

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

**KARINCA KOLONİSİ OPTİMİZASYONU VE GENETİK ALGORİTMA
TABANLI TRAMP GEMİ ROTALAMA VE ÇİZELGELEME**

YÜKSEK LİSANS TEZİ

Seher Suendam ARICI

Deniz Ulaştırma Mühendisliği Anabilim Dalı

Deniz Ulaştırma Mühendisliği Programı

TEMMUZ 2021

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

**KARINCA KOLONİSİ OPTİMİZASYONU VE GENETİK ALGORİTMA
TABANLI TRAMP GEMİ ROTALAMA VE ÇİZELGELEME**

YÜKSEK LİSANS TEZİ

**Seher Suendam ARICI
(512181012)**

Deniz Ulaştırma Mühendisliği Anabilim Dalı

Deniz Ulaştırma Mühendisliği Programı

Tez Danışmanı: Doç. Dr. Emre AKYÜZ

TEMMUZ 2021

İTÜ, Lisansüstü Eğitim Enstitüsünün 512181012 numaralı Yüksek Lisans Öğrencisi Seher Suendam ARICI, ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı “KARINCA KOLONİSİ OPTİMİZASYONU VE GENETİK ALGORİTMA TABANLI TRAMP GEMİ ROTALAMA VE ÇİZELGELEME” başlıklı tezini aşağıda imzaları olan jüri önünde başarı ile sunmuştur.

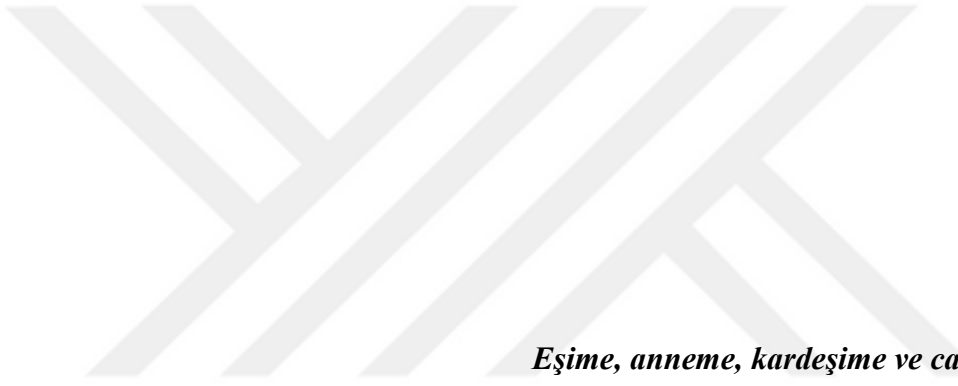
Tez Danışmanı : **Doç. Dr. Emre AKYÜZ**
İstanbul Teknik Üniversitesi

Jüri Üyeleri : **Prof. Dr. Yasin ARSLANOĞLU**
İstanbul Teknik Üniversitesi

Dr. Öğr. Üyesi Emre ÇAKMAK
Piri Reis Üniversitesi

Teslim Tarihi : 27 Haziran 2021
Savunma Tarihi : 13 Temmuz 2021





Eşime, anneme, kardeşime ve canım babama,



ÖNSÖZ

Yüksek lisans tez çalışmam süresince kıymetli bilgi, birikimi ve tecrübeleriyle bana yol gösterici olan ve desteğini esirgemeyen değerli danışman hocam Doç. Dr. Emre AKYÜZ'e teşekkür ve saygılarımı sunarım.

Tez çalışmamın her aşamasında düşünceleri ile katkıda bulunan, en büyük yardımcım, sevgili kardeşim Suat ORAL'a en içten teşekkürlerimi sunarım.

Bugünlere gelmemde en büyük paya sahip olan annem Kamuran ORAL ve babam Salih ORAL'a, ve bana her konuda destek olan eşim Hasan ARICI'ya teşekkür ederim.

Haziran 2021

Seher Suendam ARICI
(Araştırma Görevlisi)



İÇİNDEKİLER

Sayfa

ÖNSÖZ	vii
İÇİNDEKİLER	ix
KISALTMALAR	xi
SEMBOLLER	xiii
ÇİZELGE LİSTESİ	xv
ŞEKİL LİSTESİ	xvii
ÖZET	xix
SUMMARY	xixi
1. GİRİŞ	1
1.1 Tezin Amacı	3
1.2 Tezin Etkisi	4
1.3 Literatür Taraması	5
1.3.1 Genetik algoritma yaklaşımı	5
1.3.1.1 Araç rotalama problemlerinde genetik algoritma	5
1.3.1.2 Gemi rotalama problemlerinde genetik algoritma	10
1.3.2 Karınca kolonisi optimizasyonu algoritması	11
1.3.2.1 Araç rotalama problemlerinde karınca kolonisi optimizasyonu	11
1.3.2.2 Gemi rotalama problemlerinde karınca kolonisi optimizasyonu	13
1.3.3 GA ve KKOHA hibrit yaklaşımı	14
1.3.3.1 Araç rotalama problemlerinde GA ve KKOHA	14
2. TRAMP TAŞIMACILIK VE DENİZ TİCARETİNDEKİ YERİ	17
2.1 Tramp Taşımacılık ile Taşınan Yük Türleri	18
3. ARAÇ ROTALAMA PROBLEMLERİ VE ÇÖZÜM YÖNTEMLERİ	21
3.1 Gezgin Satıcı Problemi	21
3.2 Araç Rotalama Problemleri	23
3.2.1 Karma kapasiteli araç rotalama	24
3.2.2 Çoklu depoya sahip araç rotalama	24
3.2.3 Bölünmüş talebe sahip araç rotalama	24
3.2.4 Belirsiz talebe sahip araç rotalama	25
3.2.5 Geri toplamalı araç rotalama	25
3.2.6 Asimetrik araç rotalama	25
3.2.7 Zaman pencereci araç rotalama	26
3.3 Araç Rotalama Problemi Çözüm Yöntemleri	26
3.3.1 Kesin çözüm yöntemleri	27
3.3.1.1 Kesme düzlemi algoritması	28
3.3.1.2 Dal ve sınır algoritması	28
3.3.1.3 Dal ve kesme algoritması	29
3.3.2 Sezgisel yöntemler	30
3.3.2.1 Klasik sezgisel yöntemler	30
3.3.2.2 Meta-sezgisel yöntemler	37

4. YÖNTEM.....	51
4.1 Problem Tanımı	51
4.2 Modelin Matematiksel İfadesi	52
4.3 Karınca Kolonisi Optimizasyonu Algoritma Modeli	55
4.3.1 Geçiş kuralı	55
4.3.2 Feromon yenilenmesi	56
4.4 Genetik Algoritma	57
4.4.1 Temel kavramlar	57
4.4.2 Genetik algoritma adımları.....	58
4.5 Hibrit Yöntem.....	69
5. UYGULAMA.....	75
5.1 Data Seti Oluşturulması.....	75
5.2 Uygulama Çıktıları	80
5.2.1 Doğrusal programlama	80
5.2.2 Hibrit çözüm.....	82
5.2.3 Bulgular.....	83
6. SONUÇ VE ÖNERİLER.....	85
KAYNAKLAR.....	89
EKLER	109
ÖZGEÇMİŞ	125

KISALTMALAR

ARP	: Araç Rotalama Problemi
AARP	: Asimetrik Araç Rotalama Problemi
BSTSARP	: Belirsiz Talebe Sahip Araç Rotalama Problemi
BTSARP	: Bölünmüş Talebe Sahip Araç Rotalama Problemi
ÇDARP	: Çoklu Depoya Sahip Araç Rotalama Problemi
GA	: Genetik Algoritma
GSP	: Gezgin Satıcı Problemi
GTARP	: Geri Toplamalı Araç Rotalama Problemi
KARP	: Kapasite Kısıtlı Araç Rotalama Problemi
KKARP	: Karma Kapasiteli Araç Rotalama Problemi
KKOA	: Karınca Kolonisi Optimizasyonu Algoritması
NP	: Deterministik olmayan polinom
P	: Polinom
PSOA	: Parçacık Sürü Optimizasyonu Algoritması
TA	: Tabu Arama
TB	: Tavlama Benzetim
YAKA	: Yapay Arı Kolonisi Algoritması
ZPARP	: Zaman Pencereci Araç Rotalama Problemi



SEMBOLLER

C_{ijk}	: i,j,k yayı maliyet
l_{ij}	: Feromon değeri
pl_{ij}	: Koku uçma değeri
α	: Feromon katsayısı
η_{ij}	: Görünürlük değeri
β	: Görünürlük katsayısı
P	: Olasılık değeri
Δl_{ij}	: Koku değeri
(f)	: Uygunluk fonksiyonu



ÇİZELGE LİSTESİ

Sayfa

Çizelge 2.1 : 2018-2019 yılları katı dökme yük taşımacılığı.....	18
Çizelge 2.2 : 2018-2019 yılları sıvı dökme yük taşımacılığı.....	19
Çizelge 2.3 : Yıllara göre uluslararası deniz ticareti.....	19
Çizelge 3.1 : Optimizasyonun termodinamik karşılığı.	39
Çizelge 4.1 : Tek kromozom için maliyet hesabı.	72
Çizelge 5.1 : Bir gemi için maliyet tablosu.	77



ŞEKİL LİSTESİ

Sayfa

Şekil 1.1 : 2000-2021 Yılları ARP ve GA çalışmaları.	5
Şekil 1.2 : 2000-2021 Yılları ARP için KKOA çalışmaları.	11
Şekil 1.3 : 2010-2021 Yılları ARP için GA- KKOA hibrit çalışmaları.	14
Şekil 3.1 : Gezgin satıcı problemi.....	22
Şekil 3.2 : Araç rotalama problemi çözüm yöntemleri.....	27
Şekil 3.3 : En yakın komşuluk algoritması.....	31
Şekil 3.4 : Süpürme algoritması uygulama adımları.	32
Şekil 3.5 : Tasarruf oluşması.	33
Şekil 3.6 : Depo ve müşteri dağılımı.	34
Şekil 3.7 : Ters çevirme sezgiseli.	36
Şekil 3.8 : 3-Opt algoritması.....	37
Şekil 3.9 : Genetik algoritma akış şeması.....	42
Şekil 3.10 : Karınca kolonisi algoritması.	44
Şekil 3.11 : Karınca kolonisi algoritması akış şeması.	45
Şekil 3.12 : Parçacık sürü optimizasyonu algoritması akış şeması.	47
Şekil 3.13 : Yapay arı kolonisi algoritması akış şeması.	49
Şekil 4.1 : Bir gemi için sefer örneği.....	53
Şekil 4.2 : Modelin görsel ifadesi.	55
Şekil 4.3 : İkili kodlama örneği.	59
Şekil 4.4 : Değer kodlama örneği.	59
Şekil 4.5 : Permütasyon kodlama örneği.	59
Şekil 4.6 : Rulet seçim yöntemi.....	62
Şekil 4.7 : Tek-Nokta çaprazlama.	63
Şekil 4.8 : Üç-Nokta çaprazlama.	63
Şekil 4.9 : Uniform çaprazlama.	64
Şekil 4.10 : İlk çocuklar için sıralı çaprazlama.....	65
Şekil 4.11 : İkinci çocuklar için sıralı çaprazlama.....	65
Şekil 4.12 : Karıştırmalı çaprazlama 1.adım.	66
Şekil 4.13 : Karıştırmalı çaprazlama 2.adım.	66
Şekil 4.14 : Mutasyon yöntemleri.....	67
Şekil 4.15 : Hibrit yöntem işlem adımları.	69
Şekil 4.16 : Kontrat sayılarını temsil eden kromozom örneği.	70
Şekil 4.17 : Başlangıç kromozomu oluşturma.	70
Şekil 4.18 : Rota oluşturma.	70
Şekil 4.19 : Çaprazlama işlemi.	73
Şekil 4.20 : Mutasyon işlemi.	74
Şekil 5.1 : Bir örnek için doğrusal programlama ekranı.....	81
Şekil 5.2 : Bir örnek için hibrit çözüm ekranı.	83
Şekil 5.3 : Gurobi ve hibrit yöntemleri için karşılaştırma tablosu.....	84



KARINCA KOLONİSİ OPTİMİZASYONU VE GENETİK ALGORİTMA TABANLI TRAMP GEMİ ROTALAMA VE ÇİZELGELEME

ÖZET

Deniz yolu taşımacılığı, büyük hacimli yüklerin, uzak mesafelere ulaştırılma imkânı sağlaması, güvenli bir taşıma şekli olması ve diğer taşıma türleri ile karşılaştırıldığında daha uygun olması gibi nedenlerle ticarete en çok tercih edilen taşıma şeklidir.

Deniz yolu taşımacılığı, gemilerin sefer düzenleri açısından düzenli (liner) ve düzensiz (tramp) deniz taşımacılığı olarak ikiye ayrılır. Düzenli hat deniz taşımacılığında gemilerin izleyecekleri rotalar ve bu rotalara ait zaman çizelgelemeleri önceden bilinir. Bu taşımacılık şeklinde önemli olan hizmetin devamlı olmasıdır. Gemilerin söz konusu seferleri yapması için tam dolu olmaları gerekmez ve programı aksatmamak için seferi gerçekleştirirler.

Tramp taşımacılık modunda, gemilerin gidecekleri rotalar ve taşıyacakları yükler değişkenlik göstermektedir. Bu taşıma tipinde gemiler düzenli bir rotaya sahip değildirler ve taşıyacakları yükleri çoğunlukla seferden sefere bulurlar. Seferler belirli bir çizelgeye göre yapılmadığından ve gemiler karlı yük bulunan noktalara hareket ettiği için tramp taşımacılık rekabet esastır. Bu nedenle, birden çok gemisi bulunan bir filoda, gemiler tarafından hizmet verilecek kontratların, kontratlar için belirlenen zaman aralıkları içerisinde, söz konusu gemiler arasında uygun şekilde paylaştırılması önemli hale gelmektedir. Özellikle yakıt tüketiminin hız ve gidilen mesafe ile ilişkisi düşünüldüğünde, rota ve hızın optimizasyonu büyük önem içermektedir.

Gemi rotalama ve çizelgeleme problemleri, araç rotalama problemlerinin deniz yolu taşımacılığına uyarlanmış halidir. Çok bilinmezli, çok değişkenli ve karmaşık bir yapıya sahip olan büyük boyutlu problemlerdir. Bu problem tipi, çok fazla kısıt, hizmet edilmesi gereken birçok müşteri, birbiriyle uyumsuz olan amaçlar barındırmaktadır ve yapısı itibarı ile NP-hard problemlerdir.

Bu tez çalışmasında, tramp deniz taşımacılığı yapan bir gemi filosunun zaman pencereli, çok depolu ve çok araçlı rotalama ve çizelgeleme problemi incelenmiştir. Çözülen problemin amacı; taşınacak her bir yük için oluşturulan kontratlara ve bu kontratlarda belirlenen zaman kısıtlamalarına bağlı kalarak, mevcut olan her bir sözleşmeye hizmet ederken toplam operasyonel maliyeti minimuma indirmektir. Bu aşamada, her bir gemi için limana varış, limandan kalkış ve bekleme zamanları probleme dahil edilmiştir. Problemden kapasite kısıtının göz ardı edilebilmesi için tüm gemilerin aynı tip (cape-size) olduğu kabul edilmiş ve tüm gemilerin uygun zaman ve konumda olduğu sürece, tüm kontratlara hizmet götürebileceği varsayılmıştır. Ayrıca, probleme heterojenlik katmak için gemilerin hızları ve belirtilen hızlardaki yakıt harcamaları farklılaştırılmıştır.

Metodolojinin en önemli özelliği, kargo toplama ve teslim etme zaman pencerelerinin ayrılaştırılmasıdır. Bu özellik, modele çok çeşitli değişken ve kısıtın dahil edilmesi imkanı vermektedir. Çalışmada, deniz taşımacılığında temel işletme maliyeti olan

yakıt tüketimini kontrol etmek için seyir hızının ve gemiler arası optimum yük dağılımının etkisinin ölçümüne odaklanılmıştır.

Problem, taşımacılık tipinin çalışma sistemi göz önünde alınarak, karma tam sayılı doğrusal programlama modeli olarak oluşturulmuştur.

Veri toplama aşamasında, gemilerin yakıt tüketimlerine, seyir sürelerine, balastlı ya da yüklü seyir yapmalarına bağlı alınarak gerçekçi veriler oluşturulmuştur. Varış-kalkış zaman pencerelerinin probleme dahil edilmesi ile gemiler için maliyet matrisleri oluşturulmuştur.

NP-hard problemlerin çözümünde kesin çözüm yöntemleri ve sezgisel yöntemler kullanılabilmektedir. Kesin çözüm yöntemleri problem çözümünde optimum sonuçları vermektedir. Ancak, kesin çözüm yöntemlerinin çözüm kalitesi problem hacmi ile ters orantılıdır. Sezgisel yöntemler ise en iyi çözümü garanti etmez ancak büyük problem hacimleri için en iyiye yakın çözümleri makul çözüm sürelerinde verirler.

Bu amaçla çalışmada, gemi sayıları ve kontrat sayıları gibi değişkenler artırılarak farklı problem hacimleri oluşturulmuştur. Oluşturulan problemler öncelikle doğrusal programlama da Gurobi çözücüsü ile çözülmüştür. Ardından, kullanılan hibrit yöntemin karşılaştırılmasının yapılabilmesi amacıyla, model Genetik Algoritma ve Karınca Kolonisi Optimizasyonu Algoritmasının birlikte kullanıldığı hibrit meta-sezgisel yaklaşım ile çözülmüştür ve sonuçlar karşılaştırılmıştır.

Sonuçlar göstermiştir ki, her iki yöntem maliyet hesaplaması açısından birbirine çok yakın değerler vermektedir. Ancak, problem hacminin büyümesi ile kesin çözüm yöntemlerinde optimum sonucu alma süresi artarken, hibrit yöntem, problemin hacmi ve karmaşıklığı arttıkça daha iyi çözüm süreleri vermektedir.

ANT COLONY OPTIMIZATION AND GENETIC ALGORITHM BASED TRAMP SHIP ROUTING AND SCHEDULING

SUMMARY

Maritime transportation is the most preferred mode of transportation in trade, as it allows large volumes of cargo to be transported over long distances, is a safe mode of transportation, and is cheaper than other types of transportation.

Factors such as developments in maritime transport, variability in freight prices, unstable costs, closed financing structure of the maritime sector, and international operation of ships make maritime transport a highly sensitive and competitive sector.

Increasing of customer expectations day by day has started to move the maritime industry away from the concept of transportation. The changing direction of customer demands also increases the importance of correct planning for maritime companies. In this sector, which has large capital investments and high operating costs, companies aim to gain economic benefits by managing their existing fleets in the most efficient way.

There are many cost factors in the maritime industry. Some of these costs can be minimized with proper planning. In this sense, optimization of operational costs based on the operation of ships becomes important. Optimization of fuel consumption, which is a function of speed and the route traveled, has an important place in reducing operational costs.

