. Yapay sistemlerde gen, kendi başına anlamlı bilgi taşıyan en küçük birim olarak alınır.

Kromozom: Birden fazla genin belirli bir düzenle bir araya gelerek oluşturduğu diziye denir. Kromozomlar alternatif aday çözümleri gösterir.

Plan – Katar: Belirli pozisyonlarda, uygun olan alt dizileri (kromozomları) tanımlamak için kullanılan sayı katarlarına veya planlarına denir.

Populasyon: Kromozomlardan oluşan topluluğa denir. Populasyon, geçerli alternatif çözüm kümesidir. Populasyondaki (kromozom) birey sayısı genelde sabit tutulur. GA'da populasyondaki birey sayısı ile ilgili genel bir kural yoktur. Populasyondaki kromozom sayısı artıkça çözüme ulaşma süresi (iterasyon sayısı) azalır.

Standart bir GA yordamı aşağıdaki gibi verilebilir (Nabiyev, 2003):

1. Olası çözümlerin kodlandığı bir çözüm grubu oluşturulur (Çözüm grubu biyolojideki benzerliği nedeniyle populasyon, çözümlerin kodları da kromozom olarak adlandırılır). Populasyonda bulunacak birey sayısı için bir standart yoktur. Genel olarak önerilen 100-300 aralığında bir büyüklüktür. Büyüklük seçiminde, yapılan işlemlerin karmaşıklığı ve aramanın derinliği önemlidir. Populasyon bu işlemden sonra rasgele veya klasik sezgisel yöntemlerle oluşturulur. Birey sayısı belirlendikten sonra probleme bağlı olarak kromozomların kodlanması gerekmektedir.

2. Populasyondaki her kromozomun ne kadar iyi olduğu bulunur. Bu amaçla kullanılan fonksiyona uygunluk fonksiyonu denir. Bu fonksiyon yardımı ile kromozomların uygunluklarının bulunmasına ise evrimleşme adı verilir. Uygunluk fonksiyonu GA'nın beynini oluşturmaktadır. GA'da probleme özel çalışan tek kısım bu fonksiyondur. Uygunluk fonksiyonu, kromozomları problemin parametreleri haline getirerek onların bir bakıma şifresini çözmekte ve sonra bu parametrelere göre hesaplamayı yaparak kromozomların uygunluğunu bulmaktadır. Çoğu zaman GA'nın başarısı bu fonksiyonun verimli ve hassas olmasına bağlıdır.

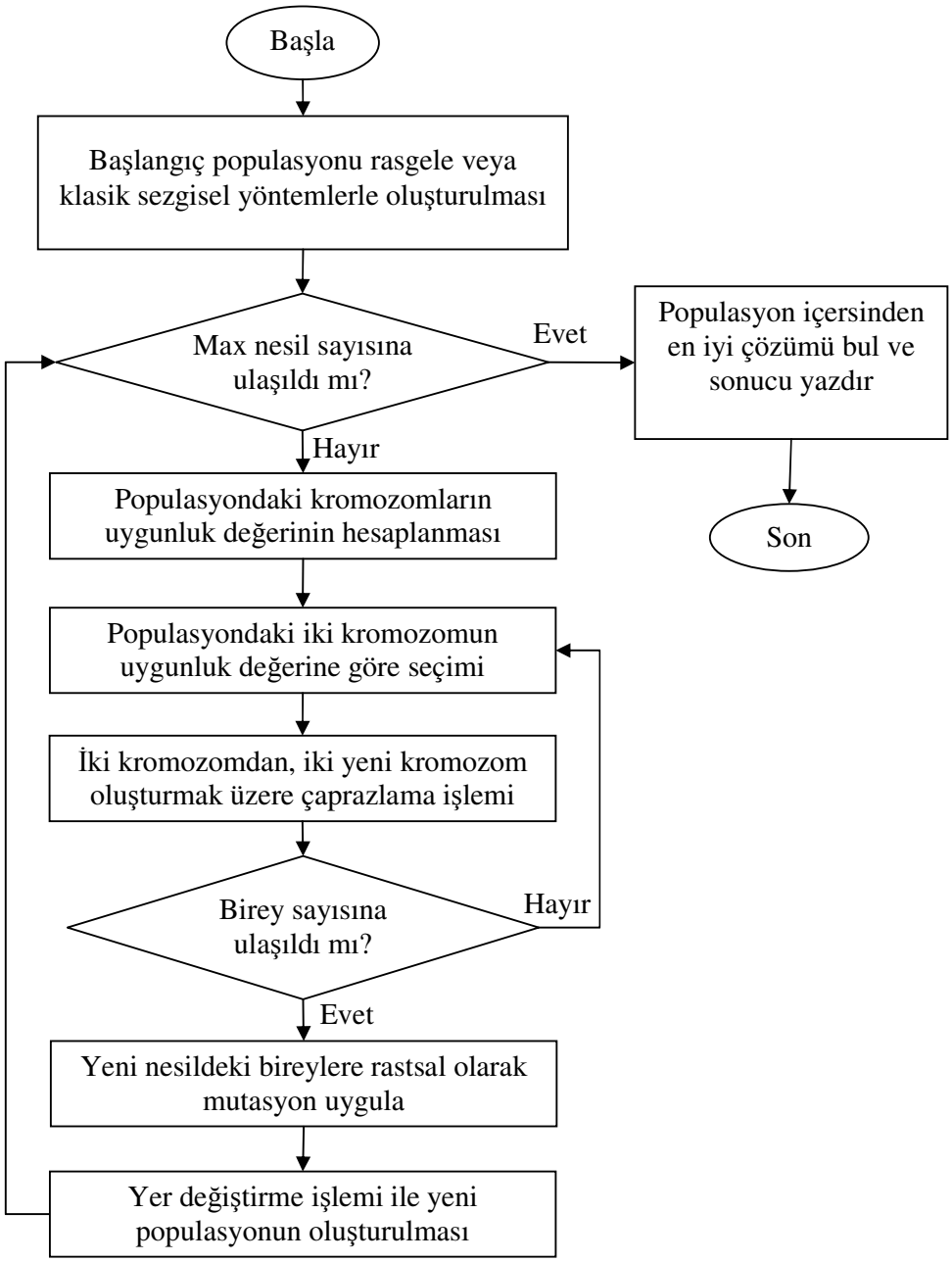
3. Populasyonda bulunan çözümler uygunluk değerlerine göre olasılıksal olarak seçilip eşlenerek çaprazlama ve mutasyon operatörleri uygulanır. Sonuçta yeni bir toplum oluşturulur. Bu seçimi yapmak için rulet tekerleği seçimi, turnuva seçimi gibi seçme yöntemleri vardır. Yeniden kopyalama genlerdeki genetik bilginin birinden diğerine geçme işlemine benzediği için çaprazlama olarak adlandırılır. Bu işlem toplumda çeşitliliği sağlar. Mutasyon ise sadece bir çözüm üzerinde uygulanarak lokal optimum noktalardan kurtulmak için yapılmaktadır.

4. Yeni kromozomlara yer açmak için eski kromozomlar çıkartılarak sabit büyüklükte bir toplum sağlanır.

5. Tüm kromozomların uygunlukları tekrar hesaplanır ve yeni toplumun başarısı bulunur.
6. İşlemler tekrarlanarak verilen nesil sayısı süresince daha iyi olan yeni nesillerin oluşturulması gerçekleştirilir.
7. Sonuçta toplumların hesaplanması sırasında en iyi bireyler bulunduğunda çözüm elde edilmiş olur.

GA, çözüm grupları ile çalışarak, daha optimal, yapılabilir sonuçlara ulaşılmada gerekli olan araştırmalara liderlik eder. Kromozom, genellikle problemdeki değişkenlerin belirli bir düzende sıralanmasıdır. Rasgele oluşturulan başlangıç topluluğu bir kez oluşturulduktan sonra seçim, çaprazlama ve mutasyondan oluşan evrim süreci başlar. Şekil 4.4'de bu süreç gösterilmektedir. GA'nın problem çözmedeki başarısı, pek çok ampirik deneyle gösterilmiştir. Yöntemin çalıştığını ispatlamak için birçok çalışma yapılmıştır. Bu çalışmaların büyük bir bölümü özel durumları içerir. GA uygulamalarının çeşitliliği ve probleme özel kullanılan yöntemler bu incelemeyi daha da zorlaştırmaktadır. GA'nın en uygun olduğu problemler geleneksel yöntemler ile çözümü mümkün olamayan ya da çözüm süresi problemin büyüklüğü ile üstel orantılı olarak artanlardır.

GA, populasyon tabanlı algoritma olduğundan özellikle gerçek hayatta çözüm üretilmesinin gerekli olduğu problem uygulamalarında önemli bir dezavantaja sahiptir. Bir algoritmanın belirli süre içerisinde çözüm geliştirme işlemini tamamlamış olması gerekir. Bu işlemi tamamlamak için populasyon tabanlı algoritmalar, lokal arama algoritmaları ile karşılaştırıldığında nispeten daha uzun süreye ihtiyaç duyarlar. Süreyi kısaltmak için az populasyonla çalışabilen ve iyi performansa sahip GA'nın geliştirilmesi önemli hale gelmiştir. Bunun yanında GA'nın bölgesel yakınsama hızının çok iyi olmaması bir diğer dezavantajıdır. Bu eksikliğini kapatmak amacıyla bazı araştırmacılar standart GA ile klasik komşuluk arama algoritmaları birleştirmişlerdir. Literatürde bu tür GA 'ya Melez Genetik Algoritma ismi verilmektedir (Karaboğa, 2004). GA'nın çözüm kalitesini artırıp hesaplama süresini düşürmek amacıyla literatürde eşzamanlı paralel olarak çalışan melez GA yöntemlerine sıkça rastlanmaktadır (Preux ve Talbi, 1999).



Şekil 4.4 Genetik Algoritma akış süreci

4.4 Karınca Kolonisi Optimizasyon Algoritması

Doğadaki bazı sosyal sistemler, sınırlı yetenekli basit bireyler tarafından oluşturulmalarına rağmen kolektif zeka davranışı sergilerler. Problemlere üretilen zeki çözümler, bu bireylerin kendi içersindeki organizasyonları ve dolaylı iletişimlerinden ortaya çıkar. Karıncalar tek başlarına basit yeteneklere sahip olmalarına rağmen, kolonin bütünü yüksek bir yapıdadır; kendilerinden çok büyük cisimleri taşımak, köprüler oluşturmak veya yuva ile yiyecek

arasındaki en kısa yolu bulmak için çözüm üretirler. Zeki davranış doğal olarak karıncalar arasındaki organizasyon ve dolaylı iletişim sonucunda ortaya çıkar (Nabiyev, 2003).

Karınca Kolonisi Optimizasyon (KKO) algoritması Dorigo tarafından 1996 yılında karıncaların yiyecek bulma mekanizmalarından ilham alınarak geliştirilen bir metasezgisel yöntemdir. KKO tekniği genel olarak bir parametrelendirilmiş olasılıksal model olan feramon modeline dayanır. Çoğu uygulamada, karıncalar uygun bir çözüm oluşturmak için kullanılsa da, olasılıksal modelin bir sonucu olarak uygun olmayan çözümler de üretilebilmektedir (Blum ve Roli, 2003).

KKO, karınca kolonilerinin yiyecek toplama prensibine göre çalışır. Doğadaki karıncalar kör olduklarından, koloniler halinde yiyecek toplamadaki en kısa yolu seçme mekanizmalarına göre algoritma oluşturulur. Her bir karınca geçiş olasılıklarına bağlı olarak hareket edeceği yönü belirler. Karınca i. düğümden j.düğüme hareket ettiğinde, j. düğüm karıncanın hafızasında tabu listesine kaydedilir. Bir sonraki adımda karınca tabu listesinde bulunan bu yöne doğru hareket etmez (Eren, 2002).

Yapay karıncalar ile doğal karıncalar arasındaki benzerlikler aşağıda belirtilmektedir (Dorigo vd., 1999):

- Yapay karıncalar, gerçek karıncalar gibi belirli bir hedefe yönelik işbirliği içinde çalışan bir koloni oluşturmaktadır. Bu işbirliği sayesinde yüksek kaliteli çözümler üretilebilmektedir.

- Yapay karıncalar geçtikleri yollara, hem kendi yönlerini bulmada hem de arkadan gelen karıncalar için kılavuzluk yapması açısından feramon maddesi bırakırlar. Feramon maddesi bir bakıma koloni içinde iletişimi sağlamaktadır ve aynı gerçek karıncalar için olduğu gibi zamanla buharlaşır.

- Gerçek ve yapay karıncaların tek ortak amacı, yuvaları ve yiyecekleri arasında en kısa başka bir deyişle en az maliyetli olan yolları bulmaktır. Gerçek karıncalar rotalarını engeller arasından geçerek, yapay karıncalar ise düğüm noktalarını seçerek belirlerler.

- Yapay karıncalar, gerçek karıncalar gibi gidecekleri yolları, yollarda bulunan feramon miktarına göre olasılıksal olarak seçerek çözüm oluştururlar. Bu sebeple seçim tamamen belirli bir zamandaki lokal verilere dayanmaktadır.

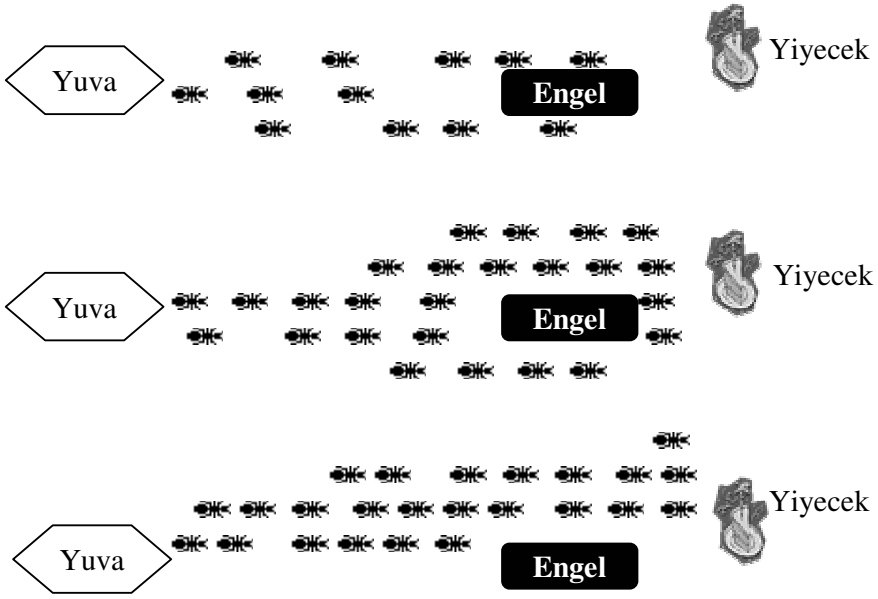
KKO algoritması gerçek karınca kolonisi davranışının matematiksel modelleri üzerine dayalı bir algoritmadır. Bu davranışların tam olarak modellenmesi yerine, yapay karınca

kolonilerinin bir optimizasyon aracı olarak değerlendirilmesinden dolayı önerilen algoritmalar gerçek karınca davranışından biraz farklı yapıdadır (Karaboğa, 2004). Yapay karıncalar ile doğal karıncalar arasındaki farklar aşağıda açıklanmaktadır:

- Yapay karıncalar, ayrık bir dünyada yaşarlar ve onların hareketleri durumdan duruma değişim gösterir.
- Yapay karıncaların, daha önce yapılan hareket bilgilerinin tutulduğu bir iç duruma (hafızaya) sahiptir.
- Yapay karıncalar, bulunan çözümün kalitesi ile orantılı miktarda feramon salgılarlar.
- Yapay karıncaların feramon salgılaması, genellikle problem için bir çözüm üretildikten sonra yapılmaktadır ve bu da gerçek karıncaların davranışı ile farklılık göstermektedir.
- KKO algoritmasının performansını artırmak için algoritmaya ek olarak görünürlük derecesi, lokal optimizasyon, geri besleme gibi özellikler ilave edilebilmekte ve çoğu KKO uygulaması lokal arama teknikleri ile melezleştirilmektedir. Oysa bu özellikler gerçek karıncalarda bulunmamaktadır.

Karıncalar yuvaları ile yiyecek arasındaki en kısa yolu şu şekilde bulmaktadırlar. Her karınca yiyecek ararken geçtiği yerlere (rotasına) değişik miktar ve yoğunluklarda karın bölgesinde yer alan feramon adlı özel bir sıvı bırakır. Bu koku karıncaya yuvasına dönmesini sağladığı gibi diğer karıncalara da yiyeceğe giden yolu göstermeye yarayan bir izdir. Karıncalar ancak feramon kokularını takip ederek doğru yolu bulabilmektedir. Ancak hiçbir karınca tek başına yiyecekle yuva arasındaki en kısa yolu bulabilme özelliğine sahip olmamaktadır. Bu süreci başarabilecek kavramaları veya değişik yolları değerlendirebilecek hafızaları yoktur. Ancak koloni davranışı, kolektif zeka sonucunda bu optimum çözümü düzenleyebilmektedir. Şekil 4.5'de yuvalarından çıkan karıncaların önlerine çıkan bir engele karşı en kısa yolu nasıl buldukları gösterilmektedir. Karıncalar ilk olarak engelle karşılaştıklarında rasgele olarak eşit ihtimallerle yollarını seçerler. Bu durumda karıncaların yarısının sağa yarısının sola döneceğini bekleyebiliriz. Alt taraftaki yolun daha uzun olmasından ve feramonun uçuculuğundan dolayı daha fazla karınca aynı sürede üst yolu izlemiş olacaktır. Bundan dolayı bu üst koldaki feramon yoğunluğu daha da artacaktır. Belirli bir süre sonra bu kollar arasındaki fark daha da artacak ve sisteme yeni karıncalar dahil olmaya başlayacaktır. Sisteme dahil olan karıncalar feramon yoğunluğunun daha fazla olduğu üst yolu tercih edeceklerdir. Böylece karıncalar yiyeceğe giden en kısa yolu bulmuş olacaklardır. Her karıncanın ortalama

aynı hızda ve aynı miktarda feramon bıraktığı düşünülürse, başlangıçta karıncaların en kısa yolu seçmeleri daha uzun zaman gerektirecektir. Fakat sonradan feramon yoğunluğunun artmasıyla yiyeceğe götüren süre (problemin çözümü) kısalacaktır (Nabiyev, 2003).

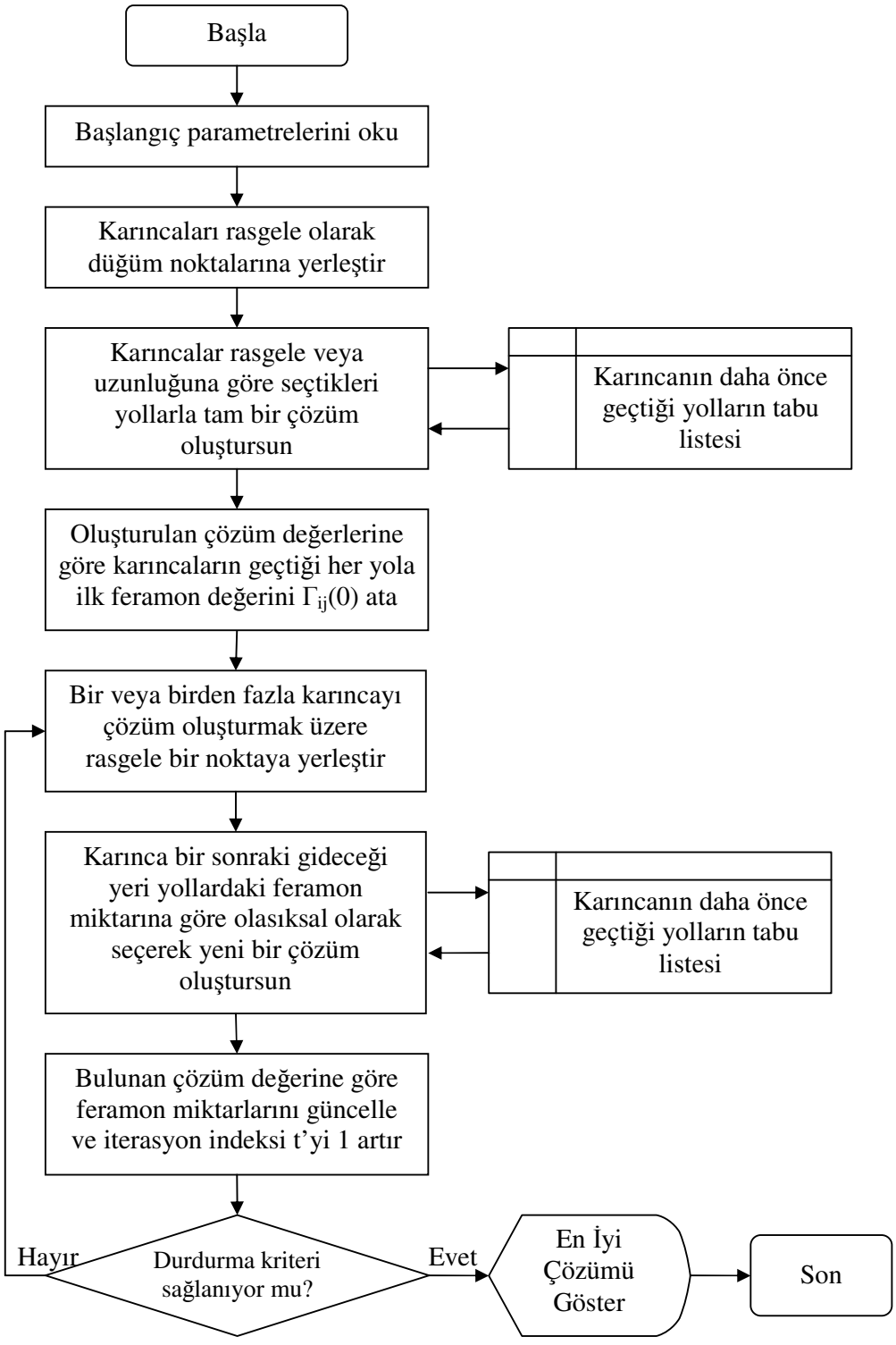


Şekil 4.5 Karıncaların en kısa yolu bulma yöntemi

4.4.1 Karınca Kolonisi Algoritmasının Gezgin Satıcı Problemine Uygulanması

Gezgin Satıcı Problemi (GSP), n tane şehir verildiğinde yapay bir gezgin satıcının her bir şehri bir kez ziyaret etmek şartıyla minimum uzunluklu kapalı bir turu oluşturma problemi olarak tanımlanabilir (Karaboğa, 2004). GSP, KKO'nun ilk olarak uygulandığı kombinatoryal optimizasyon problemidir. GSP'yi tam ağırlıklı bir G(V, U) grafında gösterebiliriz. V şehirleri temsil eden noktaları, U bu noktaları birbirine bağlayan kenarları göstermektedir. KKO algoritmasında karıncalar, GSP'deki graf yapısında şehirden şehre hareket ederek turlar yapan basit birer ajandırlar. Karıncaların çözümü oluşturan feramon izleri ve öncelikli sezgisel bilgiler tarafından yönlendirilmektedir. KKO algoritması GSP'ye uygulanırken, her kenara algoritmanın çalışması esnasında sürekli güncellenen ve feramon miktarını ifade eden sayısal bir $\tau_{ij}(t)$ değeri atanır (Burada t, iterasyon sayısını ifade etmektedir). Başlangıçta m ile temsil edilen her karınca rasgele seçilen şehirlere yerleştirilir ve yenilemeli olarak her şehre aktarmalı bir geçiş kuralı uygular. Karınca şehir i'de iken henüz ziyaret edilmemiş olan şehir j'yi, bu iki şehir arasındaki yol uzunluğunun bir fonksiyonu olan feramon izinin miktarı (τ_{ij}) ve sezgisel bilgiye bağlı olarak seçer. Makul bir sonuca ulaşabilmek için her karınca tabu

listesi adı verilen küçük çapta bir belleğe sahiptir. Karıncalar bu belleği her adımda henüz ziyaret edilmemiş olan şehirleri saptamak ve akıllıca bir sonuca ulaşmak için kullanırlar (Nabiyev, 2003). GSP için standart bir KKO akış süreci şekil 4.6'da gösterilmektedir.



Şekil 4.6 Karınca Kolonisi Optimizasyon algoritması akış şeması

Bütün karıncalar bir tur gerçekleştirdikten sonra feramonlar güncellenir. Bu iş öncelikle feramon izlerinin güçlülük derecelerinin sabit bir faktöre bağlı olarak azalmasıyla başlar. Ardından karıncalar ziyaret ettikleri kenarlar üzerinde feramon depolar. Feramon yoğunluğu daha kısa turların gerçekleştiği ve karıncalar tarafından sık ziyaret edilen kısımlarda fazlalaşır. Bunun sonucunda bu yolların seçilme olasılığı artar. Bu anlayışla feramon yoğunluğu $\tau_{ij}(t)$ karınca i şehrinde iken bir sonraki şehir olarak j'yi seçebilme yeteneği sunar. Bu olasılık aşağıda yer alan 4.4 ve 4.5 nolu formüldeki gibi hesaplanmaktadır:

$$a_{ij}(t) = \frac{\left[\tau_{ij}(t)\right]^{\alpha}\left[1/d_{ij}\right]^{\beta}}{\sum_{l \in N_i^k}\left[\tau_{ij}(t)\right]^{\alpha}\left[1/d_{ij}\right]^{\beta}} \tag{4.4}$$

$$p_{ij}^{k}(t) = \frac{a_{ij}(t)}{\sum_{l \in N_i^k} a_{il}(t)} \tag{4.5}$$

4.4 nolu formüldeki α ve β parametreleri uygun olarak feramon kokusunun ilişkili etkisini ve sezgisel bilgisini ifade eder. α = 0 ise karıncanın bulunduğu düğüme en yakın olan şehir, yolun seçimi için en uygundur. β = 0 ise yalnızca feramon yükselişi vardır ve benzer turların oluşturulması sayesinde hızlı bir şekilde durağan durumlar ortaya çıkar. Formüldeki N_i^k ifadesi k. karıncanın henüz ziyaret etmediği olası şehirleri göstermektedir.

Genelde GSP için geliştirilen KKO algoritmaları, feramon izlerinin ve bazı parametrelerin başlatılmasından sonra ana döngü içerisinde çözüm oluşturucularının belirli bir sayısı veya verilmiş bir işlemci zaman limiti olabilecek son bulma durumuna kadar tekrarlanır. Öncelikle karıncalar mümkün dolaşımlar oluşturur, daha sonra geliştirilen dolaşımlar lokal arama kullanılarak güçlendirilir. Her karınca bir sonuç ürettikten sonra feramon izlerinin güncellenmesi aşağıdaki 4.6 nolu denkleme bağlı olarak yapılmaktadır (Nabiyev, 2003):

$$\tau_{ij}(t+1) = (1-\rho).\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t) \tag{4.6}$$

Burada $0 < \rho \leq 1$ olup, feramon izinin buharlaşma oranını göstermektedir. ρ parametresi feramon birikimini sınırlayarak anlamsız çözümlerden kurtulmak için kullanılmaktadır. m toplam karınca sayısıdır. $\Delta\tau_{ij}^{k}(t)$ parametresi ise, k. karıncanın ziyaret ettiği kenara bıraktığı feramon miktarıdır ve aşağıdaki 4.7 nolu formüle göre hesaplanmaktadır:

$$\Delta\tau_{ij}^{k}(t)=\begin{cases}1/L^{k}(t), & \text{(i, j) 'den geçilirse,}\\ 0, & \text{aksi takdirde}\end{cases} \tag{4.7}$$

Burada $L^k(t)$, k. karıncanın gerçekleştirdiği tur uzunluğudur. KKO algoritmasında hiçbir merkezi karar sistemi bulunmamaktadır. Bilgi ajanlar arasında ve çevrede dağıtılır. Ajanlardan biri kazayla yok edildiğinde, sistem çalışmakta ve sonuç değişmemektedir. Sistemin uyumu bozulmamaktadır. Benzetim sırasında grafın bir kenarı silindiğinde veya yeni bir kenar eklendiğinde sistem hızla çevreye uyum gösterir ve yeni yol ortaya çıkar.

KKO algoritması literatürde sıklıkla aşağıdaki kombinatoryal optimizasyon problemlerine uygulanmıştır (Dorigo vd., 1999):

- Gezgin Satıcı Problemi
- İş Çizelgeleme Problemi
- Tesis Yeri Atama Problemi
- Araç Rotalama Problemi
- Harita Renklendirme Problemi

4.5 Metasezgisel Yöntemlerin Kombinatoryal Problemlere Uygulanması

Metasezgisel yöntemlerin bulduğu çözümlerin kalitesi ve hesaplama süresi, yöntemlerde bulunan temel kavramların probleme uygulanış biçimlerine bağlıdır. Bir metasezgisel algoritmayı bir probleme başarılı bir şekilde adapte edebilmek için bazı prensiplerin izlenmesi gerekmektedir. Lokal komşuluk tabanlı metasezgisel yöntemlerin bir kombinatoryal probleme uygulanmasında aşağıdaki faktörler göz önüne alınmalıdır (Hertz ve Widmer, 2003):

- Hazırlanan algoritmada başlangıç çözümünden komşuluk yapısı ile, kolay bir şekilde yeni çözümler üretilebilmelidir.
- Problem için çözümler arasında uygun bir komşuluk yapısı belirlenerek, bir çözümün komşu çözümlerine eş yakınlıkta ulaşılabilmelidir.
- Eğer çözüm uzayında tepeler ve çukurlar çok fazla bulunmayıp, çözüm uzayının topolojisi düz bir şekilde ise amaç fonksiyonu f, çözüm uzayının yapısını bozmadan tekrar düzenlenmeli ve çözüm uzayı topolojisi büyütülmelidir.

- Bazı durumlarda kısıtlarda yapılabilecek ihlallere göz yumularak, çözüm uzayında lokal noktalara takılmayıp daha geniş bir alanda arama sağlanmalıdır.

Populasyon tabanlı metasezgisel algoritmalarda ise modelleme önemli bir husus olup, yöntemi bir kombinatoryal optimizasyon problemine başarılı bir şekilde uygulamak için aşağıdaki prensiplere dikkat edilmelidir:

- Problem için uygun bir teknik seçilerek iki çözüm ile yeni çözümler oluşturulurken ilgili temel bilgiler yeni çözümlere aktarılmalıdır.
- İki çözüm birleştirilerek oluşturulan yeni çözüm, ebeveyn çözümlerden tamamen farklı olmamalı, ortak özellikler içermelidir.
- Populasyon içersinde çözümler arama uzayının farklı bölgelerinde olmalı, diğer bir deyişle populasyonda farklılık sağlanmalıdır. Ayrıca başlangıç populasyonu buna göre oluşturmalı ve populasyonda çözümlerin yer değiştirme işlemlerinde bu durum göz önüne alınmalıdır.

5. ARAÇ ROTALAMA PROBLEMİ İÇİN METASEZGİSEL YÖNTEMLER

Bu bölümde son 15 yılda geliştirilen ve kapasite ve mesafe kısıtlı ARP için en etkili sonuç üreten yöntemler olan metasezgisel algoritmalar işlenmektedir. Metasezgisel algoritmalar genel olarak, sezgisel metotların çeşitli tipteki problemlere uygulanmasını sağlamak için kullanılan bir grup yaklaşımdan oluşmaktadır. Diğer bir deyişle, bu yöntemler birçok optimizasyon problemine uygulanabilen kapsamlı bir algoritma çatısı olarak görülebilir. Bir metasezgisel yöntemi bir kombinatoryal problem üzerinde deneyebilmek için algoritma üzerinde çok az bir düzenlemeye gereksinim vardır (Blum ve Roli, 2003).

Literatürde bölgesel optimizasyon yöntemleri olarak da geçen klasik sezgisel yöntemlerde çözüm uzayında arama, belirlenen komşuluk yapısı ile daha iyi bir komşu çözüm bulunamadığı durumda sonlandırılmaktadır. Bu sebeple bu yöntemler lokal minimum noktalara takılmakta ve arama stratejisi kör bir şekilde uygulanmaktadır. Metasezgisel yöntemler ise lokal minimum noktalardan kurtulmak için daha kötü çözümlerin de kabul edildiği global optimizasyon yöntemleridir. Metasezgisel yöntemlerde arama, çözüm uzayının en umut verici noktalarında yapılmakta ve süreç, algoritma içine gömülü bir arama stratejisiyle yönetilmektedir. Bu yöntemlerin en büyük dezavantajı ise durdurma kriterinin doğal olarak algoritma içinde bulunmayışı, başka bir deyişle algoritmanın ne zaman duracağını bilmemesidir (Breedam, 2001).

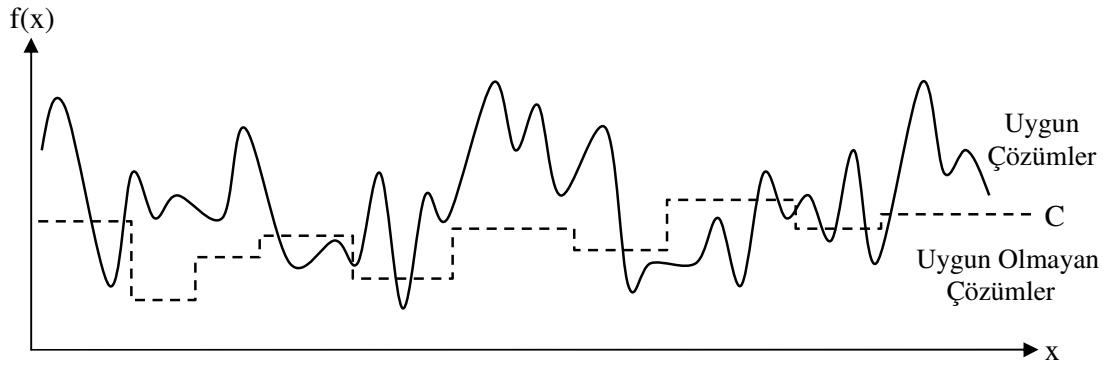
Metasezgisel yöntemlerde klasik sezgisellere göre daha kesin çözüm bulunmasına ve yeni kısıtların algoritmaya kolay bir şekilde entegre edilebilmesine rağmen algoritmaların yazılım diline kodlanması karmaşık ve çözüm süreleri oldukça uzun olmaktadır. Metasezgisel algoritmalar çözüm uzayında arama sırasında kılavuz olarak aşağıda kullandıkları stratejiler ile optimum çözüme yakın sonuçlar üretmektedir (Tarantilis vd., 2004a):

- **Çeşitlendirme Stratejisi:** Metasezgisel yöntemler genellikle iyi bir başlangıç çözümü ile algoritmaya başladıkları ve algoritma süresince belirli bir koşul koyup kötü komşu çözümleri eledikleri için çözüm uzayında iyi sonuç vermeyen çözüm bölgelerini atlamaktadır. Metasezgisel algoritmalar bu sayede kötü çözümler ile süre harcamaz. Özellikle populasyon tabanlı algoritmalar, iyi çözüm bölgeleri içinde yer alan çözümler ile ilgilenmek durumundadır.

- **Yoğunlaştırma Stratejisi:** Bu strateji belirli bir çözüm bölgesinde derin araştırma yapılarak o bölgede optimum noktanın bulunması için komşu çözümlerin taranması sürecini kapsamaktadır. Lokal komşuluk arama algoritmalarının temelinde yatan strateji

bu olup, populasyon tabanlı algoritmalarda ise genellikle saf olarak bu strateji kullanılmamakta, diğer yöntemlerle melez hale getirilerek algoritmaya dahil edilmektedir.

ARP için çözüm uzayının topolojisinde en büyük etkiyi kısıtlar yapmaktadır. Örneğin KKARP içinde yer alan araç kapasite değeri C, uygun çözümlerin yer aldığı arama uzayını Şekil 5.1'de de gösterildiği gibi daraltmaktadır. Eğer C değeri artırılırsa noktalı çizgiyle gösterilen kesme çizgisi aşağıya kayarak uygun çözümlerin yer aldığı çözüm uzayı büyüyecektir. Aksi takdirde, kapasite kısıtı düşük tutulursa belli bir noktaya kadar kesme çizgisi yukarıya çıkacak, bir eşik değerinden sonra problem için uygun bir çözüm bulunmayacaktır.



Şekil 5.1 Kapasite kısıtının çözüm uzayını daraltması

Literatürde son 15 yıldır ARP için lokal arama veya populasyon arama tabanlı metasezgisel algoritmalara sıkça rastlanmaktadır. Klasik sezgisellerden farklı olarak metasezgisel yöntemlerde, iterasyonlar sırasında daha kötü sonuç veren veya uygun olmayan çözümlerin ele alınmasına izin verilmektedir. ARP için en etkili çözümleri üreten yöntemler metasezgisel algoritmalar olup, ayrıca bu teknikler gerçek hayatta kullanılabilecek esnekliğe sahiptirler (Cordeau vd. 2002).

5.1 Araç Rotalama Problemi İçin Geliştirilen Tavlama Benzetimi Algoritmaları

KKARP için geliştirilen Tavlama Benzetimi (TB) algoritmalarında ilk olarak rasgele veya sezgisel yöntemlerle bir x_1 başlangıç çözümü üretilmektedir. Bu yöntemde t. iterasyondaki x_t çözümünün komşu çözümlerinin kümesinden ($N(x_t)$) rasgele olarak bir x çözümü ($x \in N(x_t)$) seçilir. Bir sonraki iterasyona bu çözümün geçirilip geçirilmeyeceği 5.1 nolu formüle göre belirlenmektedir:

$$x_{t+1} = \begin{cases} x & f(x) \le f(x_t) \text{ ise,} \\ x & p_t \text{ ihtimalle,} \\ x_t & 1 - p_t \text{ ihtimalle} \end{cases} \quad (5.1)$$

5.1 nolu formülde yer alan f(x), x çözümünün amaç fonksiyonu değerini gösterirken, p_t olasılık değeri ise aşağıda yer alan 5.2 nolu formüle göre hesaplanmaktadır:

$$p_t = \exp(-[f(x) - f(x_t)] / \theta_t) \quad (5.2)$$

5.2 nolu formülde θ_t, t iterasyonundaki sıcaklık değerini ifade etmekte olup her adımda α ile ($0<\alpha<1$) çarpılarak değeri bir miktar düşürülmekte başka bir deyişle soğutma işlemine tabi tutulmaktadır. Böylelikle zamanla mevcut çözümden kötü çözümlerin kabul edilme olasılığı düşürülmüş olur.

KKARP için geliştirilen Tavlama Benzetimi yöntemlerinde genel olarak 3 tür durdurma kriteri uygulanmaktadır:

- Eğer aynı çözüm için k iterasyon boyunca başka bir çözüme geçilememişse arama süreci durdurulur.
- Algoritma boyunca her iterasyon sonunda geliştirilen çözüm sayısı kontrol edilir. Eğer "Geliştirilen Çözüm Sayısı / İterasyon Sayısı" belirli bir oranın altına düşmüşse sürece son verilerek, bulunan en iyi çözüm kullanıcıya sunulur.
- Belirli bir iterasyon sayısı belirlenerek algoritma durdurulur.

ARP için Robusté, Daganzo ve Souleyrette tarafından 1990 yılında geliştirilen ilk yöntemde komşuluk yapısı olarak çeşitli teknikler denenmiştir. Bunlar bir rotanın belirli bir kısmını ters çevirip aynı rotaya eklemek, bir rotada bir parçayı alıp başka bir pozisyona yerleştirmek ve iki rota arasında noktaları yer değiştirmektir. Alfa, Heragu ve Chen tarafından 1991 yılında geliştirilen diğer bir yöntemde ise başlangıç çözümü İlk Rotala Sonra Grupla sezgisel yöntemi ile oluşturmakta ve 3-opt prosedürü ile çözümler iyileştirilmeye çalışılmaktadır. Bu algoritmanın çözüm kalitesi diğer Tavlama Benzetimi tekniklerine göre daha düşüktür (Toth ve Vigo, 2002).

5.1.1 Osman Tavlama Benzetimi Algoritması

KKARP için geliştirilen en popüler TB algoritması 1993 yılında Osman tarafından geliştirilmiştir. Bu yöntemde komşuluk yöntemi olarak λ-yer değiştirme tekniği

kullanılmaktadır. Bu teknikte rasgele olarak seçilen p ve q rotalarından $|S_p| \leq \lambda$ ve $|S_q| \leq \lambda$ olması koşuluyla yine rasgele bir şekilde S_p ve S_q müşteri kümeleri oluşturulur. İki küme arasındaki noktalar uygun bir şekilde yer değiştirilerek çözüm iyileştirilmeye çalışılmaktadır. Eğer kümelerden biri boş küme olursa, bir rotadan diğer rotaya nokta aktarımı yapılır. İki rotadan oluşturulabilecek küme kombinasyonlarının sayısı çok fazla olduğundan, bu teknik genel olarak $\lambda = 1$ veya $\lambda = 2$ olduğu durumlarda uygulanmaktadır. Arama süreci mevcut çözümden daha iyi bir çözüm elde edilinceye kadar devam eder (Tarantilis vd., 2004a).

Osman TB algoritması iki fazdan oluşmaktadır. İlk faza İniş algoritması adı verilmekte, asıl TB süreci ikinci fazda yer almaktadır. Bu yöntemin adımları aşağıda açıklanmaktadır:

Faz 1: İniş algoritması

Adım 1: Clarke ve Wright Tasarruf algoritmasını kullanarak bir başlangıç çözümü elde edilir.

Adım 2: λ-yer değişimi yöntemi ile mevcut çözümden daha iyi bir çözüm bulunur. Adım 2 çözümde iyileştirme sağlanmayıncaya dek devam ettirilir. Eğer tüm komşu çözümler incelendiği halde iyileştirme mümkün değilse algoritma durdurulur.