Maritime transportation is divided into two parts as liner shipping and tramp shipping in terms of the voyage patterns of the vessels. In liner maritime transport, the routes that the ships will follow and the time schedules for these routes are known in advance. The important thing for this mode of transportation is the continuity of the service. Ships do not need to be fully loaded to carry out their voyages. The vessels have programs and they make the voyage in order not to interrupt the program. Container and Ro-Ro (Roll on Roll of) vessels are the ship types that are frequently used in liner transportation.

Tramp transportation is a mode of maritime transport where the routes of vessels and the cargoes they carry are not stable. The voyage routes of the ships vary according to the market conditions. In this type of transport, ships often find their cargo from voyage to voyage. Tramp transportation is based on competition. Because the voyages are not predetermined and the ships move towards the ports where the cargoes will provide profit to the companies. Therefore, in a fleet with more than one ship, it becomes important to distribute the contracts to be served by the vessels appropriately among the specified vessels, within the time intervals determined for the contracts. Especially considering the relationship between fuel consumption and speed and distance traveled, optimization of route and speed is of great importance.

Ship routing and scheduling problems are the adaptation of vehicle routing problems to maritime transport. The most important point that distinguishes ship routing problems from vehicle routing problems is that there are many additional constraints in the application phase. Ship capacities, ship dimensions, ship drafts, ship-port compatibility, compatibility of transported cargoes, time windows required for receiving and delivering cargo are examples of these constraints.

These problems are large-scale problems with many unknowns, many variables and a complex structure. These types of problems have a lot of constraints, a lot of customers that need to be served, goals that are incompatible with each other. Ship routing and scheduling problems are NP-hard problems that are difficult to solve.

In this thesis, the multi-vehicle and multi-depot routing and scheduling problem of a fleet of vessels operating as tramp is investigated with time windows. The purpose of solving the problem is to serve each existing contract and to minimize the total operational cost, taking into account the contracts created for each cargo to be transported and the time constraints determined in these contracts. At this stage, arrival, departure and waiting times for each ship are integrated into the problem.

In order to ignore the capacity constraint in the problem, it is assumed that all ships have the same deadweight (cape-size) and it is assumed that all ships can serve all contracts as long as they are at the appropriate time and location. In addition, to add heterogeneity to the problem, the speeds of the ships and the fuel consumption at the specified speeds are differentiated.

The most important characteristic of the model is discretization of cargo picking and delivery time windows. For each contract, there are 3 time windows in which the cargoes will be received and 3 time windows in which they will be delivered. This feature ensures that different costs are included in the calculation according to the early or late arrival of the ships to the port. It also allows the inclusion of a wide variety of variables and constraints in the model. The thesis focuses on measuring the effect of cruising speed and contract distribution optimization between ships to control fuel consumption, which is the main operating cost in maritime transportation.

The problem was created as a mixed integer linear programming model, taking into account the operating system of the transportation mode.

During the data collection phase, the fuel consumption of the ships, voyage times, ballast or loaded voyages were taken into consideration. Realistic data were obtained. By including the arrival-departure time windows in the problem, cost matrices for the ships were created. First of all, the distances between all ports were calculated and contract-to-contract transition matrices were prepared as much as the number of contracts in order to track the contract transitions of the ships. Contracts to be serviced are listed with priority on expiry dates. The time interval for each carriage contract to be serviced was accepted as 20 days and during this period, unsuitable ships were prevented from receiving cargo. While calculating the cost, the value of the distances between the ports in terms of time, the daily fuel expenditures of the ships at the determined speed and condition (empty/full) were used. The regions where the ships operate are ignored and the price of IFO380 per ton is calculated as 540 dollars. As experimental factors; number of contracts, time window level, number of vessels and speed variations are used.

Exact solution methods and heuristic methods can be used in solving NP-hard problems. This solution methods give optimum results in problem solving.

However, the solution quality of exact solution methods is inversely proportional to the problem volume. As the size of the problem increases, the solution time also increases exponentially according to the size of the problem and sometimes a solution cannot be produced.

Heuristic methods do not guarantee the best solution, but for large problem volumes they give near-best solutions in reasonable solution times. There may be several situations that require the use of heuristic algorithms in problems. The first case is that the problem cannot be solved by exact solution techniques. In such a case, the only option is to solve the problem with heuristic methods. The second reason is that exact solution methods take a very long time to solve large volume problems.

In the thesis, different problem volumes were created by increasing the variables such as the number of ships and the number of contracts. The problems created were firstly solved in linear programming with Gurobi solver. Then, the problem is solved with a hybrid method in which Genetic Algorithm and Ant Colony Optimization Algorithm are used together. In the hybrid method, routes are created with the Ant Colony Algorithm. Then, the created routes are improved with operators such as crossover, mutation, elitism in the Genetic Algorithm stage. Thus, the solutions approach the best.

At the end of the application, the results of the linear programming and the hybrid method were compared. The results showed that both methods give very close values in terms of cost calculation. However, since the ship routing problem is an NP-hard problem, the time to get the optimum result in exact solution methods increases with the increase in the problem volume, while the hybrid method gives better solution times as the volume and complexity of the problem increase.



1. GİRİŞ

Artan teknoloji yatırımları, dünya ticaretinde yaşanan gelişmeler, bilgi aktarımının kolaylaşması, tüketimin ve ihtiyaçların artması, kaynakların yeteriz hale gelmesi gibi hayat standartlarında meydana gelen değişimler mevcut sistemleri sürekli olarak iyileşmeye zorlamaktadır. Günlük hayatta karşılaşılan problemlerin, farklı ihtiyaç gerekliliklerine göre matematiksel modeller ile ifade edilmesi ve bu problemler için mümkün olan en iyi çözümlerin bulunması optimizasyonun ana amacıdır. Kullanım alanlarına göre sonuçların, daha hızlı, daha güvenilir, daha kararlı ya da daha az maliyetli olması problemlerin ana amacı olabilir. Optimizasyon çözümleri; fizik, biyoloji, matematik, mühendislik, veri madenciliği, tıp, üretim, tasarım ve lojistik sektörleri başta olmak üzere çok geniş kullanım alanına sahiptir (Şavli, 2016, ss. 13-20).

Optimizasyon, bir sistemde belirlenen amaç veya amaçlar için belirli kısıtlar dahilinde en uygun çözümün elde edilmesi için yapılan çalışmalar bütünüdür (Özsağlam ve Çunkaş, 2008). Matematiksel olarak ifadesi ise, modelin maksimum ya da minimum noktasını belirleyen değerlerin hesaplanması olarak yapılabilir (Yıldırım, 2014, ss. 21-37). Optimizasyon problemleri, karar değişkenlerinin yapısına göre, sürekli optimizasyon problemleri ve kesikli optimizasyon problemleri, olmak üzere iki sınıfta incelenir (Korte ve diğ, 2011). Sürekli optimizasyon problemlerinde karar değişkenleri reel sayıların bir kümesi üzerinde tanımlıdır. Bu tip problemler, tüm fonksiyonlar doğrusal ise doğrusal programlama ve en az bir fonksiyon doğrusal değil ise doğrusal olmayan programlama yöntemi ile çözülür (Özbakır ve Tapkan, 2010, ss. 29-33). Kesikli optimizasyon problemlerinde karar değişkenleri tamsayı değerler aldığı anda tamsayı programlama ile, kombinatoriyal değerler aldığı anda ise kombinatoriyal optimizasyon yöntemleri ile çözülmektedir. Kombinatoriyal yöntemde, amaç fonksiyonunu optimum hale getirecek kesikli karar değişkenlerinin hesaplanması amaçlanmaktadır (Yıldırım, 2014, ss. 21-37).

Kombinatoriyal optimizasyon problemleri de, çözüm şekillerine göre, polinom türde (kolay) problemler P ve deterministik olmayan polinom (zor) NP problemler olmak üzere ikiye ayrılır (Sallabi ve diğ., 2009, ss. 389-417). Turing makinesi, Alan Turing tarafından geliştirilmiştir ve problem sınıflarının belirlenmesinde kullanılan en önemli araçlardandır. Turing makinesi deterministik bir makinedir ve aynı girdiler için her zaman aynı sonuçları üretmektedir. P sınıfı problemler Turing makinesinde polinom zamanda çözülebilirler. NP sınıfındaki problemlerin ise, polinom zamanda çözümü mümkün değildir. Bu problemler yalnızca deterministik olmayan Turing makinesi ile çözülebilirler (Kaya, 2010, ss. 7-13). Bir problem P sınıfında ise çözümü nispeten kolaydır ve optimum çözümü hesaplanabilir. NP-hard sınıfında ise çözüm zordur ve en iyi çözümü makul zamanlarda hesaplanamaz. NP-hard tipi problemlerin çözümü için gereken süre üstel olarak artış gösterdiğinden, bu problemlerin çözümünde polinom zamanlı bir algoritma bulunmaz (Özbakır ve Tapkan, 2010, ss. 29-33).

Araç rotalama problemleri (ARP), en çok çalışma yapılan NP-hard problem tiplerinden biridir. ARP çözümünde, optimum değeri tam olarak veren kesin çözüm yöntemleri kullanılabilmektedir. Ancak, problem büyüdükçe bu yöntemler probleme cevap verememekte ya da çözümü çok uzun zaman almaktadır. Son yıllarda daha esnek çözüm imkanı ve daha yüksek performans sağlayan sezgisel algoritmalar ve doğadaki olaylardan ilham alan meta-sezgisel algoritmalar bu problemlerin çözümü için tercih edilmeye başlanmıştır (Karaboğa, 2004). Sezgisel algoritmalar en iyi çözümü garanti etmez ancak daha kısa çözüm zamanı ile optimuma yakın iyi bir çözümü elde etmeyi hedefler. Bu tekniklerin kullanılma sebebi, çok büyük boyutlu problemlerin çözümünde kullanılarak, tatmin edici sonuçlar verebilmeleridir (Şavli, 2016, ss. 13-20). Araç rotalama problemlerinin bir uzantısı olan gemi rotalama ve/veya çizelgeleme problemleri, çok bilinmezli, çok değişkenli ve karmaşık bir yapıya sahip olan büyük boyutlu problemlerdir. Denizcilik sektörünün bu komplike yapısı problem çözümlerinde sezgisel yöntemlerin kullanılmasını teşvik etmektedir.

1.1 Tezin Amacı

Bu çalışmada, düzensiz hat (tramp) taşımacılık yapan bir gemi filosunun zaman pencere, çok depolu ve çok araçlı araç rotalama ve çizelgeleme problemi incelenecektir. Rotalama terimi denizcilikte, geminin seyir halinde iken denizde takip edeceği hattın çizilmesi olarak tanımlanır. Çalışmada kullanılan rotalama terimi ise, gemilerin hizmet götüreceği kontratların (yükün bir limandan alınıp, tahliye limanına götürülmesi), filoda bulunan gemiler arasında paylaştırılarak, sıralamasının belirlenmesidir (Kiremitci ve Özçakar, 2011, ss.14-25). Kontratlarla hizmet götürülebilecek zaman aralıkları kısıtlı olduğundan, zaman boyutu da önemli bir yere sahiptir (Christiansen ve diğ., 2004). Zaman faktörü içeren rotalama, çalışmada çizelgeleme olarak tanımlanacaktır.

Gemi rotalama problemlerini, ARP'den ayıran en önemli nokta, uygulamada dikkat edilmesi gereken birçok ek kısıt olmasıdır. Gemi rotalama problemlerine eklenebilecek bazı kısıtlar; gemi kapasiteleri, gemi boyutları ve gemi draftlarının hizmet götüreceği limana uygun olup olmaması, gemilerin taşıdıkları yüklerin birbiri ile uyumlu olup olmaması, yüklere ait zaman penceresi vb.dir (Fagerholt, 2004). Araç rotalama optimizasyonlarından özellikle kara taşımacılığında büyük oranda faydalanılmasına rağmen, deniz taşımacılığının, az görünür, çok bilinmezli, geleneksel ve kapalı yapısı nedeniyle sayısal yöntemlerin denizcilikte uygulanabilmesi zaman almaktadır (Kiremitci ve Özçakar, 2011, ss. 14-25). Günümüzde daha çok inter-modal, multi-modal taşımacılık gibi entegre lojistik sistemlerinin bir parçası olan gemilerin planlamasında tercih edilmektedir. Denizcilik sektöründe, taşıma ve işletme maliyetlerinin fazlalığı düşünüldüğünde, rotalama ve çizelgeleme süreçlerinin iyileştirilmesi, şirketlerin özellikle ulaştırma masraflarını düşürerek büyük faydalar elde etmelerini sağlayabilir (Ronen, 1983).

Bu çerçevede yapılacak uygulama ile aşağıdaki araştırma sorularına cevap aranacaktır:

- Gemilerin optimal taşıma rotasının belirlenmesinde hangi bilimsel yaklaşımlar daha uygundur?
- Bilimsel metot, dağıtım rotaları belirlendiğinde tüm kontratlara optimal dağıtım sürelerinde belirlenen zaman aralığında hizmet götürülebilmekte midir?
- Bilimsel metot kullanılarak optimal dağıtım rotaları belirlendiğinde dağıtım maliyetleri azalmakta mıdır?

1.2 Tezin Etkisi

Deniz taşımacılığındaki hızlı gelişmeler, navlun fiyatlarındaki değişkenlik, stabil olmayan gemi maliyetleri, denizcilik sektörünün kapalı finansman yapısı, gemilerin uluslar arası çalışıyor olması gibi nedenler, denizyolu taşımacılığını oldukça hassas ve rekabetçi bir sektör haline getirmektedir (Hoffman, 2010). Denizcilik sektörü günümüzde sadece taşımacılık kavramından uzaklaşmaya başlamıştır. Müşteri beklentileri günbegün artmaktadır. Artık denizcilik firmalarından, depolama, gümrükleme ve son taşımaya kadar tedarik zincirini üstlenmesi beklenmektedir. Müşteri taleplerinin değişen yönü denizcilik firmaları için doğru planlamanın önemini de artırmaktadır (Türe, 2021). Büyük sermaye yatırımları ve yüksek işletme maliyetleri olan bu sektörde, şirketler mevcut olan filolarını en verimli şekilde yöneterek, ekonomik faydalar elde etmeyi amaçlamaktadır. Denizcilik sektörü yapısı itibari ile içerisinde çok sayıda maliyet faktörünü barındırmaktadır. Bu maliyetlerden bazılarının kaçınılması mümkün değil iken, bazı maliyetler uygun yöntemlerin bulunması ile minimuma indirilebilir. Operasyonel maliyetler, gemilerin günlük operasyonları ile ilgili olan, yapılan seferin özelliklerine (mesafe, seyir hız, rota vb.), yakıt tüketimine, liman ücretlerine ve kanal geçiş ücretlerine göre değişkenlik gösteren maliyetlerdir (Gatica ve Miranda, 2011). Özellikle, denizcilikteki en büyük maliyet kalemlerinden biri olan yakıt tüketiminin optimize edilebilmesi ile operasyonel maliyetler azaltılarak kar arttırılabilir. Günümüzde denizcilik firmaları, filolarının rotalama ve çizelgelemeleri için nadiren optimizasyon tabanlı karar destek sistemleri kullanmaktadırlar. Ancak, gelişen dünya ve artan istekler ile birlikte ileri karar destek sistemlerine olan ihtiyaç artmaya devam edecektir.

Gemi rotalama probleminin en temeli, ARP'nin de temeli olan, birçok alanda genişletilmiş, klasik bir kombinatoriyal optimizasyon problemi olan gezgin satıcı problemidir (Remer ve Malikopoulos, 2019). Rota planlama problemlerini sürekli yinelemeler yoluyla çözen, genetik algoritma (GA) karınca kolonisi optimizasyonu algoritması (KKOA) ile elde edilen ilk çözümler kademeli olarak optimal çözümlere yaklaşmaktadır (Zhaou ve diğ, 2017). Bununla birlikte, her iki algoritmanın ayrı uygulanması ile elde edilen nihai çözümler gemi rotalaması için genellikle yerel optimal çözümlerdir (Liang ve Wang, 2020).

Özellikle, hizmet götürülen birçok liman ve karşılanması gereken birçok kısıt bulunan büyük ölçekli yol planlama problemleri açısından, hem GA hem de KKO, yerel optimal çözümlere eğilimlidir.

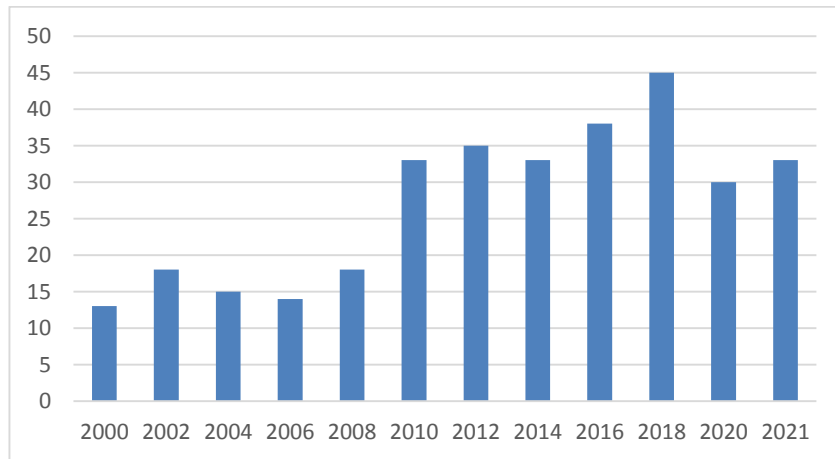
Çalışmada, yöntemlerin dezavantajların üstesinden gelmek, maliyetleri azaltmak ve her iki algoritmanın birbirini tamamlaması için zaman pencereli, çok araçlı ve çok depolu gemi rotalama problemine genetik algoritma (GA) ve karınca kolonisi optimizasyonu algoritması (KKO) meta-sezgisel yaklaşımları hibrit bir yöntem olarak uygulanacaktır.

1.3 Literatür Taraması

1.3.1 Genetik algoritma yaklaşımı

1.3.1.1 Araç rotalama problemlerinde genetik algoritma

Genetik algoritma ilk olarak, Holland (1975) tarafından tanıtılmıştır. Holland'ın algoritmayı geliştirmedeki ana düşüncesi, çevreye uyum sağlayacak bir program yaratmaktır (Goldberg, 1988). Genetik algoritmanın endüstride ilk olarak bir iletişim ağının tasarımını optimize etmek için kullanılmıştır (Davis, 1987). Genetik algoritma 70'li yıllarda geliştirilmiş bir yöntem olmasına rağmen, araç rotalama probleminin çözümünde uygulaması 90'lı yılların başına denk gelmektedir (Savelsbergh, 1992; Thangiah ve diğ., 1991; Kadaba, 1990). Yöntem zaman içerisinde farklı tip araç rotalama problemlerinin çözümünde kullanılarak en çok uygulanan tekniklerden biri haline gelmiştir. Şekil 1.1'de son yirmi yıla ait, araç rotalama problemi çözümünde genetik algoritma uygulaması için yayınlanan makalelerin sayısı gösterilmektedir.