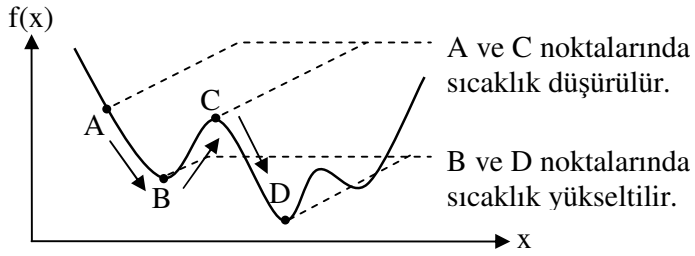
Faz 2: Tavlama Benzetimi süreci

Adım 1: Başlangıç çözümü için klasik sezgisel yöntemlerle çözüm üretilir veya faz 1'in sonunda üretilen çözüm başlangıç çözümü olarak alınır. λ-yer değişimi tekniği ile x_1 başlangıç çözümünün tüm komşu çözümlerini gözden geçirilir ve bulunan en yüksek amaç fonksiyonu fark değerini Δ_{max}'a, en düşük fark değerini Δ_{min}'e ve arama sırasında bulunan uygun çözümlerin sayısını β değişkenine atanır. Başlangıç sıcaklık değeri $\theta_1 = \Delta_{max}$ olarak ve kullanılacak diğer değişkenleri $\delta = 0$, $k = 1$, $k_3 = 3$, $t = 1$, $x^* = x_1$ olarak belirlenir (Burada x^* değişkeni bulunan en iyi çözümü, δ değişkeni x_t çözümünün lokal minimum nokta olup olmadığı bilgisini ve k değişkeni algoritma süresince bulunan lokal çözümlerin sayısını tutar. k_3 değişkeni ise kaç tane lokal minimum noktadan sonra algoritmanın sonlandırılacağını belirler).

Adım 2: Mevcut çözüm x_t'nin komşu çözümlerini λ-yer değişimi yöntemi ile taranır ve eğer $f(x) < f(x_t)$ koşulunu sağlayan bir x çözümü ile karşılaşılırsa, $x_{t+1} = x$ olur. Bu durumda bir kontrol daha yapılır: Bulunan çözüm şimdiye kadarki en iyi çözüm ($f(x) < f(x^*)$) ise $x^* = x$ ve $\theta^* = \theta_k$ atamaları yapılır. Eğer x_t'nin tüm komşu çözümlerinin amaç fonksiyon değeri x_t'den kötü ise komşu çözümlerden en iyisinin sonraki iterasyona geçme durumu 5.1 nolu formüle göre belirlenir. Eğer x çözümü sonraki iterasyona geçemezse $x_{t+1} = x_t$ olur ve $\delta = 1$ alınır.

Adım 3: Eğer $\delta = 1$ ise sıcaklık değeri $\theta_{t+1} = \max\{ \theta_t/2, \theta^* \}$ olarak artırılır ve diğer değişkenler $\delta = 0$ ve $k = k+1$ olacak şekilde ayarlanır. Eğer $\delta = 0$ ise, bir sonraki iterasyonun sıcaklık değeri $\theta_{t+1} = \theta_t / [(n\beta + n\sqrt{t})\Delta_{max}\Delta_{min}]$ yapılarak düşürülür ve $t = t + 1$ olarak atanır. Eğer $k = k_3$ ise algoritma durdurulur, aksi takdirde adım 2'ye geri dönülür.

Genel olarak Osman TB yönteminde yer alan soğutma süreci diğer TB algoritmalarına benzememektedir. Bazı lokal minimum noktalarda sıcaklık artırılarak, algoritmanın bir çözümde takılıp kalması engellenmeye çalışılmaktadır. Şekil 5.2'de bu durum açıklanmaya çalışılmaktadır. Algoritma başlangıç çözümü olan A noktası ile başlatılmakta ve sıcaklık düşürülüp kötü çözümlerin seçilme olasılığı azaltılarak, B lokal minimum noktaya erişilmektedir. Bu noktanın daha iyi bir komşu çözümü olmadığı için, normal TB yöntemlerinde iterasyonun sürekliliği bu noktada sona erebilmektedir. Bu nedenle Osman TB algoritmasında lokal minimum noktalarda sıcaklık artırılarak komşu çözümlerin en iyisinin bir sonraki iterasyona geçme olasılığı yüksek tutulmaktadır. Böylece daha etkili bir arama stratejisi uygulanmış olmaktadır (Tarantilis vd., 2004a).



Şekil 5.2 Osman TB algoritmasında sıcaklıkların belirlenmesi

Diğer TB yöntemleriyle karşılaştırıldığında Osman TB algoritması çoğunlukla iyi çözümler üretir, fakat algoritmanın hesaplama süresi oldukça uzundur (Toth ve Vigo, 2002).

5.1.2 Eşik Onaylayıcı Yöntemler

TB algoritmasının bir modifikasyonu olan Eşik Onaylayıcı yöntemlerde TB'nin kötü çözümleri kabul etmedeki stokastik kuralı devre dışı bırakılmıştır. Bu yöntemde bir eşik değeri T_t tanımlanır ve bir iterasyondaki mevcut x_t çözümünden daha kötü olan x çözümlerin kabul kuralı olarak $f(x_t) - f(x) \leq T_t$ ele alınır. TB'de sıcaklık parametresinde olduğu gibi algoritma süresince T_t parametresi düşürülerek ilerleyen iterasyonlarda kötü sonuçların kabul edilme olasılığı azaltılmaktadır (Tarantilis vd., 2004a).

Tarantilis ve arkadaşlarının 2002 yılında geliştirdikleri Eşik Onaylayıcı yöntemlerde komşuluk yapısı olarak Osman TB algoritmasında olduğu gibi λ-yer değiştirme tekniğini kullanmışlardır. Geri Dönüşlü Adaptif Eşik Onaylayıcı yöntem adını verdikleri ilk metodun süreç adımları aşağıda açıklanmaktadır:

Adım 1: Klasik sezgisel yöntemlerle uygun bir başlangıç çözümü x_1 oluştur ve eşik değerini $T_1 > 0$ olacak şekilde belirle. İterasyon indeksini t = 1, bulunan en iyi çözümü $x^* = x_1$, uygun çözüm indeksini k = 0 ve ilerleme durumunu $\delta = 0$ olarak ata.

Adım 2: Rasgele olarak bir $x \in N(x_t)$ komşu çözümü seç. Eğer $f(x_t) - f(x) \leq T_t$ ise mevcut çözümü $x_t = x$ olarak belirle. Eğer $f(x_t) > f(x)$ ise ilerleme değişkenini $\delta = 1$ olarak değiştir. Eğer bulunan çözüm değeri $f(x) < f(x^*)$ ise $x^* = x$ olarak ata. Çözüm indeksi k'yı ise 1 artır.

Adım 3: Eğer çözüm indeksi k değeri, incelenecek maksimum uygun çözüm sayısına eşit ise iterasyon indeksini t = t + 1 olarak değiştir. Aksi takdirde adım 2'ye ger dön.

Adım 4: Eğer $\delta = 1$ ise T_t eşik değerini $0 < p_d < 1$ koşulunu sağlayan bir parametre ile çarparak ($T_t = p_d\, T_{t-1}$) azalt. Aksi takdirde (lokal bir minimum noktada yer alınıyorsa) T_t eşik değerini $0 < p_i < 1$ koşulunu sağlayan bir parametreyi kullanarak $T_t = T_1 - (T_1 - T_{t-1})(1 - p_i)$ formülü ile artır. Eğer iterasyon indeksi t, maksimum iterasyon sayısına eşitse, algoritmayı durdur. Aksi takdirde çözüm indeksini k = 0 ve ilerleme indeksini $\delta = 0$ olarak değiştir.

Geri Dönüşlü Adaptif Eşik Onaylayıcı yöntemde Osman TB algoritmasında olduğu gibi lokal minimum noktalarda kötü sonuçların kabul edilme olasılığı artırılarak optimuma yakın çözüm bulunmaya çalışılmaktadır. Bu tekniğe yazarlar "geri dönüş" adını vermişlerdir. Bu şekilde eşik değerinin azaltıp artırılması (salınımlar) arama sırasında çeşitlendirme ve yoğunlaştırma arasında bir dengenin kurulmasını sağlamıştır (Tarantilis vd., 2004b). Yazarlar ARP için iyi değerler üreten bu algoritmayı aynı zamanda gerçek hayattaki bir posta taşıma problemine uygulamışlar ve başarılı sonuçlar elde etmişlerdir (Tarantilis vd., 2002).

Tarantilis ve arkadaşlarının 2002 yılında geliştirdikleri diğer bir yöntem de Liste Tabanlı Eşik Onaylayıcı algoritmasıdır. Bu yöntemde tek bir eşik parametresi kullanılmayıp, birden fazla eşik değeri göz önüne alınarak bir listede tutulmaktadır. Listedeki en büyük eşik değeri T_{max} mevcut çözümden kötü çözümlerin kabul edilip edilmeyeceğini ($[f(x_t) - f(x)]/f(x) \leq T_{max}$) formülü ile belirler. Eşik değerleri listesi iterasyon sırasında çözümlerin amaç fonksiyon değerlerindeki değişkenliğe göre algoritma boyunca otomatik olarak belirlenir (Tarantilis vd., 2004a).

5.2 Araç Rotalama Problemi İçin Geliştirilen Tabu Arama Algoritmaları

Tabu Arama (TA) algoritmasında süreç TB yöntemine benzer olup, farklardan en önemlisi iterasyonlarda bir sonraki çözüm olarak en iyi amaç fonksiyonu değerini veren komşu çözümün alınmasıdır. Daha önce kabul edilmiş çözümlerin tekrar kabul edilmesini engellemek amacıyla çözümler veya çözüm uzayındaki hareketler belirli bir iterasyon süresince bir tabu listesinde tutulur (Duncan, 1995). Eğer algoritma süresince en iyi çözüme ulaşılmışsa, tabu hareketlerine izin verilir, bu duruma aspirasyon adı verilmektedir. TA uygulamalarında performansa en fazla etki eden faktör bir çözümden başka bir çözümü üretme tekniği olan etkili bir komşuluk yapısının belirlenmesidir (Rego, 2001).

Bunların yanında, TA algoritmasında çeşitlendirme ve yoğunlaştırma stratejisi "stratejik salınım" adı verilen belirli bir düzene göre sırayla uygulandığında çözüm uzayının daha iyi bir şekilde taranabileceği ortaya çıkmıştır. Bu da belirli zamanlarda sadece uygun çözümlerin (yoğunlaştırma), belirli zamanlarda ise uygun olmayan çözümlerin de kabul edilebilmesi (çeşitlendirme) sayesinde gerçekleşmektedir (Kelly ve Xu, 1998).

5.2.1 Osman Tabu Arama Algoritması

Osman tarafından 1993'de KKARP için geliştirilen TA algoritmasında λ-yer değiştirme tekniği (λ = 2 alınarak) ile problemin komşuluk yapısı incelenmektedir. Bu yapı 2-opt tekniği, farklı rotalar arasında nokta aktarımı veya alışverişinin bir kombinasyonundan meydana gelmektedir. Geliştirilen yöntemde ilerleme süreci iki türlü ele alınmıştır. İlk olarak ele alınan çözümün tüm komşulukları taranıp en iyi kabul edilebilir komşu çözüm seçilerek yeni iterasyona geçilmekte ve diğer ilerleme sürecinde ise tarama sırasında bulunan, mevcut çözümden daha iyi ilk kabul edilebilir komşu çözüm seçilerek algoritmaya devam edilmektedir. Eğer daha iyi bir çözüm bulunamaması halinde en iyi komşu çözüm ele alınmaktadır. Bu algoritmada çözümlerin tabu listesinde kalma süresi sabit olup hiçbir yoğunlaştırma prosedürü uygulanmamıştır. Osman TA algoritması literatürde yer alan CMT problemleri için her iki ilerleme sürecinde de oldukça iyi sonuçlar elde edilmektedir (Toth ve Vigo, 2002).

5.2.2 Tabu Rota Algoritması

Gendreau, Hertz ve Laporte'nin 1994'de geliştirdikleri ve Tabu Rota adını verdikleri TA algoritmasında komşu çözümleri bulmak için bir genel ekleme prosedürü kullanılmıştır. Bu mekanizmaya göre komşu çözüm, bir rotanın bir noktasının kopartılıp başka bir rotaya

eklenmesi ile elde edilmektedir. Bu tekniğin λ-yer değiştirme yönteminden farkı ise ekleme yapılan rotanın mutlaka noktanın p yakınlıktaki en az bir noktasını içermesi gerektiğidir. Ele alınan komşuluk yapısına göre bir rota tamamen ortadan kaldırılabilmekte veya yeni bir rota oluşturulabilmektedir. Ayrıca Tabu Rota algoritması komşu çözümü bulmak için birbirine p dereceden yakın olan nokta kümelerinden de faydalanmaktadır (Laporte vd., 2000). Tabu Rota'nın diğer bir yeniliği ise kapasite veya maksimum gidilebilecek mesafe bakımından uygun olmayan çözümleri de kabul edebilmesidir. Bunun için yazarlar amaç fonksiyonuna Q kapasite aşımını ve D mesafe (süre) aşımını belirten iki ceza terimi eklemişlerdir. Bu iki terim, değerleri algoritma süresince otomatik olarak ayarlanan p ve θ parametreleri ile ağırlandırılmış olup, cezalandırılmış amaç fonksiyonu değeri $f^*(x) = f(x) + pQ(x) + \theta D(x)$ formülü ile hesaplanmaktadır. İlk olarak p ve θ parametreleri 1'e eşitlenerek algoritma başlatılmakta daha sonra algoritma boyunca ardarda bulunan 10 uygun çözüm sonunda 0,5 ile çarpılarak düşürülmekte veya ardarda bulunan 10 uygun olmayan çözüm sonunda ise 2 ile çarpılarak değerleri yükseltilmektedir. Diğer koşullarda ise parametre değerleri sabit kalmaktadır. Bu sayede amaç fonksiyon değeri değiştirilerek algoritmanın lokal minimum noktada takılıp kalması önlenmiş olur (Tarantilis vd., 2004a).

Tabu Rota yöntemi gerçek anlamda bir tabu listesi kullanmaz. Rotalar arası transfer olan noktalar hafızada tutulmakta, buna tabu etiketleri adı verilmektedir. Bu yöntemde tabu etiketleri kullanılarak iterasyonlar sırasında aynı çözümler üreten bir döngüye girmemek için şu şekilde bir kural koyulmuştur: Bir rotadan başka bir rotaya nokta taşındıktan sonra gelen k adet iterasyonda aynı noktanın ilk rotasına tekrar eklenmesi engellenmektedir. Burada k değeri 5 ile 10 arasında hesaplanan rasgele bir sayıdır. Tabu Rota algoritmasında çeşitlendirme stratejisi bir rotadan diğerine taşınan noktanın taşınma frekansı ile orantılı bir terimin amaç fonksiyonuna eklenmesi ile gerçekleştirilmektedir. Böylelikle kritik noktaların rotalar arasında alış veriş yapılma olasılığı dengelenerek çözüm uzayı daha iyi bir şekilde taranmaktadır. Yöntemin yoğunlaştırma stratejisi ise başlangıçta belli sayıda çözüm üretilip bu çözümler üzerinde arama işleminin yapılmasıdır, daha sonra ise bulunan en iyi çözüm üzerinden ana algoritma yürütülmektedir. Tabu Rota Algoritmasının temel adımları aşağıda verilmektedir (Toth ve Vigo, 2002):

Adım 1: İlk olarak $\sqrt{n}/2$ adet başlangıç çözümü oluştur. Her iterasyonda bir rotadan diğer bir rotaya aktarılabilecek müşteri kümesini $W = V\backslash\{v_0\}$ olarak belirle ve k'ya 50 değerini ata. Başlangıç çözümlerini iyileştir (k, çözümde gelişmenin sağlanmadığı iterasyon sayısını göstermektedir, ayrıca bir noktanın başka rotaya atanma olasılığı 0,9 olarak belirlenmiştir).

Adım 2: Adım 1'de elde edilen en iyi çözüm ile Tabu Rota algoritmasına başla. İterasyon sayısını k = 50n olarak ata.

Adım 3: Adım 2'den bulunan sonuç ile 50 iterasyon sayısında W kümesinin birinci ve ikinci adımda en sık taşınan noktalarını göz önüne alarak TA algoritmasını başlat.

Adım 4: Eğer mevcut çözümde herhangi bir gelişmenin sağlanmadığı iterasyon sayısını k'ya eşit ise algoritmayı durdur.

Tabu Rota algoritması ile literatürde yer alan problemlerde yüksek kaliteli sonuçlar bulunmasına rağmen hesaplama süresi uzun olmaktadır. Yöntemin bulduğu çözümlerin, en iyi bilinen çözümlerden sapma oranı 0.86% olarak bulunmuş ve bu metotla CMT problemleri için şimdiye kadar bilinen en iyi 5 çözüm elde edilmiştir. Algoritma karmaşık olmakla beraber değerlerini kullanıcının belirlemesi gerektiği 9 parametre ile etkinliği kısıtlanmaktadır. Buna rağmen yeni kısıtların oldukça kolay bir şekilde probleme dahil edilebilmesi gerçek hayattaki problemlere uygulanması kolaylaştırmaktadır (Cordeau vd., 2002).

5.2.3 Taillard Tabu Arama Algoritması

KKARP için Taillard, 1993 yılında Tabu Rota yöntemine benzer bir TA algoritması geliştirmiştir. Algoritmada komşu çözümler λ-yer değiştirme (λ = 1) tekniği ile bulunmakta ve uygun olmayan çözümler Tabu Rota yönteminin tersine kabul edilmemektedir. Taillard'ın KKARP'nin Tabu Arama ile çözümüne getirdiği en büyük yenilik problemi alt problemlere bölüp paralel hesaplamaya olanak sağlamasıdır. Bu bölme işlemi deponun merkezde bulunup müşterilerin düzgün olarak depo çevresine yayıldığı düzlemsel problemlerde şu şekilde yapılmaktadır: Alt problemler müşterilerin depo merkezli sektörlere ayrılması ile elde edilir ve sınırlamalar dinamik olarak belirlenmektedir. Daha sonra her alt problem paralel hesaplamalarla (farklı işlemci birimleriyle) bağımsız olarak çözülmeye çalışılmakta ve müşteriler yan yana bulunan bir sektörden diğerine periyodik olarak geçebilmektedir. Oluşturulan rotalar içinde yollar çapraz yer değiştirilerek, rota mesafesi kesin bir şekilde minimum yapılmaya çalışılmaktadır. Düzlemsel olmayan problemlerde veya deponun merkezde olmadığı (müşterilerin düzgün dağılmadığı) problemlerde ise bölme işlemi en kısa yollar göz önüne alınarak merkezi depoda başlayan ağaç yapısına göre yapılmaktadır (Toth ve Vigo, 2002).

Bu algoritmada yer alan her iterasyonda genel olarak sırayla aşağıdaki işlemler yapılmaktadır (Tarantilis vd., 2004a):

- Noktaları sektörlere bölme ve alt problemler oluşturma işlemi.
- Her alt problemi standart Tabu Arama algoritması ile çözme.
- TA algoritmasında belirli bir iterasyon sayısından sonra bu alt problemlerde bulunan çözümleri birleştirerek probleme bütün bir çözüm üretme.

Taillard TA algoritmasının Tabu Rota yöntemi ile benzer yönü, tabu etiketleri kullanması ve sürekli olarak çeşitlendirme stratejisini kullanmasıdır. Çeşitlendirme stratejisinde yer alan sık sık tekrarlanan çözüm hareketlerini (rotalar arasında nokta alışverişi) cezalandırmaya ek olarak bu yöntemde, alt problemlerin sınırları periyodik olarak güncellenmektedir. Bu algoritma, literatürde TA yöntemleri arasında en iyi sonuçları veren diğer bir deyişle kesinlik özelliği en yüksek olan yöntemlerden biri olup 14 CMT problemi için 12 en iyi bilinen çözümü üretmiştir. Hesaplama süreleri ise Taillard tarafından belirtilmemiş fakat algoritma Tabu Rota yönteminden daha basit olduğu için daha kısa sürelerde çözüm bulması beklenmektedir. Bu metotta komşuluk yapısında sadece uygun çözümlerin kabul edilmesinden dolayı yeni kısıtlar sürece kolayca entegre edilebilmekte, bu da yöntemin esnekliğini artırmaktadır (Cordeau vd., 2002).

5.2.4 Xu ve Kelly Tabu Arama Algoritmaları

Xu ve Kelly 1996'da geliştirdikleri TA algoritmasında KKARP için daha karmaşık bir komşuluk yapısı kullanmışlardır. Bir çözümün komşu çözümü, iki rota arasında nokta alışverişi, global yerleştirme stratejisi (tüm rotalarda yer alabilen bazı noktaları belirli rotalara yerleştirme) ve her rotanın lokal iyileştirme prosedürleri ile bulunmaktadır. Global yerleştirme stratejisi ile rotalara ekleme – çıkarma maliyetinin hesaplandığı ağ akış modeli çözülüp belirtilen sayıdaki nokta, araç kapasite değeri dikkate alınarak sıra ile farklı rotalara yerleştirilmektedir. Yer Değiştirme Zinciri adı verilen bu teknik λ-yer değiştirme yönteminin bir genellemesi olup ilk olarak ikiden fazla sayıda rota oluşturulmakta, daha sonra ise noktalar periyodik olarak komşu rotalar arasında taşınmaktadır. Belirli sayıdaki noktaları farklı rotalara düzgün bir şekilde yerleştirmek için ise yazarlar bir ağ akış modelini önermişlerdir. Bunun yanında her rota için lokal iyileştirme 3-opt tekniği ile yapılmaktadır (Toth ve Vigo, 2002). Yer Değiştirme Zinciri prosedüründe yer alan rotalara ekleme – çıkarma maliyeti sürekli bir şekilde 5.3 nolu formülde gösterildiği gibi hesaplanmaktadır (Kelly ve Xu, 1998).

$$c = c + p(\max\{0, l - L\} - \max\{0,\ l' - L\}) \tag{5.3}$$

Burada c maliyeti, l ve I' sırası ile prosedürün öncesindeki ile sonrasındaki rotanın uzunluğunu ve L, bir uzunluk parametresini göstermektedir. Formülde yer alan p ise bir ceza parametresi olup $0 < o_1 \leq o_2 \leq o_3$ olmak üzere değeri, 5.4 nolu formülde gösterildiği gibi bulunmaktadır.

$$p = \begin{cases} p_1 & \max\{0, l-L\} < o_1 \\ p_2 & o_1 \leq \max\{0, l-L\} < o_2 \\ p_3 & o_2 \leq \max\{0, l-L\} < o_3 \\ p_4 & \max\{0, l-L\} \geq o_3 \end{cases} \tag{5.4}$$

Xu ve Kelly TA algoritmasında çözümler tabu listesinde sabit sürede tutulmakta olup listeye kapasite kısıtını sağlamayan başka bir deyişle uygun olmayan çözümler de eklenebilmektedir. Algoritmadaki çeşitlendirme politikası ise şu şekilde işlemektedir: Hafızada belirli bir rotaya atanan bir noktanın frekansı tutulur ve frekans değeri taşıma maliyetine eklenerek rotalar arası daha az hareket eden bir noktanın yer değiştirilmesine öncelik tanınır. Geliştirilen algoritmada en iyi sonuç veren uygun çözümler bir havuzda tutulur ve algoritmanın ilerleyen aşamalarında periyodik olarak havuzdan rasgele seçilen bir çözüm tekrar ele alınır. Böylece parametrelere ilk değerleri verilerek yüksek kaliteli çözümler elde edilmeye çalışılır. Bu da algoritmanın yoğunlaştırma stratejisini teşkil etmektedir. Bu yöntem ile bazı CMT problemleri için en iyi bilinen çözüm elde edilse de, çok fazla hesaplamaya gereksinim olması ve parametrelerin dinamik olarak sürekli değiştirilmesi sebebiyle yöntemin diğer TA algoritmaları kadar hızlı ve etkili olduğu söylenemez (Tarantilis vd., 2004a).

Kelly ve Xu'nın 1998'de geliştirdikleri diğer bir yöntemde bir KKARP için sezgisel metotlarla bulunan birçok çözüm ele alınıp parçalanmakta ve daha iyi çözüm üretmek üzere bu parçalar birleştirilmektedir. Yazarlara göre klasik sezgisel metotlarla üretilen optimuma uzak çözümlerde dahi iyi rotalar bulunabilmektedir. Buna dayalı olarak geliştirilen iki fazlı prosedürde ilk olarak basit sezgisel yöntemlerle olabildiğince çok birbirinden farklı çözüm elde edilmekte, daha sonra bu çözümlerde yer alan iyi rotalar belirlenip belli bir kurala göre birleştirilerek iyi bir çözüm elde edilmeye çalışılmaktadır. Bununla ilgili olarak Şekil 5.3'de 10 müşteri ve 3 araçtan oluşan bir örnek verilmektedir. Şekilde klasik sezgiselle üretilen üç çözümdeki iyi rotalar kesikli çizgilerle gösterilmekte ve geliştirilen yöntemde bu rotalar birleştirilerek yeni bir çözüm üretilmektedir. Algoritma süreci aşağıda verilmektedir (Kelly ve Xu, 1998):

Faz 1: Bu bölümde sezgisel yöntemler kullanılarak N >> m olmak üzere N adet rota oluşturulmaktadır (m, araç sayısını göstermektedir). Kullanılan sezgisellerin özellikleri aşağıda verilmektedir:

- Kodlanması kolay, basit ve hızlı çalışabilir olmalıdır.
- Değişik kısıtlarla esnek bir şekilde işleyebilmelidir.
- Değişik parametre değerleri ile farklı sonuçlar üretebilmelidir.
- Belirli bir dereceye kadar uygun olmayan çözümler oluşturabilmelidir.

Yazarlar ilk fazda Clark ve Wright'ın yönteminden daha iyi sonuç veren Paessen'in Tasarruf algoritmasını ve 1996'da geliştirdikleri TA yöntemini kullanmışlardır. Paessen'in Tasarruf metodunda tasarruf $s_{ij} = d_{i0} + d_{0j} - g.d_{ij} + f\left|d_{i0} - d_{0j}\right|$ formülü ile hesaplanmakta, burada parametreler $0 < g \leq 3$ ve $0 \leq f \leq 1$ olmak üzere rasgele belirlenmektedir. Bu fazın işleyiş adımları aşağıda açıklanmaktadır:

Adım 1: Bir sezgisel metotla $K \leq m$ olmak üzere R_1, ..., R_K rotalarından oluşan bir çözüm üret. Üretilen çözümü 3-opt tekniği ile iyileştir.

Adım 2: Eğer R_i rota havuzunda yer almıyorsa onu havuza ekle (i = 1, ..., K). Eğer rota havuzunda N adet rota yer alıyorsa çık, aksi takdirde adım 1'e git.

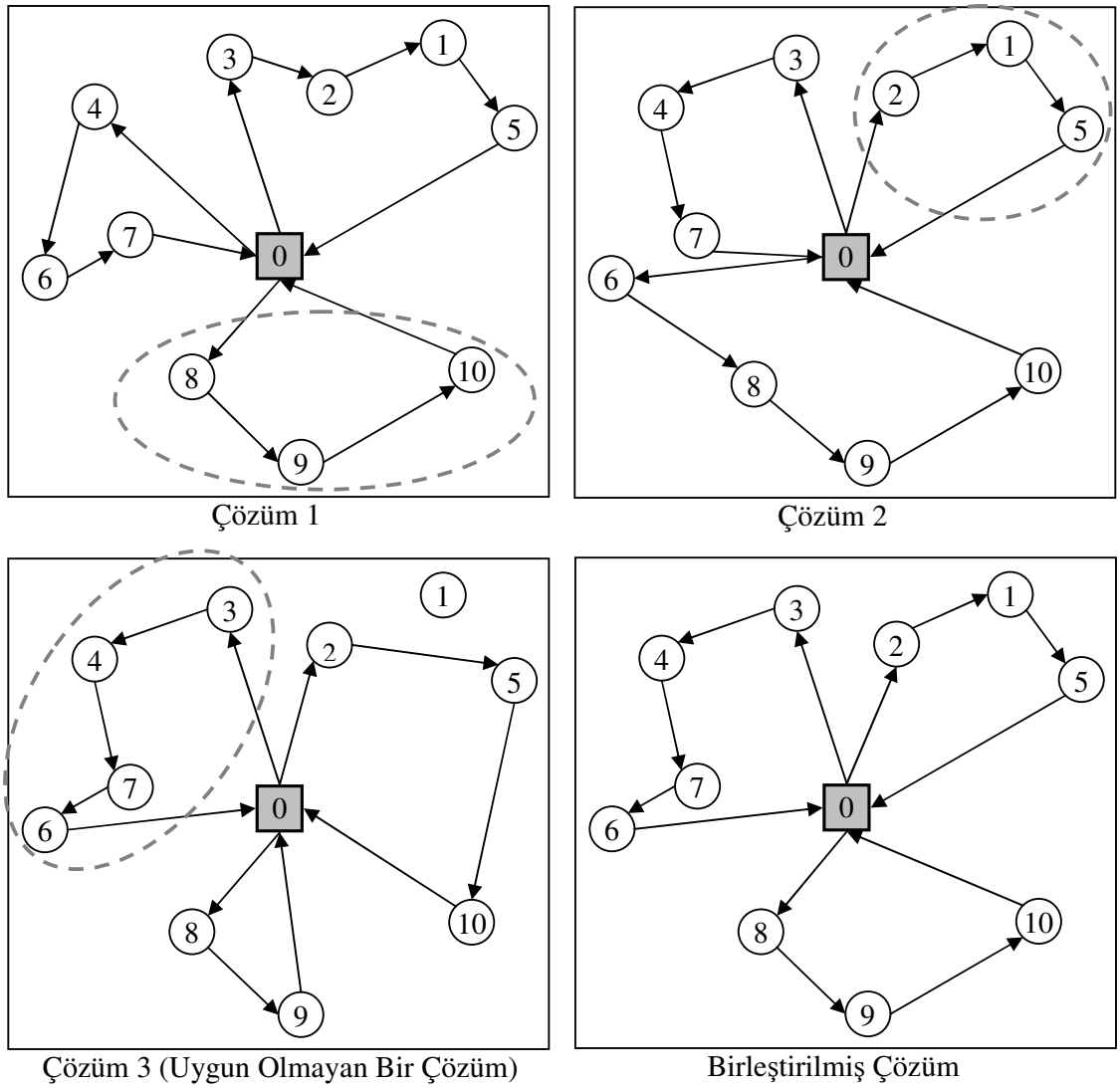
Faz 2: Bu bölümde faz 1'de bulunan rotalardan m tanesi seçilip birleştirilerek uygun bir çözüm oluşturulmaya çalışılmaktadır. Faz 1 tamamlandıktan sonra aşağıdaki tamsayı programlama modeli kurulabilir:

$$\min \sum_{j=1}^{N} c_j x_j \tag{5.5}$$

Şu kısıtlara göre:

$$\sum_{j} a_{ij} x_j = 1, \quad i = 1, \ldots, n \tag{5.6}$$

$$x_j \in \{0,1\}, \quad j = 1, \ldots, N \tag{5.7}$$



Şekil 5.3 Kelly ve Xu Tabu Arama örneği

Modeldeki c_j terimi j. rotanın uzunluğunu ve x_j, j. rotanın çözümde olup olmayacağını belirtmektedir. Eğer i. müşteri j nolu rotada yer alıyor ise a_{ij}'nin değeri 1'e eşit olmaktadır. Buna göre 5.5'de yer alan amaç fonksiyonu rotaların toplam mesafesini minimize etmeye çalışmakta ve 5.6 nolu kısıta göre ise her müşteri çözümde ancak ve ancak bir kez olması gerekmektedir. Eğer araç sayısı sabit ise aşağıdaki 5.8 nolu kısıt da modele eklenebilir:

$$\sum_{j=1}^{N} x_j = m \tag{5.8}$$

Faz 2'de kullanılan TA yöntemi başlangıç çözüm olarak faz 1'de bulunan en iyi çözümü ele alır. Algoritmada komşuluk yapısı ise mevcut çözümden bazı rotaların çıkartılıp havuzdan

yeni rota çekilmesi ile olur. Bu işlem sonunda bazı müşteriler aynı anda iki rotada yer alabilmekte veya rotalanmamış olarak kalabilmektedir. Bu durumda sonuç olarak kapasite kısıtı göz önüne alınarak müşteri toplam uzunluğun en fazla azaldığı rotadan çıkarılacak veya en az arttığı rotaya atanacaktır. Rotalardaki bu değişiklik eski çözümlere dönülmesine sebep olabilmektedir. Buna karşılık bu algoritmada açıkta kalan veya iki rotada birden yer alan noktaların düzenlenmesi tabu listesi göz önüne alınarak yapılmaktadır. Ayrıca yöntemde rota havuzunda yer alan tüm rotaların aynı oranda komşu çözümlerin oluşturulmasında kullanılması sağlanmaktadır (Kelly ve Xu, 1998).

Kelly ve Xu'nın 1998'de geliştirdikleri TA algoritması 14 değişik CMT problemi için denenmiş ve eğer başlangıç rotaları Paessen'in Tasarruf algoritması tarafından oluşturulursa 5, Xu ve Kelly'nin 1996'da hazırladıkları TA yöntemi tarafından oluşturulursa 10 en iyi bilinen çözümü üretmiştir, fakat bu yöntemin hesaplama süresi diğer algoritmalarla karşılaştırıldığında oldukça uzun olup gerçek hayatta dinamik sistemler için elverişli değildir.

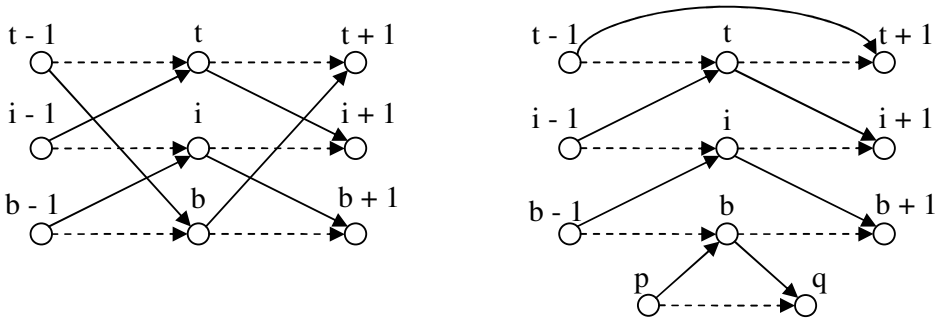
5.2.5 Rego ve Roucairol Tabu Arama Algoritmaları

KKARP'nin TA yöntemi ile çözümü için Rego ve Roucairol'un 1996'da geliştirdikleri algoritmada komşu çözüm bulma işlemi farklı bir yer değiştirme zinciri tekniği ile gerçekleştirilmektedir. Yazarlara göre bir rota zinciri, bir rotada bulunan ardışık üç noktadan oluşan (v_{i-1}, v_i, v_{i+1}) bir gruptur ve L seviyeli bir yer değiştirme zinciri prosedürü ise k. rotada bulunan $(v_{i-1}^k , v_i^k , v_{i+1}^k)$ (k = 0 ,..., L) nokta üçlüsünün $(v_{i-1}^k , v_i^{k-1}, v_{i+1}^k)$ (k = 1 ,..., L) üçlüsü ile son kalan noktaya kadar yer değiştirilmesi ile gerçekleşmektedir. Bu süreç sonunda hiçbir rotaya yerleştirilemeyen v_i^L noktası ise ele alınan birinci rotaya v_{i-1}^1 ve v_{i+1}^1 noktalarının arasına eklenmektedir. Komşu çözümler bulunurken kapasite kısıtı dikkate alınmakta, bu sayede uygun rotalar oluşturulmaktadır. Farklı rotalardan nokta zincirleri bulunurken noktaların birbirine yakınlık durumu göz önüne alınmaktadır (Toth ve Vigo, 2002). Bu algoritmada tabu listesi bir rotadan diğer rotaya atanan zincir bilgilerini tutmakta ve bu bilgilerin listede kalma süresi daha önce belirlenmiş iki parametre arasında rasgele seçilen bir değerle belirlenmektedir. Bunun yanında algoritmada yer alan başka bir hafıza yer değiştirme zinciri sırasında tüm kenarların taşınma frekansını tutmaktadır, böylece aktarmanın çözüm kalitesi üzerideki etkisi hesaplanarak en iyi hareket yapılmaya çalışılmakta ve yoğunlaştırma işlemi daha etkili bir şekilde sürdürülmektedir (Tarantilis vd., 2004a).

Rego'nun 1998'de yaptığı diğer bir çalışma 1996'daki yer aldığı çalışmanın benzeri olan Çiçek algoritmasıdır. Bu yönteme göre bir çözümün komşu çözümü çiçek referans yapısı

olarak adlandırılan alt rota yer değiştirme zincirleri ile üretilmektedir. Çiçek referans yapısı, noktaları birden fazla rotada yer alabilen nokta gruplarının yayılması olarak tanımlanmaktadır. Algoritmaya göre (v_r ,..., v_c) alt rotasına v_r bitkinin kökü, v_c çiçeğin çekirdeği olmak üzere bitkinin gövdesi denmekte ve (v_c ,..., v_i ,..., v_c) ($i \geq 1$) çevrimleri ise çiçek olarak adlandırılmaktadır. Buna göre yer değiştirme prosesi bir rotanın bir kısmı (alt-rota, bitki gövdesi) çıkarma kuralları göz önüne alınarak yeni bir rota oluşturacak şekilde gerçekleştirilir. Çıkarma kuralları noktalar arası kenarların silinmesi ve/veya eklenmesi ile ilgilidir. Bununla birlikte, bu kurallar ile bir çiçeğin bir gövdeye dönüşmesi (bir kenarı silerek) veya bir çiçeğin bir çiçek ve gövdeye dönüşmesine (bir kenarı silip başka bir kenar ekleyerek) mümkün olabilmektedirler. Bu şekilde çiçek yapısında dönüşümler gerçekleşir, fakat kök nokta (v_r) değişse bile çekirdek nokta (v_c) değişmeyecektir. Zincirde yer alan çekirdek nokta da depo olup, algoritma süresince sabittir. Bu yöntemde Tabu listesinde henüz silinmiş olan kenar bulunmakta olup bilgiler listede değişik (rasgele) sürelerle tutulabilmektedir. Algoritmada çeşitlendirme prosedürü her noktanın taşınma frekansı eşitlenmeye çalışılarak yapılmaktadır (Tarantilis vd., 2004a).

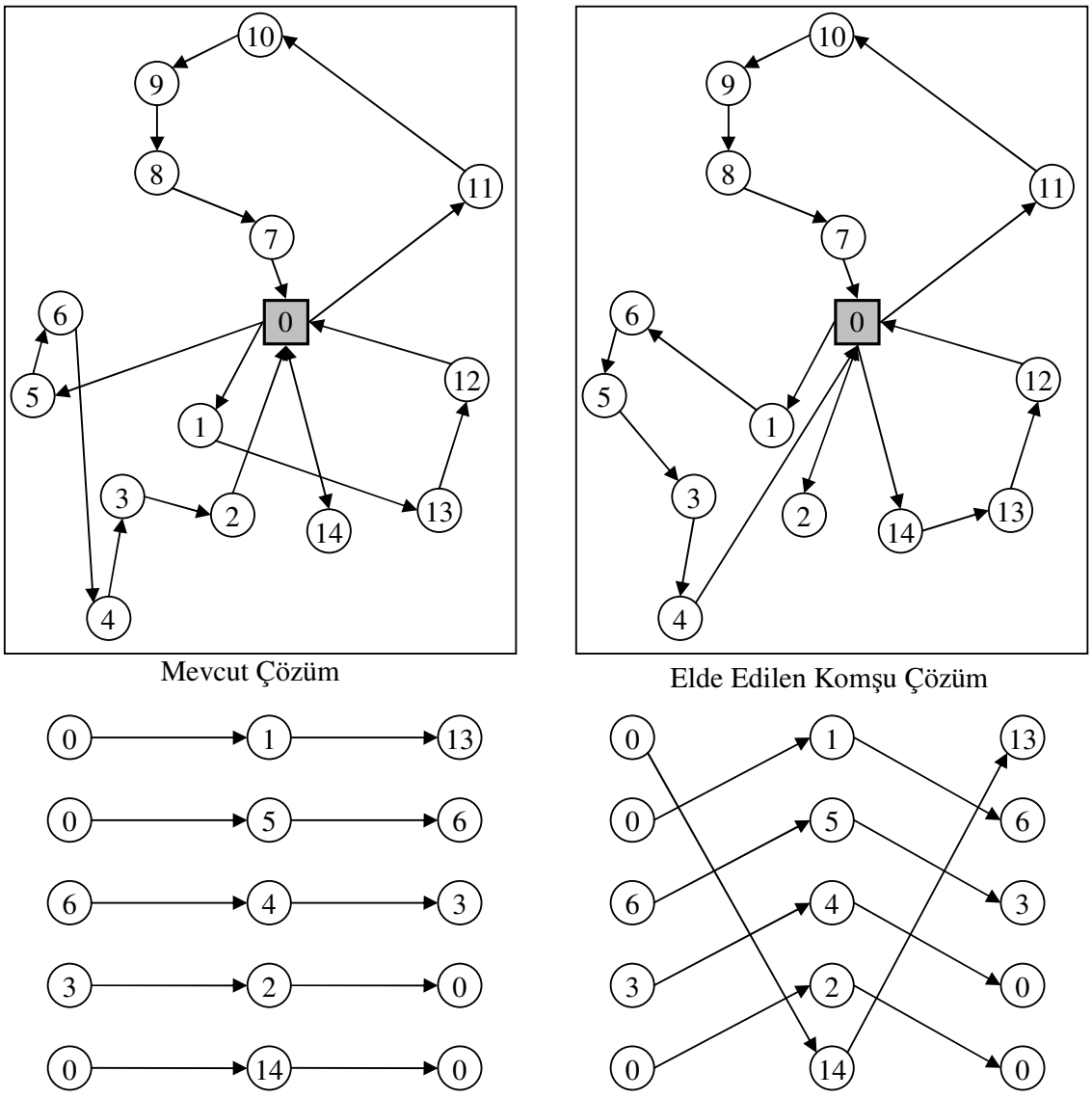
Rego'nun 2001'de yaptığı başka bir çalışmada ise komşuluk yapısı için iki tip yer değiştirme zinciri prosedürü uygulanmıştır. Yazarın gömülü komşuluk yapısı olarak nitelendiği ve Şekil 5.4'de gösterilen bu tekniklerden ilki Rego ve Roucairol'un 1996'da hazırladıkları çok noktalı yer değişim tekniğidir. Şekil 5.4'de kesikli çizgiler eski durumu, sürekli çizgiler ise yeni durumu göstermektedir (Rego, 2001).