Şekil 1.1 : 2000-2021 Yılları ARP için GA çalışmaları.

ARP için GA yaklaşımı incelendiğinde, yöntemin en çok zaman pencereli araç rotalama problemlerinin (ZPARP) çözümünde kullanıldığı görülmüştür. Jung ve Haghani (2001) zamana bağlı ARP için matematiksel bir model formüle etmiş ve çözümü için GA kullanmıştır. Tan ve diğ. (2001) zaman penceresi kısıtlamaları olan ve sınırlı kapasiteye sahip araç rotalama problemlerini çözmede dağıntık genetik algoritmalar olarak bilinen nadir bir genetik algoritma sınıfı üzerine araştırma yapmışlardır. Aynı yıl yapılan bir başka çalışmada ise, zaman pencereli ve sınırlı kapasiteye sahip araç rotalama probleminin çözümünde GA ile yerel arama yöntemi hibrit olarak kullanılmıştır (Chen ve diğ, 2001). Hwang (2002) zaman kısıtlı bir araç rotalama problemi için, GA operatörleri ve bir başlangıç popülasyonu geliştirmiştir. Sonuçlar, yöntemin ARP'nin çözümünde potansiyel olarak yararlı olabileceğini göstermiştir. Bir diğer çalışmada, zaman pencereli, 2 Boyutlu Öklid ARP için, yerel optimizasyon algoritmasının genetik algoritma ile kullanıldığı hibrit bir genetik algoritma önerilmiştir (Jung ve diğ, 2002). Zhu (2003) zaman pencereli araç rotalama problemini optimuma yakın çözümlere çözmek için uyarlanabilir bir genetik algoritma sunmuştur.

Mourkousis ve diğ. (2003) dağıtım maliyetlerini en aza indirmeyi amaçlamış, çözüm verimli hale gelene kadar çeşitli genetik operatörler, seçim ve değiştirme yöntemleri test ederek, makul bir sürede optimuma yakın çözümler sağlamak için “Yüksek Performanslı Bilgi İşlem” uygulamıştır. Alverna ve diğ. (2003) GA ve karma tamsayı programlama formülasyonu kullanarak ZPARP için sezgisel bir yaklaşım önermektedir. Bir başka makale, yumuşak zaman pencereli araç rotalama problemi için bir uygulama sunarak, çok amaçlı kombinatoriyal optimizasyon için GA'nın etkinliğinin bir araştırmasını anlatmaktadır (Geiger, 2003). Berger ve Barkaoui (2004) zamansal kısıtlama esnemesine tabi olan iki çözüm popülasyonunun eşzamanlı evrimine dayanan hibrit bir GA versiyonu sunmuştur.

Tong ve diğ. (2004) belirsiz araç sayısı ile ZPARP'nin matematiksel bir modelini sunmaktadır. Zaman pencereleri bulunan diğer bir çalışmada, dinamik toplama ve dağıtım problemini çözmek için grup genetik algoritma tekniği geliştirilmiştir (Pankratz, 2005). Ombuki ve Hanshar (2006) ZPARP'ni çok amaçlı bir problem olarak temsil etmiş ve pareto sıralama tekniğini kullanarak bir genetik algoritma çözümü sunmuşlardır. Alvarenga ve diğ. (2007) bir genetik algoritma ve bir küme bölümlleme formülasyonu aracılığıyla ana hedef olarak seyahat mesafesini kullanan

ZPARP için bir yaklaşım önermiştir. Wang ve diğ. (2008a) GA'nın başlatılmasına yardımcı olabilecek müşteri kümeleme atama algoritmasını tasarlamışlardır. Takip eden çalışmada, mevcut GA'daki erken yakınsama kusurlarının üstesinden gelmek için, yumuşak zaman pencereli araç rotalama probleminin çözümü için yeni çaprazlama operatörü ve yeni çaprazlama stratejisi ile geliştirilmiş bir GA sunulmuştur (Wang ve diğ, 2008b). Xu ve diğ. (2008) yumuşak zaman pencereleri ve bulanık talebe sahip bir ARP için GA yaklaşımını kullanmışlardır.

Cheng ve Wang (2009) ZPARP'yi bir kümeleme problemine (ana problem) ve zaman penceresi kısıtlamaları olan bir dizi gezgin satıcı problemine (alt problemlere) ayırmak için bir ayrıştırma tekniği kullanmışlardır. Dursun (2009) belirli bir zaman aralığında müşterilere hizmet vermek amacıyla hızlı ve maliyetleri minimize eden bir çözüm için problemi modelleyerek, gerçek verilerle test etmiştir. Çalışmada geliştirilen GA'ya birey kodlama özelliği eklenmiş, böylece algoritmanın hızlı ve etkili çalışması sağlanmıştır. Ghoseiri ve Ghannadpour (2010) çalışmalarında, karar vericinin hedeflere yönelik istek düzeylerini belirlediği ve bu isteklerden sapmaların en aza indirildiği, hedef programlama ve genetik algoritma kullanan zaman pencereli çok amaçlı araç rotalama problemi incelemiştir.

Jin ve Fu (2010) başlangıç popülasyonu yapmak için öncelik kuralı buluşsal yöntemiyle ve popülasyonun çeşitliliğini artırmak ve yerel aramayı geliştirmek için λ -değişim mekanizmasıyla birleştirilmiş bir hibrit genetik algoritma önermiştir. Ursani ve diğ. (2011) lokalize optimizasyon çerçevesini tanıtmıştır. Bu çerçeve, optimizasyon ve de-optimizasyon olmak üzere iki aşama arasında yinelemeli bir prosedürdür. Test etmek için, optimizasyon metodolojisi olarak bir GA ve bir etki alanı olarak ZPARP seçilmiştir. Bu yeni şemaya lokalize genetik algoritma adı verilmiştir. Nyugen ve diğ. (2011) zaman pencereli periyodik araç rotalama problemi için popülasyon tabanlı yeni bir hibrit meta-sezgisel önermiştir. Çözüm kalitesini artırmak amacıyla bir çaprazlama operatörü tarafından üretilen yavruları eğitmek için iki komşuluk tabanlı metasezgiseller kullanılmıştır.

Wang ve Chen (2012b) zaman pencerelerinde eş zamanlı teslimat ve toplama problemini incelemişlerdir. Problem için ikili karma tamsayı programlama modeli geliştirilmiş ve doğrulanmıştır. Liu (2013) araç sayısı hedefinin verimliliğini artırmak için, makalede zaman pencereli çok depolu araç rotalama problemini ele almıştır. Çok depolu araç rotalama probleminin çözüm süreci için Yapay Arı Kolonisi Algoritması

(YAKA) dayalı bir GA geliştirilmiştir. Bir diğer çalışmada, Vidal ve diğ. (2013) zaman kısıtlamalı geniş bir ARP sınıfı için gelişmiş çeşitlilik kontrolü ile verimli bir hibrit genetik arama sunmuştur. Zaman penceresi ve süre kısıtlamaları ile ilgili olarak cezalandırılmış uygulanabilir olmayan çözümleri hesaba katan ve amortismanlı sabit zamanda yay veya düğüm değişimlerine dayalı herhangi bir klasik komşuluktan hareketlerin değerlendirilmesine izin veren yeni hareket değerlendirme teknikleri önerilmiştir. Kumar ve diğ. (2014) tarafından, çok amaçlı problemin çözümü için uygunluk toplama yaklaşımına sahip genetik algoritma ve toplam uygunluk değerine dayalı seçim, toplulaştırılmış genetik algoritma adı verilen en iyi maliyetli rota çaprazlaması gibi özelleşmiş operatörler tanıtılmıştır.

Nguyen ve diğ. (2014) ZPARP çözmek için GA ve komşuluk tabanlı meta sezgisel arama yöntemlerini kullanmışlardır. Xu ve diğ. (2015) ZPARP için genetik algoritma ve parçacık sürü optimizasyonunun (PSO) bir kombinasyonu önermiştir. Bir başka çalışmada Li ve diğ. (2015) meyve sebze taşımacılığı için ZPARP uygulamıştır. Aynı zamanda yol etkisini de dikkate alan doğrusal olmayan bir matematiksel model geliştirilmiştir. Kumar ve Panneerselvam (2015) zaman pencereli araç rotalama probleminin uzantısı olan zamana bağlı araç rotalama problemi uygulanmasını, GA metasezgisel yöntemi kullanılarak gerçekleştirmiştir. Kumar ve Panneerselvam (2016) zamana bağlı araç rotalama problemi için kat edilen toplam mesafeyi en aza indirmek, kullanılan toplam araç sayısını en aza indirmek ve ayrıca trafik sıkışıklığından kaçınmak için alternatif rotalar önermek için GA tabanlı verimli bir algoritma geliştirmiştir.

Bouchra ve diğ. (2016) yumuşak zaman pencereli araç rotalama problemini statik ve dinamik zaman pencereleri kısıtlarını ekleyerek genetik yöntemin ve değişken komşuluk arama yönteminin hibridizasyonunu önermiştir. Shi ve diğ. (2017) zaman pencereli, bulanık talepli bir problemi ele almış ve bir bulanık şans kısıtı modeli tasarlamıştır. Önerilen modeli çözmek için stokastik simülasyon yöntemleri ile entegre bir hibrit genetik algoritma önermiştir. Gökçen ve diğ. (2017) müşteri talepleri bilinen, tek depolu ve özdeş araçlara sahip ZPARP'yi ele almışlardır. Genetik algoritmanın ilk popülasyon oluşturma aşamasında süpürme algoritması kullanılarak yüksek kaliteli çözüm setleri ile arama yapılmıştır. Pierre ve Zakaria (2017) GA kullanarak zaman pencereli çok amaçlı araç rotalama probleminin optimizasyonu için stokastik bir çevrimsel kaydırma operatörü sunmuştur.

Rabbani ve diğ. (2018) makalelerinde, iki onarım ve toplama aracını göz önünde bulundurarak zaman penceresi ile çok depolu araç rotalama problemini ele almaktadır. En iyi çözümü bulmak için basit genetik algoritma ve tavlama benzetim hibrit genetik algoritma olmak üzere iki meta-sezgisel algoritma kullanılmıştır.

Yaktubay (2018) ZPARP'yi çözmek için araçların toplam mesafesinin ve bekleme süresinin minimizasyonu içeren çok amaçlı bir genetik algoritma yaklaşımı önermiştir. GA adımlarında bireylerin değerlendirilmesi, sıralanması ve seçilmesinde çok amaçlı optimizasyon tekniklerinden biri olan NSGA-II kullanılmıştır. Çalışmasında, başlangıç popülasyonunun etkisini analiz etmek için çok amaçlı GA'da başlangıç popülasyonu üretimi aşamasında üç farklı yöntem kullanılmıştır. Başlangıç popülasyonları ilk olarak rasgele, ikinci olarak en yakın komşu tabanlı bir algoritma ile ve üçüncü olarak da süpürme tabanlı bir algoritma ile oluşturulmuştur. Oluşturulan üç algoritma, Solomon'un karşılaştırma problemleri üzerinde test edilmiştir. Başlangıç popülasyonu süpürme tabanlı algoritma ile oluşturulan GA ile daha etkili sonuçlara ulaşıldığı görülmüştür. Bu çalışmanın amacı, GA ile elde edilen sonuçlarda başlangıç popülasyonunun etkisini ortaya koymak ve ZPARP çözümü için karşılaştırmalı bir yaklaşım sunmaktır.

Thangiah (2019) ZPARP'yi çözmek için genetik algoritma, tavlama benzetim ve tabu arama yöntemlerini birleştiren bir hibrit arama stratejisi geliştirmiştir. Li ve Ma (2019) makalede, değiştirilmiş stokastik en yakın komşu ve ekleme tabanlı yerel aramayı içeren problem için hibrit çok amaçlı bir genetik algoritma kullanmıştır. Hesaplama sonuçları, hibridizasyonun olumlu etkisini ve meta-sezgisellerin tatmin edici performansını göstermektedir. Khoo ve diğ. (2020) iki aşamalı dağıtılmış bir yık ve yeniden oluştur genetik algoritması kullanan yeni bir sistematik çerçeve önermektedir. Zhen ve diğ. (2020) zaman pencereleri ve çıkış tarihleri ile çok depolu çok seferli bir araç rotalama problemi ele almıştır. Problem karma tamsayı programlama modeli olarak formüle edilmiştir ve problemi çözmek için hibrit PSO ve hibrit GA geliştirilmiştir. Çok amaçlı araç rotalama problemini zaman pencereleriyle çözmek, iki veya daha fazla amacın yerine getirilmesini içerir.

Khoo ve Mohammad (2021) çalışmalarında iki aşamalı hibrit bir yık ve yeniden yarat genetik algoritma uygulaması önermiştir. İbrahim ve diğ. (2021) problemin tanımına dayalı olarak, zaman pencereleri bulunan teslim alma ve teslim etme probleminde en uygun çözümü bulmak için GA kullanmışlardır.

Çaprazlama ve mutasyon aşamalarını ayarlayarak bu problemleri çözmek için optimize edilmiş bir genetik algoritma geliştirilmiştir. Genetik algoritma yaklaşımı, diğer ARP’lerde özellikle de kapasite kısıtlı araç rotalama problemlerinin çözümünde sıklıkla tercih edilmektedir (Sbai ve diğ., 2020; Ruiz ve diğ., 2019; Uyumaz, 2017; Şen, 2014; Günay, 2013; Çiçekdeş, 2011).

1.3.1.2 Gemi rotalama problemlerinde genetik algoritma

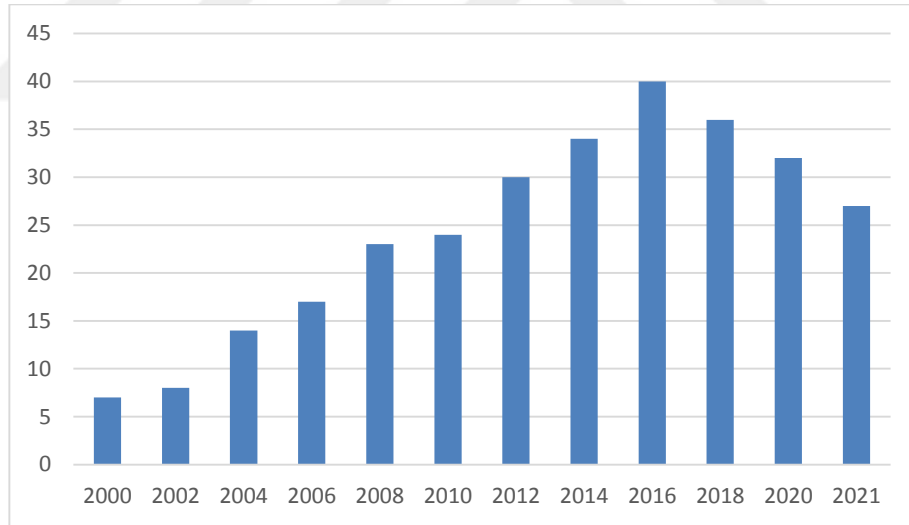
Böttner ve diğ. (2007) GA ile graph teori ve deterministik hesaplamaları birlikte kullanarak tehlike durumunda hava ve deniz koşullarına bağlı olarak en uygun rotanın bulunması için bir karar mekanizması geliştirmişlerdir. Kosmos ve diğ. (2008) optimal gemi rotası problemiyle başa çıkmak için GA ve olasılık dağılım algoritmasının tahminine dayalı hibrit bir evrimsel yöntem tasarlamış ve uygulamışlardır. Hinnenthal ve Gunther (2010) hava tahmin raporlarını kullanarak gemilerin deniz ve hava şartlarına göre hızını en az etkileyecek rotayı bulmak için GA temelli bir “robust pareto optimum” rotalama sistemi geliştirmişlerdir. Kang ve diğ. (2011) deniz taşımacılığının verimliliğini artırmak için limanlar arası rotaları çözümlemeye çalışmış bu kapsamda GA’ya dayalı bir meta-sezgisel algoritma geliştirmişlerdir. Maki ve diğ. (2011) NOAA ve Wave Watch III servislerinden aldığı hava raporlarını kullanarak Atlantik Okyanusu için test edilen bir RCGA tabanlı bir rotalama sistemi geliştirmişlerdir. Kiremitci (2011) denizyolu yük taşımacılığında karşılaşılan gemi rotalama ve çizelgeleme problemlerinin var olan araç rotalama yöntemleri içinde en çok “Çok Araçlı Zaman Pencereli Toplama ve Dağıtım Problemine” benzerlik gösterdiğini belirleyerek GA ile bir çözüm yaklaşımı geliştirmiştir. Yaptığı analizler sonucunda genetik algoritma ile %22 tasarruf sağladığını belirlemiştir. Kepaptsoglou ve diğ. (2015) GA kullanarak Ege denizi için hava ve deniz koşullarını dikkate alan filo yönlendirme ve rotalama çalışması önermişlerdir. Vettor ve Soares (2016) genetik algoritmayı kullanarak Atlantik Okyanusu ve Akdeniz için SWAN, MM5 ve WAM dalga verileri alınarak geliştirilen dalga modelleri ile bir rotalama sistemi geliştirmişlerdir. İnan (2019) yaptığı çalışmada, hava ve deniz şartlarına göre optimal rotayı belirleyebilen AGRS (Akıllı Gemi Rotalama Sistemi) geliştirmiştir. Pan ve diğ. (2021) yakıt tasarrufu için hava koşullarına bağlı uygun rotanın seçimi için gemi rotasını optimize etmek için orijinal GA’ya dayalı bir yöntem geliştirmişlerdir. Zhao ve diğ. (2021) yaptıkları çalışmada bir gemiyle ilişkili meteorolojik riski, yakıt tüketimini ve seyir süresini optimize etmeyi amaçlamış, geliştirilmiş parçacık sürüsü

optimizasyonu ve GA'ya dayalı, çok kriterli gemi rota planlama yöntemi önermişlerdir. İnan ve Baba (2021), ticari gemilerin rota optimizasyonunu ele alarak deniz ve hava şartları için GA kullanarak, Inmarsat ve Copernicus sistemlerinden aldığı veriyi CSGA algoritması ile hesaplayan bir karar destek sistemi geliştirmişlerdir.

1.3.2 Karınca kolonisi optimizasyonu algoritması

1.3.2.1 Araç rotalama problemlerinde karınca kolonisi optimizasyonu

Karıncaların besine giden en kısa yolu bulmak için gerçekleştirdikleri davranıştan esinlenerek geliştirilen KKO, ilk olarak Dorigo ve arkadaşları tarafından 1991 yılında kullanılmıştır (Dorigo ve diğ, 1991). Araç rotalama problemlerinin çözümünde ilk kez 1999 yılında Gambardella tarafından kullanılmıştır. Yaygın olarak kullanılmaya başlanması ise 2000'li yılları bulmuştur. Şekil 1.2'de 2000-2021 yıllarına ait, araç rotalama problemi çözümünde karınca kolonisi algoritması uygulaması için yayınlanan makalelerin sayısı gösterilmektedir.



Şekil 1.2: 2000-2021 Yılları ARP için KKO çalışmalarını.

Donati ve diğ. (2002) zaman pencereli araç rotalama problemlerinin çözümü için KKO tabanlı bir çözüm önerisi geliştirmişlerdir. Gökçe (2004) Eş Zamanlı Dağıtım ve Toplamalı Araç Rotalama ile Zaman Kısıtlı Araç Rotalama Problemlerini aynı özelliklere sahip araçlar için ele almıştır. Karınca kolonisi optimizasyonu ile geliştirdiği bir algoritmayı problemlere uygulayarak çözümleri test etmiştir.

Dorigo tarafından önerilen biçimde rotalar oluşturmuş elde edilen rotaları 2-opt algoritması ile iyileştirmiştir.

Zhang ve diğ. (2006) zaman pencereleri ve yeniden kullanılan araçlar ile ARP için araç sayısını ve toplam teslimat süresini en aza indirecek KKOA tabanlı çok amaçlı bir model önermiştir. Taşkiran (2006) çalışmasında, Zaman Kısıtlı Araç Rotalama Problemini, araç sayısı ve toplam kat edilen yolun minimize etmek için ele almıştır. Geliştirilen algoritma Solomon (1987) tarafından geliştirilen 56 problem üzerine uygulanarak test edilmiştir. Elde edilen sonuçlar literatürdeki araç sayısı ve araçların toplam yol uzaklıkları konusunda bir iyileştirme geliştirilememiştir.

Kılıç (2008) çalışmasında, ARP için bulanık kümeler ve olasılık teorileri ile geliştirdiği bir model önermiştir. yüksek veri işleme maliyetini minimize eden modelin çözümü için karınca kolonisi optimizasyonu ile bir algoritma geliştirmiştir. Ahmmed ve diğ. (2008) zaman pencereli dinamik ARP için KKOA tabanlı bir çözüm geliştirmişlerdir.

Yılmaz (2008) KKOA'yı birden çok depo kullanan işletmelerde araç filosunun müşterilere gitmek için kullanacağı en kısa yolu belirlemeye çalışmıştır. Yıldırım, çalışmasında zaman-bağımlı ve zaman-bağımsız, sıkı zaman kısıtlı ARP ele alarak elde edilen algoritma ilgili problemlerde test etmiş ve performansı gözlemiş, literatürdeki sonuçlarla karşılaştırma yapmıştır. Kurt (2008) Çoklu Depolu Araç Rotalama Problemlerini birden çok araç ile birçok müşteri için talepleri iletip depolarına dönecek biçimde tur planlayacak biçimde ele almıştır.

Yu ve diğ. (2009) ARP çözümü için karınca ağırlığı stratejisine ve mutasyon işlemine sahip bir KKAO temelli bir çözüm sunmuşlardır. 14 kıyaslama probleminin hesaplama sonuçları, önerilen çözüm yöntemlerinin etkili ve verimli olduğunu ortaya koymuştur. Kesintürk (2009) çalışmasında, farklı tipteki araç rotalama problemleri için global karınca koloni optimizasyonu yöntemi ile bir çözüm önermiştir. Geliştirilen algoritma, literatürdeki problemlere uygulanarak test etmiş ve sonuçları var olan karınca koloni algoritmalarıyla karşılaştırmıştır. Ekizler (2011) çalışmasında Kapasite Kısıtlı Araç Rotalama Problemini (KKARP) ele alarak KKOA ile çözüm önerisi geliştirmiştir.

Çalışkan (2011) çok depolu araç rotalama problemlerinde müşteri taleplerinin en kısa sürede, en kısa yoldan karşılanması ve uygun araç sayısının belirlenmesi problemini incelemiştir.

KKOA ile ortalama kümeleme tekniğini birleştirerek taleplerin toplam gerçekleşme süresi, toplam mesafe ve kullanılan toplam araç sayısının değişimlerini izlemiştir. Michalis ve diğ. (2012) değişikliklerin döngüsel bir modelde meydana geldiği trafik faktörü ile dinamik araç rotalama problemi için KKO A'yı ele aldıkları çalışmalarında, ortamlardan en iyi çözümleri toplamak ve bunları dinamik bir değişim meydana geldiğinde çeşitlilik oluşturmak ve bilgi aktarmak için kullanmışlardır. Sonuçlar, uygun bir bellek boyutu ve göçmen değiştirme oranı ile önerilen algoritmanın, farklı test durumlarında diğer eş KKO A algoritmalarından daha iyi performans gösterdiğini göstermiştir.

Yao ve diğ. (2014) çok depolu bir açık araç yönlendirme probleminden merkezi depolu açık araç rotalama problemine çevirerek problemin karmaşıklığını azaltan bir yöntem kullanmıştır. Ardından, sorunu çözmek için karınca kolonisi optimizasyonu kullanmıştır. Algoritmanın performansını iyileştirmek için çapraz işlem ve bazı uyarlanabilir stratejiler kullanmış, bazı problemlerde test ederek sonuçları karşılaştırmıştır. Kaya (2017) Eş Zamanlı Topla Dağıt Araç Rotalama Problemini çözmek üzere Karınca Koloni Sistemi ve Değişken Komşuluk Araması ile oluşturulan melez bir algoritma geliştirmiştir. İki yöntem karşılaştırılmış KKO A'nın çözüm kalitesi ve CPU süresi üzerinde olumlu etkili olduğu görülmüştür.

1.3.2.2 Gemi rotalama problemlerinde karınca kolonisi optimizasyonu

Li ve diğ. (2018) KKO A'yı, okyanus aşırı gemiler için en uygun rotayı hızlı, verimli ve doğru bir şekilde bulmak amacıyla kullanmışlardır. Algoritma yakınsama hızını artırmak ve yerel optimumdan kaçınmak için feromon güncelleme kuralı geliştirmişlerdir.

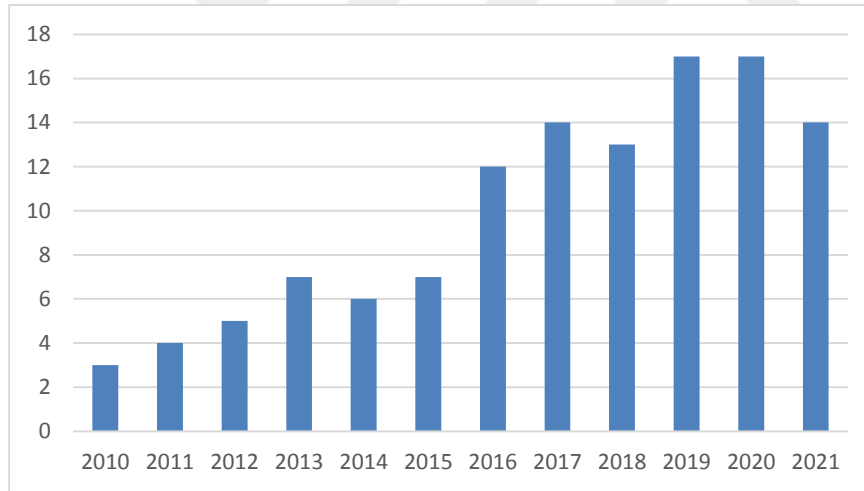
Zhang ve diğ. (2021) navigasyon güvenliği, yakıt tüketimi ve seyir süresi göz önünde bulundurularak, çok amaçlı gemi hava güzergahı optimizasyon problemini çözmek için geliştirilmiş KKO A uygulamışlardır. KKO A algoritmasının iyileştirilmesi esas olarak iki yönden yansıtılmaktadır. İlk olarak, klasik KKO A algoritmasını uzun mesafeli gemi hava güzergahı için daha uygun hale getirmek ve daha yumuşak bir rota planlamak için, algoritmanın temel parametreleri geliştirildi ve yeni kontrol faktörleri tanıtıldı. İkinci olarak, çok amaçlı problemleri çözmek için algoritma tarafından üretilen çok az sayıda Pareto baskın olmayan çözümün durumunu iyileştirmek için, genetik algoritmada çapraz geçiş, rekombinasyon ve mutasyonla ilgili işlemler,

geliştirilmiş KKO A algoritmasına dahil edilir. Nihai simülasyon sonuçları, geliştirilmiş algoritmanın çok amaçlı hava durumu yönlendirme optimizasyon problemlerini çözmedeki etkinliğini kanıtlamaktadır. Ek olarak, bir yolculuk sırasında geminin yakıt tüketimini incelemek için kara kutu modeli yöntemi kullanıldı; model yapay bir sinir ağına dayalı olarak oluşturuldu. Sinir ağı modelinin parametreleri, test gemisinin geçmiş navigasyon verileri aracılığıyla tekrar tekrar rafine edildi ve daha sonra, test gemisinin gelecekteki yakıt tüketimini tahmin etmek için eğitilmiş kara kutu modeli kullanıldı. Diğer yakıt tüketimi hesaplama yöntemleriyle karşılaştırıldığında, kara kutu modeli yöntemi daha yüksek doğruluk ve uygulanabilirlik göstermiştir.

1.3.3 GA ve KKO A hibrit yaklaşımı

1.3.3.1 Araç rotalama problemlerinde GA ve KKO A

Yeni bir yaklaşım olması sebebiyle, bu bölümde hibrit yöntemin 2010-2021 yılları arasındaki gelişimi detaylı olarak incelenmiş ve Şekil 1.3'te gösterilmiştir.



Şekil 1.3: 2010-2021 yılları ARP için GA-KKO A hibrit çalışmaları.

Silva (2005) çabuk bozulabilen malların zamanında üretimi ve dağıtımı, tedarik zinciri yönetimindeki önemi üzerine genetik algoritmaları yapıcı buluşsal meta-sezgisel yöntemlerden karınca kolonisi optimizasyon yaklaşımı ile birleştirerek yeni bir yaklaşım sunar. Simülasyon örnekleri, somut tedarik zincirinin yeni GA-KKO A algoritması ile performansı artırdığını göstermektedir. Gao ve diğ. (2010) gezgin satıcı probleminin çözümü için GA-KKO A'nun birlikte kullanıldığı bir çözüm yöntemi geliştirmiş, elde ettikleri sonuçlar ile hibrit yöntemin daha iyi sonuçlar verdiğini göstermişlerdir.

Dikmen ve diğ. (2014) yaptıkları çalışmada, araç rota planlama problemlerinden biri olan gezgin satıcı problemi (GSP) için çözüm geliştirmeye çalışmışlardır. Karınca kolonisi ve genetik algoritmalar yardımıyla geliştirilen yapay zeka teknikleri uygulanarak performansları karşılaştırılmıştır. Yapılan çalışmada Türkiye haritası kullanılmış ve en iyi rota planı için iki farklı algoritma geliştirilmiş ve uygulanmıştır. Elde edilen sonuçlar en kısa mesafeyi bulma ve rota hesaplama süresi bakımından karşılaştırılmıştır. C# tabanlı bir ara yüz tasarımıyla test edilen algoritmalar sonucunda karınca kolonisi algoritmasının rota mesafesi ve başarımlı süresi bakımından genetik algoritmadan daha iyi sonuçlar verdiği belirlenmiştir.

Kuzu ve diğ. (2014) NP-zor problem sınıfı olarak tanımladıkları gezgin satıcı problemini (GSP), metasezgisel yöntemlerle çözümlerini karşılaştırmışlardır. 8 metasezgisel yöntem literatürdeki farklı 18 gezgin satıcı problemine uygulanmış, sonuçlar karşılaştırılarak yorumlanmıştır. GA ve KKOA yöntemlerinin birlikte uygulandığı her problem için bulduğu en iyi çözümlerin ve her bir problem için optimuma en yakın sonuçları verdiği belirlenmiştir. Zhao ve diğ. (2016) tedarikçi seçim problemini çözmek için hem GA'nın hem de KKOA'nun faydalarını sağlayan yeni bir yöntem önermişlerdir. Önerilen yöntem, GA'ların evrimsel etkisini ve KKOA'nın işbirlikçi etkisini birleştirir.

Şenaras ve İnanç (2017) ARP için en kısa yol problem türlerinde düğüm sayısı fazla olduğunda problemin çözümünün imkansız olabileceğini belirterek en etkin çözüm yöntemlerinin sezgisel yöntemler olduğunu vurgulamıştır. Karınca kolonisi optimizasyonunun GSP için en başarılı yöntemlerden biri olduğu ve örnek bir kargo şirketinin 20 ilden oluşan araç rotası için uygulanmıştır. Aynı problem daha az düğüm sayısı ile karınca kolonisi optimizasyonu algoritmasıyla yeniden çözülerek optimal çözüme ne kadar yakınsadığı sezgisel olmayan başka yöntemlerle de test edilmiş ve karşılaştırılmıştır. Luan ve diğ. (2019) GA ve KKOA'na dayalı çok kriterli bir tedarikçi seçimi problemini çözmek için GA ve KKOA'nın birlikte kullanıldığı hibrit bir algoritma geliştirmiştir. Algoritmada GA'nın küresel yakınsama oranı avantajlarını ve KKOA ile paralellik ve etkili geri bildirim özelliklerini birleştirir. Parametreleri optimize etmek, orijinal ve hibrit algoritmaların performansını analiz etmek ve karşılaştırmak için yapmış oldukları deneyler sonucunda yöntemlerinin, uygulanabilirliğini ve etkililiğini doğrulayan yeni entegre algoritmanın kalite ve verimliliği iyileştirdiğini göstermişlerdir.

Khanra ve diğ. (2019) çok amaçlı optimizasyon problemlerinin ve özellikle gezgin satıcı problemlerinin çözümü için GA ve KKO'A'yu bir arada kullandıkları çalışmalarında elde ettikleri sonuçlar ile hibrit çalışmanın daha iyi bir analiz metodu olduğunu ortaya koymuşlardır. Faust ve diğ. (2020) GA ve KKO'A'yu birlikte kullanarak kapasiteli araç rotalama problemi için en uygun çözümü bulmaya çalışmışlardır.

Tsou ve Cheng (2013) Atlantik Okyanusu için Inmarsat hava raporlarını kullanarak GA ile KKO'A'yu kullanarak melez bir yöntem geliştirmiştir. Araştırmada gemiler karınca, optimum seyir rotaları yem yolları ve hava koşulları da engeller olarak kabul edilir. Karıncaların yem yolunu bulma davranışı taklit edilerek en uygun yolun bulunması için uygulamışlardır. Bu dönüştürme yaklaşımıyla, optimum gemi rotasının elde edilmesinde hesaplama verimliliğinin ve küresel optimum arama yeteneğinin geliştirildiği belirlenmiştir.

2. TRAMP TAŞIMACILIK VE DENİZ TİCARETİNDEKİ YERİ

Deniz yolu taşımacılığı, büyük hacimli yüklerin, uzak mesafelere ulaştırılması imkânını vermesi, güvenli bir taşıma şekli olması ve finansal yönden diğer taşıma modlarına göre çok daha uygun olması gibi avantajları nedeniyle ticarete en çok tercih edilen taşıma şeklidir. UNCTAD (United Nations Conference on Trade and Development), deniz ticaret hacminin 2019 yılında, 10,7 milyar tona ulaştığını belirtmektedir. 2020 yılında, (Covid-19) pandemisinin etkisiyle deniz ticaret hacmi yaklaşık, %4.1'lik bir düşüş yaşasa da, deniz taşımacılığı dünya ticaretinin bel kemiği olmaya devam etmektedir (UNCTAD, 2020).

Sefer türleri açısından deniz yolu taşımacılığı, liner (düzenli hat) ve tramp (düzensiz hat) denizyolu taşımacılığı olmak üzere ikiye ayrılmaktadır (Erol ve Dursun, 2015). Liner taşımacılık, önceden belirlenen taşıma rotalarında, gemilerin kalkış ve varış zamanları bilinerek, düzenli bir çizelgeye göre yapılan taşıma hizmetidir (Wang ve Meng, 2012a; Trivedi, 2010). Liner olarak çalışan gemiler, çok sayıda yükleyiciden, birçok farklı tipte kargo taşımaktadırlar ve taşınan yükler genellikle nihai mal ya da yarı üretim mallarıdır. Liner taşımacılıkta, zamanlama esastır. Hizmetin devamlılığı Liner taşımacılığının en belirgin özelliklerinden biridir (Kiremitci ve Özçakar, 2011, ss. 14-25). Bu hizmet türünde, gemiler yeterince yük bulamasalar dahi, programlarını aksatmamak için ilgili seferleri gerçekleştirmektedir. Konteyner ve Ro-Ro (Roll on Roll of) gemileri, liner taşımacılıkta sıklıkla kullanılan gemi tipleridir.

Tramp taşımacılığında ise gemiler, yük bulunan limanlara giderek, dünya genelinde taşımacılık yapmaktadırlar. Bu taşıma tipinde gemiler düzenli bir şekilde çalıştırılmazlar ve taşıyacakları kargoları seferden sefere bulurlar. Seferler belirli bir çizelgeye göre yapılmadığından, gemiler karlı yük bulunan noktalara gitmektedirler. Tramp taşımacılıkta gemiler ve şirketler yük bulabilmek için sürekli bir rekabet içindedirler (Clarkson Research Study, 2004, s.24). Tramp taşıma yapan şirketler, farklı yük kapasiteleri, hızları ve kargoları olan heterojen gemi filolarına sahiptirler.

Tramp taşımacılıkta genellikle gemiler bir sefer ya da belirli bir süre için kiralanarak yük taşımacılığı yapılmaktadır. Taşınacak olan her bir kargo için, her sefer karşılıklı görüşülür ve kira sözleşmesi (charter-party) imzalanır. Liner deniz taşımacılığında önemli olan hizmet kalitesi iken, tramp taşımacılık yük esastır.

Tramp taşımacılıkta genellikle taşınan yükün tek bir yükleyicisi bulunmakta ve bu yükleyiciye ait çoğunlukla homojen olan yükler, tam gemi şeklinde taşınmaktadır (Kiremitci ve Özçakar, 2011, ss. 14-25).

2.1 Tramp Taşımacılık ile Taşınan Yük Türleri

Tramp taşımacılık ile taşınan yükler, genellikle tek bir müşteriye ait olmaları ve bir çıkış noktasından, varış noktasına taşınmalarından dolayı, ölçek ekonomisine uygun yüklerdir. Bu taşıma modunun hizmet sunduğu yükler, en genel tanımı ile sıvı dökme yükler ve katı dökme yükler olmak üzere ikiye ayrılmaktadır.