Şekil 5.4 Çok Noktalı Yer Değişim ve Ekleme tekniği

Çok noktalı yer değişim yönteminde belirli sayıdaki bir grup rota içerisinden birbirlerine yakınlık durumları göz önüne alınarak ardışık nokta üçlüleri seçilir ve mesafe değerlerine göre sıralanır. Bu üçlü grupların ortalarında yer alan noktalar aşağıdan yukarıya doğru kaydırılarak diğer noktaların arasına yerleştirilmektedir. En altta kalan orta nokta ise en

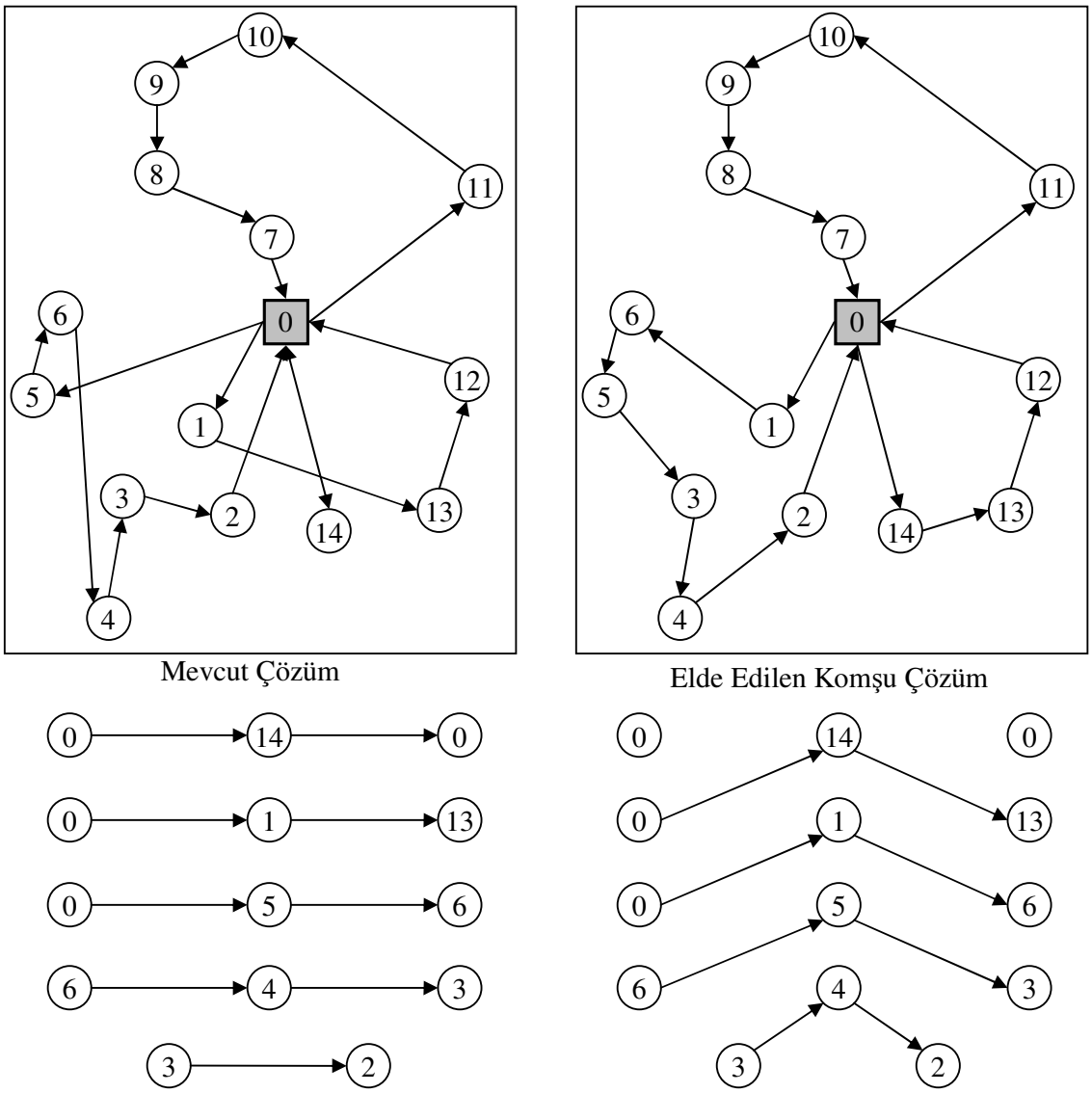
üstteki sıraya atanmakta ve böylece açıkta kalan bir müşteri kalmamaktadır. Şekil 5.5'de yer gösterilen bir örnekte mevcut çözümden daha iyi bir çözüm, 5 rota arasında T = {(0, 1, 13), (0, 5, 6), (6, 4, 3), (3, 2, 0), (0, 14, 0)} nokta üçlüleri ele alınarak ve 1, 5, 4, 2 ve 14 nolu noktaların yer değişimi yapılarak bulunmaktadır.



Şekil 5.5 L = 5 Düzeyli Yer Değiştirme Zinciri örneği

Bu algoritmada kullanılan diğer bir teknik ise şekil 5.4'de de gösterilen Rego'nun 2001'de geliştirdiği çok noktalı ekleme prosedürüdür. Bu prosedürün, çok noktalı yer değişim tekniğinden tek farkı, en altta kalan v_i^L noktasının, en üstteki ilk rotaya atanmayıp başka bir rotadaki v_p ve v_q noktalarının arasına eklenmesidir. Şekil 5.6'da bu metot kullanılarak elde edilen dört düzeyli bir çözüm örneği yer almaktadır. Örnekte 4 nolu müşteri 3 ve 2

noktalarının arasına, 14 nolu müşteri merkez depo ile 14 nolu müşterinin arasına atanmış ve bu atama sonu bir rota mevcut çözümden silinmiştir. Bu yöntemde bazı rotaların silinebileceği gibi v_p ve v_q noktaları depo noktası (0) seçilerek yeni rotalar da oluşturulabilmektedir.



Şekil 5.6 L = 4 düzeyli Çok Noktalı Ekleme Zinciri örneği

Bu yöntemde başlangıç çözümleri Paralel Tasarruf Algoritması ile bulunmakta ve tabu listesine daha önce başka rotaya atanan bir noktanın eski rotasına aktarılmasını önleme amaçlı bilgiler yazılmaktadır. Bu bilgiler tabu listesinde rasgele olarak bulunan bir iterasyon süresince kalmaktadır. Bunun yanında atama yapılan noktaların frekans bilgileri tutularak ve her noktanın atanma oranı eşitlenmeye çalışılarak arama uzayında daha fazla çözüm

taranmaya çalışılmaktadır. Başka bir deyişle çeşitlendirme stratejisi uygulanmaya çalışılmaktadır. Bu algoritmada uygun olmayan çözümler komşu çözüm olarak kabul edilmiş, sıralı ve paralel yürütülen işlemlerle çözüm uzayında arama daha hızlı ve kapsamlı olarak gerçekleşmiştir. Algoritmanın verdiği sonuçlara göre iki işlem de çoğu CMT problemi için en iyi bilinen sonuçları oluşturmuştur. Bunun yanında paralel yürütülen yöntem, sıralı yönteme göre az bir farkla daha iyi sonuçlar üretmiştir (Rego, 2001).

5.2.6 Barbarosoğlu ve Özgür'ün Tabu Arama Algoritması

Barbarosoğlu ve Özgür tarafından 1999 yılında geliştirilen TA algoritması daha önceki yöntemlerle çoğunlukla benzerlik taşısa da, çeşitlendirme stratejisinin uygulanmadığı farklı bir komşuluk yapısına sahiptir. KKARP için geliştirilen metasezgisel yöntemlerde bir çözümün komşu çözümünün bulunması genellikle rotalar arası nokta alış verişi, bir noktanın bir rotadan diğerine yer değiştirmesi, yeni bir rotanın oluşturulması veya varolan bir rotanın silinmesi şeklinde olmaktadır. Bu çalışmada komşuluk yapısı için bu tekniklerin yer aldığı fakat noktaların dağılımı ile ilgisi olmayan ve TANE ve TANEC adını verilen prosedürler kullanılmıştır. TANE prosedürü şu adımlarla işlemektedir (Barbarosoğlu ve Özgür, 1999):

Adım 1: Mevcut çözüm içerisinde rasgele olarak R_1 ve R_2 rotaları seçilir.

Adım 2: $\mu \leq \min(M, \left|R_1\right|)$ ve $\pi \leq \min(P, \left|R_2\right|)$ olmak üzere R_1 ve R_2 rotasından rasgele seçilen μ ve π sayıdaki noktalar iki rota arasında yer değiştirilir. Eğer uygun bir çözüm oluşmazsa, adım 2 tekrarlanır (Burada M ve P seçilen iki rota için yer değişimi yapılabilecek maksimum nokta sayısı ve $\left|R_1\right|$ ise R_1 rotasında yer alan nokta sayısını göstermektedir).

Adım 3: Algoritmada R_1 ve R_2 rotalarına yeni halleri ile 2-opt prosedürü uygulanarak rotalar iyileştirilir.

Adım 4: Adım 1, 2 ve 3 belirli sayıda tekrarlanır ve bu işlem sonunda en iyi amaç fonksiyonu değerini veren çözüm, komşu çözüm olarak ele alınır.

Komşuluk yapısını belirlerken noktaların yakınlık durumlarını ele alan TANEC prosedüründe ise süreç aşağıdaki gibi işlemektedir.

Adım 1: Yukarıdaki TANE süreci belli bir iterasyon süresince devam ettirilir. Mevcut çözüm içerisinde rasgele olarak R_1 ve R_2 rotaları seçilir.

Adım 2: R_2 rotasındaki tüm noktaların merkez noktası C_2 bulunur. R_1 rotası için ise bu rotada yer alan v_i noktasının dışındaki noktaların merkez noktası C_{1i} bulunur. C_{1i} ve v_i arasındaki

mesafe (d_{1i}) ve C_2 ve v_i arasındaki mesafe (d_{2i}) hesaplanarak R_1 rotasındaki noktalar d_{1i}/d_{2i} oranına göre büyükten küçüğe sıralanır.

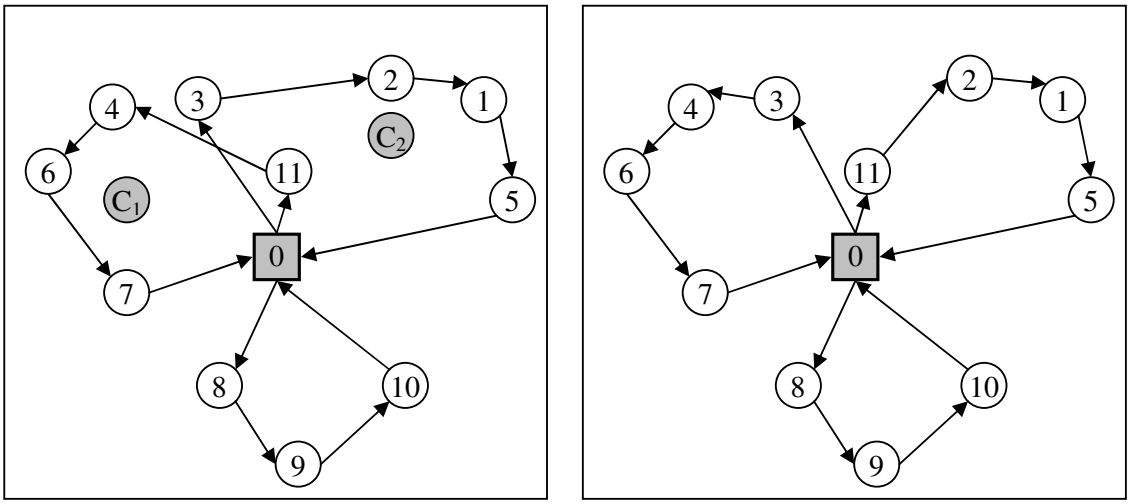
Adım 3: Adım 2, R_2 rotası için de tekrarlanarak rota içinde yer alan noktalar için bir sıralama elde edilir.

Adım 4: $\mu \leq \min(M, \left|R_1\right|)$ ve $\pi \leq \min(P, \left|R_2\right|)$ olmak üzere R_1 ve R_2 rotasından daha önce yapılan sıralamalar dikkate alınarak seçilen μ ve π sayıdaki noktalar iki rota arasında yer değiştirilir. Eğer uygun bir çözüm oluşmazsa, sıralamada daha alt sıradaki noktalar seçilerek adım 4 tekrarlanır (Burada M ve P seçilen iki rota için yer değişimi yapılabilecek maksimum nokta sayısı ve $\left|R_1\right|$ ise R_1 rotasında yer alan nokta sayısını göstermektedir).

Adım 5: R_1 ve R_2 rotalarına yeni halleri ile 2-opt prosedürü uygulanarak rotalar iyileştirilir.

Adım 6: Yukarıdaki adımlar belirli sayıda tekrarlanır ve bu işlem sonunda en iyi amaç fonksiyonu değerini veren çözüm, algoritma sonucu olarak kullanıcıya verilir.

TANEC tekniğinin temelinde, rotalar arasında nokta alış verişi yapılırken rotaların merkez noktasından uzakta bulunan bir noktanın, merkez noktasına daha yakın olduğu bir rotaya atanması fikri yatmaktadır. Şekil 5.7'de bu tekniğin uygulandığı bir örnek yer almaktadır. Örnekte C_1 ve C_2 rotaların merkez noktası olup, birinci rotadaki diğer noktalara göre C_1 noktasına uzak olan 11 nolu nokta ile ikinci noktada yer alan diğer noktalara göre C_2 noktasına uzak 3 nolu noktanın yer değişimi yapılmıştır.



Mevcut Çözüm | Elde Edilen Komşu Çözüm

Şekil 5.7 TANEC çözüm oluşturma prosedürü

Bu algoritmada TANE ve TANEC teknikleri sıra ile uygulanarak çözüm uzayı taranmaktadır. Başlangıç çözümü ise rasgele üretilen ve en kısa yol sezgiseline benzer bir yöntemle geliştirilen bir grup çözüm içinden en iyi çözüm seçilerek bulunmaktadır. Geliştirilen yöntemde tabu listesinde, ataması yapılan noktaların önceki rotasına tekrar atamasını önleme amaçlı bilgiler rasgele belirlenen bir sürede tutulmaktadır. Yoğunlaştırma stratejisi ise lokal minimum noktalara takılmadan önce bulunan çözümler iyi bir bölgede olduğu kabul edilerek bir kümede tutulmakta ve çözümler bu küme üzerinden geliştirilmektedir. CMT problemlerinde genellikle TA yöntemlerinden genellikle daha iyi sonuç vermekte, en iyi bilinen bazı çözümleri bulmaktadır (Barbarosoğlu ve Özgür, 1999).

5.2.7 Adaptif Hafıza Tabanlı Algoritmalar

1995'de Rochat ve Taillard tarafından geliştirilen Adaptif Hafıza (AH) prosedüründe, arama prosesi için otomatik olarak yoğunlaştırıcı ve çeşitlendirici işlemlerini gerçekleştirmektedir. Adaptif hafızada özel bir veri yapısı ile çözümler ve bu çözümlerin en iyi parçaları (özellikleri) dinamik olarak tutulur. Bir sonraki iterasyonda işlenecek çözüm ise, bu parçalardan geçici bir uygun çözüm yaratılması ile elde edilir. Bu geçici çözüm lokal arama metotları ile iyileştirilir ve son olarak iyileştirilen çözüm de AH'ye dahil edilir veya çözümün iyi parçaları hafızadaki parçalarla yer değiştirilir. Parçalardan yeni çözümlerin elde edilmesi Genetik Algoritmalarda çaprazlama operatörünün işlemesine benzemektedir (Genderau vd., 1999). Bu algoritmada çeşitlendirme stratejisi ilk adımlarda parça kombinasyonlarından çözüm yaratılırken gerçekleştirilmektedir. Algoritma ilerledikçe arama prosesi derece derece yoğunlaştırma stratejisine doğru kaymaktadır çünkü adaptif hafızadaki bilgi yapısı dinamik olarak çözüm uzayında belirli bölgelerde toplanıp az sayıda çözüm oluşturacak biçimde eğilim kazanmaktadır (Laporte vd., 2000). Bu algoritmanın en önemli özelliği, yeni çözümü oluşturacak parçaların adaptif hafızada yer alan çözüm parçalarından elde edilmesidir. AH prosedürü sadece TA algoritmasında kullanılmayıp diğer metasezgisel yöntemlerle entegre edilebilir. Bunun yanında metodun bir yazılım dilinde kodlanması kolay ve diğer alanlara uygulanması bakımından oldukça esnek olup, yöntem çok çeşitli alanlarda kullanılabilmektedir (Cordeau vd., 2002).

Rochat ve Taillard'ın KKARP için 1995'de geliştirdiği yöntemde, iyi kalitede bir grup çözüm bir havuzda toplanmakta ve arama süreci boyunca havuz dinamik olarak güncellenmektedir. Havuzdaki çözümlerde yer alan rotalar yeni bir çözüm oluşturmak üzere seçilirler. Bu seçim işleminde iyi kaliteli çözümlere daha fazla ağırlık değeri verilerek o çözümlerde yer alan rotaların seçilme şansı artırılmaktadır, ayrıca algoritmada yeni bir çözüm oluşturulurken aynı

çözümün iki defa kullanılması engellenmektedir. Oluşturulan yeni çözümde eksik noktalar bulunabileceğinden, bir geliştirici TA algoritması ile çözümdeki eksik parçalar tamamlanmaktadır. Bu algoritma 14 CMT probleminin tümü için en iyi bilinen çözümleri oluşturmuş ayrıca bulunamayan en iyi 2 çözümü üretmiştir (Toth ve Vigo, 2002).

Tarantilis ve Kiranoudis'ın 2002 yılında "kemik rota" adını verdikleri adaptif hafıza tabanlı algoritmada ise yeni çözümler, havuzdaki çözümlerin rotalarının birleştirilmesi ile değil, "kemik" adı verilen parçaların rasgele kombinasyonu ile elde edilmektedir (Cordeau vd., 2004). Bu yöntemde havuzdaki iyi kalite çözümlerde yer alan "kemikler", yeni çözüm oluşturmak üzere aşağıdaki kriterlere göre seçilmektedir (Tarantilis vd., 2004a):

- Seçilen parça, adaptif hafızada daha önce belirlenmiş sayıdaki çözümde yer almalıdır.
- Seçilen parça belirli sayıda nokta içermelidir.

Kemik rota yöntemi 2004 yılında Tarantilis tarafından tekrar düzenlenmiş ve yeni çözümler havuzdaki parçaların stokastik olarak değil, rasgeleliğe dayanmayan stratejilerle sistematik bir şekilde seçilerek üretilmiştir. Ayrıca yöntemde daha uzun süreli bir hafıza tutulmakta olup çeşitlendirme ve yoğunlaştırma stratejisinin daha etkili olması sağlanmaktadır. Yoğunlaştırma stratejisi kaliteli çözümlerin ortak yanlarının çözümlenerek arama prosesinin sürdürülmesi, çeşitlendirme stratejisi ise hafızada yer almayan çözümlerin belirlenerek aramanın bu yöne kaydırılması ile gerçekleştirilmektedir (Tarantilis vd., 2004a). Yazarlar tarafından 2005 yılında kemik rota yöntemi, gerçek hayatta büyük ölçekli bir lojistik problemine uygulanmış ve başarılı sonuçlar elde etmiştir (Tarantilis ve Kiranoudis, 2005).

5.2.8 Tanecikli Tabu Arama Algoritması

Toth ve Vigo'nun 1998'de geliştirdikleri Tanecikli TA algoritması çözüm uzayında iyi sonuç vermesi beklenmeyen bölgelerin dışlanması fikrine dayanır. Yazarlara göre iyi çözümlerde kenar uzunluklarının ortalaması, tüm problemdeki kenarların ortalama uzunluğundan genellikle daha küçüktür. Algoritmada bu sebeple mevcut çözümün bazı komşu çözümleri elenip sadece birbirlerine belirli yakınlıktaki noktaları içeren çözümler ele alınmaktadır. Bunu gerçekleştirmek üzere $v = \beta \bar{d}$ eşik değerini geçemeyen kenarlar elimine edilmiş ve problemin arama uzayı $E(v) = \{(v_i, v_j) \in E: d_{ij} \leq v\} \cup I$ yapılarak küçültülmüştür. Burada $\bar{d}$ problemde yer alan tüm kenarların ortalaması, β algoritma süresinde değeri [1, 2] arasında değiştirilen bir seyrekleştirme parametresi ve I ise merkez depodan çıkan yollardır (Laporte vd., 2000). β değerinin [1, 2] arasında olması problemde yer alan tüm kenarların 80% - 90% arasında

elenmesi anlamına gelmektedir. Bu da algoritmanın iyi çözüm veren arama bölgelerine yoğunlaşmasını sağlamaktadır. Ayrıca eşik değeri (v) iyi çözümler elde edildikçe dinamik olarak düşürülmekte, eğer çözüm belirli bir iterasyon süresince gelişmezse eşik değeri ilk atandığı değere geri döndürülmektedir. Böylece çözüm uzayında arama bölgesi genişletilerek yöntemin çeşitlendirme stratejisi uygulanmış olmaktadır (Cordeau vd., 2004). Bunun yanında yöntemin komşuluk yapısı bir rota içinde nokta yer değişimi veya rotalar arasında nokta alış verişi olup Tabu Rota yöntemine benzemektedir. Algoritmanın en önemli özelliği hızlı bir şekilde iyi kalitede çözümler üretebilmesi olup CMT problemleri için TA algoritmaları arasında en yüksek sapma oranı/çözüm süresi değerini veren yöntemdir (Toth ve Vigo, 2003).

5.2.9 Bütünleşik Tabu Arama Algoritması

Cordeau ve arkadaşları daha önce geliştirilmiş TA algoritmalarının bazı özelliklerini birleştirerek 2001 yılında Bütünleşik Tabu Arama adını verdikleri bir algoritma hazırlamışlar ve bu tekniği KKARP, MKARP, ZPARP ve çok depolu ARP'ye uygulamışlardır. Bütünleşik TA algoritması komşuluk yapısında Tabu Rota algoritmasındaki yapıdan yaralanmakla beraber, bir çözümden komşu çözüme geçerken ekleme prosedürü kullanılmıştır. Çözümler $B(x) = \{(i, k)$: x çözümünde v_i müşterisi k aracı tarafından ziyaret ediliyor$\}$ gösterimi ile tutulmakta olup komşuluk yapısı ise, B kümesinden bir (i, k) elemanı çıkartılıp yerine $k \neq k'$ olması koşuluyla kümeye (i, k') elemanı eklenerek elde edilmektedir. Bu şekilde bir komşu çözüme geçilmesi durumunda, daha önceki çözüme dönülmemesi için tabu listesine (i, k) bilgisi eklenmekte ve bu bilgi belirli bir iterasyon süresince listede kalmaktadır. Amaç fonksiyonunu minimize etmek için v_i noktasının k' rotasına eklenmesi ardışık iki nokta arasında gerçekleştirilmektedir. Ayrıca algoritmanın sahip olduğu cezalandırılmış amaç fonksiyonu ve kendi kendine ayarlanabilir katsayılar özelliği orta kalitedeki uygun olmayan sonuçların taranmasını ve makul bir çeşitlendirme stratejisi sağlamaktadır. Bu katsayılar bir önceki çözümün uygun olup olmadığına bağlı olarak bir faktör değerine bölünmektedir. Bütünleşik TA yönteminin kodlanması oldukça basit olmakla beraber, CMT problemleri için yüksek kalitede çözümler üretmektedir. Algoritmanın sadece bir parametresi olup ARP için geliştirilmiş TA yöntemleri içersinde en az komplike olan yöntemdir (Cordeau vd., 2002).

5.2.10 ARP İçin Tabu Arama Algoritmalarının Karşılaştırılması

Bir sıralama öncelikli gruplama problemi olan ARP'de, TA algoritması diğer metasezgisel yöntemlere göre çoğunlukla daha iyi sonuç vermektedir. Şu an en güçlü TA algoritmaları orta büyüklükteki bir ARP'yi optimum veya optimuma yakın bir şekilde çözebilmektedir.

Özellikle Tanecikli TA algoritması hızlı, basit ve dinç özelliği ile büyük boyutlu problemlerde iyi kalitede çözümler vermektedir (Laporte vd., 2000).

Çizelge 5.1'de yer alan ARP için Cordeau ve arkadaşları tarafından yapılan değerlendirmede bazı Tabu Arama algoritmalarının karakteristikleri belirtilmiştir. Buna göre genellikle klasik sezgisel yöntemlerden daha karmaşık olan bu algoritmalar, daha uzun sürelerde sonuç vermektedir. Fakat üretilen çözümlerin kalitesi sezgisel algoritmalara göre daha iyi olup gerçek hayatta bu algoritmaları farklı kısıtlara sahip problemler için kullanmak mümkündür. Özellikle Adaptif Hafıza teknikleri ve Tanecikli TA yöntemi her tip kombinatoryal probleme uygulanabilecek özelliktedir (Cordeau vd., 2002).

Çizelge 5.1 ARP için Tabu Arama algoritmalarının karakteristikleri

Yöntem	**Kesinlik**	**Hız**	**Basitlik**	**Esneklik**
Tabu Rota (1994)	Yüksek	Orta	Orta	Yüksek
Taillard Tabu Arama (1993)	Çok Yüksek	Düşük	Orta - Düşük	Yüksek
Rochat veTaillard Adaptif Hafıza (1995)	Çok Yüksek	Düşük	Orta - Düşük	Yüksek
Tanecikli Tabu Arama (1998)	Yüksek	Orta	Orta	Yüksek
Bütünleşik Tabu Arama (2002)	Yüksek	Orta	Orta	Yüksek

Çizelge 5.2'de ARP için Tabu Arama algoritmalarının temel özellikleri karşılaştırılmaktadır. Bu temel özellikler aşağıda verilmektedir (Tarantilis vd., 2004a):

- Lokal arama sırasında ilerleme türü
- Orta kalitedeki uygun olmayan çözümlerin kabul edilip edilmeyeceği
- Tabu listesinde tutulan çözüm sembolleri
- Tabu listesindeki bilgilerin tutulma süreleri (rastsal dağılım veya sabit)
- Çeşitlendirme stratejisi tekniği
- Yoğunlaştırma stratejisi tekniği

Çizelge 5.2 ARP için Tabu Arama algoritmalarının karşılaştırılması

Yöntem	İlerleme Türü	Uygun Olmayan Çözümlerin Kabulü	Tabu Listesindeki Çözüm Sembolleri	Hafızada Bilgilerin Tutulma Süreleri	Çeşitlendirme Stratejisi	Yoğunlaştırma Stratejisi
Osman (1993)	Bir veya iki adet noktanın rotalar arasında taşınması veya takası	Hayır	Rotalara atama yapılan noktalar	Sabit	Yok	Yok
Tabu Rota (1994)	Bir rotadan bir noktanın alınıp başka rotaya eklenmesi	Evet	Rotalara tekrar eklenen noktalar	Düzgün dağılımdan rasgele seçilmektedir.	Frekans tabanlı cezalandırma (sık sık yer değiştiren noktalara teşvik)	Arama bulunan en iyi sonuçtan başlar. Birden fazla nokta koparma
Taillard (1993)	İki rota arasında bir noktanın takası	Hayır	Tabu Rota ile aynı	Tabu Rota ile aynı	Tabu Rota ile aynı	2-opt ile çözüm periyodik geliştirilir
Xu ve Kelly (1996)	Çıkarma zincirleri ve iki rota arasında nokta takası	Evet	Tabu Rota ile aynı	Sabit	Tabu Rota ile aynı	Arama işlemi periyodik olarak en iyi çözümlere uygulanır
Rego ve Roucairol (1996)	Sıralı eklemelerle çıkarma zincirleri ve iki rota arasında nokta takası	Evet	Ters olarak çıkarmalar	Tabu Rota ile aynı	Frekans tabanlı cezalandırma terimi (sık sık ortaya çıkan kenarlara önlem)	Frekans tabanlı cezalandırma terimi (sık sık ortaya çıkan kenarları teşvik)
Çiçek (1998)	Kenarların eklenme ve çıkarılması ile çıkarma zincirleri	Hayır	Tekrar ekleme sonucu silinen kenarlar	İki değer arasından rasgele seçilir.	Rego ve Roucairol ile aynı	Rego ve Roucairol ve her rotaya 2-opt uygulanması
Tanecikli Tabu Arama (2003)	Aynı rotada veya farklı iki rotada kaliteli kenarların yer değiştirmesi	Evet	Rotalara tekrar eklenen kenarlar	Tabu Rota ile aynı	Eşik değeri yükseltildikçe kabul edilen kenar sayısı artar	Eşik değeri azaltıldıkça kabul edilen kenar sayısı azalır
Bütünleşik Tabu Arama (2002)	Tabu Rota ile aynı	Evet	Tabu Rota ile aynı	Sabit	Tabu Rota ile aynı	Yok

5.3 Araç Rotalama Problemi İçin Geliştirilen Genetik Algoritma Yöntemleri

Genetik Algoritmalar, populasyon arama tabanlı bir metasezgisel yöntem olup Darwin'in Evrim kuramından esinlenerek ortaya çıkarılmıştır. Algoritmanın temeli, amaç fonksiyon değerine göre seçilen, birden fazla çözüme ait bazı parçalarının birleştirilerek yeni çözümler oluşturulmasına dayanmaktadır. Yöntemde her kromozomun farklı bir çözümü temsil ettiği bir havuzda, çaprazlama ve mutasyon gibi genetik operatörlerle yeni çözümler üretilmektedir. GA'da temel olarak iyileştirici sezgiseller kullanılmadığı için yöntem geniş bir alanda kullanılmaktadır. Literatürde ARP için GA uygulamaları, çoğunlukla ZPARP, KYARP, çok depolu ARP ve okul servis aracı rotalama problemleri üzerinde rastlanmaktadır (Baker ve Ayechew, 2003).

Genel olarak GA tekniğinde akış süreci aşağıdaki gibi ilerlemektedir:

Adım 1: Rasgele olarak veya belirli bir sezgisele göre $X^1 = \{ x_1^1, ... , x_N^1 \}$ başlangıç çözümü üret ve iterasyon indeksini t = 1 olarak ata (Burada N, populasyon sayısını göstermektedir).

Adım 2: Belirli bir kurala göre populasyondan iki çözüm seç.

Adım 3: Seçilen iki ebeveyn çözümü bir çaprazlama operatörü prosedüründen geçirerek iki yeni çözüm üret.

Adım 4: Rasgele seçilen yeni çözümlere küçük bir olasılıkla mutasyon işlemi uygula. Adım 2 ve 3'ü ($k \leq N/2$) defa yürüt.

Adım 5: Bir sonraki iterasyon için ele alınacak yeni populasyonu (X^{t+1}), X^t populasyonundaki en kötü 2k çözümü, üretilen 2k adet yeni çözüm ile yer değiştirerek oluştur.

Adım 6: Algoritmayı $t \geq G$ ise şimdiye kadar üretilen çözümü kullanıcıya göster ve süreci sonlandır. Aksi halde t = t + 1 yaparak, adım 2'ye dön (Burada G nesil sayısını, başka bir deyişle iterasyon sayısını göstermektedir).

ARP için geliştirilen GA yöntemlerinde adım 2'de ebeveyn çözümler genellikle çözüm kalitesine göre seçilmekte, adım 3'deki çaprazlama işlemi çözümler arasında rotaların veya bir rotadaki nokta sıralarının yer değiştirilmesi ile yapılmaktadır. Yöntemde iyi çözümlerin bazı özelliklerinin birleştirilmesi adım 5'de oluşturulan yeni populasyonun, eski populasyona göre daha iyi çözümler içermesini sağlamaktadır. ARP için geliştirilen saf GA yöntemleri tatmin edici sonuçlar vermediğinden araştırmacılar GA yöntemini diğer lokal arama teknikleri ile birleştirilme yoluna gitmişlerdir. Literatürde geliştirilen bu tür yöntemler Melez GA veya

Genetik Lokal Arama olarak geçmektedir. Bu tekniklerde genellikle çaprazlama işleminden sonra oluşturulan çözümlere lokal arama ve iyileştirme teknikleri uygulanarak yeni populasyonun daha iyi bireylerden oluşması sağlanır (Jaszkiewicz ve Kominek, 2003).

Bu bölümde ilk olarak sıralama problemleri için kullanılan kodlama, seçim, çaprazlama ve mutasyon türleri açıklanacak, daha sonra son yıllarda ARP için geliştirilen ve başarılı sonuçlar veren beş çalışma dört başlık altında irdelenecektir.

5.3.1 Sıralama Problemleri İçin Kodlama Türleri

GA'da en önemli faktörlerden biri çözümün veya bireyin nasıl simgeleneceğidir. Kurulan genetik modelin kısa sürede ve güvenilir çalışmasını sağlamak için probleme uyan en etkili ve bellekte az yer işgal eden kodlama türünün seçilmesi gerekir. Literatürde genellikle sıfır ve birlerden oluşan ikili simgelerin yanında permutasyon kodlama gibi farklı gösterimler de kullanılabilmektedir. GA'da ARP çözümleri için kullanılabilecek kodlama türleri aşağıda belirtilmektedir:

a. İkili Kodlama: Bu teknikte kromozomdaki her gen çözümün belli bir özelliğini temsil eden ikili bir değere (0 veya 1) sahiptir. İkili kodlama yönteminin en önemli avantajı çaprazlama ve mutasyon işlemlerinin kolaylıkla gerçekleştirilebilmedir. ARP'de bu tür bir kodlama ancak küçük boyutlu problemlerde izlenecek doğru parçalarının çözümde yer alıp almaması durumuna göre yapılabilir. Çizelge 5.3'de İkili kodlanmış çözüm örnekleri yer almaktadır (Gen ve Cheng, 2000).

Çizelge 5.3 İkili kodlanmış kromozom örnekleri

Kromozom A	0 0 1 1 1 0 1 0 1 0
Kromozom B	1 0 0 1 1 0 1 0 0 0

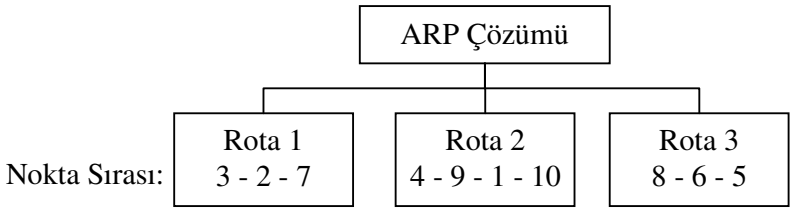
b. Permutasyon Kodlama: GSP ve iş çizelgeleme gibi sıralama problemlerinde sıklıkla kullanılan bu kodlama türünde bir değer bir kromozomda ancak bir kez bulunabilmektedir. ARP için gen kavramı, bir rotada izlenecek noktaların numaraları olup çeşitli kısıtlardan dolayı her kromozom uygun bir çözüm oluşturmayabilir. Ayrıca Permutasyon kodlamada çaprazlama ve mutasyon işlemlerinin klasik yöntemlerle yapılması oldukça zor olduğundan yeni yöntemler geliştirilmiştir. Çizelge 5.4'de Permutasyon kodlama tekniği ile kodlanmış çözümler yer almaktadır (Duncan, 1995).

Çizelge 5.4 Permutasyon kodlamalı çözümler

Kromozom I	1 2 3 4 5 6 7 8 9
Kromozom II	7 5 3 4 2 6 8 9 1

c. Grup Kodlama: Bu yöntem özellikle problemde yer alan bir elemanın bir gruba dahil edilmesi gereken kombinatoryal problemlerde kullanılmaktadır. Kromozomda yer alan bir gende, gene karşılık gelen elemanın hangi gruba dahil edileceği bilgisi taşınmaktadır, dolayısıyla bir kromozomda aynı gen birden fazla bulunabilmektedir. ARP'de noktaların hangi nolu rota içerisinde bulunacağı veya iş akış problemlerinde işin hangi makinaya atandığı bilgisi kromozomda Grup kodlama ile tutulabilir. Fakat bu tür bir kodlama ile grup içinde sıralamanın ne şekilde yapılacağı belli değildir. ARP'de rotalar içinde noktaların sıralanması genellikle 2-opt veya 3-opt iyileştirme prosedürleri ile yapılmaktadır. İkili kodlamada olduğu gibi çaprazlama ve mutasyon işlemleri, Grup kodlamada da kolay bir şekilde yapılabilmektedir (Baker ve Ayechew, 2003).

d. Genetik Araç Kodlama (GVR): Bu yöntem ARP için geliştirilmiş iki aşamalı bir gösterim tekniğidir. GVR tekniğine göre bir ARP çözümü diğer bir deyişle bir birey, birçok rotanın birleşmesinden oluşmuştur. Rotalar ise talep noktalarının belirli bir düzene göre sıralanmasından meydana gelmektedir. Problem içinde yer alan her nokta, mutlaka çözümdeki rotalardan birinde olmalıdır (Tavares vd., 2003a). GVR tekniğini kullanan GA'da bir rotada kısıtlardan biri (kapasite, mesafe, vs.) ihlal edildiğinde genellikle rota ikiye bölünerek yeni bir rota oluşturulmaktadır. GVR tekniği ile rotalara noktaların eklenmesi, rotalar arasında noktaların yer değiştirilmesi, iki nokta arasında süpürme tekniği ile rotanın oluşturulması gibi işlemler daha kolay yapılmakta ve GA, daha performanslı bir şekilde işletilebilmektedir. Şekil 5.8'de GVR tekniği ile kodlanmış bir ARP çözümü gösterilmektedir (Tavares vd., 2003b).



Şekil 5.8 Genetik Araç Kodlama gösterim tekniği

5.3.2 Sıralama Problemleri İçin Seçim Operatörü Türleri

GA'da yeni populasyonun oluşturulmasında kaç ferdin seçileceği ve hangi fertlerin çaprazlama için seçileceği seçim fonksiyonuyla sağlanır. Bu aşamada bireysel diziler amaç fonksiyonuna göre kopyalanır ve gelecek nesilde daha iyi döl verebilecek bireyler tercih edilir. Seçim operatörü, yapay bir seleksiyondur. Sıralama problemi için çaprazlanacak bireylerin seçim yöntemleri aşağıda açıklanmaktadır (Eren, 2002):

a. Rulet Tekerleği Yöntemi: Seçim yöntemleri içinde en basit ve en kolay uygulanabilir nitelikteki yöntem, Rulet Tekerleği yöntemidir. Bu yöntem aynı zamanda "değişmeli stokastik örnekleme" diye de isimlendirilmektedir. Mevcut populasyondaki en uygun kromozomlar rulet çemberi üzerine yerleştirilir, çember populasyondaki dizi sayısı kadar döndürülerek yeni nesil elde edilir. Bütün fertler bir çizgi üzerinde birbirine bitişik bölümler şeklinde dizilirler. Bununla beraber her bir ferde ait bölümün uzunluğu onun uygunluk değeri kadar olur. Rasgele sayı üretilir ve rasgele sayı hangi bölüm içine denk gelirse o bölümün ait olduğu fert seçilir. İşlem eşleşecek populasyonun gerekli adedine ulaşılıncaya kadar devam ettirilir. Bu teknik, üzerinde bölümleri olan rulet çemberine benzediği için bu şekilde isimlendirilmiştir.

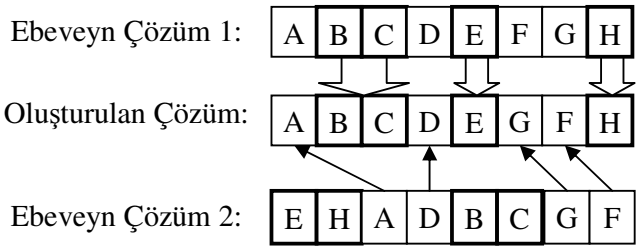
b. Rank Seçim Yöntemi: Rulet Tekerleği yöntemi eğer populasyonda uygunluk değerleri çok fazla değişiyorsa bazı problemlere yol açacaktır. Örneğin en iyi kromozom uygunluğu tüm rulet tekerleğinin %90'ı ise diğer kromozomların seçilme şansları çok düşük olacaktır. Rank seçimi ile önce populasyon uygunluk değerine göre büyükten küçüğe sıralanır ve daha sonra bu sıralama içinde en iyi ve en kötü değerdeki kromozomlardan değişik oranlarda alınır. Bir sonraki nesilde alınacak kromozomların ne kadarının iyi ve ne kadarının kötü olacağı GA modelinde başlangıçta belirlenir. Bu oran algoritma süresince sabit tutulmaktadır (Eren, 2002).

b. İkili Turnuva (Yarışma) Tekniği: Bu yöntemde, ilk önce populasyon içinde iki çözüm rasgele belirlenmekte daha sonra iki çözüm arasında uygunluk değeri daha iyi olanı ilk ebeveyn olarak seçilmektedir. Çaprazlanacak ikinci ebeveyn de populasyondan aynı kurala göre seçilmektedir. İkili Turnuva yönteminin en büyük avantajı populasyonda herhangi bir sıralama veya puanlamaya gerek kalmaksızın çözümlerin hızlı bir şekilde stokastik seçiminin sağlanmasıdır (Gen ve Cheng, 2000). ARP için en çok kullanılan seçim yöntemi İkili Turnuva yöntemidir.

5.3.3 Sıralama Problemleri İçin Çaprazlama Operatörü Türleri

Çaprazlama operatörü, GA'nın temel işlemcisi olup, GA'nın performansını en fazla etkileyen parametresidir. İkili yöntemle kodlanmış değişkenlerin yaptıkları üreme faaliyeti kromozomların çaprazlamasına benzemesi dolayısıyla böyle isimlendirilmektedir. Ancak, gerçek değerlerin kullanılmak zorunda oldukları spesifik problemlerde klasik çaprazlama yöntemi yerine daha farklı yöntemler kullanılmalıdır. ARP'de sıklıkla kullanılan Permutasyon kodlamada, kullanılan çaprazlama yöntemleri aşağıda açıklanmaktadır (Eren, 2002):

a. Pozisyona Dayalı Çaprazlama: Şekil 5.9'da gösterildiği gibi rastsal olarak seçilmiş genler bir ebeveynden çocuğa kalıtsallaştırılır. Boş yerlere ise diğer ebeveyndeki işlerin sırasında diğer işler yerleştirilir. Pozisyondaki sayılar, [1, n] rastsal tamsayılar olarak düzenlenir, daha sonra bu pozisyonlar rastsal olarak seçilir. Her pozisyonun çaprazlama olasılığı %50'dir.