UNCTAD (2020), deniz taşımacılığı raporları, katı dökme yükleri, majör dökme yükler (demir cevheri, tahıl, kömür) ve minör dökme yükler (orman ürünleri, çelik ürünleri) olmak üzere ikiye ayırmaktadır. 2018- 2019 yılları arası majör ve minör katı yüklerin taşınma miktarları ve yüzde değişimleri, milyon ton cinsinden Çizelge 2.1’ de verilmektedir. Tabloda, demir cevherinin, %1,5 ve çelik ürünlerinin, % 4,4 oranında taşıma miktarlarının azaldığı, ancak toplam kuru yük taşımacılığın, 5 milyar 225 milyon tondan, 5 milyar 253 milyon tona çıkarak, yıllık yüzde değişiminin 0.5 arttığı görülmektedir.

Çizelge 2.1: 2018-2019 yılları katı dökme yük taşımacılığı (UNCTAD,2020).

	2018	2019	Yüzde değişimi 2018-2019
Majör dökme yükler	3215.0	3225.0	0.3
-Kömür	1263.0	1293.0	2.4
-Tahıl	475.0	477.0	0.4
-Demir cevheri	1477.0	1455.0	-1.5
Minör dökme yükler	2010.0	2028.0	0.9
-Orman ürünleri	380.0	382.0	0.5
-Çelik ürünleri	388.0	371.0	-4.4
Toplam kuru yük	5225.0	5253.0	0.5

UNCTAD (2020) sıvı dökme yükleri ham petrol, diğer tanker yükleri (petrol türevi, kimyasal ürünler, nebati yağlar vb.) ve sıvılaştırılmış petrol gazı (LPG) ve sıvılaştırılmış doğal gaz (LNG) yükleri olarak sınıflandırılmaktadır. Çizelge 2.2’de, 2018- 2019 yılları arası sıvı dökme yüklerin taşınma miktarları ve yüzde değişimleri, milyon ton cinsinden verilmektedir. Toplam sıvı yük taşımacılığı 2019 yılında negatif yönde değişmiş olsa dahi, 3 milyar 169 milyon ton ile deniz ticaretinde önemli bir konuma sahip olmaya devam etmektedir.

Çizelge 2.2: 2018-2019 yılları sıvı dökme yük taşımacılığı (UNCTAD, 2020).

Sıvı dökme yükler	2018	2019	Yüzde değişimi 2018-2019
Ham petrol	1881	1860	-1.1
Diğer tanker yükleri	1320	1308	-0.9
Gaz	416	461	10.8
Toplam tanker ticareti	3201	3169	-1.0

UNCTAD raporundan elde edilen bilgilere göre ve aynı zamanda Çizelge 2.3’de de gösterildiği üzere deniz ticaretinin ana gruplarını sıvı dökme yükler, ana dökme yükler ve diğer kuru yükler oluşturmaktadır. Raporda, diğer kuru yükler sütünü, toplam minör dökme yükler, konteyner yükleri ve genel kargo yüklerini kapsamaktadır, tüm değerler milyon ton cinsinden verilmiştir (UNCTAD, 2020).

Çizelge 2.3: Yıllara göre uluslararası deniz ticareti (UNCTAD, 2020).

Yıl	Sıvı dökme yük	Ana dökme yük	Diğer kuru yük	TOPLAM
2005	2422	1579	3108	7109
2006	2698	1676	3328	7702
2007	2747	1811	3478	8036
2008	2742	1911	3578	8231
2009	2641	1998	3218	7857
2010	2752	2232	3423	8408
2011	2785	2364	3626	8775
2012	2840	2564	3791	9195
2013	2828	2734	3951	9513
2014	2825	2964	4054	9842
2015	2932	2930	4161	10023
2016	3058	3009	4228	10295
2017	3146	3151	4419	10716
2018	3201	3215	4603	11019
2019	3169	3225	4682	11076

2019 yılı tramp olarak taşınan yük miktarını, toplam taşınan yük miktarı ile oranladığımızda %57,7 değeri görülmektedir. Tüm bu değerler, dünya ticaretinde tramp taşımacılığının önemli bir yere sahip olduğunu ve sektörde yaşanan yoğun rekabeti açıkça göstermektedir.



3. ARAÇ ROTALAMA PROBLEMLERİ VE ÇÖZÜM YÖNTEMLERİ

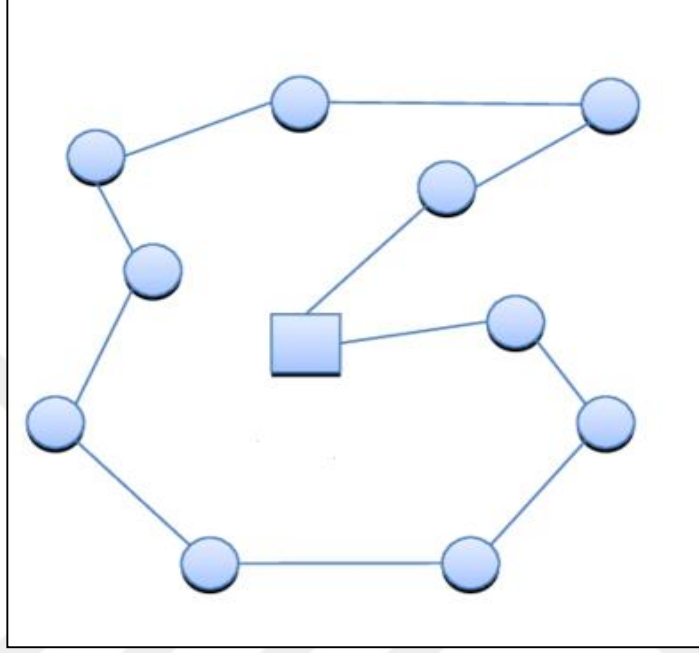
Problem tipleri kategorilere ayrılırken, polinom zamanda çözülebilen ve polinom zamanda çözülemeyen problemler olarak sınıflandırılmaktadır. Polinom zamanda çözülemeyen ve çözümü zor olan problemlere NP-hard ismi verilmektedir (Çetin, 2007, ss. 9-15). Araç grupları için en uygun rotayı bulma problemi olan ARP, NP-hard kombinatorial problemler sınıfına girmektedir. Temel amaç, minimum araç sayısını, minimum seyahat süresini, seyahat edilen rotaların minimum maliyetini bulmaktır (Carić ve diğ., 2008, s.142). ARP, coğrafi olarak farklı alanlarda bulunan ve talepleri bilinen çok sayıda müşteriye hizmet sunan araç filosu için, müşterilerin isteklerini karşılayacak şekilde, minimum maliyetli dağıtım rota kümesinin belirlenmesi problemidir (Kiremitçi ve Özçakar, 2011, ss. 14-25).

ARP’de, sevkiyat yapacak araçlar başlangıç noktasından harekete geçer, rotaya bağlı kalarak ve araç kapasitelerini aşmayacak şekilde tüm müşterilerin ürünlerini teslim eder ve yine başlangıç noktasına geri döner (Baker ve Ayeche, 2003). Araç rotalama problemine müşteri talepleri/ yapılan sözleşmeler dahilinde, teslim zaman aralığı veya zaman pencereleri eklendiğinde problem, araç rotalama ve çizelgeleme problemine dönüşür. Birleşik rotalama ve çizelgeleme problemleri, çoğunlukla gerçek hayatta müşteri beklentilerini karşılayan problem tipleridir. Araçların etkin bir şekilde rota ve çizelgelenmesi, verimliliği artırır ve uzun dönemli maliyet planlamasına yardımcı olur. Gereksiz maliyetlerin önüne geçerek, tasarruf elde edilmesini sağlar (Bodin ve Golden, 1981).

3.1 Gezgin Satıcı Problemi

Gezgin satıcı problemi (GSP), araç rotalama problemlerinin temelini oluşturmaktadır. Gezgin satıcı probleminin ARP ile ilişkisini açıklamak için, problem tek araçlı ARP olarak tanımlanabilir.

GSP, sonlu ve belirli sayıdaki şehir adedi ve her iki şehir arası uzaklık bilindiğinde, bir başlangıç noktasından başlanarak, aynı noktaya dönmek üzere tüm şehirlerin dolaşılması ile minimum maliyetli yolun bulunması olarak ifade edilir. (Applegate ve diğ, 1998). GSP, mesafenin en küçüklenmesi için uygulanan kesikli optimizasyon problemidir (Çetin, 2007, ss. 9-15).



Şekil 3.1 : Gezgin satıcı problemi (Liu ve diğ, 2014).

GSP’ de satıcı n adet noktayı başladığı noktaya geri gelecek şekilde minimum mesafe ile gezmelidir. Şekil 3.1’teki noktalar şehirleri temsil etmektedir. Satıcı noktaların (şehirlerin) tümüne uğramak zorundadır (Liu ve diğ, 2014). Ayrıca, bir noktadan yalnız bir kere geçebilir. Problemin amaç fonksiyonu denklem 3.1’de ifade edilmektedir (Dantzig ve diğ, 1954). n adet noktadan (şehirden) meydana gelen bir GSP’de tur sayısı $(n-1)!/2$ olarak hesaplanabilir. GSP model formülasyonu aşağıda tanımlanmaktadır.

Amaç fonksiyonu;

$$\min z = \sum_{i=1}^n \sum_{j=1, i \neq j}^n c_{ij} x_{ij} \quad (3.1)$$

kısıtlar;

$$\sum_{i=1, i \neq j}^n x_{ij} = 1, \quad j = 1, 2, \dots, n \quad (3.2)$$

$$\sum_{j=1, j \neq i}^n x_{ij} = 1, \quad i = 1, 2, \dots, n \quad (3.3)$$

$$\sum_{i, j \in S, i \neq j}^n x_{ij} \leq |S| - 1, \quad \forall S = \{1, 2, \dots, n\} \quad (3.4)$$

$$x_{ij} = \begin{cases} 1 & \text{i noktasından j noktasına gidiliyor ise,} \\ 0 & \text{aksi durumda,} \end{cases} \quad (3.5)$$

Formülde, c_{ij} ; i ve j noktaları arasındaki mesafeyi, x_{ij} ; i noktasından j noktasına gidilip gidilmediğini gösterir. i noktasından, j noktasına gidiliyor ise 1, aksi durumda ise 0 değerini almaktadır. Denklem 1.2 ve 1.3'te verilen kısıtlar her noktaya yalnız bir kez uğranılmasını sağlamak amacıyla kullanılmaktadır. Olası alt turlardan kurtulmak amacıyla denklem 1.4'te verilen kısıt kullanılmaktadır.

3.2 Araç Rotalama Problemleri

Geleneksel ARP'ye ilave kısıtlar eklenerek, kullanım alanlarına/amaçlarına uygun olarak özelleşmiş araç rotalama problemleri geliştirilmiştir. Gerçek hayattaki özel durumlara göre özelleşmiş araç rotalama problemlerine, karma kapasiteli araç rotalama (KKARP), çoklu depoya sahip araç rotalama (ÇDARP), bölünmüş talebe sahip araç rotalama (BTSARP), belirsiz talebe sahip araç rotalama (BSTSARP), geri toplaması olan araç rotalama (GTARP), zaman pencereci araç rotalama (ZPARP) ve asimetric araç rotalama problemleri (AARP) örnek gösterilebilir (Düzakın ve Demircioğlu, 2009).

3.2.1 Karma kapasiteli araç rotalama

Kapasite kısıtlı araç rotalama problemi (KARP), sınırlı taşıma kapasitesine sahip araçların belirli konumlardaki ürünleri alması veya teslim etmesi gereken bir araç rotalama problemidir. Taşınan yüklerin, ağırlık veya hacim gibi bir miktarı vardır ve araçlar taşıyabilecekleri maksimum kapasiteye sahiptir.

Amaç, araçların kapasitesini aşmadan taşınan birimleri en düşük maliyetle almak veya teslim etmektir (Altabeeb ve diğ, 2019). KKARP'de KARP'den farklı olarak, araçların taşıma kapasiteleri birbirlerinden farklıdır. Bu problem modellenirken, araç tipine göre araç kümeleri oluşturulur ve her bir araç kümesine ayrı ayrı kapasite kısıtları verilir (Düzakın ve Demircioğlu, 2009).

3.2.2 Çoklu depoya sahip araç rotalama

Dağıtım firmasının talepleri karşılamak amacıyla birden fazla deposunun olması durumudur. Bu problemde, iki yaklaşım kullanılabilir. Dağıtımın yapılacağı müşteriler depoların etrafında ve küme halinde ise, problem ayrı birer ARP olarak modellenebilir. Bu yaklaşımda ilk olarak depolar kendileri için en yakın olan müşteriye atanır ardından her bir depo için ARP çözülür (Salhi ve Sari, 1997). Ancak, müşteriler ve depoların yerleri küme halinde değil ve karışık ise, çoklu depoya sahip araç rotalama probleminin çözülmesi gerekmektedir (Yılmaz, 2008, s.21). Bu durumda, tüm dağıtım sisteminin aynı anda rotalanması gerekir. Müşterilerin hem depolara atanması hem de dağıtım rotasının oluşturulması aynı anda yapılır. Problemin amacı, seyahat mesafesini ve araç sayısını minimize etmektir. Klasik ARP yaklaşımında olduğu gibi her araç ayrıldığı depoya geri dönmelidir (Tansini ve diğ, 2001).

3.2.3 Bölünmüş talebe sahip araç rotalama

Bölünmüş talebe sahip araç rotalama probleminde, klasik ARP'de bulunan, ürünlerin müşterilere tek seferde teslim edilmesi kısıtı esnek haldedir. Bu yaklaşımda farklı olarak en az bir müşteriye götürülmesi gereken ürün miktarı, araç kapasitesinden büyüktür (Koç ve Karaoğlu, 2014). Bu nedenle, müşterinin talebi tek seferde gönderilmez.

Aynı müşteriye, farklı araçlar tarafından birden fazla kez uğranabilir ve müşteri aynı anda birden fazla rotanın içinde olabilir. Araç kapasitelerinin aşılması ve bir müşteriye birçok kez uğranması söz konusu olduğundan, en iyi sonucu bulmak ARP'ye oranla çok daha zordur. Bu problemde amaç, araç akışını ve müşterilerin taleplerini karşılamak için, tüm rotalardaki toplam maliyet en küçükmektir (Chen ve diğ., 2017).

3.2.4 Belirsiz talebe sahip araç rotalama

Bu problem yaklaşımında, önceki problem türlerinden farklı olarak müşterilerin talepleri bilinmemektedir. Dağıtım aracı müşteriye ulaşıncaya kadar, müşterinin talebinin ne olacağı belli değildir. Taşıma aracı kapasitesi planlaması yapılamaması, problemin çözümünü daha da karmaşık hale getirmektedir (Moghaddam ve diğ., 2012). Talepteki belirsizlik, ARP' de müşteri isteklerine cevap veremeyerek kar kaybına neden olabilir. BSTSARP'de amaç, güvenilir bir plan ile karşılanamayan talep maliyetini önemli ölçüde azaltarak müşteri memnuniyeti sağlamaktır.

3.2.5 Geri toplamalı araç rotalama

GTARP, tersine taşımacılık için kullanılan problem çeşididir. İade durumu gerektiren ürünlere uygulanan araç rotalama problemleridir (Alakaş ve diğ., 2018). Müşterilerden geri dönen her türlü taşıma paketleri (ambalaj, palet ve koruyucu materyalleri), imha edilmesi gereken ürünler, iade ürünler, geri dönüşüm için fabrikalara getirilen ürünler ve bu ürünlerin taşınmasında kullanılan koruyucu malzemelerin geri taşınması bu probleme örnek verilebilmektedir (Clarke ve Wright, 1964). Araç kapasiteleri, müşterilerin iade edebileceği ürünler göz önünde bulundurularak hesaplanır. Geri dönen ürünlerin taşınması, yeni bir rota yaratmamak ve verimliliği arttırmak için dağıtıma çıkan araçlarla yapılmaktadır. Kısaca, dağıtım ve toplama aynı anda gerçekleştirilmektedir. İlk aşamada müşteriler toplama ve dağıtımın yapılacak müşteriler olmak üzere ikiye ayrılır. Araç ilk olarak tüm müşterilere ürün dağıtımını gerçekleştirir, ardından aynı rotayı takip ederek geri toplama işlemi yapar ve depoya döner (Keçeci, 2008, ss. 10-12).

3.2.6 Asimetrik araç rotalama

AARP, depodan müşteriye kadar olan mesafe ile müşteriden depoya olan mesafenin eşit olmadığı problem yaklaşımıdır. Bu durumda mesafeye bağlı maliyet matrisi

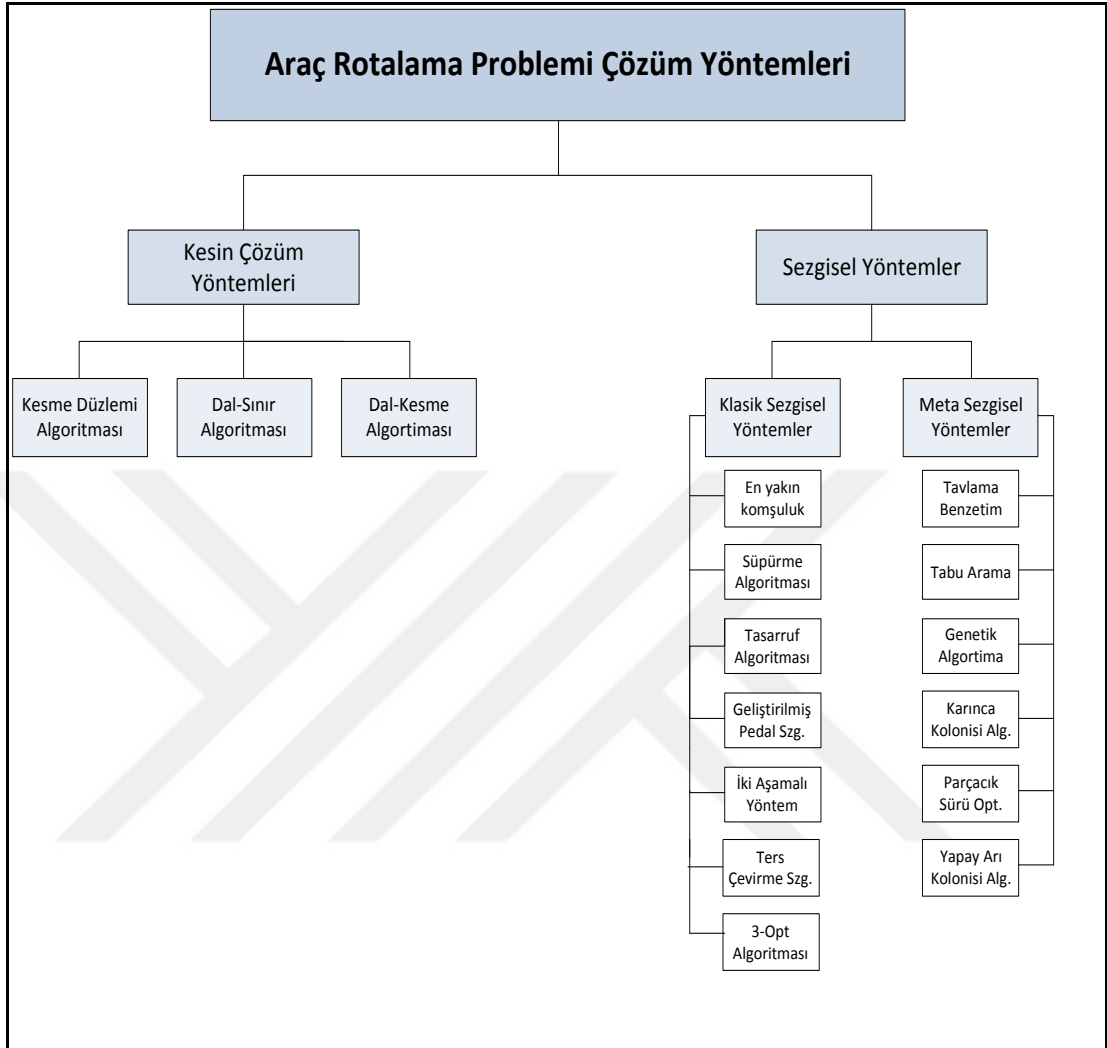
simetrik değildir. İki nokta arasında bulunan geliş-gidiş mesafesi aynı olmadığı için taşıma maliyeti hesaplaması klasik ARP yaklaşımına oranla daha zordur. Asimetrik araç yönlendirme probleminin amacı, tüm müşterileri araçlarla bir depoya bağlayarak toplam kat edilen mesafeyi en küçükmektir (Almoustafa ve diğ, 2009).

3.2.7 Zaman pencereci araç rotalama

ZPARP, diğar araç rotalama problemlerinden daha özel ve zor yapan kısıt, her müşteriye yalnız belirli bir zaman diliminde hizmet götürülebilmesidir (Gökçen ve diğ, 2018). Problemde, verilen zaman aralığı dışında teslimat yapılamamaktadır. Her müşteriye, müşteri için belirlenen zaman penceresi içerisinde hizmete başlanması ve yine belirli bir hizmet süresi kadar zamanda teslimat yapılabilmesi gerekmektedir (Schneider ve diğ, 2014). Bu problem yaklaşımında amaç, müşteriler tarafından verilen zaman aralığını göz önünde bulunduran en düşük dağıtım maliyetli araç rotasının hesaplanmasıdır. Söz konusu dağıtım maliyeti tüm müşterilere talepleri doğrultusunda hizmet götürebilmek için gerekli araç sayısı ve araçların aldığı toplam mesafedir (Ho ve diğ, 2001). Toplam maliyeti en küçükmek için zaman pencereci araç rotalama probleminde, toplam gidilecek yolu en küçükmek, toplam sefer süresini en küçükmek, araç sayısını en aza indirmek, araçların bekleme sürelerini en aza indirmek ve araçların alan kullanımını en büyükmek amaçlarından biri veya birkaçı seçilebilir (Gökçen ve diğ, 2018).

3.3 Araç Rotalama Problemi Çözüm Yöntemleri

ARP'ni en verimli şekilde çözebilmek için yıllar boyunca birçok teknik geliştirilmiştir. Temel olarak bu yöntemler kesin çözüm yöntemleri ve sezgisel çözüm yöntemleri olmak üzere ikiye ayrılmaktadır. Sezgisel yöntemler de kendi arasında klasik sezgisel yöntemler ve meta-sezgisel yöntemler olmak üzere ikiye ayrılmaktadır. Kesin çözüm yöntemleri optimum çözümü verseler de, sezgisel yöntemler ile daha büyük problemler çözülebilir ve optimuma yakın sonuçlar daha kısa sürelerde bulunabilir. Ayrıca son yıllarda, problemlerin bilgisayar ortamında taklit edilmesini sağlayan simülasyon yöntemleri de ARP çözümlerinde kullanılmaktadır (Braekers ve diğ, 2016). Şekil 3.2'de kesin çözüm yöntemlerinin ve sezgisel yöntemlerin detaylı gösterimi verilmiştir.



Şekil 3.2 : Araç rotalama problemi çözüm yöntemleri.

3.3.1 Kesin çözüm yöntemleri

ARP'nin çözümünde kullanılan kesin çözüm yöntemleri temel araç rotalama problemi olan GSP'den geliştirilmiştir. En belirgin özellikleri problemlere kesin çözüm sağlayabilmeleri ve optimum sonuçların elde edilebilmesidir. Bu yöntemlerinin ürettiği sonuçlardan daha iyi bir çözüm bulunmamaktadır. Ancak, kesin çözüm yöntemleri yalnızca az sayıda müşterinin olduğu problemlerde uygulanabilir. ARP'nin boyutu arttıkça, çözüm süresi de problemin boyutuna göre üstel olarak artmakta ve bazen de çözüm üretilmemektedir (Düzakın ve Demircioğlu, 2009).

Bu nedene, kesin çözüm yöntemleri küçük ve orta boyutlu problemlerin pratik şekilde çözümü için uygundur. Bir kesin çözüm yönteminin tüm problemlerde uygulanması mümkün değildir.

Nasıl ki Araç Rotalama Problemleri amaç ve yapı olarak birbirinden farklılık gösteriyorsa, çözüm yöntemleri de o problem türüne ait olarak gelişir (Ropke, 2005). Bu kesin çözüm yöntemleri; kesme düzlemi, dal-sınır algoritması ve dal- kesme yöntemi algoritmalarıdır.

3.3.1.1 Kesme düzlemi algoritması

1958 yılında Gomory tarafından geliştirilen bu hesaplama yönteminde, doğrusal programlama problemlerinin tam sayılı çözümleri elde edilmektedir. Bu nedenle, kesme düzlemi algoritmasına tam sayılı algoritma da denilmektedir (Özder, 2009, ss. 34-38). Kesme düzlemi algoritması yönteminde, ARP öncelikli olarak simpleks algoritması kullanılarak çözülür. Simpleks Metot, amaç fonksiyonunu en iyi (optimum) yapacak çözüme adım adım yaklaşan hesaplama yöntemidir. Daha detaylı belirtmek gerekirse, probleme bir uç noktasından başlayarak sonsuz sayıdaki çözüm seçeneğinden, sonlu sayıdaki çözüm alternatiflerine oradan optimuma en yakın değerlere ve neticede tek bir en iyi çözüme ulaşmayı sağlamaktadır (Cibej ve Mihelič, 2017). İkinci aşama, simpleks algoritma ile elde edilen sonuçtan parçalar kesilmesidir. Parça kesilmesi olarak adlandırılan bu aşamada, matematiksel modele çözüm kümesini yalnızca tam sayılı değerleri içeride bırakacak şekilde kesen yeni kısıtlar ilave edilir. Eklenen kısıtlar, çözüm kümesini küçülterek bir ya da daha fazla uygun olan ya da olmayan tamsayı noktasını kesmelidir. Değişkenler hala kesirli ise, tam sayılı çözüm bulunana kadar kesme işlemi yapılmaya ve problem çözmeye devam edilir (Toth ve Vigo, 2002). Tüm değişkenler tam sayı değeri aldığı anda problemin çözümüne ulaşılır.

3.3.1.2 Dal ve sınır algoritması

Çözüm algoritmasına eklenen kısıtlara rağmen optimum bir çözümün elde edilememesi durumunda uygulanan çözüm yöntemlerinden biri de dal ve sınır algoritması yöntemidir (Özder, 2009, ss. 34-39). Dal ve sınır algoritması Land ve Doig tarafından geliştirilmiş, 1980 yılından bu yana aktif olarak kullanılan bir tam sayılı doğrusal programlama yöntemidir. Yöntem, kesme düzlemi algoritmasına göre daha etkin çalışmaktadır (Önsel, 2021).

Bu algoritmada ilk olarak problem daha küçük alt parçalara bölünür. Daha sonra, alt kümelerdeki en iyi çözümler belirlenip, bu çözümler uygun ama optimum değil ise bu alt parçalar işlem dışı bırakılır (Mutlu, 2010, ss. 39-45). Tanımı detaylandırmak gerekirse, dal ve sınır algoritmasında, araç rotalama problemi alt turlarını (depoda başlayıp bitmeyen) engellemek için modele engelleyici kısıtlar ilave edilir ve problem atama problemi haline getirilir.

Atama problemi haline dönüşen ARP, satır ve sütun elemesi temeline dayanan Macar algoritması ile çözülmeye başlanır ve rotalar belirlenmeye çalışılır (Düzakın ve Demircioğlu, 2009). Alt turların oluşması durumunda, probleme alt tur oluşmasını engelleyecek yeni kısıtlar eklenir ve gidilmemesi istenen rotalara atama yapılmasını önlemek amacıyla mesafe matrisine atamayı engelleyecek yüksek ceza katsayılı M sayısı atanır. Yeni mesafe matrisine göre problem baştan çözülür ve tüm dallar için aday çözümler belirlenir. Alt turlar oluşmayana kadar tüm problem dallarına aynı iterasyonlar tekrar tekrar uygulanarak en iyi çözüme karar verilir (Keskintürk ve diğ., 2015). Dal ve sınır algoritmasının altındaki temel fikir, uygun çözüm alanı içerisindeki bütün tam sayılı çözüm çiftlerini inceleyerek en uygun çözüm bulunana kadar karşılaştırmaktır. Bu nedenle, çözümün optimum olduğunu doğrulama yani onaylama süreci çok zaman alıcı olabilmektedir (Özder, 2009, ss. 34-39).

3.3.1.3 Dal ve kesme algoritması

Dal ve kesme yöntemi küçük ve orta boyuttaki ARP'nin çözümü için etkin bir yöntemdir. Bu yöntemde kesme düzlemi algoritması ile dal ve sınır algoritması yöntemleri beraber kullanılmaktadır. (Araque ve diğ., 1994). Yöntem, tüm tam sayılı programlama problemlerinde olduğu gibi, araç rotalama probleminin doğrusal programlama çözümü ile başlar. Bu aşamada amaç fonksiyonu ve kısıtlar belirlenerek model oluşturulur (Mitchell ve diğ., 2009). Ardından, dal ve sınır algoritmasına benzer şekilde problem alt kümelere (problemlere) ayrılır, alt turların oluşmasını engelleyecek kısıtlar eklenerek, alt turlar sıfıra eşitlenir ve dallandırma yapılarak uygun çözümlere götürmeyen çözüm seçenekleri elenir. Eğer dallarda elde edilen çözümler tam sayılı değilse, probleme kesme düzlemi algoritması ile bir doğrusal kısıtlayıcı eklenir ve uygun sonuç üretmeyen çözümler uzaklaştırılır.

Bu sayede, problemin çözüm alanı küçülür ve kesme düzlemi algoritmasının katkısı ile dal ve sınır algoritması büyük oranda hızlandırılır (Başkaya ve Öztürk, 2005). Optimum sonucun bulunamaması durumunda, probleme yeni kesme eklenebilir ya da dallarına ayrılabilir.

3.3.2 Sezgisel yöntemler

ARP çözümünde kullanılan sezgisel yöntemler, klasik sezgisel yöntemler ve meta-sezgisel yöntemler olarak iki grupta toplanmaktadır. Sezgisel Optimizasyon Algoritmaları en iyi çözümü garanti etmez. Optimuma en yakın sonuçlara klasik yöntemlere göre çok daha kısa sürede ulaşma imkanı sağlar. Son yıllarda özellikle meta-sezgisel yöntemler, ARP için umut verici çözüm yöntemi olarak ortaya çıkmaktadır (Mijwel, 2015). Problemlerde sezgisel algoritma kullanımı gerektiren birkaç durum meydana gelebilir. İlk durum; kesin çözüm teknikleriyle çözilemeyen problem durumlarıdır. Böyle bir durumda, sezgisel yöntem kullanılarak problem çözümü tek seçenektir. İkinci neden, kesin çözüm yöntemleri bilinse bile, hesaplama açısından yoğun olan ve çözüm süresi uzun olan problemler için kullanılamamalarıdır. Üçüncü durum olarak, sezgisel algoritmalar, dallanma ve sınır çözüm algoritma süreçlerinde optimal çözüme ilişkin sınırları hesaplamak için de kullanılabilir (Eiselt ve Sandblom, 2013, ss. 229-260). Sezgisel algoritmalar, Kullanım amaçlarına göre temel olarak üç kategoriye ayrılırlar. Bunlar; sıfırdan bir tur inşa edilmesi gereken durumlarda yapı sezgiseli, turun bilinmesi ancak iyileştirilmesi gereken zamanlarda iyileştirme sezgiseli, alt turların oluştuğu ve bu alt turların birleştirilmesi gereken durumlarda birleştirme sezgiselidir.

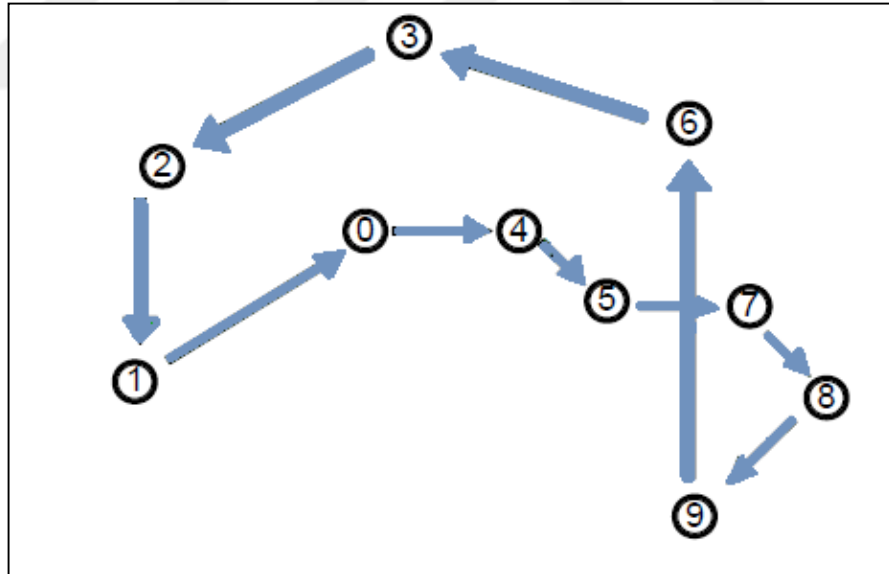
3.3.2.1 Klasik sezgisel yöntemler

Klasik sezgisel yöntemler, çözülmek istenen probleme göre uyarlanmış algoritmalarıdır. Klasik sezgisel yöntemler, turların kurumu ve geliştirilmesi amacıyla kullanılmaktadır. Bu algoritmalarda, önceden belirlenmiş adımlar izlenerek, son verme (bitiş) kriterine ulaşıncaya kadar bu adımlar tekrar edilir. Bitiş kriterine ulaştığında rota kurulmuş veya geliştirilmiş olur. Tur kurucu algoritmaların ARP'ye uygulanması ile araç rotaları oluşturulur. Tur kurucu sezgisel algoritmalar; en yakın komşuluk sezgiseli, süpürme algoritması, tasarruf algoritması, iki aşamalı yöntem, geliştirilmiş pedal sezgiselidir (Keskintürk ve diğ, 2015).

Tur kurucu algoritmalarından elde edilen rotaları başlangıç çözümü olarak alan ve geliştiren Tur geliştirici algoritmalar ise; tersine çevirme ve 3-Opt algoritması örnek gösterilebilir.

- En yakın komşuluk sezgiseli

Bellmore and Nemhauser (1968) tarafından geliştirilen en yakın komşuluk algoritması, ARP'lerin çok hızlı şekilde çözüme ulaşmasını sağlamaktadır. Algoritmanın ilk adımında öncelikli olarak bir başlangıç noktası seçilir. Ardından, başlangıç noktasına komşu olan noktalar belirlenir. Bu komşu noktalardan en yakın olanı tercih edilerek ilk olarak rotaya eklenir (Chen ve diğ, 2016). Sonraki adımlarda ise sırayla bir sonraki en yakın komşu noktaya harekete geçilir. Bu işleme kapasite ya da zaman penceresi gibi kısıtlar ihlal edilmediği sürece, tüm noktalar rotaya eklenene kadar devam edilir ve en sonunda başlangıç noktasına dönülür. Kısıtlar ihlal edilmeden eklenecek nokta bulunamadığı durumda başlangıç noktasına dönülerek yeni bir rota başlatılır (Solomon, 1987). Şekil 3.3'te en yakın komşu algoritmasına bir örnek gösterilmiştir (Keskintürk ve diğ, 2015).

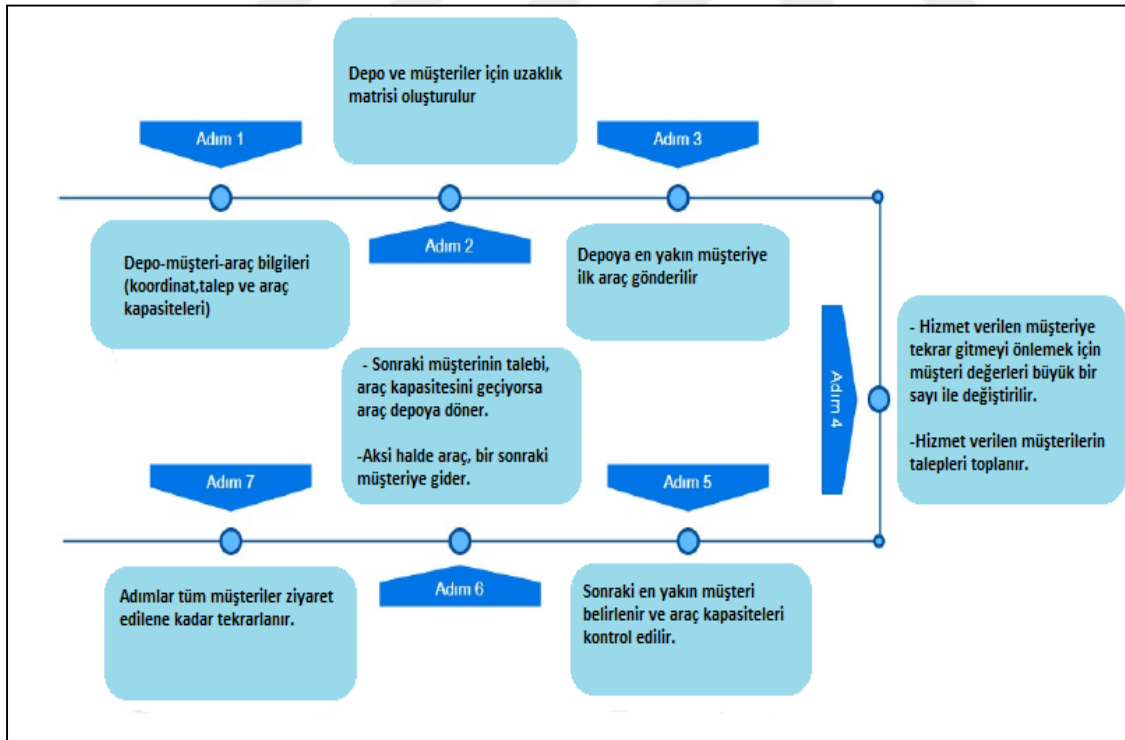


Şekil 3.3 : En yakın komşuluk algoritması (Keskintürk ve diğ, 2015).