Şekil 5.9 Pozisyona Dayalı Çaprazlama

b. Sıraya Dayalı Çaprazlama: İlk olarak Davis tarafından önerilmiştir. Bu yöntemle bir grup nokta rasgele seçilir. Birinci kromozomun seçilen noktalara karşılık gelen karakterleri aynen yerlerini korur. İkinci kromozomun seçilen noktalara ait karakterleri birinci kromozomun aynı noktalarındaki karakterlerin önüne getirilir. Geriye kalan boş pozisyonlara ikinci kromozomdan aktarılan yeni karakterler de göz önünde bulundurularak ilk kromozomun kullanılmayan karakterleri tarafından sıra ile (soldan sağa) yerleştirilerek yeni bir kromozom elde edilir. Bu tür çaprazlama kromozomu oluşturan karakterlerin sayı ve sıralarının önem taşıdığı durumlarda kullanılır. Bu çaprazlama işlemine ait birer çaprazlama örneği şekil 5.10'da verilmiştir (Eren, 2002).

Çaprazlamadan Önce	Çaprazlamadan Sonra
A B C D E F G	A G C D E F B
↕ ↕	
G F E D C B A	G A E D C B F

Şekil 5.10 Sıraya Dayalı Çaprazlama

c. Kısmi Planlı Çaprazlama (PMX): Goldberg tarafından geliştirilen bu çaprazlama ilk olarak GSP'de kullanılmıştır. PMX prosedürünü için aşağıda verilen 8 noktalı GSP örneği için çaprazlanacak bireyler şunlardır:

A = 2 8 6 4 5 7 1 3

B = 8 7 2 1 3 4 5 6

PMX operatörü uygulandığında A ve B'den ortak bir aralık rastsal şekilde seçilir, daha sonraki seçilmiş iki aralıktaki yeni bireylerin planları belirlenir. Bu örnekte, seçilmiş aralıklar arasındaki plan 6'ya 2, 4'e 1 ve 5'e 3'tür. İkinci olarak A ve B'deki iki aralıkları değiştirilir. Bir dizide aynı işlem birden fazla olduğundan her iki yapının da uygun olmadığı belirlenir.

$$A = \begin{array}{ccc|ccc|ccc} 2 & 8 & & 2 & 1 & 3 & 7 & 1 & 3 \\ & & & & & & & & \\ 8 & 7 & & 6 & 4 & 5 & 4 & 6 & 5 \end{array}$$

Bundan dolayı yeni yapılar uygun olmayan A ve B'de seçilmiş aralıkların da, yerleştirilemeyen daha önceki adımda belirlenen elementlerin planının değiştirilmesi gerekir. Bu örnekte, A yapısının 1, 7 ve 8 pozisyonlarında; 2, 1 ve 3 ile sırasıyla 6, 4 ve 5 ile değiştirilir ve aşağıda çözümler elde edilir:

A^I = 6 8 | 2 1 3 | 7 4 5

B^I = 8 7 | 6 4 5 | 1 2 3

d. Dairesel Çaprazlama (CX): Dairesel çaprazlama yöntemi de, Davis, Goldberg ve Lingle tarafından geliştirilmiş bir yöntemdir. İşlem adımlarını aşağıda verilen C ve D kromozomları üzerinde göstermek gerekirse bu teknikte, çaprazlama haritası yerine C kromozomundan (ilk bireyden) en sondaki değer seçilir (Eren, 2002).

C = 9 8 2 1 7 4 5 10 6 3

D = 1 2 3 4 5 6 7 8 9 10

C^I = 9 - - - - - ifadesindeki gibi C bireyinden 9 değeri seçildiğinde, 9'un karşılığı, D bireyinde 1 olmaktadır. C'de 1 geni yerine yazılır, 1'in karşılığı D'de 4 olmaktadır, 4 geni yerine yazılır. C^I = 9 - - 1 - 4 - - 6 – elde edilir. CX çaprazlama yönteminin sonrasında, C^I çözümünde kalan boşluklara D kromozomunda bulunan diğer genler yazılır. Aynı prosedür D kromozomuna da uygulanır. Bu durumda çaprazlama sonrasında yeni kromozomlar aşağıdaki gibi olur:

C^I = 9 2 3 1 5 4 7 8 6 10

D^I = 1 8 2 4 7 6 5 10 9 3

e. Sıralı Çaprazlama (OX): Bu yöntem Davis, Goldberg ve Lingle tarafından geliştirilmiştir. Aşağıda bu çaprazlama ile ilgili bir örnek verilmektedir (Eren, 2002):

A = 9 8 4 | 5 6 7 | 1 3 2 10

B = 8 7 1 | 2 3 10 | 9 5 4 6

Sıralı çaprazlama yöntemine göre, 5, 6, 7 genleri yerine 2, 3, 10 genleri atanır ve A kromozomunda daha önce 2, 3 ve 10 bulunan yerlere H yazılır, buna göre şu ifade elde edilir:

B = 8 H 1 | 2 3 10 | 9 H 4 H

H yerine dizide olmayan genler eklendiğinde ise aşağıdaki yeni kromozomlar bulunur:

A^I = 5 6 7 | 2 3 10 | 1 9 8 4

B^I = 2 3 10 | 5 6 7 | 9 4 8 1

f. Doğrusal Sıralı Çaprazlama (LOX): Falkenauer ve Bouffouix tarafından geliştirilen bu yöntem Dairesel çaprazlamanın bir varyantıdır. İşlem adımları aşağıda belirtilmektedir:

- Mevcut populasyon içerisinden rastsal olarak iki ebeveyn seç.
- Seçilen bu iki dizi (kromozom) üzerinde rastsal olarak iki alt dizi seç.
- P_1'den seçilen alt diziyi çözümden kopar, boş kalan yerlere H yaz, benzer şekilde P_2 dizisinde aynı işlemi gerçekleştir ve birinci alt diziyi P_1'e ve ikinci alt diziyi P_2'e yerleştir.

LOX çaprazlama operatörü OX yöntemine kıyasla KUARP ve TSP gibi kapalı uçlu problemlerde iyi sonuçlar vermemekle birlikte, bu yöntem daha çok bir çevrimin olmadığı AUARP, HYP ve makina çizelgeleme problemlerinde başarılı olmaktadır (Prins, 2004).

g. Alt Değişimli Çaprazlama (SXX): Kobayashi, Brady ve Mühlenbein tarafından farklı zamanlarda yapılan çalışmalar ile GSP için geliştirilmiş bir çaprazlama yöntemidir. SXX çaprazlama yöntemi ile ilgili bir örnek aşağıda yer almaktadır (Eren, 2002):

A: 7 4 1 8 5 2 9 6 3

B: 3 5 7 6 8 2 4 9 1

Rasgele olarak seçilen genler 1, 3 ve 5 olsun. A kromozomunda bu noktalar (1 5 3) sırasıyla, B kromozomunda ise (3 5 1) noktalar bulunmaktadır. Buna göre alt değişimli çaprazlama sonucunda şu bireyleri elde ederiz:

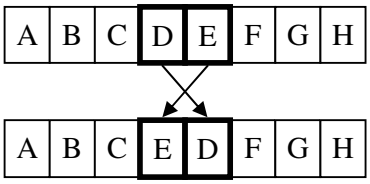
A^1 = 7 4 3 8 5 2 9 6 1

B^1 = 1 5 7 6 8 2 4 9 3

5.3.4 Sıralama Problemleri Mutasyon Operatörü Türleri

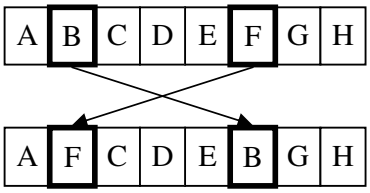
Genetik algoritmalarda önemli rol oynayan faktörlerden biri de mutasyon operatörüdür. Doğal populasyonlarda mutasyon oranı çok düşük olduğundan GA'da da genelde düşük seçilir. Son on yılda çok çeşitli mutasyon yöntemleri geliştirilmiştir. Permutasyon kodlama kullanılan sıralama problemleri için geliştirilen yöntemler aşağıda açıklanmaktadır (Eren, 2002):

a. Komşu İki Noktayı Değiştirme: Bu teknikte şekil 5.11'de gösterildiği gibi rastsal olarak seçilen iki komşu nokta yer değiştirilmektedir.



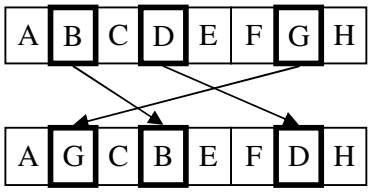
Şekil 5.11 Komşu İki Noktayı Değiştirme

b. Keyfi İki Nokta Değiştirme: Şekil 5.12'de gösterildiği üzere rastsal olarak seçilen iki nokta yer değiştirilir.



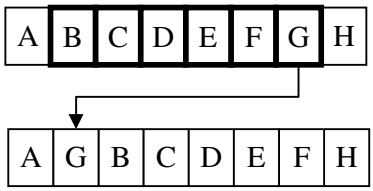
Şekil 5.12 Keyfi İki Nokta Değiştirme

c. Keyfi Üç Nokta Yer Değiştirme: Şekil 5.13'de gösterildiği gibi rastsal olarak seçilen üç nokta keyfi olarak yer değiştirilir.



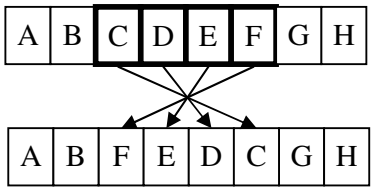
Şekil 5.13 Keyfi Üç Nokta Değiştirme

d. Araya Ekleyerek (Kaydırarak) Değiştirme (RAR): Bu mutasyonda, şekil 5.14'de gösterildiği gibi kromozomdaki bir gen kaydırılır ve diğer bir pozisyona yerleştirilir. Kaydırma noktası gebelikle rastsal olarak seçilir. Bu teknik komşu iki nokta değiştirmenin özel bir durumudur.



Şekil 5.14 Araya Ekleyerek Değiştirme

e. Ters Çevirerek Mutasyon: ARP'de yer alan rotalarda ve GSP için uygulanabilir olan bu yöntemde kromozomda rasgele olarak belirlenen bir alt dizi ters çevrilerek aynı pozisyona eklenir. Bu yönteme ait bir örnek şekil 5.15'de gösterilmektedir (Eren, 2002).



Şekil 5.15 Ters Çevirerek Mutasyon

5.3.5 Baker ve Ayechew'in Saf ve Melez Genetik Algoritması

Baker ve Ayechew tarafından 2003 yılında geliştirilen yöntemde çözümler genel atama probleminde olduğu gibi kodlanmıştır. Çözümlerin kodlanmasında rotalardaki noktaların sıralama bilgisine yer verilmemekte, sadece noktanın kaç nolu rotada bulunduğu (hangi araca atandığı) bilgisi yer almaktadır. Buna göre n müşterili ve m araçlı bir problemde kromozomun uzunluğu n olup, genler [1, m] arasında bir değer almaktadır. Dolayısıyla bir arama sürecinde

tüm çözümler için noktalar ve araçlar aynı notasyon ile numaralandırılmalıdır. Bu tür bir kodlama ile açık bir çözüm oluşturulamasa da çaprazlama ve mutasyon işlemlerinde oldukça kolaylıklar sağlanmaktadır. Yapılan çalışmada, kromozomdaki noktalar hangi rotada yer alacağı belirlendikten sonra her rota ayrı bir GSP gibi çözülerek amaç fonksiyon değeri bulunmaktadır (Baker ve Ayechew, 2003).

Hazırlanan algoritmada başlangıç çözümleri rasgele olarak ve iki klasik sezgisel yöntem kullanılarak üretilmektedir. Kullanılan tekniklerden biri Süpürme sezgiselidir. Bu yöntemde araca atanabilmesi mümkün son noktanın atanıp atanmayacağı çizelge 5.5'e göre belirlenmektedir. Çizelge 5.5'de yer alan R_c terimi süpürme sırasında araçta kalan atıl kapasitenin son müşteri talebine bölümü ile, R_d ise MKARP'de aracın gidebileceği son mesafe değerinin son müşterinin atanması için gerekli mesafeye bölümü ile elde edilmektedir. Ayrıca tablodaki problem tipi kavramı müşterilerin toplam talebinin toplam araç kapasitesine bölümü ile bulunan "sıkılık oranı" ile ilgilidir.

Çizelge 5.5 Süpürme yönteminde son noktanın kabul edilmesi durumları

Problem Tipi	**Sadece Kapasite Kısıtlı Problemlerde**	**Kapasite ve Mesafe Kısıtlı Problemlerde**
Gevşek	(sıkılık oranı < 0,95) $R_c \geq 0,9$	(sıkılık oranı < 0,8) $R_d \geq 0,9$
Sıkı	(sıkılık oranı ≥ 0,95) $R_c \geq 0,75$	(sıkılık oranı ≥ 0,8) $R_d \geq 0,9$

Başlangıç çözümü için kullanılan diğer bir yöntem de tohum noktalarını göz önüne alarak çözüm oluşturan Fisher ve Jaikumar atama tabanlı sezgisel yöntemdir. Geliştirilen GA'da küçük boyutlu test problemleri (50 müşteri) için populasyon büyüklüğü 30, daha büyükleri için 50 alınmış, başlangıç çözümlerinin yarısı Süpürme, diğer yarısı Fisher ve Jaikumar'ın yöntemi ile oluşturulmuştur. Populasyondan çaprazlanmak üzere bireyler İkili Yarışma tekniği ile seçilmekte olup, ilk önce iki çözüm rasgele belirlenmekte daha sonra iki çözüm arasında uygunluk değeri daha iyi olan çözüm ilk ebeveyn olarak seçilmektedir. İkinci ebeveyn de aynı yöntemle belirlenmektedir.

Algoritmada çaprazlama operatörü olarak İki Noktalı Çaprazlama tekniği kullanılmıştır. Şekil 5.16'da bu teknikle ilgili 20 müşterili, 4 araçlı bir örnek gösterilmektedir. Şekle göre O1 ve O2 çözümleri, P1 ve P2 ebeveynlerinin altıncı ve on beşinci noktalardan sonra kesilip ara

parçaların yer değiştirmesi ile elde edilmektedir. Böylece bazı rotalar arasında rasgele seçilen noktalar yer değiştirerek yeni çözümler oluşturulmaktadır. Bu tip bir çaprazlamanın önemli bir sakıncası, noktaların atandığı rotaların veya rotalarda bulunan noktaların her çözümde aynı yapıyı ifade edemiyor olabilmesidir. Bunun için çaprazlamadan sonra rotalarda bulunan noktaların polar açıları göz önüne alınarak rotaların tekrar numaralandırması yapılmaktadır. Böylece bir sonraki çaprazlama için mantıklı bir gruplama yapılmış olur. Geliştirilen yöntemde mutasyon işlemi ise basit olarak rasgele seçilen iki genin (farklı olması koşuluyla) yer değişimi ile yapılmaktadır (Baker ve Ayechew, 2003).

No:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
P1:	1	1	1	1	2	2	2	3	3	2	3	3	3	3	4	4	4	4	4	1
P1:	1	1	2	2	1	2	2	2	2	3	3	3	3	3	4	4	4	4	1	4

No:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
O1:	1	1	1	1	2	2	2	2	2	3	3	3	3	3	4	4	4	4	4	1
O1:	1	1	2	2	1	2	2	3	3	2	3	3	3	3	4	4	4	4	1	4

Şekil 5.16 İki Noktalı Çaprazlama operatörü

Yöntemde bir sonraki populasyona uygun olmayan çözümler sokulabilmekte, populasyonun yavru bireylerle yenilenmesi "sınıf yer değişimi" adı verilen belirli bir kurala göre yapılmaktadır. Buna göre bir çözümün uygunluk değeri olduğu gibi, uygunsuzluk değeri de olabilmektedir. Bir bireyin uygunluk değeri (f) onun kat edilen toplam mesafesi, uygunsuzluk değeri (u) ise rotalarda kapasite veya mesafe kısıtlarını geçen toplam miktarıdır. Bu yönteme göre, bir P populasyonu 4 alt kümeye ayrılmaktadır:

$S_1 = \{ p \in P: f(p) \geq f(o), u(p) \geq u(o)\}$

$S_2 = \{ p \in P: f(p) < f(o), u(p) \geq u(o)\}$

$S_3 = \{ p \in P: f(p) \geq f(o), u(p) < u(o)\}$

$S_4 = \{ p \in P: f(p) < f(o), u(p) < u(o)\}$

Burada p, ebeveyn çözümleri, o ise yavru çözümleri ifade etmekte, populasyon ebeveyn ve yavru çözümlerdeki uygunluk değerlerinin durumlarına göre gruplara ayrılmaktadır. Buna göre S_1 kümesinde yavru çözümler hem uygunluk hem de uygunsuzluk açısından ebeveyn çözümlerden daha iyi olduğu için bu grup istenen bir durumdur. Yeni populasyon sırası ile S_1, S_2 ve S_3 kümesinde yer alan yavru bireyler ile oluşturulmakta, eğer yavru çözüm

populasyonda zaten varsa, yeni çözüm populasyona sokulmamaktadır. Ayrıca algoritma süresince populasyon büyüklüğünün sabitliği korunmaktadır.

Yazarlar geliştirdiği GA'nın optimuma yakınsamasını artırmak için yoğunlaştırma stratejisi olarak aşağıdaki işlemleri yapmaktadırlar, böylece algoritma komşuluk arama yapısı ile melez bir hal almıştır (Baker ve Ayechew, 2003):

1) Yeni bir birey oluşturulduğunda, bireyin uygunsuzluk değeri 0'dan büyükse, uygunsuzluk yaratan araçtan bir müşteri kopartılıp, komşu araca atanmaktadır. Bu atama toplam kat edilen mesafede en az artışa sebebiyet verecek biçimde yapılmaktadır. Daha sonra rotalar 2-opt metodu ile iyileştirilmektedir.

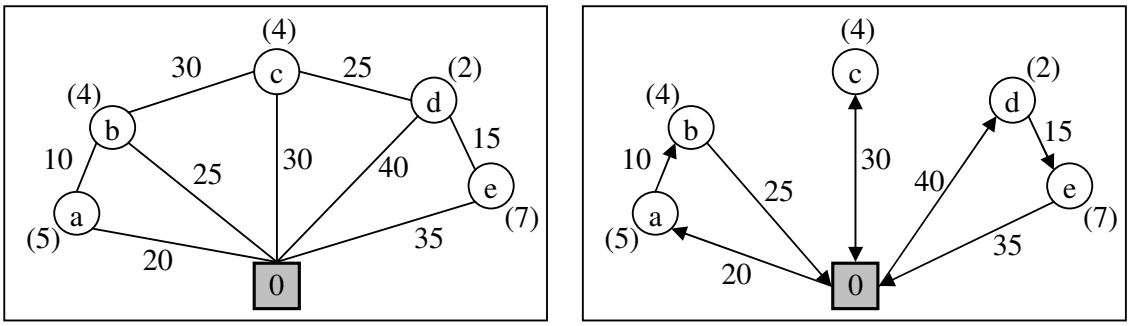
2) Başlangıçta ve her 10000 iterasyonda bir, populasyondaki her birey için periyodik olarak çözümlerin komşuları taranmaktadır. Bu da bir müşteriyi bir rotadan alıp komşu rotaya atama ve λ-yer değiştirme yöntemi (İki komşu rota üzerinde müşterilerin değiş tokuş edilmesi) ile yapılmaktadır.

Baker ve Ayechew'in geliştirdiği saf GA iyi kalitede çözümler verse de TA yöntemleri kadar performanslı değildir. Fakat komşuluk arama yapısı ile melez hale getirilen yöntem, daha hızlı bir şekilde CMT problemlerinin en iyi bilinen çözümlerine 0,5% yakınlıkta sonuçlar vermektedir.

5.3.6 Prins ve Lacomme'un Melez Genetik Algoritma Yöntemleri

KKARP için Prins tarafından 2004 yılında geliştirilen melez GA yönteminde çözüm kodlama tekniği olarak çoğu zaman GSP'de uygulanan permutasyon kodlama kullanılmaktadır. Bu teknikte genler arasında rotaların belirlendiği ayraç yer almamakta, rotalar kromozomda yer alan noktaların sırası ile oluşturulmaktadır. Bu da eğer kısıtların olmaması durumunda bir aracın tüm müşterileri dolaşacağı olarak yorumlanabilir. Bu nedenle geliştirilen bir ayırma prosedürü ile kromozomdaki nokta sıraları ele alınırken kapasite veya mesafe kısıtı ihlal edildiğinde yeni rotaya geçilerek güzergahlar belirlenmektedir. Ayrıca algoritmada kromozomda bir nokta veya ardışık duran 2'den C/q_{min} sayıya kadar olan nokta sıra göz önüne alınıp kaydırmalı bir şekilde grup oluşturularak rota alternatifleri denenmektedir. Böylece bir kromozom için uygun rotalar oluşturularak en iyi değer seçilip kesin bir çözüm değeri hesaplanmakta ve bu tip basit bir kodlama ile çaprazlama ve mutasyon işlemleri kolaylaşmaktadır. Geliştirilen ayırma prosedürüne ve kromozomun uygunluk değerinin hesaplanmasına ait bir örnek şekil 5.17'de gösterilmektedir. Soldaki şekilde noktalar arası

uzaklıklar ve müşteri talep miktarları (parantez içinde) verilmekte, sağdaki şekilde ise araç kapasitesi 10 birim alınan problemde S = {a, b, c, d, e} kromozomunun ayırma prosedürü ile ürettiği çözüm gösterilmektedir. Ayırma prosedürüne göre 10 / 2 = 5 nokta kaydırmalı bir şekilde grup oluşturabilir. Çözüm oluşturulurken dikkate alınan uygun rotalar şunlardır: (a), (b), (c), (d), (e), (a, b), (b, c), (b, c, d), (c, d) ve (d, e). Bu rotalar tek tek denendiğinde elde edilen en iyi çözüm ise 205 değeri ile şekilde sağda yer almaktadır (Prins, 2004).



Uzaklıklar ve Talep Miktarları Üretilen Çözüm

Şekil 5.17 Prins Ayırma prosedürü örneği

Geliştirilen algoritmada başlangıç populasyonu rastsal ve sezgisel yöntemlerle oluşturulmakta olup, algoritma süresince populasyon büyüklüğü sabit tutulmaktadır. Kullanılan sezgisel teknikler Paralel Tasarruf, Mole ve Jameson ile Gillet ve Miller yöntemleridir. Populasyonda çaprazlanmak üzere bireyler İkili Yarışma yöntemi ile seçilmekte, yeni çözümlerin populasyona atanması ise Δ-kuralı ile gerçekleşmektedir. Bu kurala göre yeni çözümün populasyona dahil edilebilmesi için yeni çözüm ile ebeveyn çözüm arasındaki farkın belirli bir değerden büyük olması gerekmektedir. Böylece populasyonda aynı bireylerin bulunma olasılığı azaltılmış ve populasyonda çeşitlilik sağlanmış olur. Yazar, çaprazlama operatörü olarak OX (Sıralı çaprazlama) ve LOX (Doğrusal Sıralı çaprazlama) denemiş, dairesel permutasyonda daha iyi sonuçlar veren OX yönteminde karar kılmıştır. Algoritmanın melez deyimini kazanmasını sağlayan özelliği ise mutasyon işleminin kaldırılarak yerine bir lokal arama prosedürü uygulanmasıdır. Buna göre çaprazlamadan sonra bir birey belirli bir olasılıkla lokal arama prosedürüne sokulmaktadır. Bu prosedürde farklı yöntemler ile bir çözümün tüm $O(n^2)$ komşulukları taranıp bulunan ilk daha iyi çözümde durularak mevcut çözüm geliştirilmeye çalışılmaktadır. Böylece GA'nın yapısında olmayan yoğunlaştırma stratejisi bu yolla elde edilmektedir. Uygulanan lokal arama prosedüründe, bir çözümde yer alan iki nokta u, v ve x, y noktaları sıra ile rotalarda bu noktalardan sonra gelen noktalar ise incelenen komşuluk yapıları aşağıda verilmektedir (Prins, 2004):

- Eğer u ile v komşu noktalar ise, u noktasını v noktasından sonraki pozisyona ekle.
- Eğer u ve x komşu noktalar ise, (u, x)'i v'den sonra ekle.
- Eğer u ve x komşu noktalar ise, (x, u)'yu v'den sonra ekle.
- Eğer u ile v komşu noktalar ise u ile v'yi yer değiştir.
- Eğer u, x ve v noktaları komşu ise, (u, x) ile v'yi yer değiştir.
- Eğer (u, x) ve (v, y) komşu ise, (u, x) ve (v, y)'yi yer değiştir.
- Eğer u ve v noktası aynı rota üzerinde bulunuyorsa, (u, x) ve (v, y) yerine sıra ile (u, v) ve (x, y) noktalarını koy (Bu klasik 2-opt prosedürüdür).
- Eğer u ve v noktası farklı rota üzerinde bulunuyorsa, (u, x) ve (v, y) noktaları yerine sıra ile (u, v) ve (x, y) noktalarını koy.
- Eğer u ve v noktası farklı rota üzerinde bulunuyorsa, (u, x) ve (v, y) noktaları yerine sıra ile (u, y) ve (x, v) noktalarını koy.

Prins'in algoritması ile yapılan denemelerde CMT problemleri için genelde en iyi bilinen sonuçlar bulunmuş fakat algoritmanın asıl büyük başarısı büyük boyutlu problemlerde olmuştur. Özellikle nokta sayısının 200'den büyük olduğu ARP'de algoritma, şimdiye kadar bulunan en iyi çözümleri üretmiştir. Bu da yöntemin GA'nın yapısında bulunan çeşitlendirme stratejisi ile lokal arama prosedürü ile gelen yoğunlaştırma stratejisini iyi bir şekilde birleştirmesinin bir sonucudur. Algoritmanın dezavantajı ise ayırma prosedüründe özellikle C/q_{min} değerinin yüksek olduğu problemlerde iterasyonların büyümesi sonucu hesaplama süresinin uzun olması ve bu sebeple populasyon büyüklüğünün düşük tutulmasıdır.

Lacomme ve arkadaşları 2005 yılında bu yönteme benzer bir tekniği kapasite kısıtlı yol rotalama problemine uygulamışlardır. Bu problemde talepler, yolların kesiştiği noktalarda olmayıp direkt yol üzerinde yer almaktadır. Bu problem rotalama probleminden çok çizelgeleme problemine benzemekte ve gerçek hayatta yol temizliği, çöp toplama, yolların asfaltlanması gibi işlerde karşılaşılabilmektedir. Geliştirilen GA'da çözümlerin kodlanma tekniği Prins'in 2005 yılında yaptığı çalışmadan alınmış ve ayırma prosedürü sayesinde kromozomda rota ayraçları kullanılmamıştır. Ayrıca seçim yöntemi İkili Yarışma tekniği olarak belirlenmiş, çaprazlama yöntemi olarak ise Doğrusal Sıralı çaprazlama (LOX) tekniği uygulanmıştır. Prins'in 2005 yılında çalışmaya paralel olarak mutasyon operatörü yerine yine 2-opt ve nokta yer değişimi teknikleri uygulanarak mevcut çözümden daha iyi çözümler

bulunmaya çalışılmıştır. Çeşitli büyüklükteki problemlere uygulanan algoritma başarılı sonuçlar vermiştir (Lacomme vd., 2005).

5.3.7 Alba ve Dorronsoro'nun Hücresel Melez Genetik Algoritma Yöntemi

Hücresel GA'da normal GA'dan farklı olarak populasyon, çözüm uzayının belirli bir bölgesinde oluşturulmakta ve genetik operatörler sadece her bireyin komşuluklarında uygulanmaktadır. Alba ve Dorronsoro tarafından 2005 yılında geliştirilen yöntemde, GA prosedürüne 2-opt ve λ-yer değiştirme teknikleri eklenerek melez hale getirilmiştir. Algoritmada çözümler permutasyon kodlama ile gösterilmekte olup, kromozomda rotaların belirlendiği ayraçlar kullanılmaktadır. Bu kodlama yapısına göre k rota barındıran bir kromozomda n + k – 2 adet birbirinden farklı gen bulunmakta ve talep noktaları [1, n – 1], rota ayraçları ise [n – 1, n – 1 + k] aralığındaki sayılarla gösterilmektedir. Geliştirilen yöntemin işleyiş adımları aşağıda belirtilmektedir (Alba ve Dorronsoro, 2005):

Adım 1: GA parametrelerine ilk değerlerini ata (Yöntemde populasyon büyüklüğü 100, çaprazlama olasılığı 0,65 ve mutasyon olasılığı 0,85 olarak alınmıştır).

Adım 2: Başlangıç populasyonunu belirli bir topolojide oluştur. Populasyonda bulunan her S çözümü için $f_{uyg}(S)$ uygunluk değerini aşağıdaki formüllere göre hesapla.

$$f(S) = F(S) + \lambda O_c(S) + \mu O_t(S) \tag{5.8}$$

$$f_{uyg}(S) = f_{max} - f(S) \tag{5.9}$$

Formülde yer alan F(S), S çözümü için toplam kat edilen mesafeyi, $O_c(S)$ ve $O_t(S)$ ise eğer S uygun bir çözüm değil ise, kapasite ve süre kısıtlarının ne kadar ihlal edildiği bilgisini tutmaktadır. Kısıtların önemliliğini belirten λ ve μ katsayıları ise algoritmada 1000 olarak alınmıştır. Yöntemde f_{max} algoritma süresince bulunan en kötü f değeri olup, amaç $f_{uyg}(S)$'i maksimize etmektir.

Adım 3: İkili Yarış yöntemi ile rasgele bir çözüm seç. Seçilen çözümün komşu çözümlerinden birini ele alarak, çaprazlama yapılacak diğer çözümü belirle.

Adım 4: Seçilen iki birey için rasgele belirlenen rotalarda yer alan kenarları iki çözüm arasında yer değiştirerek çaprazlama yap ve yeni çözümler oluştur (Bu tür çaprazlamaya ERX denmektedir).

Adım 5: Yeni oluşturulan çözümlere rasgele olarak İki Nokta Değiştirme ve Araya Ekleyerek Değiştirme ve Ters Çevirerek Mutasyon yöntemlerinden birini uygula.

Adım 6: 2-opt ve λ-yer değiştirme yöntemi (Çalışmada λ = 1 alınmıştır.) ile mevcut çözümü iyileştir ve en iyi çözümü seç.

Adım 7: Eğer yeni çözümler, ebeveyn çözümlerden daha iyi sonuçlar veriyorsa, yeni çözümleri populasyona ekle.

Adım 8: Eğer durdurma kriteri sağlıyorsa (belirli bir nesil sayısına ulaşılmışsa) algoritmayı durdur, aksi takdirde adım 3'e git.

Alba ve Dorronsoro geliştirdikleri GA'yı çeşitli ARP örnekleri üzerinde denemişler ve başarılı sonuçlar elde etmişlerdir. Özellikle büyük boyutlu bazı problemler için şimdiye kadar bulunan en iyi sonuçları veren algoritma, hesaplama süresi açısından diğer yöntemlere göre zayıf kalmaktadır. Bu durum yöntemin GA'nın yapısında bulunan çeşitlendirme stratejisini yoğun olarak kullanmasından kaynaklanmaktadır (Alba ve Dorronsoro, 2005).

5.3.8 Jaszkiewicz ve Kominek'in Melez Genetik Algoritması

Jaszkiewicz ve Kominek'in 2003 yılında geliştirdikleri yöntemin en önemli özelliği global dışbükeylik testine bağlı olarak yapılan mesafe korumalı çaprazlama tekniği olup, yazarlar bu yöntemi gerçek hayatta karşılaştıkları büyük boyutlu bir ARP'ye uygulamışlardır. Bir problemde global dışbükeylik özelliğinin olup olmadığı, problemin iyi çözümleri arasındaki benzerliğe bakılarak anlaşılabilmektedir. Birbirine aşırı derecede benzer iyi kalitede çözümleri olan bir problemde global dışbükeylik özelliği var kabul edilmektedir. Yazarlara göre ARP çözümleri dört açıdan birbirine benzeyebilmektedir (Jaszkiewicz ve Kominek, 2003):

- İki çözümde ortak bulunan kenarların oranı.
- Araçlara atanan ortak noktaların oranı.
- Araçlara atanan ortak kenarların oranı.
- Daha önce belirlenen nokta çiftlerinin aynı rotada bulunması oranı.

Genel olarak ARP ve GSP gibi sıralama tabanlı kombinatoryal problemlerde global dışbükeylik özelliği bulunduğundan çözüm uzayı taranırken belirli bir bölgede çok sayıda lokal minimum noktaya rastlanmaktadır. Mesafe korumalı çaprazlama tekniği bu bulgudan yola çıkarak, yavru çözümleri ebeveyn çözümlerin aynı niteliklerinden oluşturmakta, farklı olduğu yerlerde rasgeleliği uygulayarak eksik kısımları tamamlamaktadır. Çalışmada global

dışbükeylik testi şu şekilde yapılmaktadır: Yazarlar üzerinde çalıştıkları problem üzerinde 500 lokal minimum çözüm belirlemişler ve bu çözümler arasında yukarıdaki maddelere göre benzerlik oranını ve bu benzerlik ölçülerinin çözüm kalitesi ile korelasyonunu hesaplamışlardır, bu sayede çaprazlama işlemi sırasında hangi benzerlik kuralına göre çözümlerin oluşturulacağı bulunacaktır. Geliştirilen yöntemde benzerlik ölçülerine göre 3 tip mesafe korumalı çaprazlama operatörü oluşturulmuştur. İlk çaprazlama operatörü araçlarda ve çözümlerde ortak bulunan kenarların dışındaki benzerlik ölçüleri için olup işlem adımları aşağıda açıklanmaktadır (Jaszkiewicz ve Kominek, 2003):

Adım 1: P1 ebeveyn çözümündeki her rotayı 0,5 olasılıkla yeni çözümlere aktar.

Adım 2: P2 ebeveyn çözümündeki pozisyonundan farklı olan yeni çözümdeki noktaları kaldır.

Adım 3: Yeni çözümde boş kalan yerleri amaç fonksiyonunda minimum artış sağlanacak şekilde ekleme sezgiseli kullanarak doldur.

İkinci mesafe korumalı çaprazlama operatörü, belirli nokta çiftlerinin aynı rotada bulunması dışındaki benzerlikler için geliştirilmiş olup işlem adımları aşağıda belirtilmektedir:

Adım 1: P1 ebeveyn çözümündeki her rotayı 0,5 olasılıkla yeni çözümlere aktar.

Adım 2: P2 ebeveyn çözümünden rasgele olarak seçilen noktaları ele al. Eğer seçilen nokta yeni çözümde yer almıyorsa, yeni çözümde noktayı P2 çözümünde noktanın bulunduğu pozisyona ekle. Eğer ekleme sonucunda kapasite kısıtı aşılıyorsa, yeni bir rota oluştur ve rotaları tekrar düzenle.

Geliştirilen üçüncü mesafe korumalı çaprazlama operatörü ise araçlara ortak atanan noktaları dikkate almayıp diğer üç benzerliği ele almaktadır. Bu çaprazlama operatörünün işlem adımları aşağıda açıklanmaktadır:

Adım 1: P1 ebeveyn çözümündeki her rotayı 0,5 olasılıkla yeni çözümlere aktar.

Adım 2: P2 ebeveyn çözümündeki pozisyonundan farklı olan yeni çözümdeki noktaları sil.

Adım 3: Yeni çözümde boş kalan yerleri P2 ebeveyn çözümdeki noktaların pozisyonunu dikkate alarak amaç fonksiyonunda minimum artış sağlanacak şekilde ekleme sezgiseli kullanarak doldur, gerekirse yeni bir rota oluştur.

Ele alınan problemde incelenen 500 iyi çözüm için dört benzerlik ölçüsü de amaç fonksiyonu ile önemli oranda ilişkili bulunduğu için yukarıda belirtilen üç çaprazlama yöntemi de kullanılarak, ebeveyn çözümler ile yeni oluşturulan çözümler arasında benzerlikler korunmaya çalışılmaktadır. Geliştirilen yöntemde başlangıç çözümleri rasgele oluşturulmakta ve daha sonra 2-opt ve ekleme sezgiselleri ile çözümler iyileştirilmektedir. Ayrıca 40 bireyden oluşan populasyon içinden bireyler de rasgele olarak seçilmekte, bu da yazarlara göre çeşitlendirme faktörünün etkisini azaltmaktadır. Çalışmada üç çaprazlama operatörünü kullanan GA yöntemleri, ikili kodlamayı kullanan klasik bir GA ve bir TB algoritması ayrı ayrı gerçek hayattaki probleme uygulanmış ve aşağıdaki sonuçlar gözlenmiştir (Jaszkiewicz ve Kominek, 2003):

- Geliştirilen çaprazlama operatörlerini kullanan GA yöntemleri diğer metotlara göre daha iyi sonuçlar bulmuştur.
- İlk çaprazlama operatörü en iyi çözümleri bulmakla beraber, farklı denemelerde birbirine çok yakın değerler üretmiştir. İkinci çaprazlama operatörü ise iyi kalitede çözümler geliştirmiş fakat farklı denemelerde birbirine uzak sonuçlar vermiştir.
- Uzun süreli hesaplamalarda üçüncü çaprazlama operatörü, klasik GA'ya yakın sonuçlar vermekte, TB algoritmasından ise daha iyi sonuçlar bulmaktadır.

Jaszkiewicz ve Kominek'in yaptığı çalışmadan çıkartılacak en önemli sonuç, ele alınan ARP'de araçlara atanan ortak noktaların veya nokta çiftlerinin korunarak yapılan çaprazlamanın diğer yöntemlere göre daha iyi sonuçlar vermesi ve farklı denemelerde birbirine çok yakın çözümler üretmesidir.

5.4 Araç Rotalama Problemi İçin Geliştirilen Karınca Kolonisi Yöntemleri

Doğadan esinlenerek geliştirilen diğer bir metasezgisel algoritma olan Karınca Kolonisi Optimizasyon (KKO) yöntemi, karıncaların yiyecek bulmada bıraktıkları feramon maddesinin yiyeceğe yakın olan yollarda daha çok birikmesi, arkadan gelen karıncaların kararını buna göre belirlemesi ve böylece en kısa yolun bulunabileceği fikrine dayanır. Yollarda bulunan feramon miktarları daha önceki iterasyonlarda, karıncaların o yolu kullanarak ürettiği çözümlerin kalitesini vurgulamaktadır (Dorigo vd., 1999).

KKO algoritması genellikle diğer metasezgisel yöntemler kadar iyi kaliteli sonuçlar üretememektedir fakat algoritma bazı lokal arama yöntemleri ile birleştirilerek üretilen

çözümler iyileştirilebilir (Toth ve Vigo, 2002). ARP için geliştirilen KKO algoritmaları genel olarak aşağıdaki adımlarla çalışırlar (Tarantilis vd., 2004a):

Adım 1: Karıncaların ileriyi görüş derecesi ve feramon miktarını göz önüne alarak çözümlerin oluşturulması.

Adım 2: Üretilen çözümlerin lokal arama yöntemleri ile iyileştirilmesi.

Adım 3: Her kenarda bulunan feramon miktarının güncellenmesi (buharlaştırma).

Bunun yanında bazı algoritmalar aşağıdaki adımı uygulayarak çözüm kalitesini yükseltmeye çalışmaktadır:

Adım 4: Uygun nokta kombinasyonlarının bulunduğu listenin, oluşturulan her çözüm sonrası güncellenmesi (yeni kombinasyonların eklenmesi, bazılarının çıkartılması).

5.4.1 Bullnheimer, Hartl ve Strauss Karınca Kolonisi Optimizasyon Algoritmaları

ARP için ilk KKO algoritması 1997 yılında Bullnheimer ve arkadaşları tarafından geliştirilmiştir. Ele alınan ARP kapasite ve mesafe kısıtlı olup içinde tek depo bulunmaktadır ve problemde her araç aynı kapasite değerine sahiptir. Yazarlar çalışmalarında ARP'yi aynı düzlemde bulunan birden fazla GSP gibi görmüşler, bu sebeple daha önce GSP'ye uygulanan teknikleri uygulamışlardır (Dorigo vd., 1999). Aradaki en önemli fark bir talep noktasında bulunan bir karınca, gideceği bir noktayı seçerken kapasite veya mesafe kısıtının ihlal edilip edilmediğinin göz önüne alınması ve eğer kısıtlar sağlamıyorsa ya başka bir talep noktasının ya da merkez noktanın bir sonraki gidilecek nokta olarak seçilmesidir. Geliştirilen algoritmada i noktasında bulunan bir karıncanın gideceği bir sonraki noktanın j noktası olma olasılığı hesaplanırken kapasite ve mesafe kısıtları aşağıdaki 5.10 nolu formülde gösterildiği gibi göz önüne alınmaktadır (Bullnheimer vd., 1997a):

$$p_{ij} = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha}[1/d_{ij}]^{\beta}[\mu_{ij}]^{\nu}[\mathrm{K}_{ij}]^{\lambda}}{\sum_{h \in N}[\tau_{ih}]^{\alpha}[1/d_{ih}]^{\beta}[\mu_{ih}]^{\nu}[\mathrm{K}_{ih}]^{\lambda}} & \mathrm{v_j} \in N_i \text{ ise} \\ 0 & \text{aksi takdirde} \end{cases} \qquad (5.10)$$

Burada τ_{ij}, i ve j noktası arasında yer alan yol üzerindeki feramon miktarını, N_i ise i noktasından sonra gidilebilmesi mümkün noktalar kümesini ifade etmektedir. Bunun yanında formülde yer alan μ_{ij} ifadesi $\mu_{ij} = d_{i0} + d_{0j} - d_{ij}$ ile hesaplanmakta olup i ve j noktalarının birleştirilmesinin sağlayacağı tasarruf miktarını, K_{ij} ise $K_{ij} = (Q_i + q_j)/C$ ile hesaplanıp i

noktasından sonra j noktası geldiği takdirde kapasite kullanım oranını belirtmektedir. Bu formülde yer alan Q_i ifadesi ilgili rota için kullanılan mevcut kapasite miktarını göstermektedir. 5.10 nolu formüldeki α, β, ν ve λ parametreleri ise sırasıyla yol üzerindeki feramon miktarı, karıncanın ileriyi görüş derecesi, tasarruf miktarı ve kapasite kullanım oranının ağırlıklarının belirlenmesinde kullanılmaktadır.

Karıncalar tarafından uygun çözümler oluşturulduktan sonra, rotalara 2-opt sezgiseli uygulanarak iyileştirme sağlanır. Çözüm üretildikten sonra tüm yolların feramon miktarlarında 5.11 nolu formüldeki gibi değişiklik yapılır:

$$\tau_{ij}^{yeni} = \rho.\tau_{ij}^{eski} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} + \sigma\Delta\tau_{ij}^{*} \qquad (5.11)$$

Burada $0 < \rho \leq 1$ olup feramon izinin buharlaşma oranını ve m, toplam karınca sayısını göstermektedir. $\Delta\tau_{ij}^{k}$ parametresi ise, k. karıncanın ziyaret ettiği kenara bıraktığı feramon miktarıdır ve aşağıdaki 5.12 nolu formüle göre hesaplanmaktadır (Bullnheimer vd., 1997a):

$$\Delta\tau_{ij}^{k} = \begin{cases} 1/L^{k}, & \text{(i, j) 'den geçilirse,} \\ 0, & \text{aksi takdirde} \end{cases} \qquad (5.12)$$

5.12 nolu formülde L^k, k. karıncanın gerçekleştirdiği turun uzunluğudur. 5.11 nolu formülde yer alan $\Delta\tau_{ij}^{*}$ ifadesi ise algoritma süresince iyi bir sonucun bulunması halinde, çözümü oluşturan yollardaki feramon miktarını daha da artırılmasıyla ilgilidir. Yazarlar iyi çözümü bulacağı kabul edilen bu karıncalara "seçkin karınca" demektedir ve seçkin karıncanın geçtiği yollar $\Delta\tau_{ij}^{*}$ terimi ile vurgulanmaktadır. σ parametresi 5.11 nolu formülde algoritma süresince seçkin karıncaların sayısını belirtmekte olup $\Delta\tau_{ij}^{*}$ ise aşağıdaki gibi hesaplanmaktadır:

$$\Delta\tau_{ij}^{*} = \begin{cases} 1/L^{*}, & \text{(i, j) 'den geçilirse,} \\ 0, & \text{aksi takdirde} \end{cases} \qquad (5.13)$$

Bu formülde L^*, seçkin karıncanın ürettiği çözümün amaç fonksiyon değeridir. Geliştirilen yöntemde GSP'de uygulandığı gibi her talep noktasına bir karınca yerleştirilerek süreç başlatılmakta ve algoritma genel olarak rotaların oluşturulması ve feramon izlerinin güncellenmesi işlemleriyle iki aşamada iteratif olarak sürdürülmektedir.

Yazarlar saf KKO ve 2-opt ile melezleştirdikleri KKO algoritmasını değişik parametreler ile 14 CMT problemi üzerinde uygulamışlardır. Denemelerin hepsi TB ve TA algoritmaları kadar

başarılı sonuçlar vermese de, en iyi çözümü parametrelere $\alpha = \beta = \nu = \lambda = 5$ olarak atama yapıldığında elde etmişlerdir. Ayrıca melez KKO, saf KKO'dan 10% daha iyi çözüm üretmiş bu da KKO algoritmasının ARP için tek başına yeterli bir yöntem olmadığını göstermiştir. Çalışmada ayrıca, seçkin karıncaların çözüm kalitesi üzerindeki etkisi de araştırılmıştır. 50 talep noktasından oluşan ilk CMT probleminde en iyi sonuçlar, belirlenen 50 seçkin karınca ile bulunmuş, seçkin karınca sayısı artırılması halinde sürecin başlarında aramanın lokal optimum noktalara takılması sebebiyle algoritmanın performansı düşmüştür. Bunların yanında geliştirilen KKO algoritması ile diğer bazı testler yapılmış ve en iyi sonuçlar ρ buharlaştırma katsayısına 0,75 atanmasıyla, karınca sayısı m = n alınıp karıncaların çözüm oluşturmaya talep noktasından başlatılmasıyla, seçkin karınca sayısının nokta sayısı n'ye eşitlenmesiyle elde edilmiştir (Bullnheimer vd., 1997a).

Bullnheimer ve arkadaşlarının 1997'de yaptığı diğer bir çalışmada hesaplama güçlüğünden dolayı, karıncaların yönlerini belirledikleri fonksiyondan tasarruf miktarı μ_{ij} ve kapasite kullanım oranı K_{ij} terimlerini çıkartmışlardır. Böylece i noktasında bulunan bir karınca j noktasını seçme olasılığı aşağıdaki 5.14 nolu formüle göre hesaplanmaktadır (Bullnheimer vd., 1997b):

$$p_{ij} = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{h \in N}[\tau_{ih}]^{\alpha}[\eta_{ih}]^{\beta}} & v_j \in N_i \text{ ise} \\ 0 & \text{aksi takdirde} \end{cases} \qquad (5.14)$$

Dorigo ve arkadaşlarının GSP için geliştirdiği KKO algoritmasında ve Bullnheimer ve arkadaşlarının daha önceki çalışmalarında "görünürlük" kavramı (η_{ij}) yolun uzunluğunun bire bölümü ile elde edilmekteydi. Geliştirilen yöntemde ise μ_{ij} ifadesinden esinlenilerek görünürlük kavramı değiştirilerek $\eta_{ij} = d_{i0} + d_{0j} - g.d_{ij} + f.\left|d_{i0} - d_{0j}\right|$ ile hesaplanmaktadır. Burada f ve g parametreleri, sırasıyla iki noktanın merkez depoya pozitif uzaklık farkının ve iki nokta arasındaki mesafenin ağırlıklarıdır.

Yazarlar ayrıca yaptıkları çalışmada arama prosesini hızlandırmak için bir noktadan diğer noktaya giderken o noktanın belirli yakınlıktaki komşu noktalarını göz önüne almışlar ve bu komşu noktaları "aday noktalar" olarak ifade etmişlerdir. Bunun yanında algoritmanın feramonların güncellenmesi aşamasında, karıncalar ürettikleri çözümlerin kalitesine göre sıralanıp sadece en iyi çözümü üreten ($\sigma - 1$) karıncaya göre yollardaki feramon miktarı ayarlanmaktadır. Bu da aşağıdaki 5.15 nolu formüle göre yapılmaktadır:

$$\tau_{ij}^{yeni} = \rho.\tau_{ij}^{eski} + \sum_{k=1}^{\sigma-1} \Delta\tau_{ij}^{k} + \sigma\Delta\tau_{ij}^{*} \tag{5.15}$$

Yukarıdaki 5.15 nolu formüle göre feramon miktarını belirleyen iki önemli terim ($\Delta\tau_{ij}^{k}$ ve $\Delta\tau_{ij}^{*}$) vardır. Bunlardan $\Delta\tau_{ij}^{k}$ ifadesi, (σ – 1) en iyi çözümü üreten karınca içersinden r. sırada yer alan karınca için aşağıdaki formül kullanılarak hesaplanmaktadır.

$$\Delta\tau_{ij}^{k} = \begin{cases} (\sigma - r)/L^{k}, & \text{(i, j) 'den geçilirse,} \\ 0, & \text{aksi takdirde} \end{cases} \tag{5.16}$$

$\Delta\tau_{ij}^{*}$ terimi ise daha önceki çalışmadaki olduğu gibi iyi çözümlerde bulunan yolları vurgulamak için formülde yer almakta ve 5.13 nolu formül ile hesaplanmaktadır. Geliştirilen yöntemde σ = 6 alınmış, diğer bir deyişle her iterasyonda 5 en iyi çözümün yollardaki feramon miktarını belirlemesine izin verilmiştir. Ayrıca feramon ve görünürlük katsayılarına α = β = 5, buharlaşma oranına ρ = 0,75, tasarruf oranlarına f = g = 2 değerleri atandığında ve sadece n/4 sayıdaki en yakın komşu nokta, aday nokta olarak belirlendiğinde 14 CMT probleminde daha kısa sürelerde daha iyi sonuçlar elde edilmiştir. Fakat üretilen sonuçlar ve algoritma performansı TA ve TB algoritmaları ile yarışacak düzeyde değildir (Bullnheimer vd., 1997b).

5.4.2 Tasarruf Tabanlı Karınca Kolonisi Optimizasyon Algoritmaları

Doerner ve arkadaşları tarafından 2002 yılında geliştirilen algoritmada ana fikir Tasarruf algoritması ile gelen eşzamanlı tur oluşturma mekanizması ile derecelendirme tabanlı bir KKO yöntemi oluşturmaktır. Klasik Tasarruf yönteminde yollar $\mu_{ij} = d_{0i} + d_{0j} - d_{ij}$ ile hesaplanan tasarruf değerlerine göre büyükten küçüğe sıralanmakta ve bu sıralamaya göre kapasite kısıtını aşmayacak şekilde birleştirilerek rotalar oluşturulmaktadır. Geliştirilen yöntemde ise yollar $\xi_{ij} = [\mu_{ij}]^{\beta}[\tau_{ij}]^{\alpha}$ değerlerine göre büyükten küçüğe sıralanmaktadır. Formülde α ve β parametreleri sıra ile yollardaki feramon miktarı ve tasarruf değerinin ağırlıkları olup algoritmanın başlangıcında tüm yollardaki feramon miktarları eşit kabul edilerek sıralama yapılmaktadır. Buna göre i noktasında bulunan bir karıncanın y noktasını seçme olasılığı aşağıdaki 5.17 nolu formüle göre belirlenmektedir (Doerner vd., 2002):

$$p_{ij} = \begin{cases} \dfrac{\xi_{ij}}{\sum_{(h,l)\in N_i} \xi_{hl}} & \text{v}_j \in N_i \text{ ise} \\ 0 & \text{aksi takdirde} \end{cases} \tag{5.17}$$

Hesaplama süresini azaltmak için, sadece belirli sayıdaki büyük ξ_{ij} değerleri dikkate alınmaktadır. Bu sebeple 5.17 nolu formülde N_i terimi, i noktasından sonra gidilebilmesi mümkün olan ve büyük ξ_{ij} değerine sahip komşu noktalar kümesini ifade etmektedir. Bir rotanın oluşturulması, i noktasından sonra gidilebilecek bir noktanın bulunmadığı durumda (N_i boş küme ise) tamamlanır. Çözümler oluşturulduktan sonra rotalar 2-opt sezgiseli iyileştirilmekte ve yollardaki feramonlar Bullnheimer ve arkadaşlarının 1997(b)'de yaptığı çalışmadaki gibi 5.15 nolu formüle göre güncellenmektedir. Geliştirilen bu algoritmanın en önemli özelliği bu adımdan sonra ortaya çıkmaktadır. Güncellenen feramon miktarına göre ξ_{ij} değerleri tekrar hesaplanmakta ve yollar ξ_{ij}'e göre tekrar büyükten küçüğe sıralanmaktadır. Böylece iyi gözüken bazı yolların önemliliği azalabilmekte, bazılarının ise artarak iyi çözümlerin bulunma olasılığı yükseltilmekte ve algoritma dinamiklik kazanmaktadır.

Yazarlar geliştirdikleri KKO algoritmasını 14 CMT problemi üzerinde uygulamışlar ve başarılı sonuçlar elde etmişlerdir fakat yaptıkları çalışmada hesaplama sürelerini belirtmemişlerdir. Uygulamada parametre değerleri $\alpha = \beta = 5$ ve $\sigma = 6$ olarak belirlenmiştir. Ayrıca daha önceki çalışmalara kıyasla buharlaştırma katsayısı $\rho = 0{,}75$ yerine $\rho = 0{,}95$ alınarak daha iyi sonuçlar üretilmiştir. Bunun yanında algoritmada bir karıncanın gideceği diğer nokta belirlenirken, bulunduğu noktanın sadece n/4 adet yakınlıktaki komşu noktası dikkate alınmaktadır (Doerner vd., 2002).

Reimann, Doerner ve arkadaşlarının 2004'de yaptığı diğer bir çalışma yukarıda açıklanan algoritmanın devamı niteliğindedir. Geliştirilen yöntemin en önemli avantajı "böl ve yönet" yaklaşımı ile gelen hızlılıktır. Buna göre ARP ilk önce turlardan oluşan alt parçalara ayrılıp, bu parçalar yakınlık ilişkisine göre çözülerek birleştirilmektedir. Bu alt parçalar ise Doerner ve arkadaşlarının 2002 yılında geliştirdiği yöntem ile çözülmektedir. Yöntemin işleyiş adımları aşağıda belirtilmektedir (Tarantilis vd., 2004a):

Adım 1: Problemi belirli bir iterasyon sayısıyla Doerner ve arkadaşlarının 2002 yılında geliştirdiği Tasarruf tabanlı KKO algoritması ile çöz.

Adım 2: Bulunan en iyi sonuçta yer alan her turun merkez noktasını hesaplayarak turlar arasında yakınlık ilişkisi kur.

Adım 3: Süpürme sezgisel algoritmasını kullanarak rotaları yakınlıklarına göre grupla.

Adım 4: Rota gruplarını tekrar belirli bir iterasyon sayısıyla Doerner ve arkadaşlarının 2002 yılında geliştirdiği Tasarruf tabanlı KKO algoritması ile ayrı ayrı çöz.

Adım 5: Rota gruplarında oluşan turları birleştirerek ana problem için çözümü oluştur.

Geliştirilen algoritma CMT problemleri için oldukça kısa zamanlarda iyi kalitede sonuçlar vermiştir.

Reimann ve Laumanns 2006 yılında Tasarruf tabanlı KKO yaklaşımını KKARP'ye benzer bir problem olan ve özellikle telekomünikasyon ağlarında sıkça rastlanan kapasite kısıtlı minimum ağaç problemi üzerinde denemişlerdir. Çalışılan problemin KKARP'den tek farkı nokta gruplarının bir HYP'de olduğu gibi sıralamasının yapılmasına gerek olmaması, sadece gruplandırılarak bir ağaç oluşturulmasının yeterli olmasıdır. Geliştirilen yöntemin işleyiş adımları aşağıda belirtilmektedir (Reimann ve Laumanns, 2006):

Adım 1: Tasarruf tabanlı KKO algoritması ile problemi KKARP gibi ele alıp başlangıç çözümleri oluştur.

Adım 2: Oluşturulan en iyi çözümü $\lambda = 1$ olduğu nokta yer değiştirme sezgiseli ile iyileştir ve sonucu ağaç problemi çözümüne dönüştür.

Adım 3: Elde edilen amaç fonksiyon değerine göre yollardaki feramonları güncelle.

5.4.3 Ekleme Tabanlı Karınca Kolonisi Optimizasyon Algoritması

Reimann ve arkadaşlarının 2003 yılında yaptıkları çalışmada KKO algoritmasının farklı tip ARP üzerinde "dinçlik" faktörünü, diğer bir deyişle parametre değişikliklerinin problem tipinden bağımsızlığı konusunu ele almışlardır. Bunun için KKARP, ZPARP, KYARP ve karma yüklemeli ZPARP için ekleme tabanlı bütünleştirilmiş bir KKO algoritması geliştirilmiştir. Yöntemde ilk önce merkez depodan uzaklıklarına göre bir nokta, rota oluşturmak üzere rasgele seçilir (Uzak olan noktaların seçilme şansı daha fazladır). Daha sonraki noktaların rotaya eklenme olasılığı aşağıdaki formüle göre belirlenmektedir (Reimann vd., 2003):

$$p_i = \begin{cases} \dfrac{\kappa_i}{\sum_{h|\kappa_h>0} \kappa_h} & \kappa_i > 0 \text{ ise} \\ 0 & \text{aksi takdirde} \end{cases} \tag{5.18}$$

5.18 nolu formülde yer alan κ_i değerlendirme fonksiyonu, rotalanmamış bir i noktasının mevcut rotada en uygun yere eklenmesi ile ilgilidir ve aşağıdaki formüle göre hesaplanmaktadır:

$$\kappa_i = \max\left\{0, \max_{j \in R_i}\left[\eta_{ij}\frac{\tau_{ji}+\tau_{ik}}{2\tau_{jk}}\right]\right\} \qquad \forall i \in N_u \tag{5.19}$$

5.19 nolu formülde N_u rotalanmamış noktalar kümesini, τ_{ij} ise i ve j nolu noktalar arasındaki yoldaki feramon miktarını belirtmektedir. η_{ij} ifadesi ise i nolu noktayı rotaya, j nolu noktadan sonra eklemenin maliyet değeri olup aşağıdaki formülle hesaplanmaktadır:

$$\eta_{ij} = \max\left\{0, \alpha.d_{0i} - \beta(d_{ji}+d_{ik}-d_{jk}) - (1-\beta)(b_k^i - b_k) + \delta(\gamma.type_i + (1-\gamma)(1-type_i))\right\}$$

$$\forall i \in N_u, \forall j \in R_i \tag{5.20}$$

5.20 nolu formülde k noktası, mevcut rotada j noktasından sonra gelen noktayı, R_i ise mevcut rotada, i noktasının ondan sonra eklenebileceği noktalar kümesini ifade etmektedir. Formülde yer alan müşteriye geliş zamanı b_k, müşteri tipi $type_i$ ve bunların önem katsayıları δ ve γ ise karma yüklemeli ve zaman pencereli ARP ile ilgili olup, KKARP için değerleri 0 alınmaktadır. Formülde yer alan α katsayısı merkez depodan uzakta bulunan noktaların eklenme olasılığını, β katsayısı ise j ve k noktalarına yakında bulunan noktaların eklenme olasılığını belirlemektedir.

Geliştirilen algoritmada bir karınca tarafından bir çözüm oluşturulduğunda, λ yer değiştirme yöntemi ile çözüm iyileştirilmektedir. Buna göre bir nokta bulunduğu rotadan başka bir rotaya taşınmakta veya iki rota arasında noktalar yer değiştirilmektedir. Ayrıca yöntemde yollardaki feramonların güncellenmesi Bullnheimer ve arkadaşlarının 1997(b)'de yaptığı çalışmadaki gibi 5.15 nolu formüle göre hesaplanmaktadır. Uygulamada karınca sayısı m = n/2, buharlaşma katsayısı ρ = 0,95 ve seçkin karınca sayısı σ = 4 olarak sabit alınmış, diğer parametreler ise değişken tutulmuştur. Algoritma her problem için 10 dakika sabit çalıştırılmış ve her problem tipi için en iyi çözümü veren parametre değerleri ise α = 2, β = 1, δ = 1 ve γ = 0,33 olarak bulunmuştur. Yöntemin çözüm kalitesi ve performansı TA ve TB algoritmaları kadar iyi olmasa da yöntem, dört tip ARP için parametre değerlerinin değiştirilmesine gerek kalmaksızın esnek ve uygulanabilir niteliktedir (Reimann vd., 2003).

5.4.4 Çoklu Karınca Kolonisi Optimizasyon Algoritması

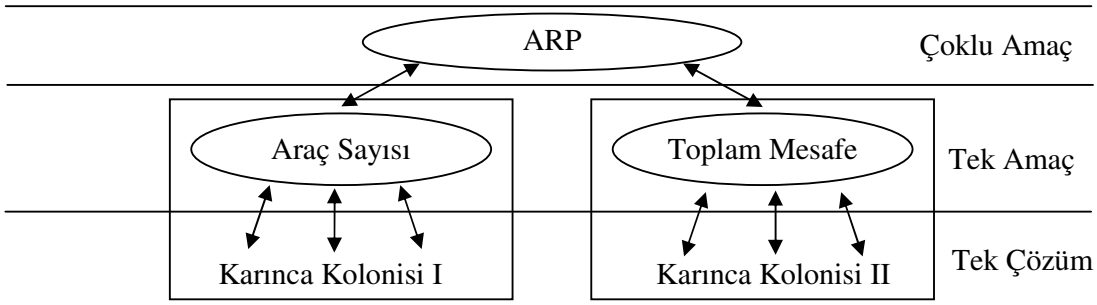
Bell ve McMullen'in 2004 yılında yaptığı çalışmada geliştirilen tekli ve çoklu KKO algoritmaları ARP üzerinde denenmektedir. Çoklu KKO algoritmasında birden fazla koloni olup her kolonin amacı farklıdır. Doğada karınca kolonilerinde de farklı amaçlarda karıncalar bulunmakta ve bunlar arasında besin bulma, yuva kurma ve yuvayı savunma gibi farklı tipte görevler yer almaktadır. Yazarlar bundan esinlenerek ARP için her kolonin bir rota oluşturduğu çok kolonili bir KKO algoritması geliştirmişler ve özellikle nokta sayısı n'in çok olduğu ARP'de bu yöntemin tek kolonili bir KKO yönteminden daha iyi olduğunu öngörmüşlerdir. Buna göre çoklu KKO'da her koloni yollara farklı feramon maddesi bırakmakta ve böylece bir kolonin yola bıraktığı feramon diğer kolonideki karıncaların yol seçimini etkilememektedir. Ayrıca yapılan çalışmada farklı bir yol seçim fonksiyonu kullanılmakta olup i noktasında bulunan bir karıncanın hangi noktayı seçeceği aşağıdaki formüle göre belirlenmektedir (Bell ve McMullen, 2004):

$$\begin{aligned} &q \le q_0 \text{ ise} \quad u \notin M_k \text{ olmak üzere} \quad j = \arg\max\left\{(\tau_{iu})(1/d_{iu})^{\beta}\right\} \\ &q > q_0 \text{ ise} \quad p_{ij} = \begin{cases} \dfrac{[\tau_{ij}]\,[1/d_{ij}]^{\beta}}{\sum_{u \notin M_k}[\tau_{iu}]\,[1/d_{iu}]^{\beta}} & u \notin M_k \text{ ise} \\ 0 & \text{aksi takdirde} \end{cases} \end{aligned} \tag{5.21}$$

Yukarıdaki formülde u ifadesi M_k rotalanmış noktalar kümesi içinde yer almayan bir noktayı, τ_{ij} ise noktalar arasındaki feramon miktarını göstermektedir. Ayrıca q_0, bir sabit, q, [0, 1] aralığında rasgele belirlenen bir değer ve β görünebilirlik katsayısıdır. Formüle göre karınca gideceği bir sonraki nokta olarak, belirli bir olasılıkla en büyük $(\tau_{iu})(1/d_{iu})^{\beta}$ değerini veren noktayı seçecektir. Bunun yanında karınca gideceği diğer noktayı seçerken bulunduğu noktanın belirli sayıda yakın olan komşu noktalarını dikkate almaktadır. Geliştirilen algoritmada karıncalar tarafından oluşturulan rotalar 2-opt lokal arama sezgiseli iyileştirilmekte olup feramonların güncellenmesi ise klasik KKO algoritmasındaki gibi yapılmaktadır.

Geliştirilen yöntem, buharlaşma oranı ρ = 0,9, görünürlük katsayısı β = 2,3, yol seçimi olasılığı için sabit parametre q_0 = 0,9 ve her kolonide yer alan karınca sayısı m = 25 alınarak 3 CMT problemi için denenmiş ve algoritma 5000 iterasyon sayısı ile çalıştırılmıştır. Uygulamada çoklu KKO ile daha iyi sonuçlar elde edilse de özellikle büyük problemlerde bulunan değerler TA algoritmaları kadar iyi kalitede değildir (Bell ve McMullen, 2004).

Gambardella ve arkadaşlarının 1999 yılında geliştirdiği KKO yönteminde ise karıncalar belirli bir hiyerarşiye uygun davranarak ARP için çözüm üretmektedirler. Yöntemde ilk koloni problem için gerekli olan araç sayısını optimum yapmaya, başka bir koloni ise toplam gidilen mesafeyi minimize etmeye çalışarak ilk koloni ile koordineli bir şekilde çalışmaktadır. Koloniler arasındaki işbirliği ise feramonların güncellenmesi sırasında gerçekleşmektedir. Şekil 5.18'de algoritmanın temellerini oluşturan hiyerarşik yapı gösterilmektedir (Gambardella vd., 1999):



Şekil 5.18 Çoklu Karınca Kolonisi algoritması hiyerarşi yapısı

Çoklu KKO algoritması işleyiş adımları aşağıda açıklanmaktadır:

Adım 1: En Yakın Komşu sezgiseli ile sınırsız sayıda araç kullanarak uygun bir başlangıç çözümü üret ve ilk uygun araç sayısını bul.

Adım 2: Araç sayısını minimize etmeye çalışan ilk koloniyi aktif hale getir. Koloniyi mevcut araç sayısının bir eksiğini ele alarak çalıştır.

Adım 3: İkinci koloniyi toplam mesafeyi optimum yapmak üzere çalıştır.

Adım 4: Birinci ve ikinci koloniden mevcut sonuçlardan daha iyi bir sonuç gelinceye kadar bekle. Eğer araç sayısı için çalıştırılan birinci kolonide daha iyi bir sonuç bulunursa, iki koloniyi de aynı anda durdur. Bulunan çözümü yer değiştirme ve ekleme lokal arama prosedürlerine sokarak iyileştirmeye çalış.

Adım 5: Eğer durdurma kriteri sağlıyorsa bulunan sonucu kullanıcıya ver, aksi takdirde yeni araç sayısını dikkate alarak adım 2'ye dön.

Geliştirilen yöntemde feramon güncellemesi klasik KKO algoritmasındaki gibi yapılmakta olup koloniler paralel, diğer bir deyişle aynı anda çalıştırılmakta ve yollardaki feramonlar iki koloni için aynı anda değiştirilmektedir. Problemde kat edilen yolun yanında araç sayısı da

minimize edilmeye uğraşılmakta ve ikinci koloni daha çok birinci koloninin bulduğu sonucu iyileştirmek üzere çalıştırılmaktadır. Geliştirilen algoritma çoğu ARP türleri için uygulanabilir durumdadır, fakat yazarlar algoritmayı ZPARP üzerinde uygulamışlar ve başarılı sonuçlar elde etmişlerdir (Gambardella vd., 1999).

5.4.5 Mazzeo ve Loisseau'nun Karınca Kolonisi Optimizasyon Algoritması

Mazzeo ve Loisseau'nun 2004 senesinde yaptığı çalışmada ARP için farklı tiplerde KKO algoritmaları geliştirilmiş ve yöntemler problemler üzerinde uygulanarak karşılaştırılmıştır. Yazarlara göre KKARP için geliştirilen KKO algoritmalarının temel karakteristikleri aşağıda açıklanmaktadır (Mazzeo ve Loisseau, 2004):

- **Rotaların Oluşturması:** Algoritmalarda genel olarak her iterasyonda uygun bir çözüm oluşturulmaktadır. Yazarlara göre bu işlem iki türlü yapılabilmektedir:
  - **Sıralı:** Bir karınca aynı anda sadece bir araç için rota oluşturabiliyor ve kapasite kısıtı ihlal edildiğinde karınca başka bir araç için rota oluşturmaya başlıyor. Ayrıca her karınca farklı bir noktadan algoritmaya giriyor.
  - **Paralel:** Bir karınca aynı anda tüm araçlar için rotaları paralel olarak oluşturuyor. Tüm çözüm işlemi karıncanın bir noktaya yerleştirilmesi ile başlıyor.
- **Karıncaların Güzergahını Belirlemesi:** Karıncalar gideceği bir sonraki noktayı iki kurala göre belirleyebilmektedir:
  - **Rasgele Olasılıksal Kural:** Bu kurala göre bir karınca gideceği diğer noktayı 5.14 nolu formüle göre seçerek, güzergahını belirlemektedir.
  - **Pseudo Rasgele Olasılıksal Kural:** Bu kuralda i noktasında bulunan bir karınca gideceği diğer bir nokta olan j noktasını seçerken 0 ile 1 arasında bir rasgele sayı q üretilir ve bu sayı q_0 sabit sayısı ile karşılaştırılarak en iyi $[\tau_{iu}]^{\alpha}[1/d_{iu}]^{\alpha}$'a sahip nokta veya rasgele bir nokta belirlenir.

$$j = \begin{cases} \max_{u \in N_i} [\tau_{iu}]^{\alpha}[1/d_{iu}]^{\alpha} & q \le q_0 \text{ ise} \\ J & \text{aksi takdirde} \end{cases} \qquad (5.22)$$

5.22 nolu formülde N_i, i noktasından sonra gidilebilmesi mümkün noktalar kümesini, j ise rasgele seçilen bir noktayı belirtmektedir.

- **Feramon Miktarlarının Düzenlenmesi:** Yapılan çalışmada aşağıdaki alternatifler denenmektedir:
  - **Global Ayarlama:** Algoritmada iterasyonlardan sonra üç türlü yapılabilmektedir:
    - Her iterasyon sonunda her karınca rotasını oluşturduktan sonra çözüm üretildiğinde feramonlar güncellenebilir.
    - Sadece algoritma süresince en iyi çözümleri üreten seçkin karıncalar dikkate alınarak yollardaki feramon miktarı 5.15 nolu formüle göre belirlenebilir.
    - Feramon miktarı iterasyon içinde en iyi çözümü üreten karıncaya göre güncellenebilir.
  - **Lokal Ayarlama:** Bir karıncanın bir noktadan diğerine her gidişinde feramonlar aşağıdaki formüle göre lokal olarak düzenlenmektedir:

$$\tau(i,j) = \rho.\tau(i,j) + (1-\rho).\Delta\tau(i,j) \tag{5.23}$$

    Bu formülde yer alan $\Delta\tau(i,j)$ ifadesi ise aşağıdaki yöntemlerden biriyle belirlenir:
    - $(0 \le \gamma \le 1)$ olması koşulu ile $\Delta\tau(i,j) = \gamma.\max_{z\in\Gamma(j)} \tau(j,z)$ formülü kullanılarak ilk feramon miktarları ayarlanır.
    - $\Delta\tau(i,j) = \tau_0$ ile feramonlara başlangıç değeri atanır.
    - $\Delta\tau(i,j) = \tau_0 / d_{ij}$ ile feramonlara uzaklıkla ters orantılı olarak değer atanır.
    - $\Delta\tau(i,j) = 0$ alınarak sadece buharlaştırma uygulanarak güncelleme yapılabilir.
- **Azaltılmış Komşu Listeleri:** Bir karınca gideceği noktayı belirlerken tüm noktalar dikkat alınmaz. sadece bulunduğu noktaya belirli yakınlıkta olan noktalar değerlendirilmektedir. Böylece arama uzayı belirli bir oranda daraltılır.
- **Geliştirilmiş Sezgiseller:** Algoritma boyunca iterasyonların sonunda bulunan her çözümü iyileştirmek için, 2-opt, yer değiştirme ve ekleme yöntemleri gibi klasik sezgisel yöntemler rotalara ve tüm probleme uygulanabilir.
- **Durdurma Kuralı:** KKO algoritması, n-max sayıdaki iterasyon sonucunda mevcut çözümde bir iyileşme sağlanamazsa veya daha önce belirlenen bir iterasyon sayısına erişilmişse durdurulabilir.

Geliştirilen yöntemler, literatürde yer alan farklı tiplerdeki ARP üzerinde uygulanmış ve küçük boyutlu problemlerde oldukça iyi sonuçlar vermiştir. Fakat büyük boyutlu problemlerde diğer metasezgisel yöntemlerle rekabet edecek çözümler üretilememektedir. Yazarlar farklı özellikteki KKO algoritmalarını çözüm kalitesi ve hesaplama hızı yönünden karşılaştırarak ARP için en iyi KKO yöntemi özellikleri için aşağıdaki değerlendirmeyi yapmışlardır (Mazzeo ve Loisseau, 2004):

- Rotaları paralel olarak oluşturan KKO algoritması, sıralı olarak oluşturan yöntemlerden daha iyi sonuç vermektedir.
- Feramon güncellemesini en iyi çözümle karşılaşınca yapan algoritmalar, diğer global düzenlemelere göre daha iyi çözüm üretmektedir.
- Feramonların lokal olarak ayarlanması, çözümü iyileştirmemektedir.
- Geliştirilmiş sezgiseller uygulanarak KKO algoritması ile daha iyi çözümler elde edilmektedir.
- n/4 oranında uygulanan azaltılmış komşu listeleri ile sonuçlar daha hızlı bir şekilde bulunabilmektedir.

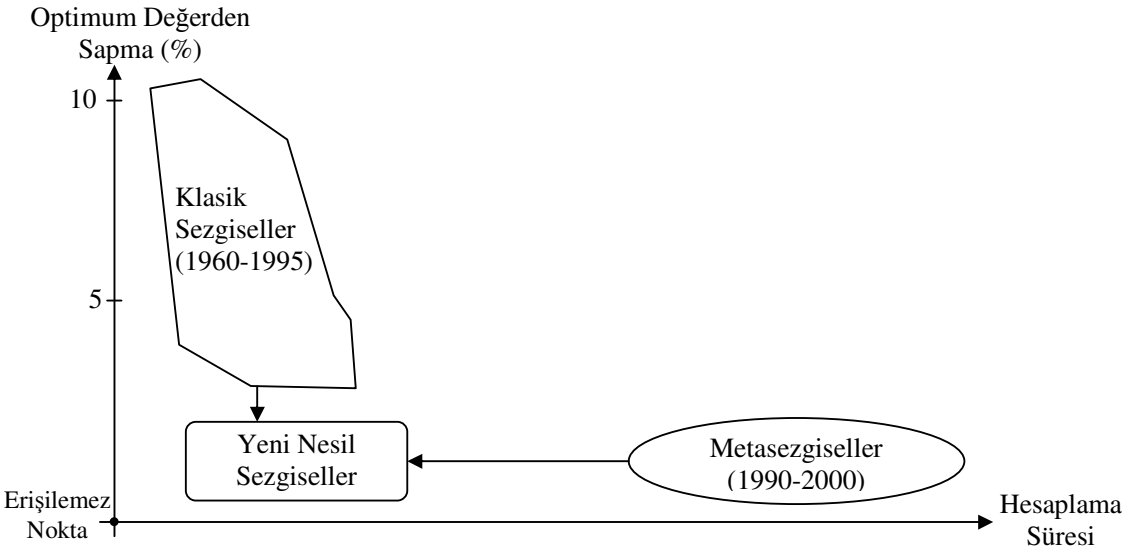
5.5 Araç Rotalama Problemleri İçin Metasezgisel Yöntemlerin Karşılaştırılması

Metasezgisel algoritmalar müşteri sayısı birkaç yüz adete kadar olan ARP için makul sürelerde yüksek kaliteli optimuma yakın sonuçlar üretmektedir. Klasik sezgisel yöntemlere kıyasla çözüm süresi biraz daha uzun sürmesine rağmen, metasezgisel yöntemlerle üretilen çözümler tatmin edici düzeyde olup, metasezgisel yöntemler arasında TA algoritmaları diğer yöntemlere göre hesaplama süresi ve çözüm kalitesi açısından çoğunlukla daha iyi sonuçlar vermektedir. Şekil 5.19'da ARP için geliştirilen sezgisel ve metasezgisel yöntemlerin çözüm kaliteleri ve hesaplama süreleri arasındaki ilişki gösterilmektedir. Algoritmaların daha etkili hale getirilmesi, daha kısa sürelerde üretilen iyi kalitede çözümler ile gerçekleştirilecektir. Literatürde ARP için GA ve KKO ile araştırmalar tam olarak tamamlanmadığı için gelecekte büyük bir ihtimalle melez GA ve KKO yöntemleri TA algoritmasının performansını yakalayacaktır (Toth ve Vigo, 2002).

ARP için özellikle Rochat ve Taillard'ın 1995 yılında geliştirdiği Adaptif Hafıza tabanlı yöntem tüm CMT problemleri için en iyi bilinen sonuçları üretmiştir. Tabu Rota ve Taillard'ın TA algoritmaları ise diğer iyi kalitede çözüm üreten yöntemlerdir. Gerçek

hayattaki uygulamalar için en uygun metot ise Toth ve Vigo'nun 1998 yılında geliştirdiği Tanecikli TA yöntemidir. Bunların yanında Doerner ve arkadaşlarının geliştirdiği melez KKO ile Baker ve Ayechew'in 2003 yılında, Prins'in 2004 yılında geliştirdiği melez GA yöntemleri de iyi çözümler vaat eden tekniklerdir.

Gerçek hayatta metasezgisel yöntemlerin lojistik sistemlerine entegre edilebilmesi için yöntemlerin esnek hale getirilmesi ve hesaplamaların kısa sürelerde yapılması gerekmektedir. Bu tekniklerde hesaplama süresini ise süreçten bağımsız olarak algoritmanın tasarımı, veri yapıları, seçilen programlama dili ve bilgisayar donanımı gibi faktörler oldukça etkilemekte ve sadece çözüm kalitesine göre karşılaştırma yapmayı zorlaştırmaktadır (Tarantilis vd., 2004a).



Şekil 5.19 ARP için geliştirilen sezgisel yöntemler

Metasezgisel yöntemlerin çeşitli kriterler üzerinden değerlendirilmesinde karşılaşılan bazı zorluklar ve konu ile ilgili literatürde yer alan çalışmalarda bulunması gereken özellikler aşağıda açıklanmaktadır (Laporte vd., 2000):

- Algoritmalar genellikle birçok parametre ile işletildiğinden optimuma yakın sonuç almak için problemin büyüklüğüne, noktaların lokasyonuna göre farklı ayarlamalar yapmak gerekebilmektedir.
- Literatürde yer alan çalışmalarda farklı parametre değerlerine göre alınan sonuçlar da verilmelidir, bu sayede parametrenin çözüm kalitesi üzerindeki etkisi anlaşılabilir.

- Paralel işlem yapan algoritmalarda hesaplama süresinin dolayısıyla algoritmanın hızının yorumlanması zordur.
- Yapılan her çalışmada geliştirilen algoritmanın CMT problemleri gibi ortak bulunan problemlerdeki çözüm değerleri ve hesaplama süreleri verilmelidir.

6. ARAÇ ROTALAMA PROBLEMİ İÇİN BİR METASEZGİSEL YÖNTEM

Bu bölümde daha önce açıklanan metasezgisel algoritmalardan esinlenerek geliştirilen populasyon ve komşuluk tabanlı bir algoritma tanıtılmaktadır. Geliştirilen yöntem tam olarak bir metasezgisel algoritma özelliği göstermekte olup, sadece GA veya TB gibi bir tip algoritma içerisinde yer aldığı söylenemez. Bu bölümde ilk olarak yöntemin temel yapısı ve kavramlar açıklanmakta, daha sonra algoritma süreci belirtilerek literatürde yer alan ARP örnekleri için uygulama sonuçları irdelenmektedir.

6.1 Algoritmanın Temel Yapısı

Önerilen algoritmanın en önemli özelliği populasyon tabanlı tekniklerden yola çıkılarak, çözümlerin komşuluk mantığı ile oluşturulmasıdır. Geliştirilen yöntemde GA'da olduğu gibi iki çözüm ele alınmakta fakat yeni çözümler, bu iki çözüm arasında kalan çözümlerin ekleme ve λ-yer değiştirme metodu ile taranması ile elde edilmektedir. Rotalar üzerinde yer değişimi yapılacak noktalar ise seçilen "iki çözüm arasında uzaklık" kavramı ile belirtilen bir nokta kümesi içinde bulunmaktadır. Bu yöntem geliştirilirken metasezgisel algoritmaların aşağıdaki özellikleri göz önüne alınmıştır:

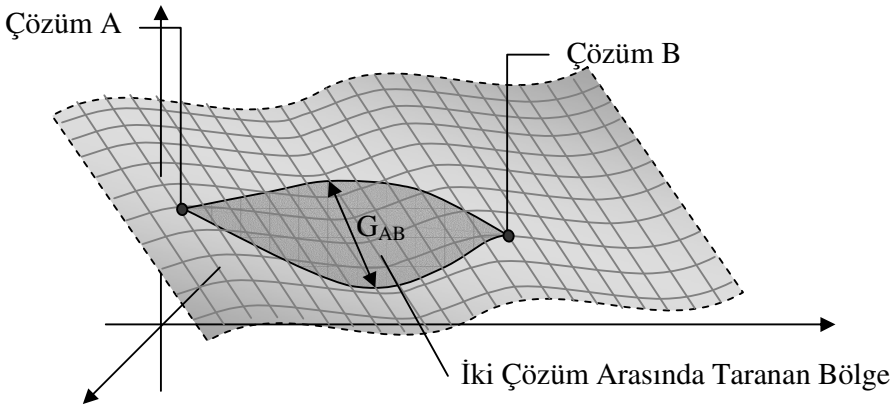
- Klasik sezgisel yöntemlerle oluşturulan başlangıç çözümleriyle metasezgisel algoritmaların kısa sürelerde oldukça iyi çözümler üretmesi.
- GA ve KKO gibi populasyon tabanlı algoritmaların çözüm uzayında çeşitli noktalardan hareket ederek çeşitlendirme stratejisini etkin bir şekilde kullanması fakat en iyi bilinen çözümlere uzak, diğer bir deyişle kesin olmayan çözümler oluşturması.
- Populasyon tabanlı algoritmalarda başlangıç çözümleri, arama uzayının belirli bölgelerinde toplanmayıp arama uzayında homojen bir şekilde dağılması ile yöntemin performansının yükselmesi.
- TA ve TB gibi komşuluk tabanlı algoritmaların yoğunlaştırma stratejisi ile çok iyi çözümler üretmesi fakat büyük boyutlu problemlerde çeşitlendirme stratejisini başarılı bir şekilde uygulayamamaları.
- Az sayıda parametre içeren metasezgisel algoritmaların daha fazla kullanılabilir ve esnek olması.

Yöntemde yukarıdaki maddelerde belirtilen metasezgisel algoritmaların iyi yönleri birleştirilerek kapasite ve mesafe kısıtlı ARP için çözüm üreten ve stokastik arama

yöntemlerine dayanan bir algoritma geliştirilmiştir. Algoritma mantığı her türden ARP'ye uygulanabilir olup, gerekli düzenlemelerin yapılması halinde diğer kombinatoryal problemlerde de denenebilir esnekliktedir. Geliştirilen metodun dayandığı temel nokta, arama uzayında aralarında yeterince uzaklık bulunan iki iyi çözüm örneği arasında bu iki çözümden daha iyi bir çözüm yer alabileceği varsayımıdır. Bu varsayımdan yola çıkarak ortaya atılan "çözümler arası uzaklık" kavramı ise daha çok komşuluk yapısı ile ilgili olup problemden probleme farklılık gösterebilir.