- Süpürme algoritması

Süpürme Algoritması Gillet ve Miller (1974) tarafından, kapasiteli araç rotalama problemlerinin çözümü için sunulan bir sezgisel algoritma türüdür (Yıldırım ve Çebi, 2019). Bu sezgisel algoritmada, depo orijine alınarak, tüm müşteriler x-y koordinat sistemine yerleştirilir.

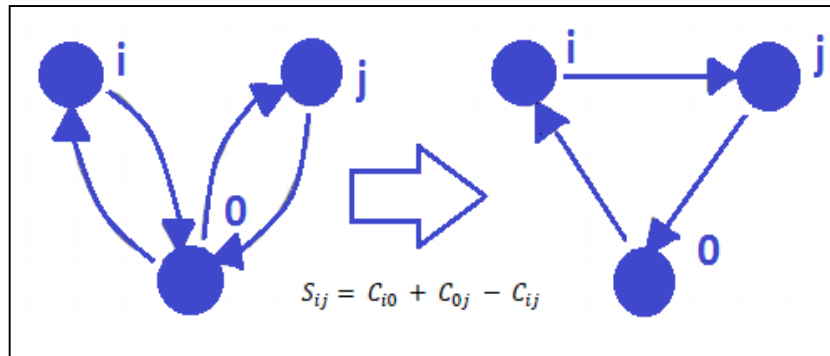
Daha sonra müşteriler ile depo arasındaki açılar hesaplanır. +x ekseniiyle müşterilerin arasındaki açılar küçükten büyüğe doğru sıralanır. Rotalama yapılmamış ve en küçük açı değerine sahip müşteriden başlanarak rota oluşturulur (Hertrich ve diğ, 2019). Depodan ilk müşteriye rota belirlenmesinin ardından, aynı yönde bir sonraki en küçük açıya sahip olan müşteri belirlenir ve gidilir. Müşterilere rota oluşturulurken, müşterilerin talep kapasiteleri toplanır ve araç kapasitesi ile karşılaştırılır. Araç kapasitesi dolduğunda ve rotalanmamış müşteriler bulunduğunda yeni araç dağıtımına çıkar ve tüm müşterilere hizmet götürüne (Son müşteri rotaya ekleninceye) kadar algoritma adımları aynı şekilde uygulanır (Laporte, 1992). Süpürme algoritmasının uygulama adımları Şekil 3.4’te verilmektedir (Yıldırım ve Çebi, 2019).



Şekil 3.4 : Süpürme algoritması uygulama adımları (Yıldırım ve Çebi, 2019).

- Tasarruf algoritması

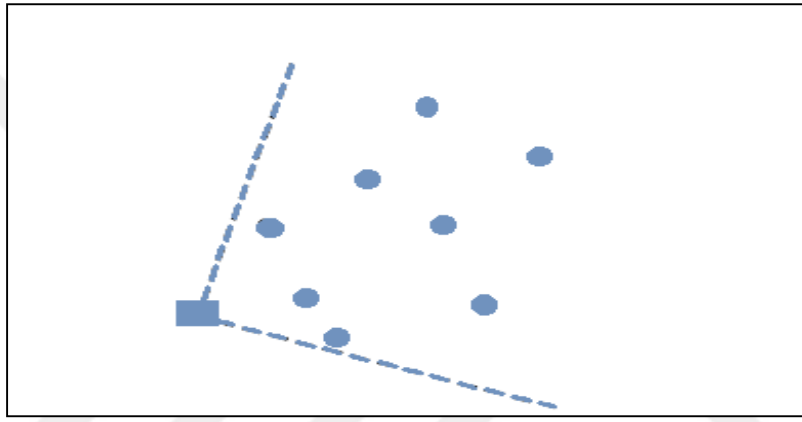
Tasarruf algoritması, 1964 yılında Clarke ve Wright tarafından geliştirilmiş, bir tur kurucu algoritmadır (Clarke ve Wright, 1964). Tur kurucu sezgiseller arasında en çok kullanılan algoritmadır. Bu yöntem, her adımda oluşturulan tur setinin daha iyi bir set elde etmek üzere değiştirilmesi temeline dayanır (Toth ve Vigo, 2002). İlk adımda her bir araç için ayrı bir rota oluşturulur. Talepleri ve konumları bilinen her bir müşteriye ayrı bir araç hizmet götürür ve bütün araçların geri döneceği bir depo bulunur. Herhangi iki müşteri birbirine bağlanmadığında, depodan çıkan araç, müşterilerin her birine ayrı ayrı uğrayıp depoya geri döner. Birbirine bağlandığında ise araç, sırasıyla birinci ve ikinci müşteriye gider ve depoya döner. Böylece iki müşterinin bağlanması ve bağlanmaması arasındaki farka göre bir tasarruf oluşur (Kosif ve Ekmekçi, 2012). Tasarruf ile ifade edilen; i ve j düğümlerine iki farklı araç yerine bir tek araç ile hizmet verilmesi durumunda azalan taşıma maliyetidir (Halim ve Yoanita, 2015). Bu tasarruflar tüm müşteri çiftleri için hesaplanarak büyükten küçüğe doğru sıralanır. En küçük tasarruf değerine sahip müşteri çifti ilk rotaya atanır. Ardından bir sonraki tasarruf değerine bakılır ve bu tasarrufa ait müşteri ikilisinden biri herhangi bir rotaya eklenmemişse yeni rota oluşturulur. Eğer bu ikiliden birisi rotaya atanmışsa bir sonraki müşterinin de bu rotaya eklenmesi denir. Eklenemezse, söz konusu müşteri ikilisi bırakılır ve bir sonraki müşteri ikilisine geçilir. Bu tasarruf sırasına göre tüm müşteriler rotalara atanıncaya kadar algoritma devam eder (Acar ve Çakmak, 2018). Tasarruf algoritmasının oluşması Şekil 3.5'te gösterilmiştir (Halim ve Yoanita, 2015).



Şekil 3.5 : Tasarrufun oluşması (Halim ve Yoanita, 2015).

- Geliştirilmiş petal sezgiseli

Petal sezgiseli ARP için ilk olarak Foster ve Ryan (1976) tarafından önerilmiştir. Daha sonra 1996 yılında Renaud ve arkadaşları tarafından geliştirilmiş ve şuan ki kullanım şeklini almıştır (Renaud ve diğ, 1996). Şekil 3.6’da görüldüğü gibi, geliştirilen yöntemde, belirli bir sektörde bulunan veya depoda merkezlenmiş tüm müşterilere toplu olarak hizmet veren bir dizi rota kümeleri oluşturulmaktadır. İki aşamalı yöntemin ilk aşamasında araç tur rotaları oluşturulur. İkinci aşamada, daha önce oluşturulmuş rotalar kolon yenileme işlemi kullanılarak geliştirilir ve en düşük maliyetli rotalar oluşturulur (Renaud ve diğ, 1996).



Şekil 3.6 : Depo ve müşteri dağılımı (Renaud ve diğ, 1996).

1-petal sezgiseli ile tek bir dağıtım noktası seçilir. Ardından, her bir müşteriden tam olarak bir kere geçip başlangıç düğümüne dönmeyi sağlayan Hamilton turu oluşturulur. İlk tur oluşturulduktan sonra kalan noktalar tura eklenir ve sonunda 4-Opt kenar değişimi ile tur geliştirilir. 2-petal sezgiseli aşamasında ise birbirlerine en uzak iki nokta ile iki başlangıç turu geliştirilir. Diğer noktalar ise en ucuz ve en uygun yerleştirme şeklinde yerleştirilir. Eklenmemiş noktalar için ise, kısmi turlar geliştirilir. Tüm noktalar turlara yerleştirildiğinde 4-Opt yöntemine göre turlar optimize edilir. Bu sezgisel yöntemin en önemli özelliği kısa sürede en iyiye yakın sonuç üretmesidir (Düzakın ve Demircioğlu, 2009).

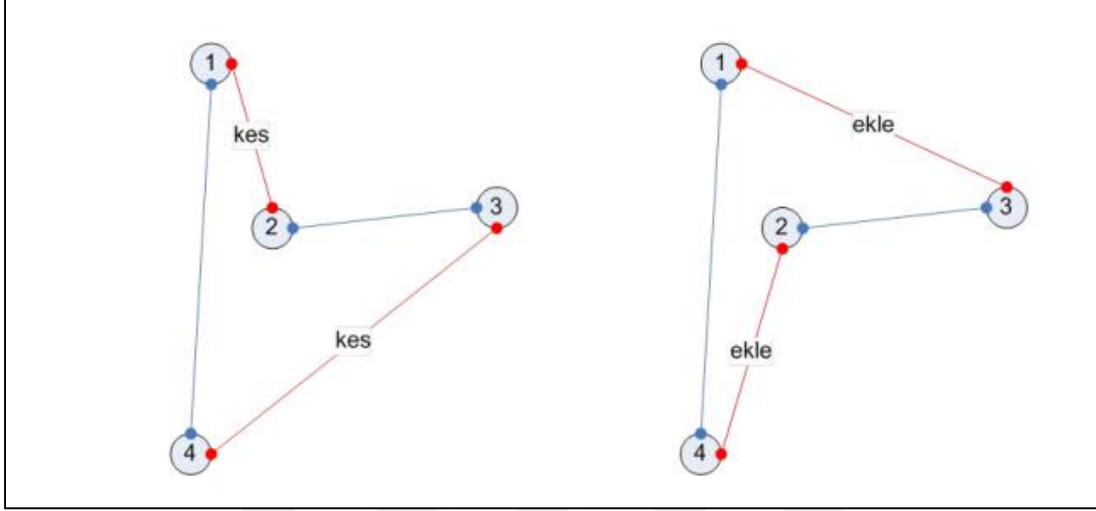
- İki aşamalı yöntem

Christofides Mingozzi ve Toth (1979) tarafından geliştirilen iki aşamalı yöntem, zaman, mesafe, kapasite kısıtlı araç rotalama (KARP) problemlerinin çözümünde kullanılmaktadır. Bu yöntemde, rota kurulması ve iyileştirmesi prosedürleri beraber gerçekleştirilir. Yöntem bu nedenle iki aşamadan meydana gelmektedir. İlk adımda, bir dizi uygun rota belirlemek için seri bir ekleme algoritması kullanılır (Laporte, 1992). Bu aşamada, daha önceden rotada olmayan müşteriler için değer üretilir. Hesaplanan değerler içinde en az değere sahip müşteri rotaya eklenir. Bu işlem, tüm noktalar rotaya ilave edilinceye kadar devam eder. İkinci adım ise, paralel rota ekleme yaklaşımıdır. Bu aşamaya, tüm müşterilerin rotaya eklenmesinin ardından geçilir. Bu aşamada rota iyileştirilmesi yapılır. İlk adımda belirlenen her rota için, bir temsili müşteri seçilir ve bu müşterilere bir dizi tek müşterili rota başlatılır. Geri kalan rotalanmamış müşteriler daha sonra en iyi ve ikinci en iyi ekleme maliyeti arasındaki farkın dikkate alındığı bir kriter kullanılarak eklenir ve kısmi yollar 3-opt prosedürü ile iyileştirilir (Cordeau ve diğ., 2007). Tüm olası iyileştirmeler yapınca kadar hesaplamalar devam eder ve rotalar iyileştirilir.

- Ters çevirme sezgiseli (2-Opt Algoritması)

Croes (1958) tarafından gezgin satıcı problemlerinin çözümü için geliştirilen yöntem bir tur geliştirme sezgiselidir. Yöntem, iteratif olarak mevcut çözümü iyileştirmeye çalışmaktadır. Tur geliştirici sezgisellerin başlangıç çözümü genellikle tesadüfi olarak belirlenmektedir. Ayrıca, süpürme algoritması, en yakın komşuluk algoritması gibi bazı tur kurucu sezgisellerle oluşturulan rotaların iyileştirilmesi için de kullanılmaktadır (Keskintürk ve diğ., 2016). 2-Opt sezgiselinin uygulama aşamasında, önceden belirlenmiş araç rotalarındaki bazı noktalar ters çevrilerek yeni rota oluşturulur. Eğer, ters çevrilme sonucu oluşturulmuş rota, eski rotadan daha iyiye yeni rota tutulur. Aksi halde bir önceki çözümle devam edilir. Metot, rotada bulunan iki kenarın silinmesi ve iki parçaya ayrılan rotanın maliyetleri düşürecek şekilde, yeniden bağlanması şeklinde de tanımlanabilir (Gutin ve Punnen, 2002). İlk aşamada var olan çözüm üzerinden iki nokta çifti seçilir (1-2, 3-4). Sonraki aşamada turu bozmayacak şekilde iki noktanın yeri değiştirilir (1-3, 2-4).

Yeni rota oluşurken yalnızca iki noktanın yeri değişir ve rotadaki diğer noktalar sabit kalır. Bu işlem, ikili nokta değişimlerinin çözümü daha iyi hale getiremediği noktaya kadar devam ettirilir (Keskintürk ve diğ, 2016). Şekil 3.7’de ters çevrilmenin etkisi şematik olarak göstermektedir (Yıldırım, 2014, ss. 21-37).

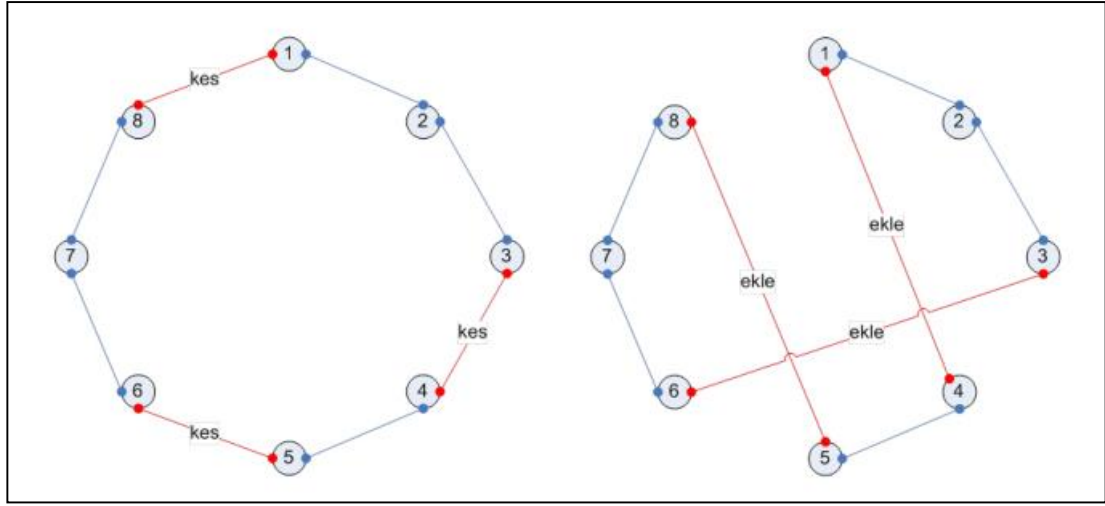


Şekil 3.7 : Ters çevirme sezgiseli (Yıldırım, 2014, ss. 21-37).

Ters çevirme algoritması ile en iyiye çok yakın bir değer hızlı bir şekilde bulunabilir. Ancak, deneme sayısının az olması denenmemiş rotalarda bulunan daha iyi sonuçların gözden kaçırılmasına neden olabilir (Yıldırım ve Çebi, 2019).

- 3 Opt algoritması

3-Opt algoritması ilk olarak, Bock (1958) tarafından uygulanan bir tur geliştirme algoritmasıdır. 3-Opt algoritması, k-Opt sezgiselinin bir alt çeşididir ve 2-Opt sezgisel algoritmasına benzer şekilde çalışır. Fakat bu sefer 2 nokta yerine 3 noktanın yerleri değiştirilir (Gülcü ve diğ, 2018). Uygulama aşamasında, başlangıç çözümü bilinen bir rotadaki üç kenarı siler, böylece önceden hesaplanmış rotayı üç yola bölerek bu yolları diğer olası yollarla yeniden bağlar ve en iyisini seçer. 3-Opt sezgiseli, nokta değişimlerinin çözümü daha iyi hale getiremediği ana kadar rotaları kaldırıp, yeniden bağlamaya devam eder (Johnson ve McGeoch, 1997). Şekil 3.8’de, (1-8), (3-4) ve (5-6) bağlantıları kesilerek (5-8), (1-4) ve (3-6) bağlantıları oluşturulmuştur (Yıldırım, 2014, ss. 21-37).



Şekil 3.8 : 3-Opt algoritması (Yıldırım, 2014, ss. 21-37).

3.3.2.2 Meta-sezgisel yöntemler

Meta-sezgisel algoritmalar, farklı tip problemleri çözmek için kullanılabilen stratejik problem çözme çerçeveleridir. Meta-sezgisel çerçeve, doğal veya fiziksel süreçlere dayanan, bir dizi problem çözme ilkesini ve bir dizi kontrol parametresini içerir. Yaklaşım, kesin çözüm teknikleri ile çözülemeyen ya da çözülmesi uzun zaman alan problemleri doğadaki canlılardan veya olaylardan esinlenerek çözer (Gandomi ve diğ, 2013a). Meta-sezgisel yöntemler, bir problem tipi üzerine özelleşmiş buluşsal yöntemlerden daha iyi performans göstermez. Burada asıl amaç, özel bir algoritmanın olmadığı zor problemlere çözüm getirmektir (Eiselt, 2013). Çözülmesi gereken gerçek hayattaki problem ilk olarak matematiksel bir modele dönüştürülür. Böylelikle, en iyi çözümün kısa sürede bulunması amaçlanır (Mijwel, 2015). Meta-sezgisel yöntemler, alt seviyede hareketlere izin vererek çözüm uzayında optimum çözümün daha verimli bir şekilde aranmasına imkân verirler. Bu yöntemin klasik sezgisellere göre güçlü yanı, problem tipine göre, algoritma adımlarında küçük değişiklikler yapılarak farklı problemlerin çözümünde kullanılabilmesidir (Yıldırım, 2014).

Yöntem, karmaşık yapıli problemlere makul zamanda deneme yanılma yoluyla kabul edilebilir çözümler üretmektedir. En iyi çözümlerin bulunabileceğinin garantisi yoktur (Yang, 2008).