6.2 Çözüm Uzayında İki Çözüm Arasındaki Uzaklık Kavramı

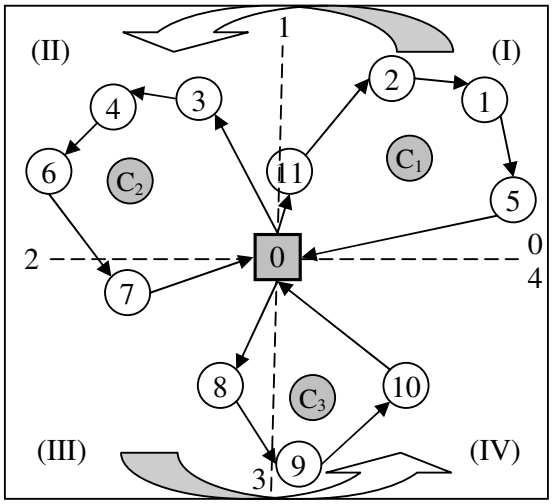
Karar değişkenleri permutasyon bazlı olan kombinatoryal optimizasyon problemleri için uygun şekilde tanımlanan komşuluk yapısı ile iki çözüm arasında kalan alan, şekil 6.1'de gösterildiği gibi bir G_{AB} genişlik değeri ile belirtilebilir. Bu tip problemlerde çözümler arasındaki uzaklık ne kadar fazla olursa, G_{AB} değeri de bununla orantılı olarak büyüyecektir.



Şekil 6.1 Çözüm uzayı ve arama genişliği

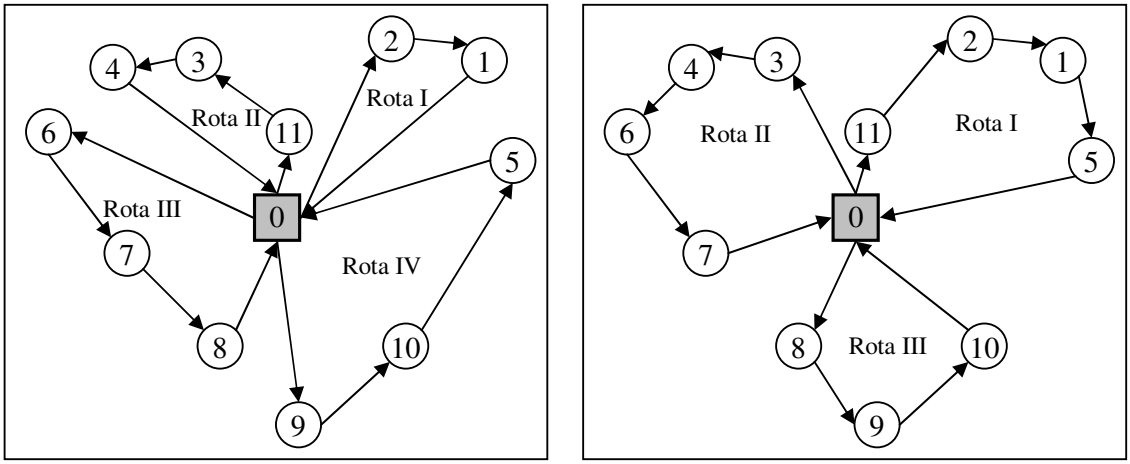
Karar değişkenleri permutasyon bazlı bir problem olan ARP için iki çözüm örneği arasındaki uzaklık, rotalarda diğer bir deyişle nokta gruplarında farklı olan noktalar ile belirlenir. Bunu yapabilmek için iki çözüm arasında rotaların eşleştirilmesi diğer bir deyişle bir çözümde yer alan bir rotanın diğerinde hangi rotaya denk düştüğünün bulunması gereklidir. Bu çalışmada, rotaların eşleştirilmesi işlemi, rotaların merkez noktalarının polar değerlerini hesaplayıp rotaları sıralayarak yapılmaktadır. Bir rotanın merkez noktası ise rota üzerinde bulunan tüm noktaların koordinatlarının aritmetik ortalaması alınarak bulunmaktadır. Bu teknik ile rotalara, merkez noktayı depo ile birleştiren doğrunun x ekseni ile yaptığı açının sinüs değeri kullanılarak noktanın bulunduğu bölgeye göre [0, 4) aralığında bir değer verilmektedir.

Böylece şekil 6.2'de gösterildiği gibi bir çözüm için rotaların, saat yönünün tersine sıralaması yapılmaktadır. Şekle göre C_1, C_2 ve C_3 rotaların merkez noktalarını göstermekte ve merkez noktası I. bölgede bulunan rotanın polar değeri (0, 1) aralığında, II. bölgedeki rotanın polar değeri (1, 2) aralığında ve son rotanın polar değeri (3, 4) aralığında olmaktadır. Buna göre çözüm, ilk olarak birinci rota, daha sonra ikinci ve üçüncü rota olarak kodlanacaktır. Bu sayede bu çözümdeki rotalar başka bir çözümdeki rotalarla eşleştirilirken maksimum tutarlılık sağlanmış olur. İki ARP çözümü arasındaki uzaklık miktarı ise eşleştirilen bu rota gruplarındaki farklı olan noktaların sayısı ile belirlenmektedir. Farklı noktaların çok olması, çözüm uzayında iki çözüm arasındaki uzaklığın büyük olması anlamına gelmektedir.



Şekil 6.2 Rotaların polar değerlerinin belirlenmesi

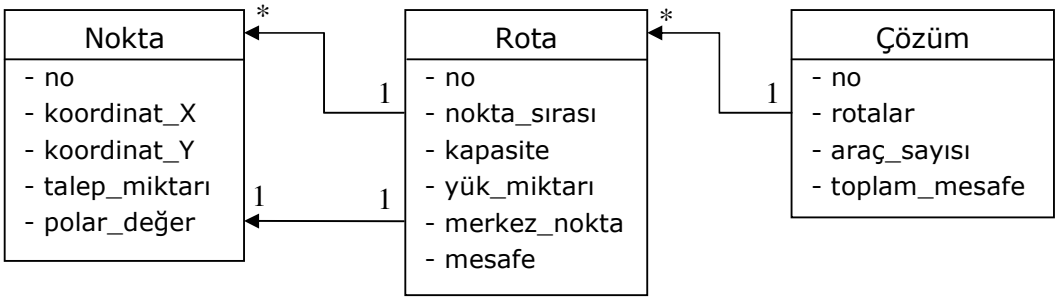
Araç sayılarının farklı olduğu ARP çözümlerinde rota eşleştirmesi yapılırken araç sayısı fazla olan çözümün bir veya birden fazla rotası diğer çözümdeki rotalar ile denk düşmeyebilmektedir. Böyle durumlarda seçilen iki çözüm arasında uzaklık miktarı hesaplanırken, fazla olan bu rotaların diğer çözümdeki en büyük polar değere sahip rota ile eşleştirilmesi yapılmakta ve bu sebeple araç sayısı az olan çözümdeki en büyük nolu rota, diğer çözümdeki birden fazla rota ile eş tutulmaktadır. Aynı ARP için şekil 6.3'de gösterilen araç sayısı farklı olan çözüm A ve çözüm B arasındaki uzaklık bulunurken rotalardaki farklı noktalar şu şekilde bulunmaktadır. Rota I için farklı noktalar; 11, 5, rota II için farklı noktalar; 11, 6, 7, rota 3 için; 6, 7, 9, 10 ve rota 4 için; 8, 5 olmak üzere, rotalardaki tüm farklı noktalar 5, 6, 7, 8, 9, 10, 11'dir. Böylece çözüm A ve B arasındaki uzaklık miktarı, farklı noktaların sayısı olan 7 değeridir.



Şekil 6.3. Araç sayısı farklı olan iki çözüm

6.3 Çözümlerin Kodlanması

Geliştirilen yöntemde çözümler için GVR tekniğine benzer bir kodlama tekniği kullanılmıştır. ARP çözümlerinin nokta, rota ve çözüm olmak üzere üç kademeli olarak gösterimi yapılmaktadır. Şekil 6.4'de de gösterildiği gibi her aşamada, aşamaya ait özel bilgiler tutulmaktadır.



Şekil 6.4 Çözüm, rota ve nokta ilişki diyagramı

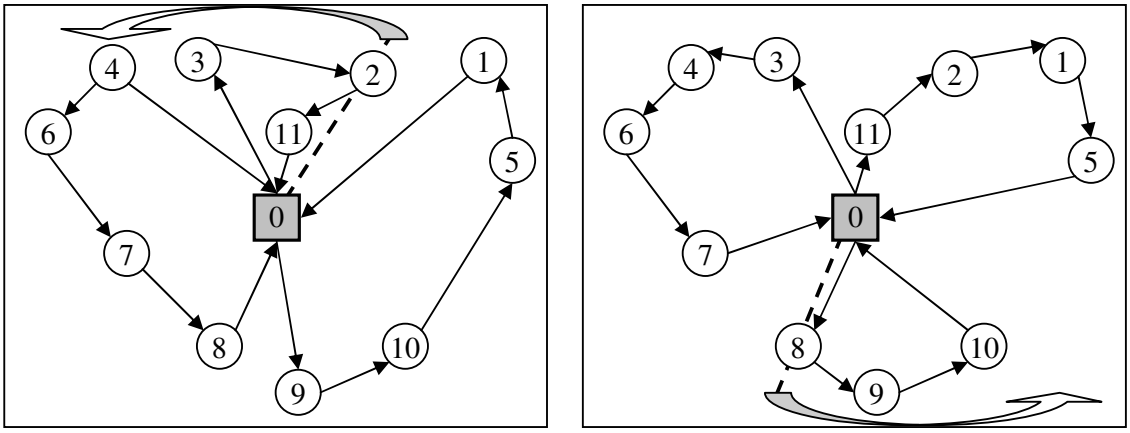
Bir çözüm, içerdiği "rotalar" özelliği ile birden fazla "Rota" nesnesini, rotaların polar değerlerine göre yapılan sıralamada referans etmekte ve bu rota nesnelerinde "nokta_sırası" özelliği ile ziyaret edilme sırasında "Nokta" nesneleri tutulmaktadır. Ayrıca rotalarda yer alan "merkez_nokta" özelliği de bir "Nokta" nesnesini referans etmektedir. Çizelge 6.1'de bu aşamalara ait tüm özelliklerin açıklaması verilmektedir.

Çizelge 6.1 Çözüm kodlamada aşamalara ait özellik açıklamaları

Aşama	Özellik	Açıklama
Nokta	no	Problem dosyasında verildiği şekilde nokta numarası bilgisini tutar. Merkez deponun nokta nosu çoğunlukla 0'dır.
	koordinat_X	Noktanın x eksenindeki koordinat değerini verir.
	koordinat_Y	Noktanın y eksenindeki koordinat değerini verir.
	talep_miktarı	Müşteri noktaları için müşteriye ait talep miktarı bilgisini tutar.
	polar_değer	Noktayı merkez depoyla birleştiren doğrunun polar değerini içerir.
Rota	no	Bir çözümde yer alan rotanın sırasıdır. Merkez noktasının polar değeri en küçük olan rota, çözüm içerisinde ilk sırada yer alır ve no bilgisinde 0 değeri bulunur.
	nokta_sırası	Rotada ziyaret edilecek noktalar sıra ile bu özellik içinde tutulur. Burada merkez depo noktasına yer verilmez çünkü her rotanın merkez depo ile başlayacağı ve biteceği varsayılmıştır.
	kapasite	Rota için araç taşıma kapasitesi bilgisi tutulur. KKARP için bu bilgi çoğunlukla tüm rotalar için aynıdır.
	yük_miktarı	Bu özellik rotaya atanan araca yüklenecek toplam yükün miktarını verir. Diğer bir deyişle "nokta_sırası" özelliğindeki noktaların toplam talep miktarıdır.
	merkez_nokta	Rotadaki tüm noktaların koordinat değerlerinin aritmetik ortalaması ile bulunan rotanın orta noktasıdır. Bu noktanın polar değeri, rotanın polar değerini verir.
	mesafe	Rotaya atanan aracın merkez depodan çıkıp, merkez depoya dönünceye kadar kat edeceği toplam mesafe bilgisini tutar.
Çözüm	no	Populasyonda çözümün hangi sırada yer aldığı bilgisidir. Populasyondan çözüm no bilgisi referansıyla çözüm seçilmektedir.
	rotalar	Çözümde yer alan rotalar polar değerlerine göre yapılan sıralamada bu özellik içersinde tutulmaktadır.
	araç_sayısı	Çözümde kullanılan araç sayısı (rota sayısı) bilgisini içerir.
	toplam_mesafe	Çözümde kat edilen toplam mesafeyi verir, başka bir deyişle çözümün amaç fonksiyonu değeridir.

6.4 Başlangıç Çözümlerinin Oluşturulması

Metasezgisel yöntemler çoğunlukla rasgele olarak veya klasik sezgisel yöntemlerle oluşturulan başlangıç çözümlerini baz alarak çözüm üretmeye başlamaktadır. Özellikle populasyon tabanlı metasezgisel yöntemlerde başlangıç populasyonu, algoritma performansını ve çözüm kalitesini etkileyen önemli faktörlerden birini teşkil etmektedir. Populasyondaki çözümlerin birbirinden farklı olması ve çözüm uzayında belirli bir alanda toplanmayıp umut vaat eden her bölgeye orantılı olarak dağılması, algoritmanın çeşitlendirme stratejisini en iyi şekilde uygulamasını sağlamaktadır. Böylece çözüm uzayının daha geniş bir bölümünde hızlı bir şekilde arama gerçekleştirilebilmekte ve iyi çözümlerin bulunma olasılığı artırılmaktadır. Bu ifadeler dikkate alınarak ARP için geliştirilen metasezgisel yöntemde başlangıç populasyonu dört klasik sezgisel metot ile oluşturulmuştur. Bu tekniklerin her uygulanışında değişik çözümlerin üretilebilmesi amacıyla tekniklerde bazı stokastik düzenlemeler yapılmıştır. Başlangıç çözümü üretmek için kullanılan ilk yöntem olan Süpürme yönteminde ilk olarak sabit bir nokta ele alınmayıp, rasgele bir nokta seçilmekte ve bu noktadan başlayıp saat yönünün tersine dönülerek rotalar oluşturulmaktadır. Şekil 6.5'de aynı ARP için rasgele olarak seçilen 2 ve 8 nolu noktalar ile oluşturulan iki çözüm gösterilmektedir.



Şekil 6.5 Süpürme yöntemi ile oluşturulan iki çözüm

En Yakın Nokta yöntemi (EYK), başlangıç populasyonunu üretmek için kullanılan başka bir yöntemdir. Bu klasik sezgisel yöntemin her kullanıldığında farklı çözümler üretmesini sağlamak için seçilen ilk nokta rasgele olarak belirlenmekte ve oluşturulan rotalar 2-opt sezgiseli ile iyileştirilmektedir.

Başlangıç çözümlerini üretmek için kullanılan diğer yöntemler ise Paralel ve Sıralı Tasarruf yöntemleridir. Bu tekniklerde nokta çiftleri ($s_{ij} = d_{i0} + d_{j0} - d_{ij}$) tasarruf değerlerine göre büyükten küçüğe sıralanmakta ve bu sıraya göre uygun olarak birleştirilerek güzergahlar

belirlenmektedir. Tasarruf yöntemi ile farklı çözümler elde etmek için sıralamada en üstte yer alan nokta çiftinden başlanmamakta, en üstteki n ikiliden rasgele bir nokta çifti seçilerek rotalar oluşturulmaktadır. Nokta çiftinin sıralamada en üstte yer alan n adet ikiliden seçilmesinin sebebi daha yüksek tasarruf değerleri ile çözümlerin daha kaliteli olmasının istenmesidir. Bu sayede, iyi kalitede ve çözüm uzayında orantılı olarak yayılmış başlangıç çözümlerinin üretilmesi mümkün kılınmıştır. Geliştirilen yöntemde başlangıç populasyonunun Süpürme, En Yakın Nokta ve Tasarruf tekniklerinin hangisi ile oluşturulacağı ise algoritma süresince rasgele olarak belirlenmektedir. Ayrıca başlangıç populasyonu olasılıksal olarak oluşturulduğu için, aynı amaç fonksiyonu değerine bireyler populasyon içersinde bulunabilmektedir. Çözüm geliştirme sırasında aşamasında ise üretilen çözümün, çözüm havuzuna atanabilmesi için çözüm havuzunda o çözümden bulunmaması gerekmektedir.

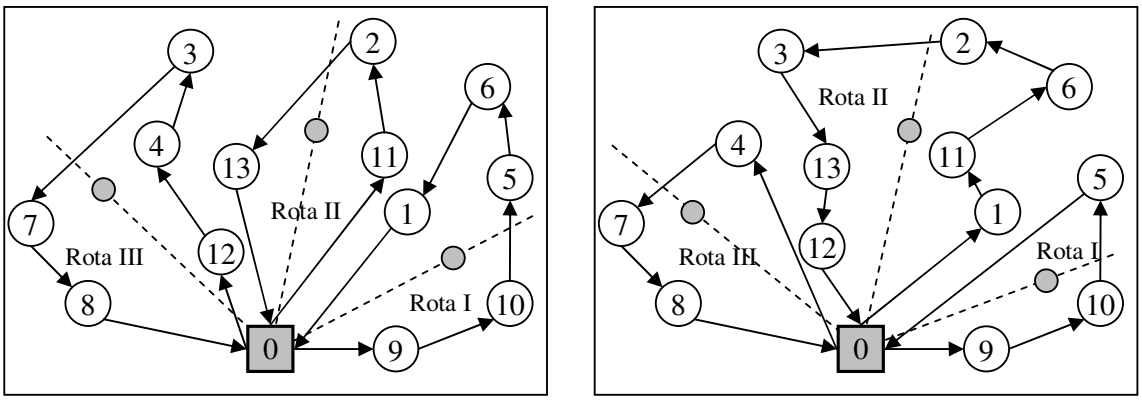
6.5 Populasyondan Bireylerin Seçilme İşlemi

Populasyon tabanlı algoritmalarda genel olarak bir havuz içerisinden seçilen çözümlerle yeni çözümler geliştirilir. Yeni oluşturulan çözüm, populasyondan seçilen ebeveyn çözümlerin çoğu özelliğini içerdiğinden dolayı seçim yöntemi algoritmanın ürettiği sonuçların kalitesini önemli ölçüde etkiler. ARP için geliştirilen metasezgisel yöntemde seçim yöntemi olarak İkili Turnuva tekniği kullanılmıştır. Bu teknikte ilk olarak populasyondan rasgele olarak farklı iki çözüm seçilmekte, daha sonra iki çözüm karşılaştırılarak hangisi daha iyi bir amaç fonksiyonu değerine sahipse o çözüm, yeni çözümü geliştirmek üzere ele alınmaktadır. Geliştirilen populasyon ve komşuluk tabanlı yöntemde, seçilecek iki çözümden ilki bu prosedür ile seçilmektedir. Diğer çözüm ise buna benzer, fakat amaç fonksiyon değeri göz önüne alınmayıp seçilen ilk çözüme uzaklık değerleri dikkate alınarak elde edilmektedir. Diğer bir deyişle çözüm havuzundan rasgele olarak iki çözüm seçilmekte ve iki çözümden önceki seçilen çözüme en uzak çözüm, yeni çözüm oluşturulmak üzere elde tutulmaktadır. Böylece iki uzak çözüm arasında daha fazla seçenek taranabilmekte ve iyi kalitedeki çözümlerin bulunma olasılığı yükseltilmektedir.

6.6 İki Çözüm Arasındaki Çözümlerin Taranması

TA ve TB gibi komşuluk tabanlı metasezgisel algoritmaların en önemli özelliği, belirli bir bölgede yoğun bir şekilde arama yaparak o bölge için lokal optimum noktayı kısa sürede açığa çıkarmasıdır. Bu sebepten dolayı bu yöntemler çözüm uzayı geniş olmayan küçük

boyutlu problemlerde hızlı ve başarılı sonuçlar vermekte, büyük problemlerde ise ancak belirli bir bölgedeki en iyi çözümü bulabilmektedir. Bu bilgilerden yola çıkarak ARP için hazırlanan yöntemde populasyon havuzundan seçilen çözümlerden yeni çözümler üretilmesi, komşuluk tabanlı algoritmalara benzer olarak yapılmıştır. Seçilen iki çözüm arasındaki çözümler komşuluk mantığı ile rasgele bir şekilde taranmakta ve ebeveyn çözümden daha iyi kalitede bulunan ilk uygun çözüm yavru çözüm olarak ele alınmaktadır. İki çözüm arasındaki çözümler ise rotalar arasındaki farklı noktaların, rotalar arasında yer değiştirilmesi ile elde edilmektedir. Algoritma sürecini hızlandırmak ve iki çözüm arasındaki bölgede stokastik arama işlemini daha da etkinleştirmek için bu yer değiştirme işlemi belirli bir kurala göre yapılmaktadır. Bu kurala göre bir rotaya eklenebilecek iki farklı nokta kümesi bulunmakta ve bu küme noktanın diğer çözümdeki pozisyonuna göre saptanmaktadır. Şekil 6.6'da gösterilen çözümlerde koyu renk daireler rotaların merkez noktasını ve kesikli çizgiler ise rotalarda yer alan noktaların sağda veya solda olma durumunu belirtmektedir. Populasyondan seçilen iki birey, çözüm A ve B olması halinde çözüm A'da yer alan II nolu rota için yer değişimi yapılabilecek farklı noktalar şu şekilde belirlenmektedir: İlk olarak çözüm A'da rota II için farklı olan 1, 6, 3 ve 2 noktaları saptanmakta ve bu noktalar çözüm B'deki II nolu rotada merkez noktanın solunda ve sağında 3, 12 ve 1, 6 olarak bulunmaktadır. Buna göre, çözüm A'da yer alan II nolu rotanın farklı nokta kümelerinden ilki {3, 12} diğeri ise {1, 6} olmaktadır. Bu kümeler, rota II'ye bir nokta eklenmesi halinde hangi noktanın hangi rotadan eklenebileceğini belirler. Ayrıca belirli bir yönde rotalar arası nokta yer değiştirme işlemleri sırasında süreci kolaylaştırmaktadır.



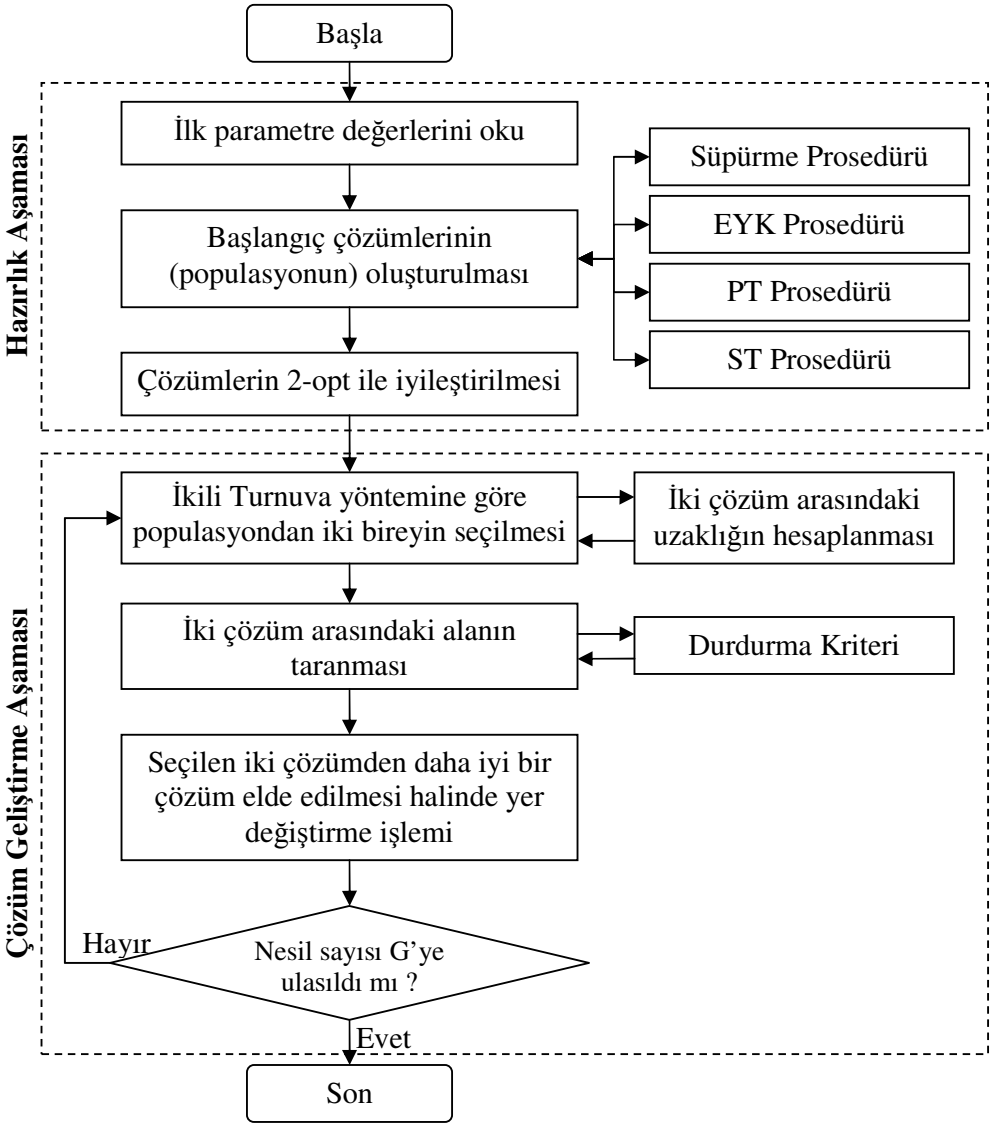
Çözüm A Çözüm B

Şekil 6.6 Rotalar için farklı noktaların bulunması

İki çözüm arasındaki alan taranırken ilk önce farklı nokta kümeleri boş olmayan rasgele bir rota seçilmekte ve bu rotaya kümeler içersinden bir nokta eklenmektedir. Eklenen nokta herhangi bir kısıtı ihlal ediyorsa, noktanın önceki rotasına rasgele bir nokta eklenerek yer değiştirme işlemi yapılmaktadır. Her ekleme ve çıkarma işleminden sonra kısıtlar kontrol edilmekte ve sürecin uzunluğu bir parametre tarafından belirlenmektedir. Bu parametre lokal arama prosesinin durdurma kriteridir. Eğer durdurma kriteri ile arama sonlandırılıp uygun bir çözüm bulunamaması halinde ebeveyn çözümler bırakılarak İkili Turnuva yöntemi ile yeni çözümler seçilmekte, arama yeni çözümler arasında devam ettirilmektedir. Arama sırasında oluşturulan uygun çözümlerin populasyondaki çözümlerle yer değiştirilmesi ise şu şekilde yapılmaktadır: Seçilen çözümler arasında daha iyi bir çözüm bulunduğunda ilk olarak bu çözümün populasyon içinde bulunup bulunmadığı kontrol edilmekte, eğer populasyonda yoksa, ebeveyn çözüm çıkartılarak populasyona yeni çözüm eklenmektedir. Böylelikle populasyonun güncel kalması sağlanmış olup tarama sırasında daha iyi çözümlerin bulunma olasılığı artırılmaktadır. Geliştirilen yöntemin işleyiş adımları detaylı olarak bölüm 6.7'de açıklanmaktadır.

6.7 Algoritmanın Akış Süreci

Metasezgisel algoritmaların süreçleri genellikle, hazırlık ve çözüm geliştirme olmak üzere iki aşamadan oluşmaktadır. Hazırlık aşamasında algoritmanın çözüm geliştirmesi için gerekli bazı koşullar yerine getirilmekte ve çözüm geliştirme aşamasında ise arama uzayında lokal optimum noktalara takılmadan çoğunlukla rasgele bir şekilde çözümler irdelenmektedir. Buna göre ilk parametre değerlerinin okunması, başlangıç çözümlerinin oluşturulması gibi işlemler hazırlık aşamasında, bir veya birden fazla çözümden populasyon veya komşuluk mantığı ile yeni çözümlerin üretilmesi ise çözüm geliştirme aşamasında yapılmaktadır. Metasezgisel algoritmaların ana süreçlerinin bulunduğu arama stratejileri, çözüm geliştirme safhasında bulunmaktadır. Bunun yanında hazırlık aşamasında, özellikle başlangıç çözümlerinin nasıl oluşturulduğu yöntemin performansını ve çözüm kalitesini büyük oranda etkilemektedir. Geliştirilen yöntemin bu hususlar dikkate alınarak hazırlanan akış şeması şekil 6.7'de gösterilmektedir.



Şekil 6.7 Algoritma akış şeması

Geliştirilen algoritmanın işleyiş adımları ise detaylı bir şekilde aşağıda açıklanmaktadır:

Adım 1: Algoritma parametrelerini oku ve klasik sezgisel yöntemlerle (Süpürme, En Yakın Nokta ve Tasarruf algoritmaları) N adet başlangıç çözümü (populasyon) oluştur.

1.1. Her noktanın polar değerini aşağıdaki 6.1 nolu formüle göre hesapla:

$$p_i = \begin{cases} |y_i - y_0| / d_{i0} & x_i \geq x_0 \text{ ve } y_i \geq y_0 \text{ ise} \\ 1 + |x_i - x_0| / d_{i0} & x_i < x_0 \text{ ve } y_i \geq y_0 \text{ ise} \\ 2 + |y_i - y_0| / d_{i0} & x_i < x_0 \text{ ve } y_i < y_0 \text{ ise} \\ 3 + |x_i - x_0| / d_{i0} & x_i \geq x_0 \text{ ve } y_i < y_0 \text{ ise} \end{cases} \quad (6.1)$$

1.2. Noktaları polar değerlerine göre büyükten küçüğe sırala.

1.3. Problemde yer alan dağıtım merkezi dışındaki tüm nokta ikililerinin tasarruf değerini $s_{ij} = d_{i0} + d_{j0} + d_{ij}$ formülü ile hesapla ve tasarruf değerlerine göre nokta çiftlerini büyükten küçüğe sırala.

1.4. Eğer çözüm Süpürme algoritması ile oluşturulacaksa aşağıdaki adımları çalıştır.

1.4.1. Rasgele bir nokta seçerek rota oluşturmaya başla. Dönülecek yönü rasgele belirleyerek rotaya diğer noktaları ekle.

1.4.2. Eğer nokta eklemeleri sırasında herhangi bir kısıt ihlal edilirse bir sonraki rotanın oluşturulmasına geç.

1.4.3. Yukarıdaki 1.4.1 ve 1.4.2 adımlarını problemde rotalanmayan nokta kalmayıncaya kadar sürdür ve uygun bir çözüm üret.

1.5. Eğer çözüm Sıralı Tasarruf Algoritması ile oluşturulacaksa aşağıdaki adımları çalıştır.

1.5.1. Tasarruf sıralamasında en üstte yer alan n nokta çiftinden rasgele olarak birini seç ve sıralamayı dikkate alıp diğer uygun nokta çiftleri ile birleştirerek rota oluştur.

1.5.2. Eğer nokta eklemeleri sırasında herhangi bir kısıt ihlal edilirse bir sonraki rotanın oluşturulmasına geç.

1.5.3. Yukarıdaki 1.5.1 ve 1.5.2 adımlarını problemde rotalanmayan nokta kalmayıncaya kadar sürdür ve uygun bir çözüm üret.

1.6. Eğer çözüm Paralel Tasarruf Algoritması ile oluşturulacaksa aşağıdaki adımları çalıştır.

1.6.1. Tasarruf sıralamasında en üstte yer alan n nokta çiftinden rasgele birini seç ve rota oluşturmaya başla.

1.6.2. Tasarruf sıralamasında bir sonraki nokta çiftini, oluşturulan rotalara mümkünse ekle. Eğer eklemek mümkün değilse nokta çiftinden yeni rota oluştur.

1.6.3. Yukarıdaki 1.6.1 ve 1.6.2 adımlarını problemde rotalanmayan nokta kalmayıncaya kadar sürdür ve uygun bir çözüm üret.

1.7. Eğer çözüm En Yakın Nokta Yöntemi ile oluşturulacaksa aşağıdaki adımları çalıştır.

1.7.1. Rasgele bir nokta seç ve bu noktayı eklenmesi mümkün en yakın nokta ile

birleştirerek rota oluşturmaya başla.

1.7.2. Eğer ekleme işlemi kısıtlara uymuyorsa, yeni rotanın oluşturmasına başla.

1.7.3. Yukarıdaki 1.7.1 ve 1.7.2 adımlarını problemde rotalanmayan nokta kalmayıncaya kadar sürdür ve uygun bir çözüm üret.

1.8. Oluşturulan çözümü 2-opt sezgiseli uygulayarak iyileştir. Populasyondaki en iyi çözümü belirle.

1.9. Çözümde yer alan rotaları saat yönünün tersi yönünde numaralandır.

Adım 2: İkili Turnuva yöntemine göre çözüm havuzundan iki çözüm seç.

2.1. Populasyondan rasgele olarak iki çözüm seç. Amaç fonksiyon değeri daha iyi olan çözümü ilk ebeveyn çözüm olarak belirle.

2.2. Populasyondan ilk ebeveyn çözümden farklı rasgele olarak iki çözüm seç. Ebeveyn çözümle, seçilen çözümler arasındaki uzaklıkları numaralandırılmış rotalar arasında farklı noktaları göz önüne alarak hesapla ve ebeveyn çözümden en uzakta bulunan çözümü seçilecek ikinci çözüm olarak belirle.

Adım 3: Seçilen iki çözüm arasında kalan çözümleri, aşağıda yer alan adımları N/2 defa tekrarlayarak stokastik bir şekilde tara.

3.1. İki çözüm arasında rota bazında farklı noktaları belirle. Her rota için farklı noktaları, noktanın diğer çözümdeki pozisyonuna göre iki kümeye ayır.

3.2. Farklı nokta kümeleri boş olmayan bir rotayı rasgele seç ve rotaya kümeler içersinden rasgele bir nokta ekle.

3.3. Eğer ekleme sonucunda kısıtlar ihlal ediliyorsa, ilk noktanın çıkartıldığı rotaya küme içersinden rasgele bir nokta ekle.

3.4. Yukarıdaki 3.2 ve 3.3 nolu adımları uygun bir çözüm bulununcaya kadar belirli bir iterasyon sayısında tekrarla, eğer uygun bir çözüm bulunamazsa çözüm havuzundan yeni çözümler seç.

3.5. Eğer tarama sırasında ebeveyn çözümlerden daha iyi çözüm elde edilirse, bunu yeni oluşturulacak populasyona ekle, eğer yeni çözümler eski çözümlerden daha iyi ise eski çözümleri yeni populasyona ekle.

Adım 4: Belirli bir G, nesil sayısına kadar adım 2 ve 3'ü iteratif olarak çalıştır. Nesil sayısına ulaşılınca bulunan en iyi çözümü kullanıcıya çıktı olarak ver.

6.8 Geliştirilen Yöntemle Bir Uygulama

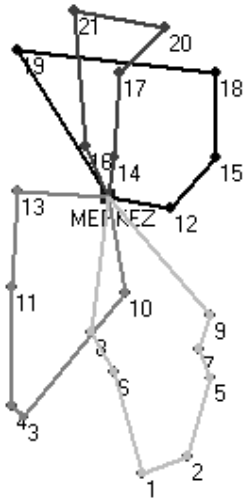
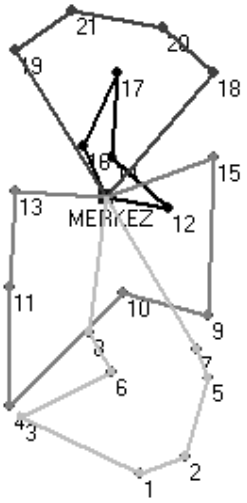
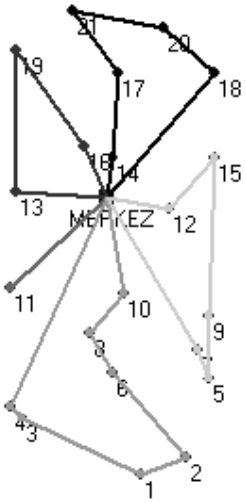
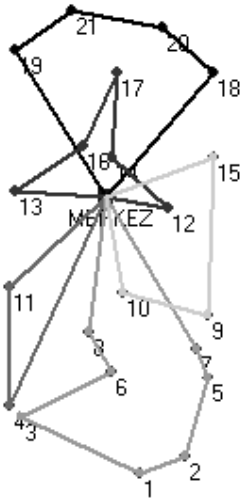
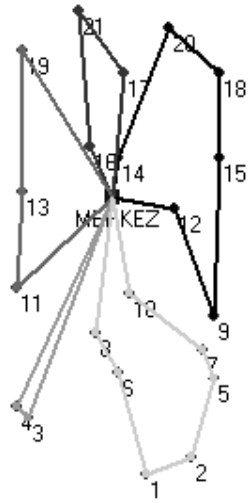
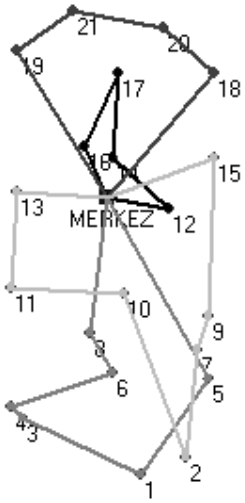
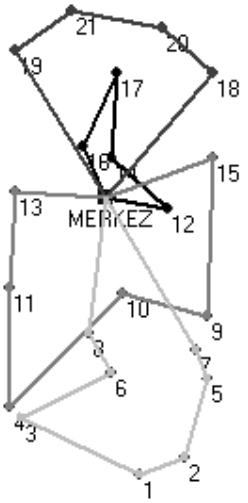
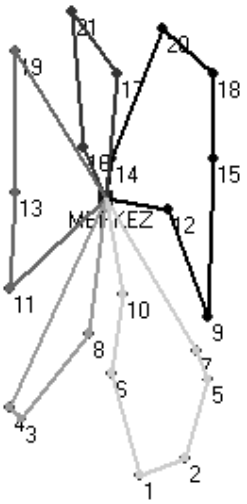
Bu bölümde geliştirilen populasyon ve komşuluk tabanlı yöntemin literatürde Eil-22 adıyla geçen ve İnternet'ten ([1], [2]) bulunabilen bir KKARP üzerinde uygulanması adım adım açıklanmaktadır. Christofides ve Eilon tarafından hazırlanan problemde 22 nokta bulunmakta ve araç kapasitesi 6000 olarak alınmaktadır. Çizelge 6.2'de probleme ait nokta koordinatları, talep değerleri ve hesaplanan polar değerler verilmektedir. Tabloya göre 0 nolu nokta, merkez depo noktasıdır. Şekil 6.8'de ise problem Öklidyen düzlem üzerine çizdirilmiştir. Şekilde sol üst köşe (0, 0) koordinatı olarak alınmış olup, sağa ve aşağıya doğru koordinat değerleri büyümektedir.

Çizelge 6.2 Eil-22 problemi nokta bilgileri

Nokta No	Koordinat	Talep	Polar Değer	Nokta No	Koordinat	Talep	Polar Değer
0	(145, 215)	0	-	11	(128, 231)	1200	2,68
1	(151, 264)	1100	3,12	12	(156, 217)	1300	3,98
2	(159, 261)	700	3,29	13	(129, 214)	1300	1,99
3	(130, 254)	800	2,93	14	(146, 208)	300	0,98
4	(128, 252)	1400	2,90	15	(164, 208)	900	0,34
5	(163, 247)	2100	3,49	16	(141, 206)	2100	1,40
6	(146, 246)	400	3,03	17	(147, 193)	1000	0,99
7	(161, 242)	800	3,50	18	(164, 193)	900	0,75
8	(142, 239)	100	2,99	19	(129, 189)	2500	1,52
9	(163, 236)	500	3,65	20	(155, 185)	1800	0,94
10	(148, 232)	600	3,17	21	(139, 182)	700	1,17

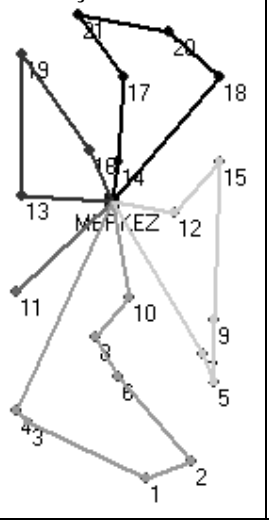
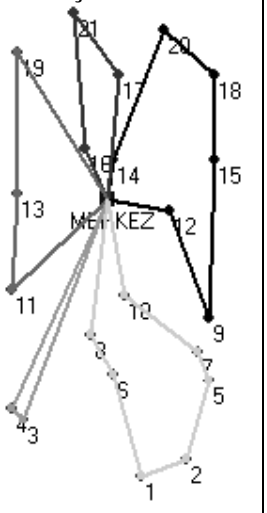
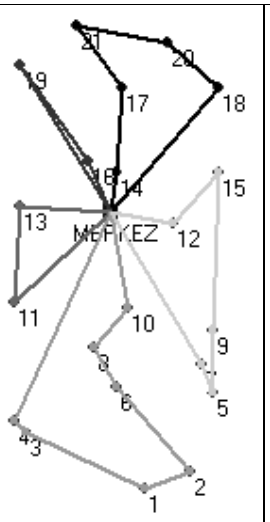
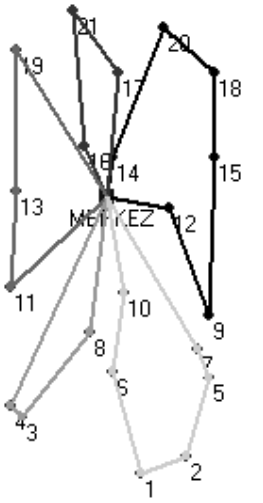
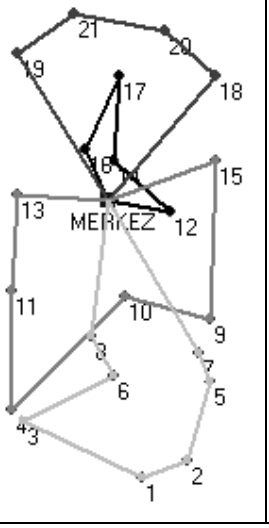
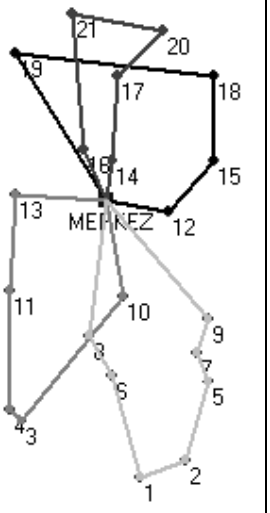
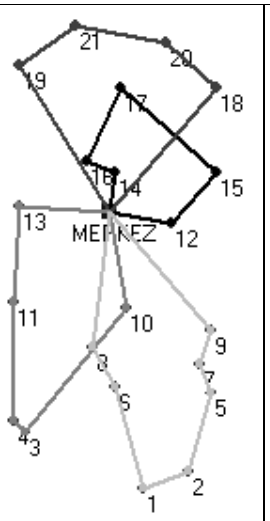
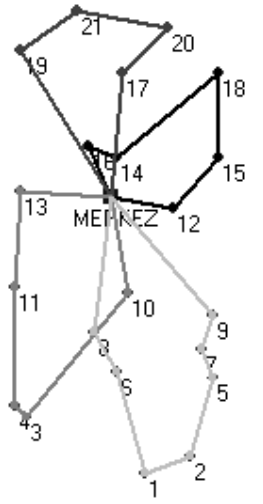
Uygulamada populasyon sayısı N = 8, nesil sayısı G = 4 belirlenmiş ve başlangıç çözümleri Süpürme, En Yakın Komşu (EYK), Paralel Tasarruf (PT) ve Sıralı Tasarruf (ST) yöntemlerinden biri rasgele seçilerek oluşturulmuştur. İki çözüm arasında tarama durdurma iterasyon sayısı ise 15 olarak alınmıştır. Bulunan başlangıç çözümleri, kullanılan sezgisel yöntem, seçilen nokta veya nokta ikilisi ve sonuçlar çizelge 6.3'de yer almaktadır:

Çizelge 6.3 Eil-22 için oluşturulan başlangıç çözümleri

Çözüm 1	Çözüm 2	Çözüm 3	Çözüm 4
 Yöntem: EYK Seçilen Nokta: 17 Sonuç: 403,58	 Yöntem: ST Seçilen İkili: 1 – 2 Sonuç: 443,36	 Yöntem: Süpürme Seçilen Nokta: 4 Sonuç: 440,22	 Yöntem: EYK Seçilen Nokta: 2 Sonuç: 483,95
Çözüm 5	**Çözüm 6**	**Çözüm 7**	**Çözüm 8**
 Yöntem: Süpürme Seçilen Nokta: 8 Sonuç: 482,20	 Yöntem: EYK Seçilen Nokta: 1 Sonuç: 456,59	 Yöntem: ST Seçilen İkili: 1 – 2 Sonuç: 443,36	 Yöntem: Süpürme Seçilen Nokta: 9 Sonuç: 480,68

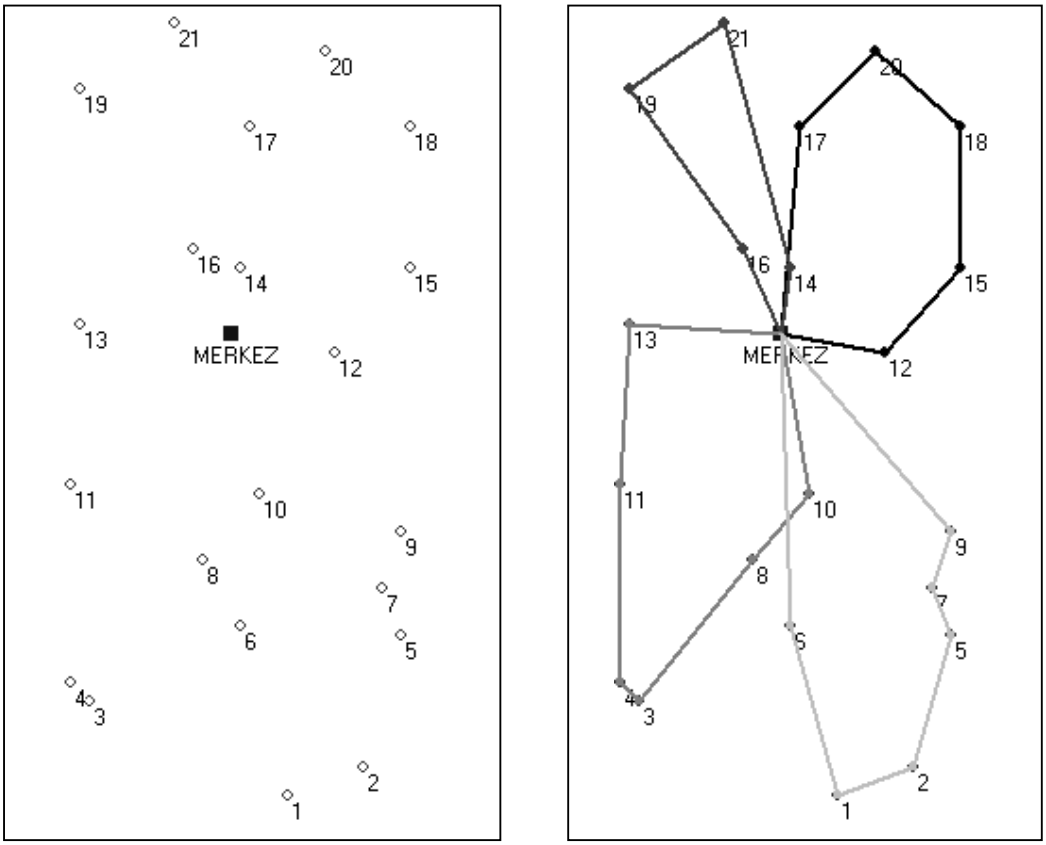
Üretilen başlangıç çözümleri içersinde çözüm 2 ve çözüm 7 aynı oluşturulmuştur. Ayrıca süreç stokastik bir şekilde işlediğinden ve örnek olması amacıyla N düşük tutulduğundan Paralel Tasarruf (PT) yöntemi ile çözüm üretilememiştir. Algoritmanın ilerleyiş süreci ve bulduğu sonuçlar ise adım çizelge 6.4'de gösterilmektedir:

Çizelge 6.4 Eil-22 için geliştirilen çözümler

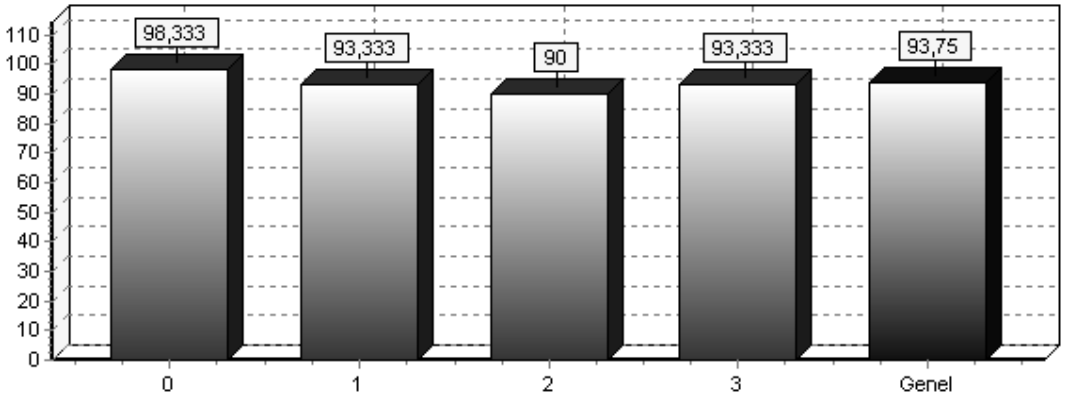
<table>
<tr><th colspan="2">Seçilen Çözümler</th><th colspan="2">Geliştirilen Çözümler</th></tr>
<tr><td>Çözüm 3
  </td><td>Çözüm 5
  </td><td> 
Sonuç: 439,44</td><td> 
Sonuç: 480,68</td></tr>
<tr><td colspan="2">Farklı Noktalar
17, 21, 12, 9, 19, 13, 10, 8, 6, 2, 1</td><td>Çözüm 3’de 13 nolu noktayı 2 nolu rotaya eklenmesi</td><td>Çözüm 5’de 8 nolu noktayı 3. rotaya eklenmesi</td></tr>
<tr><th colspan="2">Seçilen Çözümler</th><th colspan="2">Geliştirilen Çözümler</th></tr>
<tr><td>Çözüm2
  </td><td>Çözüm1
  </td><td> 
Sonuç: 388,77</td><td> 
Sonuç: 385,28</td></tr>
<tr><td colspan="2">Farklı Noktalar
16, 17, 14, 15, 18, 19, 9, 3</td><td>Çözüm 2’de 15 nolu noktanın rota 1’e atanması</td><td>Çözüm 1’de rota 0 ve rota 1 arasında 19 ve 16 nolu noktaların yer değiştirmesi</td></tr>
</table>

Çizelge 6.4’de ilk nesilde seçilen çözümler arasında farklı noktalar ve çözümlerin hangi değişikliğin yapılarak geliştirildiği açıklanmaktadır. Eil-22 probleminin optimum çözümü ise şekil 6.8’de gösterilmektedir. Optimum çözümün amaç fonksiyon değeri 375,27’dir. Bu

çözüme göre 0 nolu rotada sıralama 17, 20, 18, 15, 12 noktaları ile toplam yükleme 5900 birim; 1. rotada sıralama 16, 19, 21, 14 ile toplam yükleme 5600 birim; 2 nolu rotada sıralama 10, 8, 3, 4, 11, 13 ile toplam yükleme 5400 birim ve 3 nolu rotada sıralama 6, 1, 2, 5, 7, 9 ile toplam yükleme 5600 birim olmaktadır. Yükleme oranları şekil 6.9'da gösterilmektedir.



Şekil 6.8 Öklidyen düzlemde Eil-22 problemi ve optimum çözümü



Şekil 6.9 Eil-22 probleminin optimum çözümüne ait araç yükleme oranları

6.9 Uygulama Sonuçları

Geliştirilen yöntem literatürde sıkça rastlanan ve İnternet'ten ([1], [2]) yüklenebilen iki problem grubu üzerinde denenmiştir. İlk problem grubu Christofides ve Eilon tarafından hazırlanmıştır ve literatürde problemler "Eil problemleri" olarak geçmektedir. Uygulanan Eil grubunda 22 ile 33 arasında değişen nokta sayılı 4 KKARP bulunmaktadır. Diğer problem grubu ise Christofides, Mingozzi and Toth (CMT) tarafından hazırlanmıştır ve problemler "CMT" ön eki ile anılmaktadır. Çizelge 6.5'de problemlerin özellikleri gösterilmektedir.

Çizelge 6.5 Eil ve CMT problemleri

Problem	Nokta Sayısı	Kapasite	Mesafe Kısıtı	Servis Süresi	En İyi Çözüm
Eil22-k4	22	6000	∞	0	375,27
Eil23-k3	23	4500	∞	0	568,56
Eil30-k4	30	4500	∞	0	505,01
Eil33-k4	33	8000	∞	0	837,67
CMT1	51	160	∞	0	524,61
CMT2	76	140	∞	0	835,26
CMT3	101	200	∞	0	826,14
CMT4	151	200	∞	0	1028,42
CMT5	200	200	∞	0	1291,45
CMT6	51	160	200	10	555,43
CMT7	76	140	160	10	909,68
CMT8	101	200	230	10	865,94
CMT9	151	200	200	10	1162,55
CMT10	200	200	200	10	1395,85
CMT11	120	200	∞	0	1042,11
CMT12	100	200	∞	0	819,56
CMT13	120	200	720	50	1541,14
CMT14	100	200	1040	90	866,37

Eil problemleri tamamen kapasite kısıtlı olup, mesafe ve süre kısıtına sahip değillerdir. Bu sebeple mesafe kısıtı ∞ ve servis süresi 0 olarak belirtilmişlerdir. CMT grubunda ise 51 ile

200 arasında noktaya sahip 14 problem bulunmakta, kapasite kısıtının yanında mesafe ve süre kısıtı da yer almaktadır. CMT problemlerin ilk 10 tanesinde noktalar düzleme rastsal ve diğer 4 problemde ise gruplaşarak dağıtılmıştır. Ayrıca ilk 5 problem ile sonraki 5 problemde noktalar aynı lokasyonda bulunmakta, tek fark sonraki 5 problemde kapasite kısıtının yanında mesafe ve süre kısıtının da bulunmasıdır. Aynı durum CMT11 ve CMT12 problemleri ile CMT13 ve CMT14 problemleri arasında da görülmektedir. Geliştirilen populasyon ve komşuluk tabanlı algoritma Borland Delphi 7.0 dilinde kodlanarak bir program haline getirilmiştir. Problem grupları için programda alınan sonuçlar, atanan parametre değerleri ve hesap süreleri çizelge 6.6'da gösterilmektedir.

Çizelge 6.6 Geliştirilen yöntemin problemlere uygulanması

Problem	**Populasyon Sayısı (N)**	**Nesil Sayısı (G)**	**Durdurma Kriteri**	**Hesap Süresi***	**En İyi Çözüm**	**Üretilen Çözüm**	**Sapma Oranı**
Eil22-k4	20	20	5	1	375,27	**375,27**	0,0
Eil23-k3	20	20	5	1	568,56	**568,56**	0,0
Eil30-k4	30	30	5	2	505,01	**505,01**	0,0
Eil33-k4	40	30	5	5	837,67	**837,67**	0,0
CMT1	80	50	10	24	524,61	**524,61**	0,0
CMT2	150	150	10	263	835,26	**835,26**	0,0
CMT3	200	250	15	571	826,14	**826,14**	0,0
CMT4	300	400	15	1252	1028,42	1047,28	0,018
CMT5	400	500	15	2351	1291,45	1338,28	0,036
CMT6	100	80	10	188	555,43	**555,43**	0,0
CMT7	200	250	10	445	909,68	**909,68**	0,0
CMT8	250	300	15	854	865,94	**865,94**	0,0
CMT9	300	400	15	1842	1162,55	1173,48	0,009
CMT10	400	500	15	3141	1395,85	1440,27	0,031
CMT11	250	250	15	575	1042,11	1042,97	0,0008
CMT12	200	250	10	166	819,56	**819,56**	0,0
CMT13	250	350	15	1391	1541,14	1543,73	0,001
CMT14	200	300	10	92	866,37	**866,37**	0,0

* Hesap süreleri saniye cinsindendir.

Çizelge 6.6'da gösterilen değerler, programın AMD 1800 Mhz işlemcili bir bilgisayarda çalıştırılması sonucu elde edilmiştir. İki nokta arasındaki mesafe değerleri 10^{-8}'e kadar duyarlı olup, hiçbir yuvarlama işlemi yapılmamıştır. Tablodaki durdurma kriteri seçilen iki çözüm arasında rasgele taramayla kaç iterasyon boyunca çözüm oluşturulacağını göstermektedir, başka bir ifadeyle arama iterasyon sayısıdır. Tabloda belirtilen hesap süresi ise algoritma süresince bulunan en son çözüme programın başlatılmasından itibaren ne kadar zamanda ulaşıldığını belirtmektedir.

Geliştirilen populasyon ve komşuluk tabanlı yöntem küçük boyutlu Eil problemlerinde oldukça hızlı ve kaliteli çözüm üretirken, büyük boyutlu CMT problemlerinde en iyi bilinen çözümlere %3 oranında yaklaşabilmektedir. Ayrıca önerilen algoritma noktaların gruplaşarak dağıtıldığı CMT problemlerinde, rastsal dağıtılan problemlere göre daha iyi sonuçlar üretilmektedir. Mesafe ve süre kısıtının olduğu CMT problemlerinde ise algoritma esnekliğini ve güçlü arama stratejisini göstermekte, çözüm kalitesinden ödün vermemektedir.

Çizelge 6.7'de kapasite ve mesafe kısıtlı ARP için önerilen algoritmanın, geliştirilen diğer yöntemlerle karşılaştırılması yapılmaktadır. Tabloda verilen değerler tekniklerin ürettikleri çözümlerin, problemin en iyi çözümünden sapma oranıdır. Buna göre geliştirilen populasyon ve komşuluk tabanlı yöntem, 14 CMT problemi için 8 en iyi bilinen çözümü üretmiş ve çoğu algoritmadan daha iyi sonuçlar bulmuştur. Hafıza tabanlı yöntem olması dolayısıyla bilgisayar belleğini çok fazla kullanan Taillard'ın TA yöntemi ve Rochat ve Taillard'ın AH tabanlı algoritması ile mutasyon operatörü adı altında 9 farklı komşuluk yapısıyla çözümü iyileştiren Prins'in melez GA'sı dışında, geliştirilen algoritmanın çözüm kalitesi oldukça iyidir. Yöntemin en büyük dezavantajı ise büyük boyutlu problemler için sapma oranının yüksek olmasıdır. Bu da başlangıç çözümünün değişik klasik sezgisel yöntemlerle üretilmesiyle veya Prins'in yaptığı çalışmadaki (2004) gibi farklı komşuluk yapılarının ele alınması ile giderilebilir. Ayrıca uygulanan yoğunlaştırma stratejisinin 2-opt işlemi ile sadece rota bazlı olması, yöntemin populasyon tabanlı melez algoritma kategorisine girmemesini sağlamaktadır. Yöntemin en önemli özelliği ise tamamen stokastik yapısıyla az parametre içermesi ve sürecin oldukça basit, kolay anlaşılabilir ve esnek olmasıdır.

Çizelge 6.7 Geliştirilen algoritmanın diğer yöntemlerle karşılaştırılması

Problem	Osman TB[a]	Osman TA[a]	Taillard TA[b]	Rochat ve Taillard AH[b]	Tabu Rota[b]	Tanecikli TA[b]	Bütün. TA[b]	Barbaros. ve Özgür TA[c]	Baker ve Ayechew Saf GA[a]	Baker ve Ayechew Melez GA[a]	Prins Melez GA[d]	Bullnheimer vd. KKO[e]	Doerner vd. Tasarruf KKO[f]	Önerilen Yöntem
cmt1	0,00646	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**	0,00038	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**
cmt2	0,00328	0,01046	**0,0**	**0,0**	0,00007	0,00399	0,00022	0,00173	0,01737	0,00434	**0,0**	0,04229	0,00399	**0,0**
cmt3	0,00346	0,01435	**0,0**	**0,0**	**0,0**	0,00292	0,00399	0,00312	0,01764	0,00403	**0,0**	0,0645	0,01481	**0,0**
cmt4	0,02876	0,01514	**0,0**	**0,0**	0,00257	0,00465	0,00974	0,01504	0,02667	0,00620	0,00198	0,1157	0,01209	0,01833
cmt5	0,06546	0,03294	0,00056	**0,0**	0,01540	0,02075	0,01116	0,01139	0,06758	0,02813	0,00382	0,14089	0,01264	0,03626
cmt6	**0,0**	0,00077	**0,0**	**0,0**	0,00005	**0,0**	**0,0**	-	0,00875	**0,0**	**0,0**	0,0135	**0,0**	**0,0**
cmt7	**0,0**	0,00145	**0,0**	**0,0**	**0,0**	0,01213	**0,0**	-	0,00489	**0,0**	**0,0**	0,0423	0,00366	**0,0**
cmt8	0,00006	0,01392	**0,0**	**0,0**	**0,0**	0,00408	0,00051	-	0,00794	0,00201	**0,0**	0,02336	0,00480	**0,0**
cmt9	0,00124	0,01845	**0,0**	**0,0**	0,00029	0,00909	0,00796	-	0,02623	0,00319	**0,0**	0,03394	0,00935	0,00940
cmt10	0,01586	0,03234	0,00149	**0,0**	0,00637	0,02857	0,01400	-	0,06247	0,02107	0,00494	0,07805	0,03071	0,03182
cmt11	0,12848	0,00085	**0,0**	**0,0**	**0,0**	0,00072	0,03072	0,0087	0,01739	0,00461	**0,0**	0,02911	0,00170	0,00082
cmt12	0,00785	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**	0,07106	**0,0**	**0,0**	0,00045	**0,0**	**0,0**
cmt13	0,00250	0,00250	**0,0**	**0,0**	0,00310	0,00283	0,01801	-	0,01369	0,00335	0,00111	0,03204	0,00454	0,00168
cmt14	0,02727	**0,0**	**0,0**	**0,0**	**0,0**	**0,0**	0,00018	-	0,00689	0,00087	**0,0**	0,00403	**0,0**	**0,0**

[a](Baker ve Ayechew, 2003), [b](Cordeau vd., 2002), [c](Barbarosoğlu ve Özgür, 1999), [d](Prins, 2004), [e](Bullnheimer vd., 1997a), [f](Doerner vd., 2002)

7. SONUÇ

Son yıllarda optimizasyon kavramı işletmelerde planlama, tasarım, çizelgeleme ve maliyetlendirme faaliyetlerinde oldukça önemli bir konuma gelmiştir, zira optimizasyon işlemi kaynakların verimli bir şekilde kullanılmasını, zaman tasarrufu ve ürün kalitesinin artırılmasını sağlamaktadır. Özellikle büyük boyutlu sistemlerde, etkinliklerin insan emeği ile optimize edilmesi zor olduğundan, bu tür işlemler algoritmalar yoluyla bilgisayar programları ile gerçekleştirilebilir. Metasezgisel algoritmalar günümüzde optimizasyon uygulamaları için en etkili sonuç veren yöntemlerden biridir. Tavlama Benzetimi, Tabu Arama, Genetik Algoritmalar, Karınca Kolonisi Optimizasyon Algoritması ve Yapay Sinir Ağları uygulamalarda başarısını kanıtlamış metasezgisel yöntemlerdir. Kombinatoryal problemlere iyi kalitede çözümler oluşturan bu tekniklerde çoğunlukla bir veya birden fazla çözüm ele alınıp, bu çözümlerden iteratif olarak daha iyi çözüm üretilmektedir. Bu yöntemlerde belirlenen komşuluk yapısı ile bir çözümün komşu çözümleri göz önüne alınarak algoritma, bir arama stratejisi ile yürütülmekte ve belirlenen bir durdurma kriteri ile süreç sonlandırılıp, üretilen çözüm kullanıcıya sunulmaktadır.

Araç Rotalama Problemi, işletmelerde özellikle dağıtım kanallarında rota planlarının yapılması sırasında karşılaşılan ve lojistik maliyetlerini önemli ölçüde belirleyen, sıralama ve gruplamanın yapıldığı NP-zor kombinatoryal optimizasyon problemidir. Çok farklı kısıtların yer alabildiği problemin çözüm yöntemleri, Kesin ve Yaklaşık yöntemler olmak üzere iki grup olarak belirtilmektedir. Kesin yöntemler optimum sonuç vermesine rağmen sadece küçük boyutlu problemlere uygulanabilmekte ve çok fazla işlem gereksinimi olduğu için çözüm süreci uzun sürede gerçekleşmektedir. Yaklaşık yöntemler ise büyük problemlerde daha az işlemle ve kısa hesaplama süresiyle optimuma yakın, iyi kalitede çözümler üretmektedir. Yaklaşık yöntemler, klasik sezgisel ve metasezgisel yöntemler olmak üzere ikiye ayrılmaktadır. Klasik sezgisel yöntemler hızlı bir şekilde orta kalitede çözümler oluşturabilmektedir ve bu yöntemlerin en önemlileri Tasarruf, Süpürme ve Taç Yaprağı yöntemidir. ARP için en etkili sonuçlar veren algoritmalar ise metasezgisel algoritmalardır. Klasik sezgisel algoritmalara göre işlem süresi daha uzun olsa da hafıza tabanlı algoritmalardan TA ve AH prosedürleri, ARP için en iyi çözümleri bulan tekniklerdir. Literatürde çoğu metasezgisel yöntemin ARP için uygulaması bulunmaktadır.

Metasezgisel algoritmaların, lokal komşuluk tabanlı ve populasyon tabanlı olmak üzere iki türü bulunmaktadır. TA ve TB gibi komşuluk tabanlı algoritmalar arama sürecini bir çözüm ile yürütmekte ve belirlenen komşuluk yapısı ile mevcut çözümü iyileştirmeye

çalışmaktadırlar. Bu tür yöntemlerin en önemli özelliği belirli bir bölgede yoğun bir şekilde arama yaparak, o bölge için lokal optimum noktayı kısa sürede açığa çıkarmasıdır. Fakat bu yöntemler büyük problemlerde başlangıç çözümüne oldukça bağlı olup, genellikle lokal bir noktaya takılarak uzun süre çalıştırılsalar dahi iyi çözümler bulamayabilir. GA ve KKO gibi populasyon tabanlı algoritmalar ise aynı anda birden fazla çözümü ele alıp, bu çözümlerin özelliklerinden faydalanarak yeni çözümler üretmektedirler. Populasyon tabanlı algoritmalar çözüm uzayında çeşitli noktalardan hareket ederek lokal noktalara takılmadan çeşitlendirme stratejisini etkin bir şekilde kullanmaktadır, fakat büyük boyutlu problemlerde en iyi bilinen çözümlere uzak çözümler oluşturmaları en büyük dezavantajlarıdır.

Bu çalışmada lokal komşuluk ve populasyon tabanlı algoritmaların iyi yönleri birleştirilerek yeni bir algoritma geliştirilmiştir. Önerilen algoritmanın en önemli özelliği populasyon tabanlı tekniklerden yola çıkılarak, çözümlerin komşuluk mantığı ile oluşturulmasıdır. Geliştirilen yöntemde GA'da olduğu gibi iki çözüm ele alınmakta fakat yeni çözümler, bu iki çözüm arasında kalan çözümlerin ekleme ve λ-yer değiştirme metodu ile taranması ile elde edilmektedir. Rotalar üzerinde yer değişimi yapılacak noktalar ise seçilen "iki çözüm arasında uzaklık" kavramı ile belirtilen bir nokta kümesi içinde bulunmaktadır. Hem populasyon hem de komşuluk bazlı metasezgisel algoritma özelliği gösteren yöntemin temel özellikleri aşağıda açıklanmaktadır:

- **Algoritmanın yoğunlaştırma stratejisi:** Süreç boyunca oluşturulan rotaların 2-opt prosedürü ile iyileştirilmesi.
- **Algoritmanın çeşitlendirme stratejisi:** Başlangıç çözümlerinin farklı klasik yöntemlerle, stokastik bir şekilde oluşturulması ve populasyon içersinden mümkün olduğunca uzak çözümler arasında yer alan geniş bölgelerin ele alınması.
- **Çözümler arasında uzaklık kavramı:** Çözümler rota grubu gibi ele alındığı ve rotaların numaralandırılması yapıldığı takdirde, hangi noktanın hangi rotada olduğuna bakılarak farklı olan nokta sayısı ile çözümler arasındaki uzaklığın belirlenmesi.

Geliştirilen metodun dayandığı temel nokta, arama uzayında aralarında yeterince uzaklık bulunan iki iyi çözüm örneği arasında bu iki çözümden daha iyi bir çözüm yer alabileceği varsayımıdır. Algoritma mantığı her türden ARP'ye uygulanabilir olup, gerekli düzenlemelerin yapılması halinde diğer kombinatoryal optimizasyon problemlerine de denenebilir esnekliktedir. Yöntemin en önemli özelliği ise tamamen stokastik yapısıyla az parametre içermesi ve sürecin oldukça basit, kolay anlaşılabilir ve esnek olmasıdır. Ayrıca

geliştirilen yöntemin bilgisayarda kodlanması ile literatürde bulunan 18 kapasite ve mesafe kısıtlı ARP üzerinde yapılan denemelerde, 12 en iyi bilinen çözüm üretilmiş ve çoğu metasezgisel algoritmadan daha iyi sonuçlar elde edilmiştir.

Metasezgisel yöntemler zor kombinatoryal optimizasyon problemlerinde iyi sonuçlar üretmek için doğadan esinlenerek geliştirilmiş ileri seviye sezgisel prosedürlerdir. Metasezgisel yaklaşım çözüm uzayında stokastik fakat bilinçli bir şekilde arama yaparak optimum çözümü bulmaya çalışır. Metasezgisel yöntemlerin en önemli özelliği sadece belli bir tip problem için değil, tüm kombinatoryal problemlere uygulanabilir esneklikte olmasıdır. Bu sebeple literatürde bu tekniklerin birçok başarı hikayesine rastlanmaktadır. Metasezgisel yöntemlerin gerçek katkısı ise günümüzde yaşanılan problemlere uygulanması ile elde edilecektir.

KAYNAKLAR

Alabaş, Ç., Dengiz, B., (2004), "Yerel Arama Yöntemlerinde Yöre Yapısı: Araç Rotalama Problemine Bir Uygulama", Yöneylem Araştırması/Endüstri Mühendisliği 24. Ulusal Kongresi, Gaziantep – Adana, 333 – 335

Alba, E., Dorronsoro, B., (2005), "Computing Nine New Best-So-Far Solutions For Capacitated VRP With A Cellular Genetic Algorithm", Information Processing Letters

Baker, B., M., Ayechew, M., A., (2003), "A Genetic Algorithm For The Vehicle Routing Problem", Computers & Operations Research, 30, 787-800

Barbarosoğlu, G., Özgür, D., (1999), "A Tabu Search Algorithm For The Vehicle Routing Problem", Computers & Operations Research, 26, 255-270

Bell, J., E., McMullen, P., R., (2004), "Ant Colony Optimization Techniques For The Vehicle Routing Problem", Advanced Engineering Informatics, 18, 41-48

Bektaş, T., (2004), "The Multiple Traveling Salesman Problem An Overview of Formulations and Solution Procedures", Omega The International Journal of Management Science

Blum, C., Roli, A., (2003), "Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison", ACM Computing Surveys, 35, 3, 268-308

Brandao, J., (2004), "A Tabu Search Algorithm for the Open Vehicle Routing Problem", European Journal of Operation Research, 157, 552-564

Breedam, A., V., (2001), "Comparing Descent Heuristics and Metaheuristics for the Vehicle Routing Problem", Computers & Operations Research, 28, 289-315

Breedam, A., V., (2002), "A Parametric Analysis of Heuristics for the Vehicle Routing Problem", European Journal of Operation Research, 137, 348-370

Bullnheimer, B., Hartl, R. F., Strauss, C., (1997), "Applying The Ant System To The Vehicle Routing Problem", 2nd International Conference on Metaheuristics, Sophia-Antipolis, France

Bullnheimer, B., Hartl, R. F., Strauss, C., (1997), "An Improved Ant system Algorithm for the Vehicle Routing Problem", Sixth Viennese workshop on Optimal Control, Dynamic Games, Nonlinear Dynamics and Adaptive Systems, Vienna, Austria

Cordeau, J. F., Gendreau, M., Laporte, G., Potvin, J.Y., Semet, F., (2002), "A Guide To Vehicle Routing Heuristics", Journal of the Operational Research Society, 53, 512-522

Cordeau, J. F., Gendreau, M., Hertz, A., Laporte, G., Sormany, J.S., (2004), "New Heuristics for the Vehicle Routing Problem", Les Chairs du Gerad, G-2004-33

Çölkesen, R., (2002), Veri Yapıları ve Algoritmalar, Papatya Yayıncılık, İstanbul

Doerner, K., Gronalt, M., Hartl, R., Reimann, M., Strauss, C., Stummer, M., (2002), "SavingsAnts For The Vehicle Routing Problem", EvoWorkshops02, LNCS 2279, Springer-Verlag, 11–20

Dorigo, M., Caro, G. D., Gambardella, L., M., (1999), "Ant Algorithms for Discrete Optimization", Artificial Life, 5, 137-172

Duncan, T., (1995), "Experiments in the use of Neighbourhood Search Techniques for Vehicle Routing", Expert Systems, Alai-Tr-176

Erel, R., (1995), "Taşıt Rotalaması ve Çizelgelemesi: Otobüsle Kentlerarası Yolcu Taşımacılığı İçin Bir Model", Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü İnşaat Mühendisliği Bölümü Doktora Tezi, İstanbul

Eren, H., (2002), "Akış Tipi Çizelgeleme Problemlerinin Genetik Algoritma(GA) İle Çözüm Performansının Artırılmasında Deney Tasarımı Uygulaması", İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Bölümü Yüksek Lisans Tezi, İstanbul

Gambardella, L. M., Taillard, E., Agazzi, G., (1999), "MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows", New Ideas in Optimization, McGraw-Hill

Gen, M., Cheng, R., (2000), "Genetic Algorithms and Engineering Optimization", John Wiley Sons, Inc, USA

Gendreau, M., Guertin, F., Potvin, J. Y., Taillard, E., (1999), "Parallel Tabu Search for Real-Time Vehicle Routing and Dispatching", Transportation Science, 33, 4, 381

Hertz, A., Widmer, M., (2004), "Guidelines For The Use Of Meta-Heuristics In Combinatorial Optimization", European Journal of Operation Research, 151, 247-252

Jaskiewicz, A., Kominek, P., (2003), "Genetic Local Search With Distance Preserving Recombination Operator For A Vehicle Routing Problem", European Journal of Operation Research, 151, 352-364

Karaboğa, D., (2004), "Yapay Zeka Optimizasyon Algoritmaları", Atlas Yayın Dağıtım, Cağaloğlu, İstanbul

Karahan, A., (2003), "Tedarik Zinciri Yönetiminde Dağıtım Faaliyetlerinin Optimize Edilmesine Yönelik Bir Model Tasarımı", Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Ana Bilim Dalı Yüksek Lisans Tezi, İstanbul

Kelly, J. P., Xu, J., (1999), "A Set-Partitioning-Based Heuristic for the Vehicle Routing Problem", Informs Journal on Computing, Vol. 11, No. 2

Lacomme, P., Prins, C., Cherif, W., R., (2005), "Evolutionary Algorithms For Periodic Arc Routing Problems", European Journal of Operation Research, 165, 535-553

Laporte, G., Gendreau, M., Potvin, J. Y., Semet, F., (2000), "Classical And Modern Heuristics For The Vehicle Routing Problem", International Transactions in Operation Research, 7, 285-300

Laporte, G., Palekar, U., (2002), "Some Applications Of The Clustered Travelling Salesman Problem", Journal of the Operational Research Society, 53, 972-976

Mazzeo, S., Loiseau, I., (2004), "An Ant Colony Algorithm for the Capacitated Vehicle Routing", Electronic Notes in Discrete Mathematics, 18, 181-186

Nabiyev, V. V., (2003), "Yapay Zeka – Problemler Yöntemler Algoritmalar", Seçkin Yayınevi, Ankara

Potvin, J. Y., Xu, Y., Benyahia, I., (2004), "Vehicle routing and scheduling with dynamic travel times", Computers & Operations Research, 0305 - 0548

Preux, P., Talbi, E. G., (1999), "Towards Hybrid Evolutionary Algorithms", International Transactions in Operation Research, 6, 557-570

Prins, C., (2004), "A Simple And Effective Evolutionary Algorithm For The Vehicle Routing Problem", Computers & Operations Research, 31, 1985-2002

Rego, C., (2001), "Node-Ejection Chains For The Vehicle Routing Problem Sequential And Parallel Algorithms", Parallel Computing, 27, 201-222

Reimann, M., Doerner, K., Hartl R.F., (2003), "Analyzing a Unified Ant System for the VRP and Some of Its Variants", EvoWorkshops 2003, LNCS 2611, 300–310

Reimann, M., Laumanns, M., (2006), "Savings Based Ant Colony Optimization For The Capacitated Minimum Spanning Tree Problem", Computers & Operations Research, 33, 1794-1822

Ruelle, D., (2004), "Rastlantı ve Kaos", Tübitak Popüler Bilim Kitapları, Dönmez Ofset, Ankara

Tarantilis, C. D., Kiranoudis, C. T., Markatos, N. C., (2002), "Use of the BATA Algorithm and MIS To Solve The Mail Carrier Problem", Applied Mathematical Modelling, 26, 481-500

Tarantilis, C. D., Ioannou, G., Prastacos, G., (2004), "Advanced Vehicle Routing Algorithms For Complex Operations Management Problems", Journal of Food Engineering, Elsevier Ltd.

Tarantilis, C. D., Kiranoudis, C. T., Vassiliadis, V., S., (2004), "A Threshold Accepting Metaheuristic For The Heterogeneous Fixed Fleet Vehicle Routing Problem", European Journal of Operation Research, 152, 148-158

Tarantilis, C. D., Kiranoudis, C. T., (2005), "A Flexible Adaptive Memory-Based Algorithm For Real-Life Transportation Operations: Two Case Studies From Dairy And Construction Sector", European Journal of Operation Research, Elsevier

Tavares, J., Pereira, F. B., Machado, P., Costa, E., (2003), "On the Influence of GVR in Vehicle Routing", Proceedings of the ACM Symposium On Applied Computing (SAC 2003)

Tavares, J., Pereira, F. B., Machado, P., Costa, E., (2003), "Crossover and Diversity: A Study About GVR", Proceedings of the 2003 Analysis and Design of Representations and Operators

Toth, P., Vigo, D., (2002), "The Vehicle Routing Problem", Society for Industrial and Applied Mathematics, Philadelphia

Toth, P., Vigo, D., (2003), "The Granular Tabu Search and Its Application to the Vehicle-Routing Problem", Informs Journal on Computing, 15, 4, 333

İNTERNET KAYNAKLARI

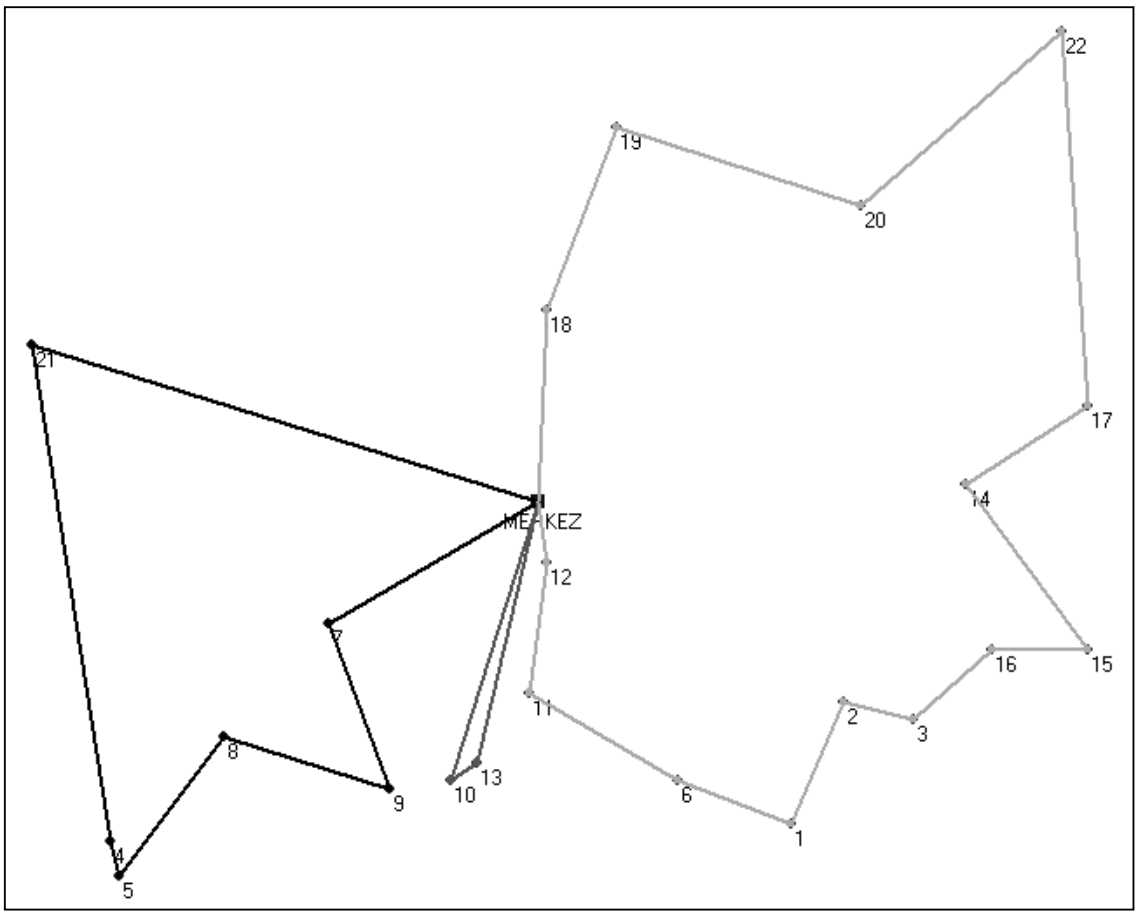
[1] http://www. neo.lcc.uma.es/radi-aeb/webvrp/index

[2] http://www.branchandcut.org/VRP/data

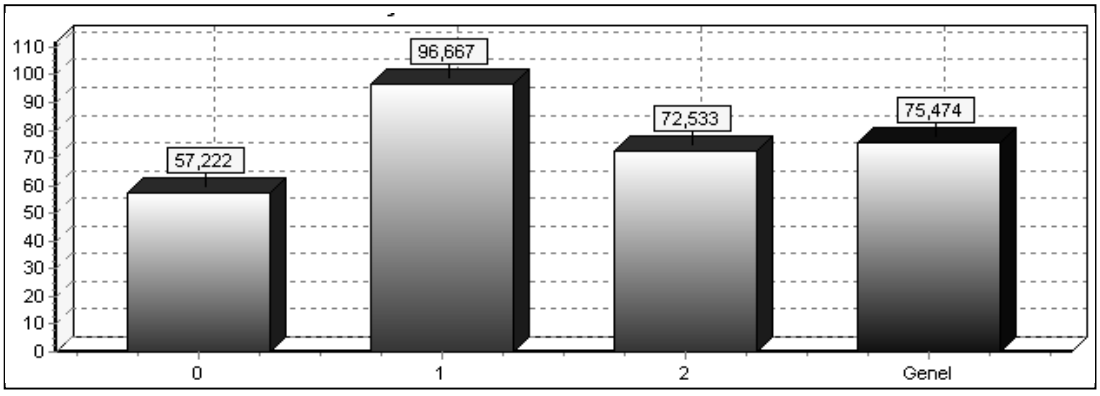
EKLER

Ek A Bazı Araç Rotalama Problemleri için en iyi bilinen çözümler

1. Eil23-k3 Problemi için bulunan çözüm değeri 568,56 olup çözüm şekil A.1'de, araç yükleme oranları ise şekil A.2'de gösterilmektedir.

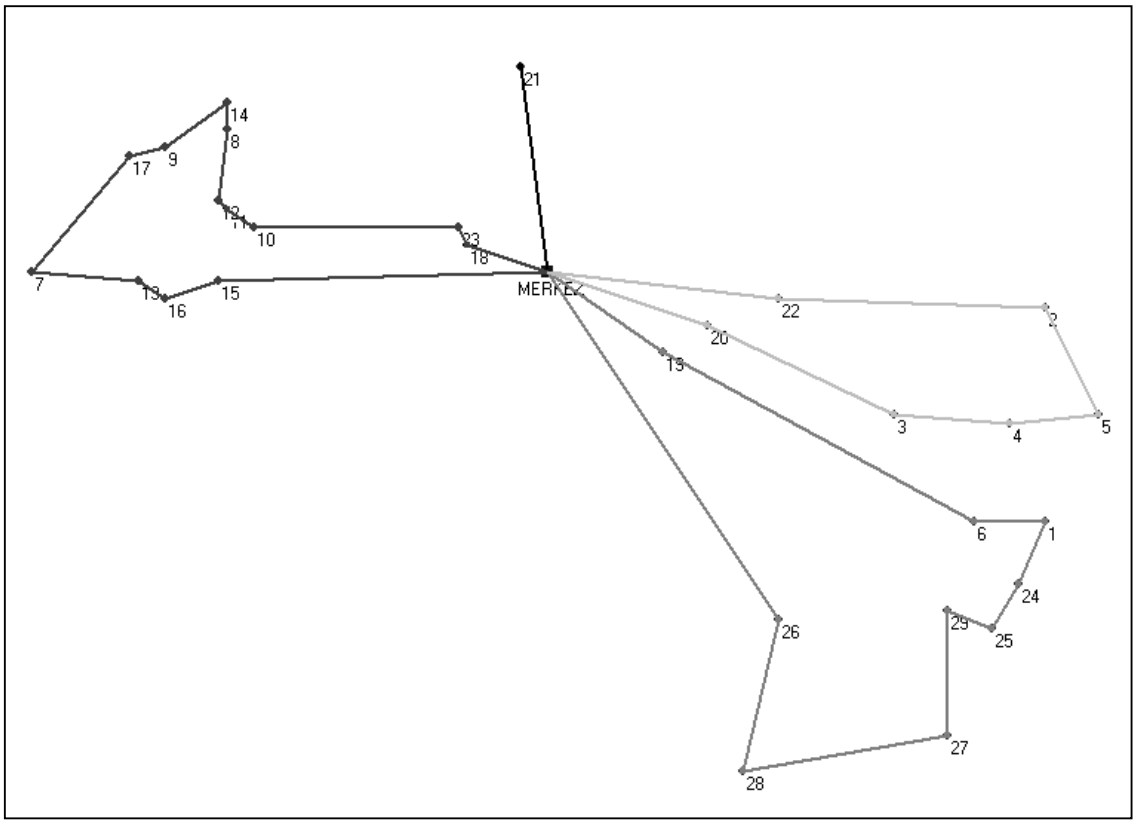


Şekil A.1 Eil23-k3 problemi için bulunan optimum çözüm

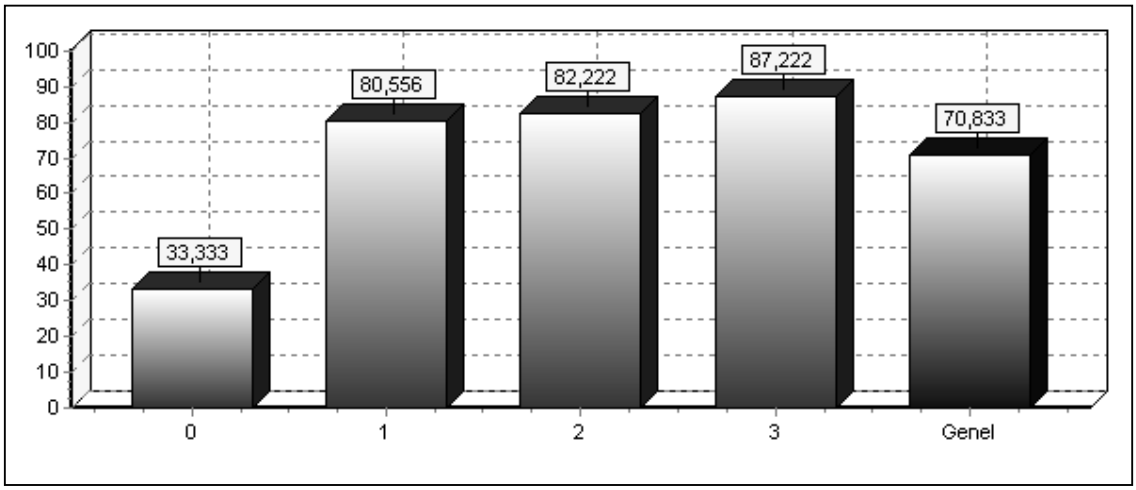


Şekil A.2 Eil23-k3 problemi için araç yükleme oranları

2. Eil30-k4 Problemi için bulunan çözüm değeri 568,56 olup çözüm şekil A.3'de, araç yükleme oranları ise şekil A.4'de gösterilmektedir.

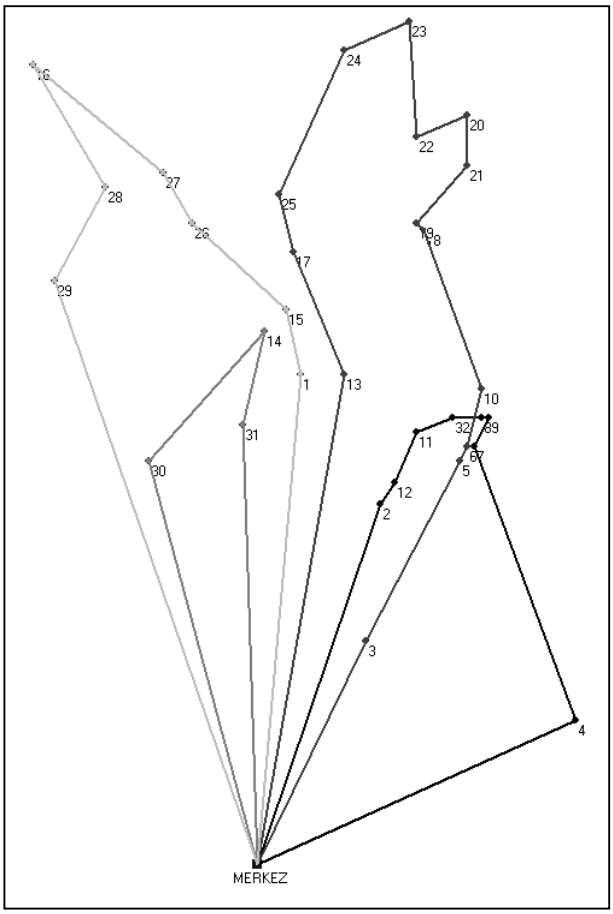


Şekil A.3 Eil30-k4 problemi için bulunan optimum çözüm

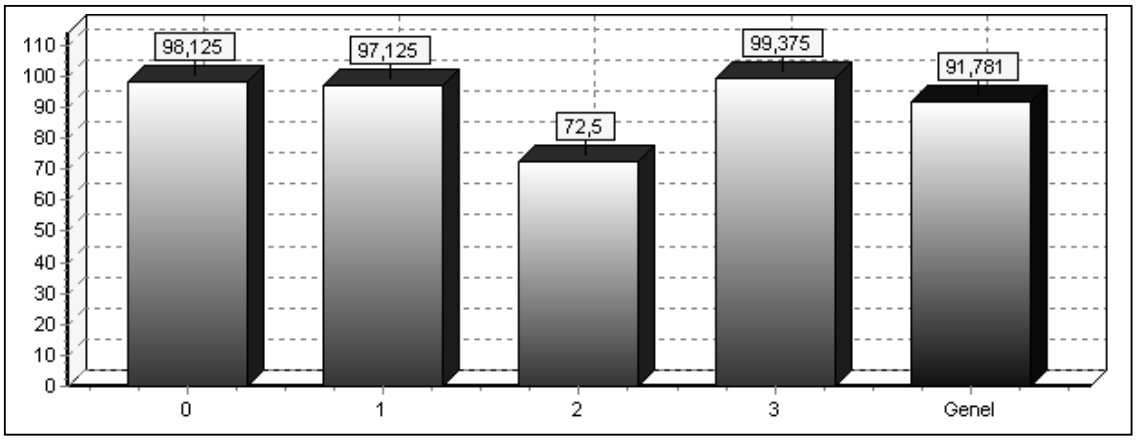


Şekil A.4 Eil30-k4 problemi için araç yükleme oranları

3. Eil33-k4 Problemi için bulunan çözüm değeri 837,67 olup çözüm şekil A.5'de, araç yükleme oranları ise şekil A.6'da gösterilmektedir.

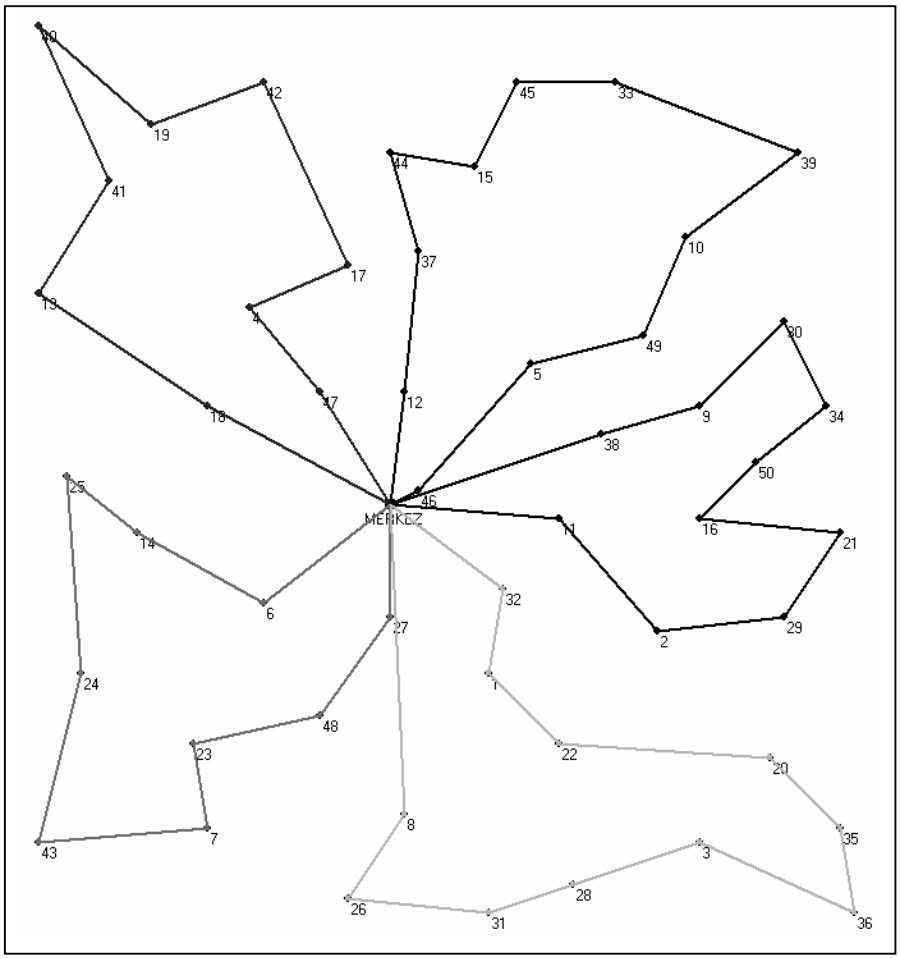


Şekil A.5 Eil33-k4 problemi için bulunan optimum çözüm

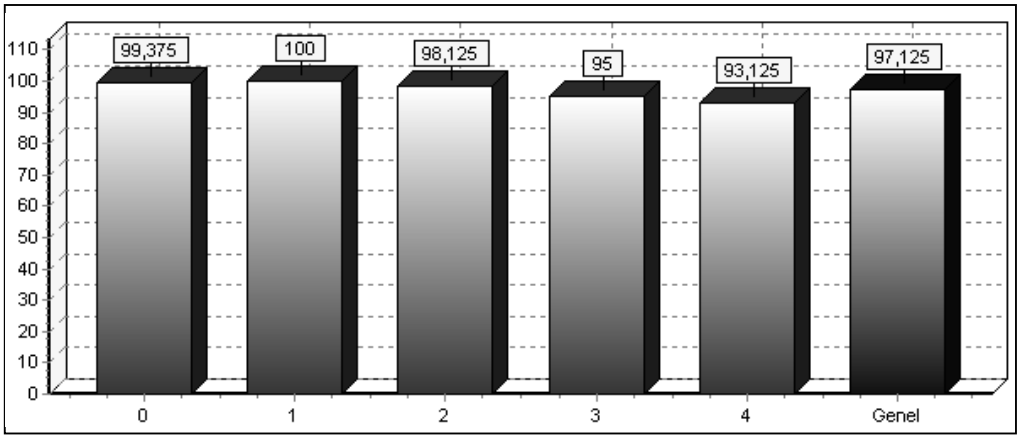


Şekil A.6 Eil33-k4 problemi için araç yükleme oranları

4. CMT1 Problemi için bulunan çözüm değeri 524,61 olup çözüm şekil A.7'de, araç yükleme oranları ise şekil A.8'de gösterilmektedir.

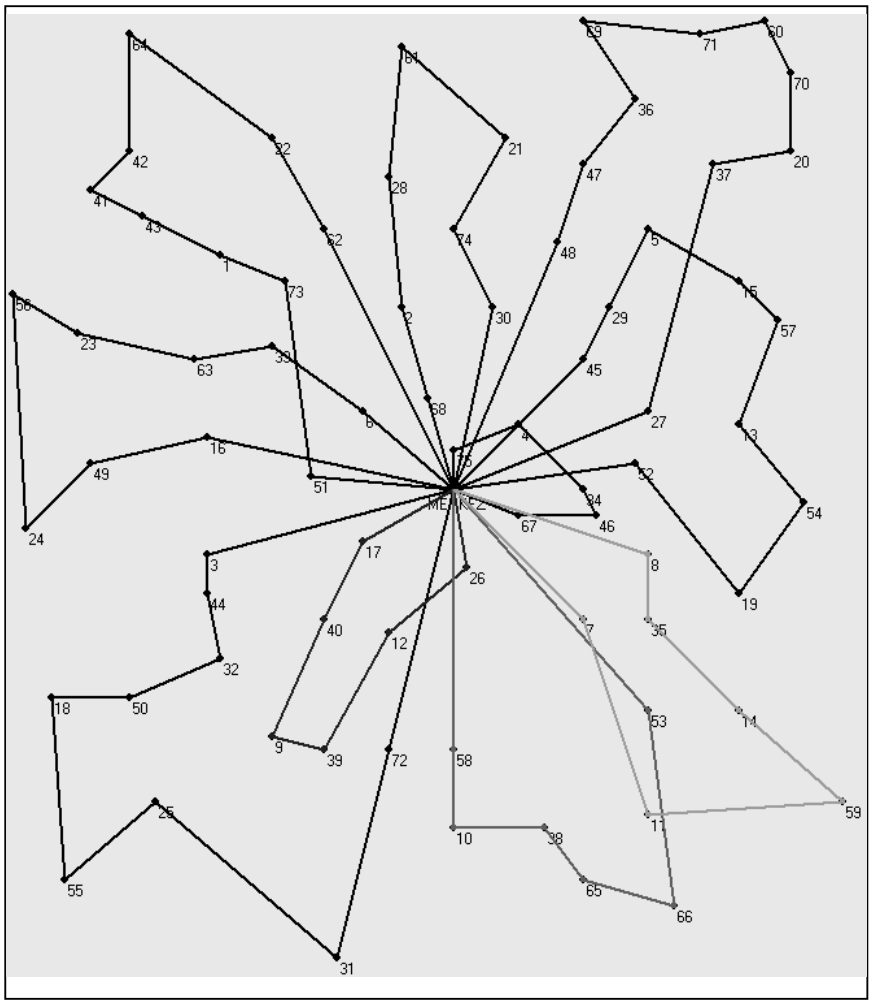


Şekil A.7 CMT1 problemi için bulunan optimum çözüm

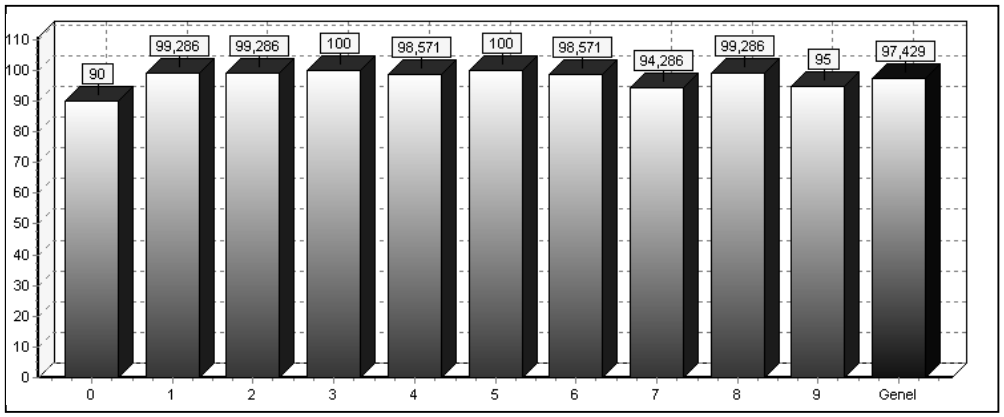


Şekil A.8 CMT1 problemi için araç yükleme oranları

5. CMT2 Problemi için bulunan çözüm değeri 835,26 olup çözüm şekil A.9'da, araç yükleme oranları ise şekil A.11'de gösterilmektedir.

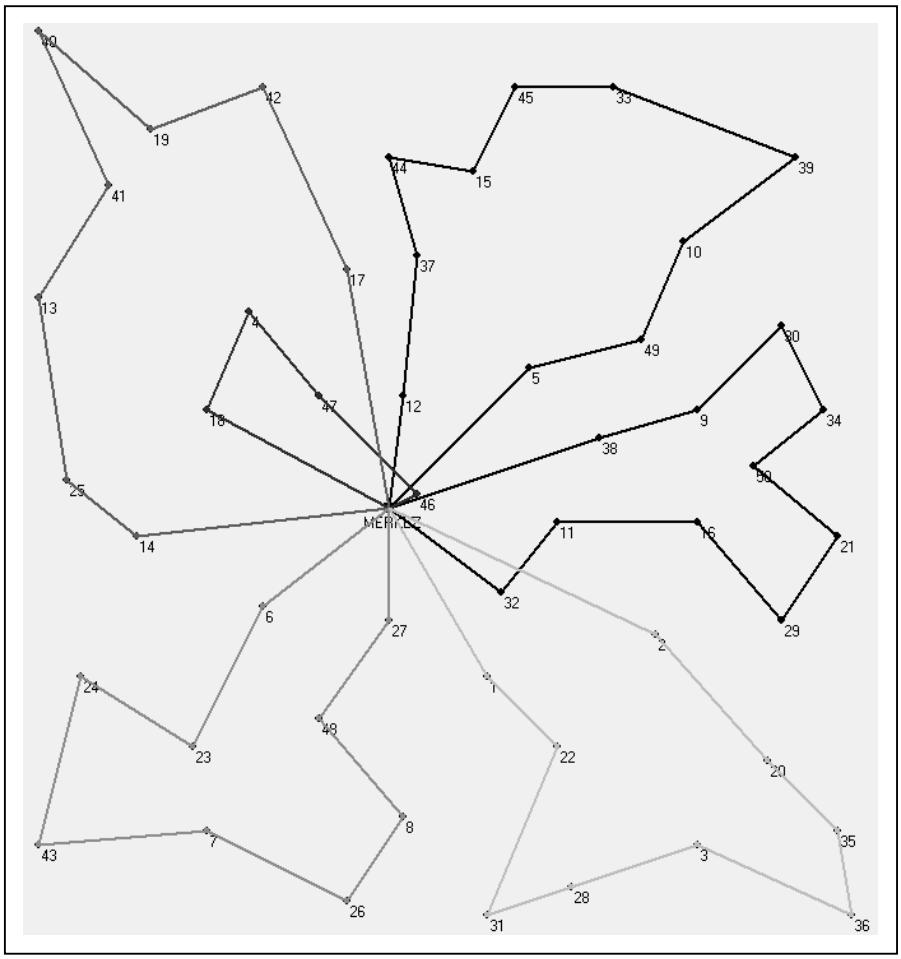


Şekil A.9 CMT2 problemi için bulunan optimum çözüm

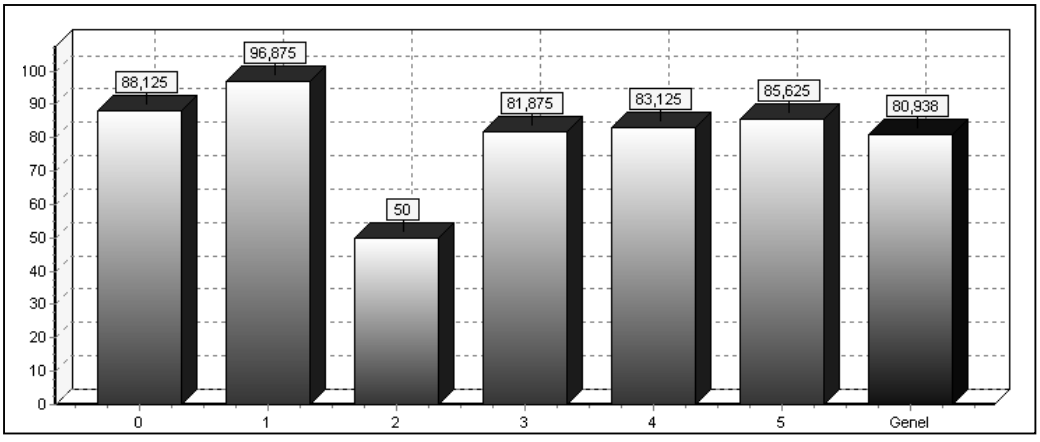


Şekil A.10 CMT2 problemi için araç yükleme oranları

6. CMT6 Problemi için bulunan çözüm değeri 555,43 olup çözüm şekil A.11'de, araç yükleme oranları ise şekil A.12'de gösterilmektedir.

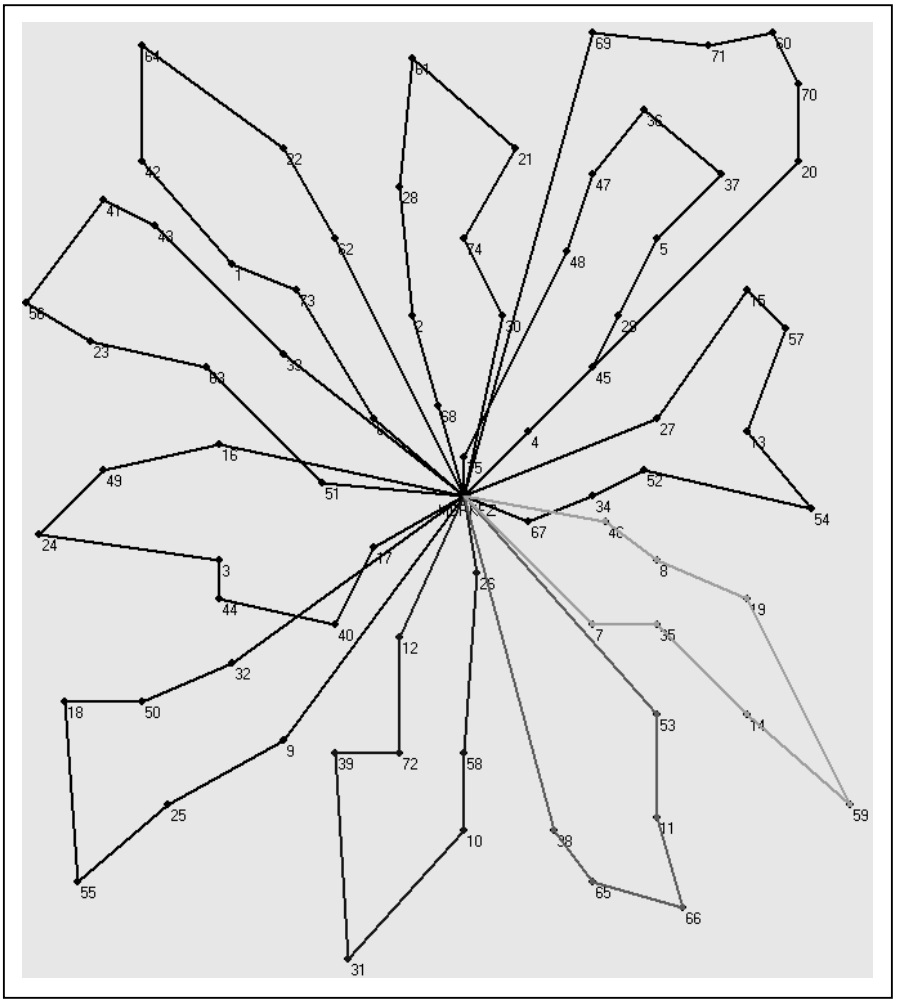


Şekil A.11 CMT6 problemi için bulunan optimum çözüm

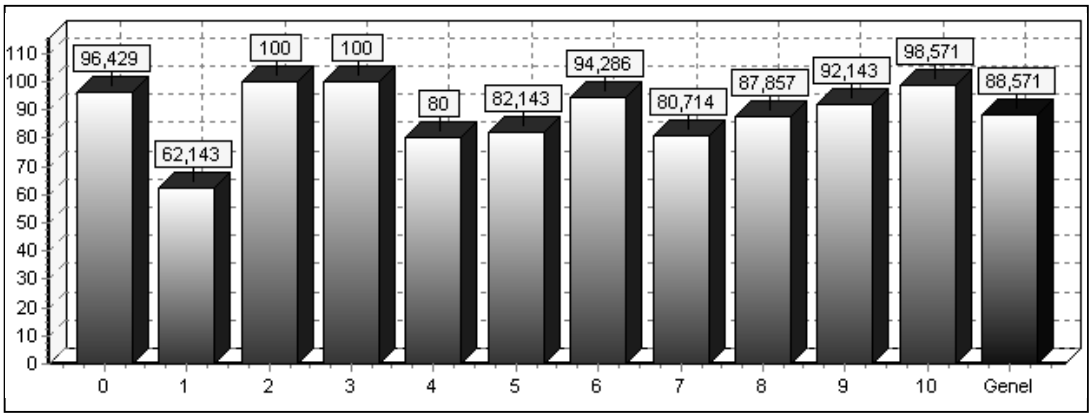


Şekil A.12 CMT6 problemi için araç yükleme oranları

7. CMT7 Problemi için bulunan çözüm değeri 909,67 olup çözüm şekil A.13'de, araç yükleme oranları ise şekil A.14'de gösterilmektedir.

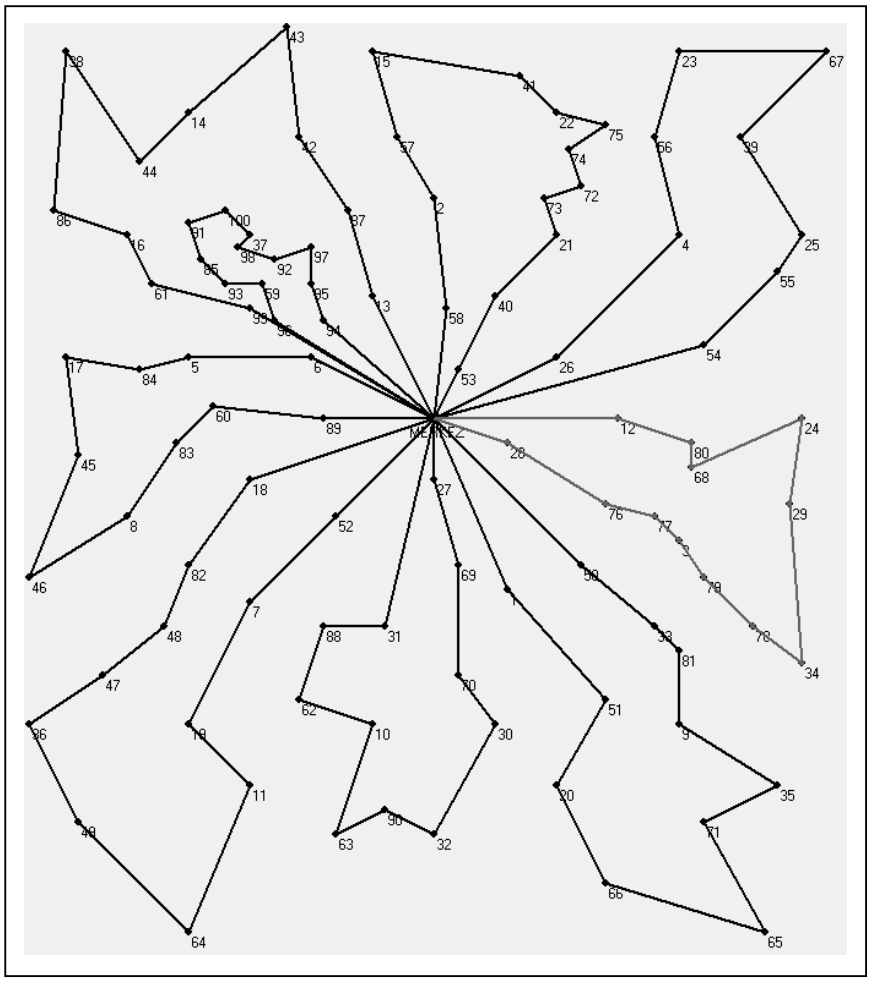


Şekil A.13 CMT7 problemi için bulunan optimum çözüm

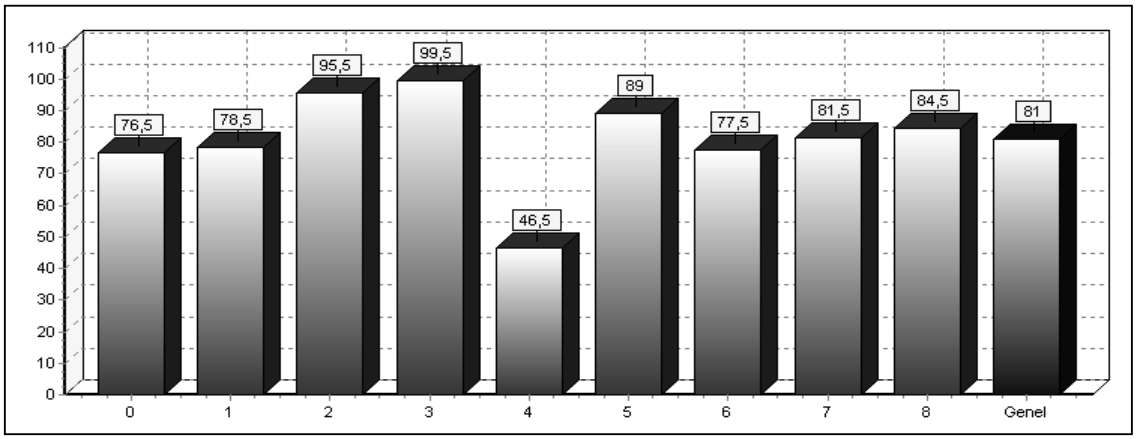


Şekil A.14 CMT7 problemi için araç yükleme oranları

8. CMT8 Problemi için bulunan çözüm değeri 865,94 olup çözüm şekil A.15'de, araç yükleme oranları ise şekil A.16'da gösterilmektedir.

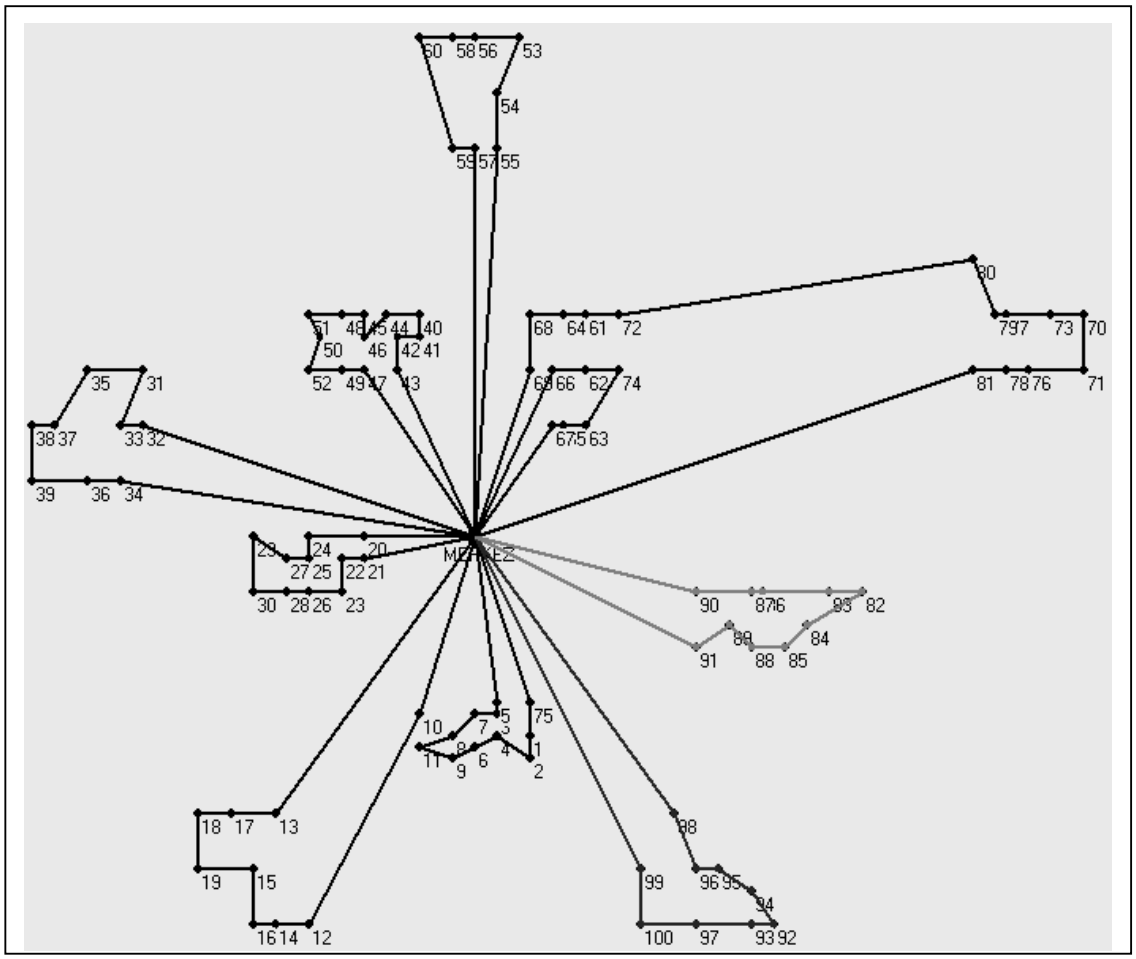


Şekil A.15 CMT8 problemi için bulunan optimum çözüm

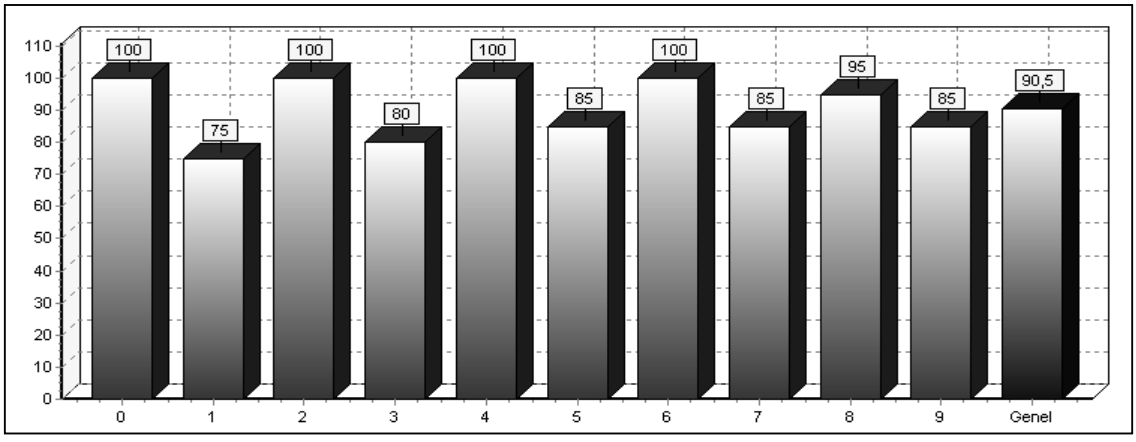


Şekil A.16 CMT8 problemi için araç yükleme oranları

9. CMT12 Problemi için bulunan çözüm değeri 819,55 olup çözüm şekil A.17'de, araç yükleme oranları ise şekil A.18'de gösterilmektedir.

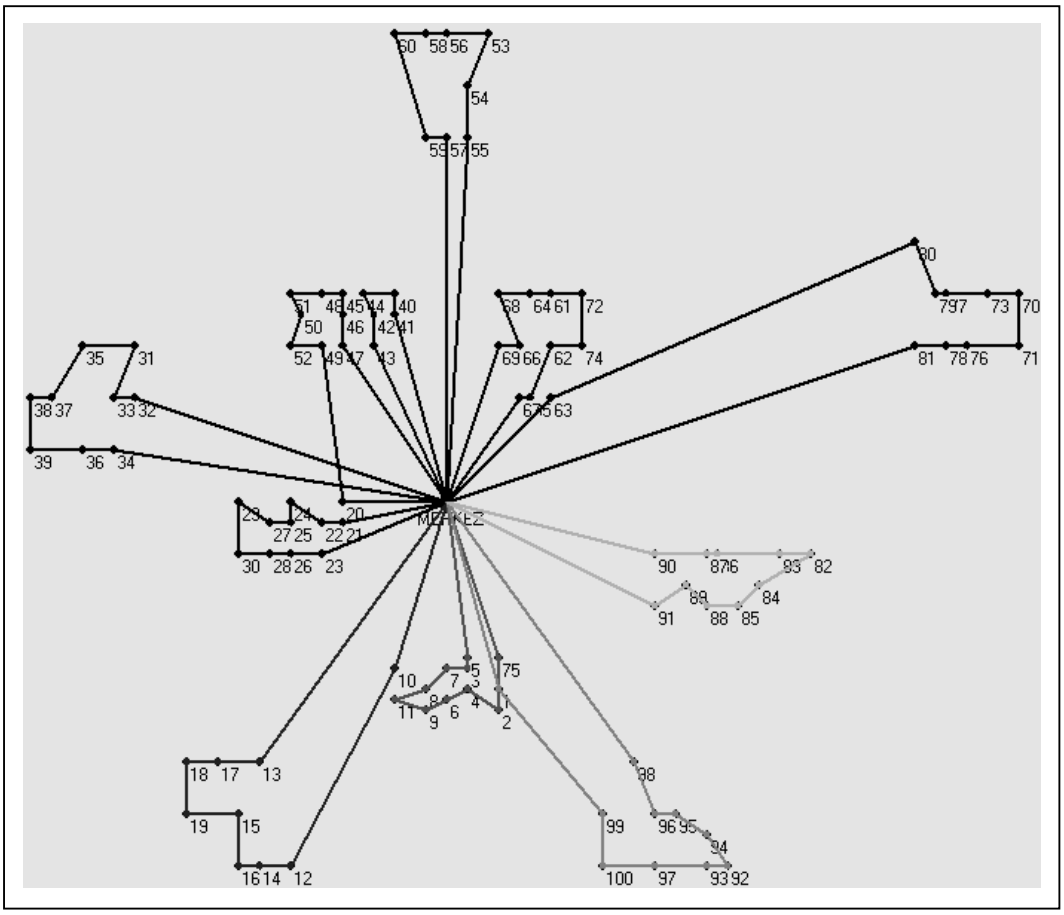


Şekil A.17 CMT12 problemi için bulunan optimum çözüm

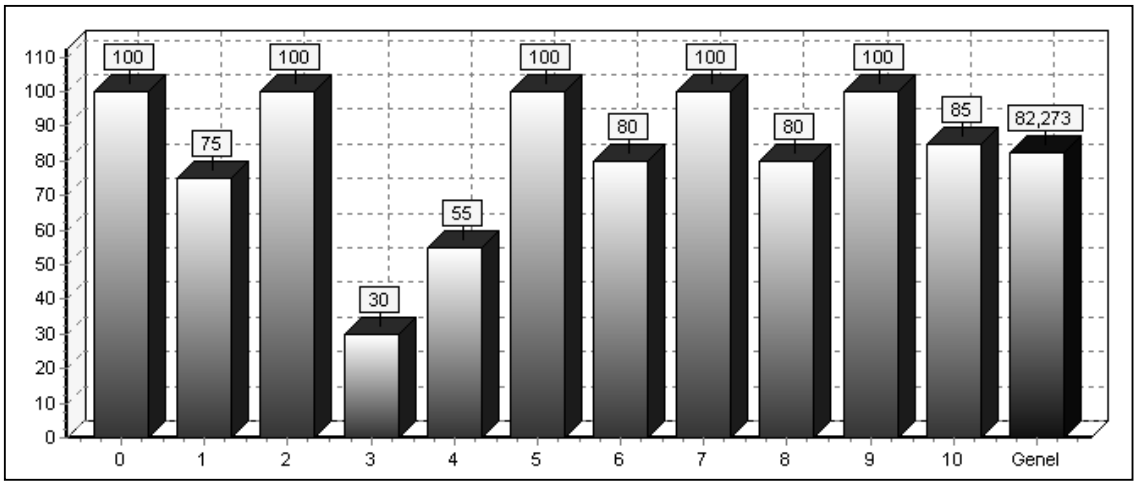


Şekil A.18 CMT12 problemi için araç yükleme oranları

10. CMT14 Problemi için bulunan çözüm değeri 866,36 olup çözüm şekil A.19'da, araç yükleme oranları ise şekil A.20'de gösterilmektedir.



Şekil A.19 CMT14 problemi için bulunan optimum çözüm



Şekil A.20 CMT14 problemi için araç yükleme oranları

ÖZGEÇMİŞ

Doğum tarihi	25.08.1981	
Doğum yeri	Denizli	
Lise	1996-1999	Denizli Anadolu Lisesi
Lisans	1999-2003	İstanbul Teknik Üniversitesi İşletme Fakültesi Endüstri Mühendisliği Bölümü
Yüksek Lisans	2003-2006	Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Anabilim Dalı Sistem Mühendisliği Programı

Çalıştığı kurumlar:

2003-2004	Uytes Danışmanlık Ltd. Şti.
2004-2005	Çözüm Yazılım ve Bilgisayar Ltd Şti.
2005-	Sistek Yazılım ve Danışmanlık Ltd. Şti.