Buradaki fikir, çoğu zaman karmaşık yapıli problemler için işe yarayacak ve kaliteli çözümler üretebilecek verimli ve pratik bir algoritmaya sahip olmaktır. Bu nedenle

özellikle büyük ve karmaşık problemlerin çözümü için en pratik yol olarak kabul edilirler (Gandomi ve diğ, 2013a). Bu yöntemler, tüm ARP'ler için kullanılabilir. Doğadaki canlılardan veya olaylardan ilham alarak, optimuma yakın sonucu makul sürede üretebilmelerinden dolayı günümüzde problem çözümünde yaygın olarak tercih edilmektedir (Şahin ve Eroğlu, 2014; Gendreau ve diğ, 2008; Cordeau ve diğ, 2002). Meta-sezgisel algoritmalar, iterasyonda elde edilen sonuçlara göre tek sonuçlu ve çok sonuçlu çözüm yapan algoritmalar olarak ikiye ayrılır. Uygulamada en çok tercih edilen tek sonuçlu arama yapan algoritmalar; tavlama benzetim (TB) ve tabu arama algoritmalarıdır (TA). Çok sonuçlu çözüm yapan algoritmalar ise, genetik algoritma (GA), karınca kolonisi algoritması (KKOA), parçacık sürü optimizasyon (PSOA) algoritması ve yapay arı kolonisi algoritmasıdır (YAKA) (Aarts ve Lenstra, 1997).

- Tavlama benzetim

TB ilk olarak Kirkpatrick Gelatt ve Vecchi (1983) tarafından GSP çözümü için kullanılmıştır. Tavlama benzetim algoritması katı maddelerin fiziksel olarak tavlansından esinlenerek oluşturulmuş bir arama algoritmasıdır (Johnson ve diğ, 1989). Tavlama, belirli bir sisteme göre katı halde bulunan metali yeterince yüksek bir sıcaklıkta ilk önce ısıtmanın ve çok yavaş soğutmanın fiziksel işlemidir. Kontrollü olarak azaltılan sıcaklıkta, sistemin kararlı duruma gelmesi için yeterli zaman verilir. Sıcaklık ne kadar düşükse, verilen süre o kadar yüksek olur (Malani ve diğ, 2011). Isıtma sıcaklığı, kristallerin boyutunu artırmak ve kusurlarını azaltmak için yeterince yüksekse ve soğutma işlemi termal dengeyi sağlamak için yeterince yavaşsa, o zaman atomlar kendilerini mükemmel bir kristalin global minimum enerjisine karşılık gelen bir modele yerleştirir (Jünger ve diğ, 1995). TB, kesin çözüm algoritmalarının başarısız olduğu çok zor problemlerin optimizasyonu için kullanılabilir; Genelde en iyiye yakın bir çözüm sağlamamaktadır. Bir fonksiyonun global optimumunu yaklaşık olarak tahmin etmek için kullanılan olasılıksal bir tekniktir.

Yöntemin amaç fonksiyonu fiziki sistemde enerjiye, uygun çözümler ise sistemin kararlı haline karşılık gelir Bahsi geçen termodinamik benzetimlerin, algoritma karşılıkları Çizelge 3.1'de gösterilmektedir (Yıldırım, 2014, ss. 21-37).

Algoritma, katı maddenin yüksek sıcaklığa çıkarılmasıyla başlar ve sıcaklık kontrollü olarak komşu çözümlere doğru düşürülür. Her adımda, çözümü belirleyen

değişkenlerden birinin değeri bozulur. Eğer yeni komşu çözümün performansı daha iyiye, yeni yapılandırma kabul edilir. Eğer iyileşme gerçekleştirilemezse, mevcut iyi çözüm değiştirilmez. Başlangıç sıcaklığının yüksekliği, probleme ilk aşamada geniş bir uzayda arama yapma imkanı verir. Sıcaklığın kademeli olarak düşmesiyle kötü çözümler elenir ve arama uzayı daraltılır (Gülsün ve diğ., 2008). Maliyet fonksiyonunda daha fazla iyileştirme mümkün olmayana veya sistem tarafından kararlı bir duruma ulaşılan kadar algoritma adımları tekrarlanır ve çözüm elde edilir (Malani ve diğ., 2011).

Çizelge 3.1 : Optimizasyonun termodinamik karşılığı (Yıldırım, 2014, ss. 21-37).

Termodinamik Karşılık	Optimizasyon
Sistem kararlı bir hal alır	Uygun çözüm bulunur
Enerji	Maliyet
Durum değişikliği	Komşu çözüm
Sıcaklık	Kontrol parametreleri
Donmuş hal	Sezgisel çözüm

- Tabu arama

Yöntem, Glover (1986) tarafından geliştirilmiştir. TA algoritması yerel optimum tuzaklarından kaçarak global optimumu bulan meta-sezgisel bir arama yöntemidir (Chelouah, ve Siarry, 2000). Algoritma, kısa ve uzun süreli hafızalara sahiptir ve komşuluğa dayalıdır (Dreo ve diğ., 2006, ss. 47-55). Kısa süreli hafıza, yeni çözüm üretmek için en doğru hamlenin seçilmesini sağlar ve bazı çözümlerin üretilmesini engeller. Yöntemin çözüm alanı, son bulunan çözüm, ziyaret edilmeyen en iyi komşu çözümlerle değiştirilerek araştırılır (Glover, 1989, ss. 190-206). Farklı problem tiplerinde kullanılabilen bu yöntem, esnek yapısı sayesinde optimuma en yakın çözümleri üretebilmektedir. TA algoritması, problemin tüm kısıtlarını karşılayan bir başlangıç çözümü ile başlar.

Her bir iterasyonda, başlangıç çözümünün komşu çözümlerinden bir aday liste oluşturulur ve bu adaylar değerlendirilir (Chelouah ve Siarry, 2000). Neticede,

değerlendirilen aday daha iyi bir sonuç verirse, yeni çözüm olur. Çözüm değişimi sırasında algoritmanın geriye dönüp eski çözümleri tekrar tekrar üretmemesi için iyileştirilmiş davranışın tersi engellenir. Bu engellemeler tabu olarak adlandırılır, tabu listesi yasaklı hareketler listedir (Zhou ve diğ, 2013). Tabuların amacı, yöntemi yerel optimumlardan kurtararak global optimumlara ulaşmasını sağlamaktır. Bu listedeki hareketler belli bir süre boyunca yasaklı kalır ve o süre zarfında bu listedeki hareketlerde arama yapılmaz. Bazı durumlarda bu tabular yıkılarak, yasaklı hareketler listeden çıkar ve algoritmanın yeni çözüm oluşturamaması durumuna engel olunur (Gürbüz, 2015, ss. 12-18). Kısa süreli hafıza listesine alınarak hareketlerin yasaklanması, algoritmanın mevcut çözümünün çevresinde dolaşarak daha iyi arama yapmasını ve optimal sonuca yaklaşmasını sağlamaktadır (Nowicki ve Smutnicki, 1996). Algoritma tarafından bulunan en iyi çözüm uzun süreli hafızada tutulur, hesaplanan tüm yeni çözümler, hafızadaki çözüm ile kıyaslanarak değerlendirilir. Tabu arama algoritması, belirli bir iterasyon sayısına ulaşıldığında, yöntem daha iyi bir çözüm bulamadığında ve optimum sonucu bilinen problemin optimum değerine ulaşıldığında durdurulur (Ahmed ve diğ, 2020; Gürbüz, 2015, ss. 12-18).

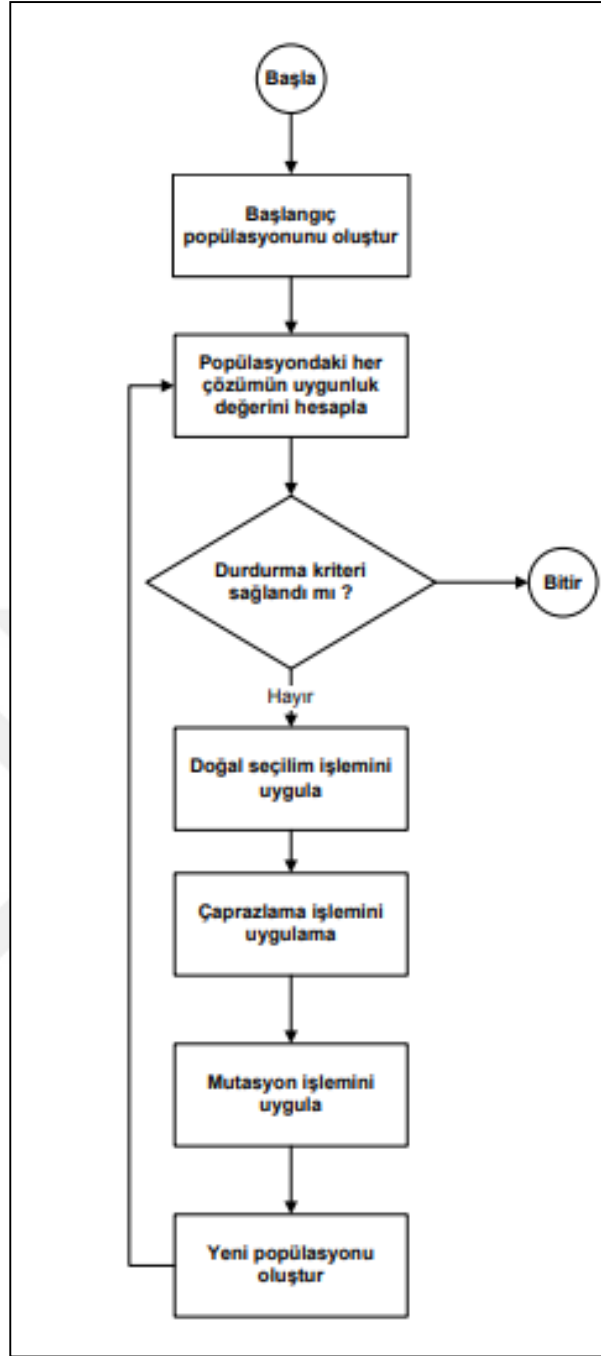
- Genetik algoritma

GA, evrimsel algoritmaların bir alt kategorisidir ve en yaygın kullanılan dalıdır. Yöntem, klasik çözüm yöntemleri ile çözülemeyen veya çözümü uzun zaman alan problemler için kullanılmaktadır. Mutasyon, gen çaprazlama ve seçim gibi özelliklerle evrim teorisinden esinlenen popülasyon tabanlı meta-sezgisel algoritmadır (Beasley ve diğ, 1993). Parametre ve sistem tanımı, kontrol sistemleri, robotik uygulamaları, mühendislik tasarımları, planlama, yapay zeka uygulamaları, ağ tasarım problemleri, rotalama problemleri, çizelgeleme problemleri, kombinatoriyal optimizasyon problemleri, sosyal ve ekonomik planlama problemleri gibi birçok farklı alanda kullanılmaktadır (Mijwel, 2015). GA, diğer sezgisellerden elde edilen bir başlangıç çözümü (popülasyonu) ile başlar. Yöntem popülasyona dayalı olduğundan, her bir iterasyonda çok sayıda çözüm elde edilir (Gandomi ve diğ, 2013b) Başlangıç çözümünün her bir parametresi gen, çözümün tamamı ise kromozom olarak kabul edilir.

Yeni çözümler bu kromozomların üzerine işlenir ve popülasyonlar oluşturulur (Sallabi ve diğ. 2009, ss. 389-417). Kromozomların yapısı, canlılardaki gen transferini taklit

etmektedir. İlk kromozomların oluşturulmasının ardından, mevcut iki kromozomun çaprazlanmasıyla, kromozomun mutasyona uğramasıyla ya da inversiyon ile yeni nesil kromozomlar üretilir. Çaprazlama ile iki kromozomdan yavru bireyler oluşturulur. Mutasyon ile kromozom üzerindeki bazı noktalar kalıcı olarak değiştirilir. İnversiyonda ise, kromozomun bazı noktalardaki parçalar değiştirilir ya da birleştirilir (Mitchell, 1998, ss. 15-31). Üretilen tüm yeni kromozomlar uygunluk fonksiyonu ile değerlendirilir. Değerlendirme sonucunda amaç fonksiyonuna en uzak kromozomlar elenir (seçilim) ve daha iyi uygunluk değerine sahip olanlar sonraki nesillere aktarılır (İşçi ve Korukoğlu, 2003). GA, bu iterasyonlar ile en iyi kromozomun yani optimum çözümün bulunması amaçlar (Yüksel ve diğ, 2019; Keskindürk ve diğ, 2015). Genetik Algoritma akış şeması, Şekil 3.9’da gösterilmiştir (Yıldırım, 2014).





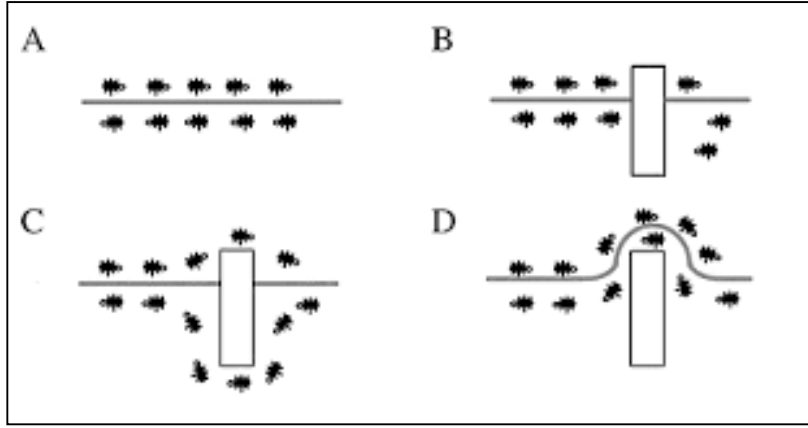
Şekil 3.9 : Genetik algoritma akış şeması (Yıldırım, 2014).

Uygunluk fonksiyonunun formülasyonu, popülasyon büyüklüğünün tanımı, mutasyon ve geçiş oranı gibi önemli parametrelerin seçimi dikkatlice yapılmalıdır. Herhangi bir uygunsuz seçim, algoritmanın yakınsamasını ve anlamlı sonuçlar üretilmesini zorlaştırır. GA, en iyi çözümü bulmayı garanti etmez. Yöntem, ilk iterasyonlarda hızlı bir şekilde iyileşir ve optimuma yaklaştıkça iyileşme üstel olarak yavaşlar (Keskindürk ve diğ, 2015).

Dikkatli davranılması gereken noktalar bulunmasına rağmen genetik algoritmalar, doğrusal olmayan optimizasyon problemleri için yaygın olarak kullanılan optimizasyon algoritmalarıdır (Yang, 2020, ss. 90-100). Yöntem, ARP çözümünde de sıklıkla kullanılmaktadır (Şahin ve Eroğlu, 2014).

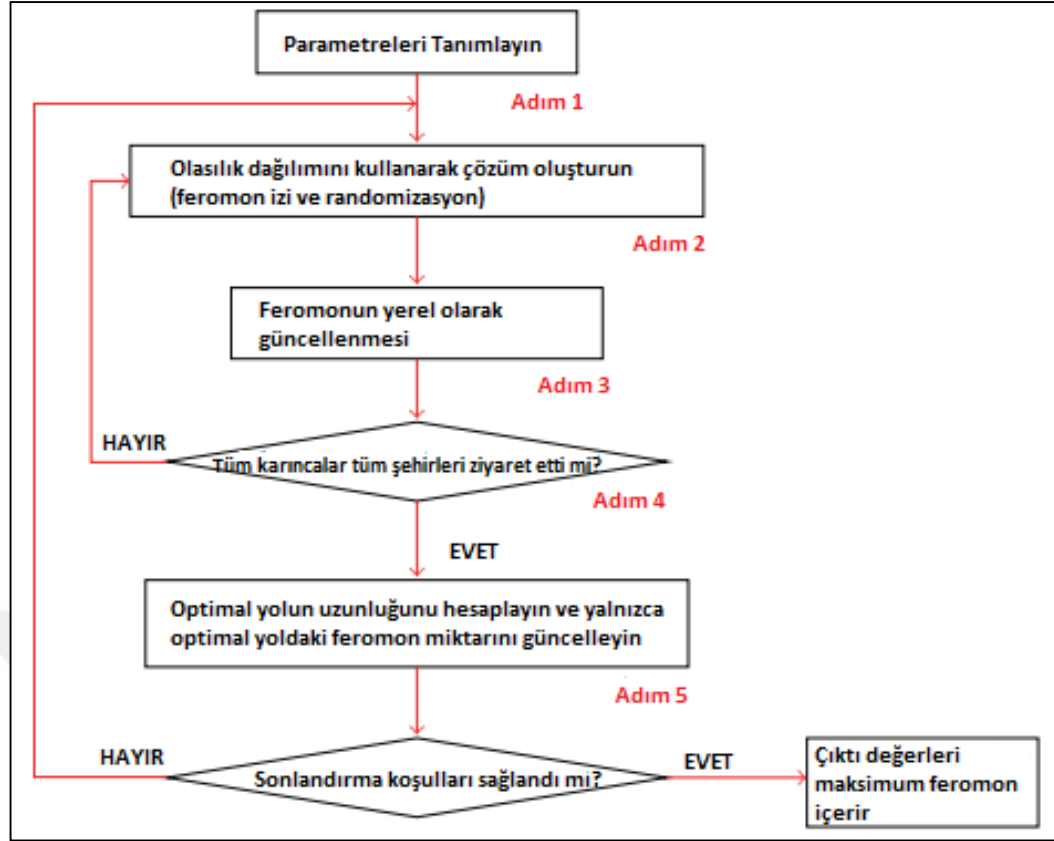
- Karınca kolonisi optimizasyonu algoritması

KKOA ile ilgili ilk çalışma Dorigo ve arkadaşları tarafından 1991 yılında yapılmıştır (Dorigo ve diğ, 1991). Karıncaların belirli bir sisteme göre hareket ettiği fikri ile yola çıkılan çalışmada, 200 adet karınca ile deneyler yapılmıştır. Karıncaların önlerine yiyecek ve engeller konularak, hareketleri incelenmiştir. İlk olarak “Karınca Sistemi” olarak tanımlanan bu gözlemler sonucunda karıncaların en kısa yolu buldukları görülmüştür (Düzakın ve Demircioğlu, 2009). Kısacası yöntem, karıncaların doğadaki yiyecek arama ve yuvaya dönecek en kısa yolu bulabilme davranışlarının taklit edilmesiyle geliştirilmiş popülasyon tabanlı bir meta-sezgisel algoritmadır (Caric ve Gold, 2008, ss. 57-71). Doğal yaşamda karıncalar, birbirleri ile bilgi aktarımını feromon denen kimyasal bir madde ile yapmaktadır. Karıncalar gıdaya giderken ve gıdayı yuvaya taşıırken geçtikleri yollara bir miktar feromon maddesi bırakır. Bu davranış, kendilerinden sonra gelen karıncaların feromonları takip ederek gıdaya giden en kısa yolu bulmalarını sağlamaktadır (Dorigo ve Gambardella, 1997). Feromon izlerinin yoğun ve güncel olması, bu yolun kaliteli olduğunu gösterir ve tercih olasılığını artırır. Feromon maddesi o kaynağa giden yoldaki karınca sayısının azalmasıyla, zaman içerisinde uçmakta ve kaybolmaktadır (Karaboğa, 2017). Şekil 3.10’ da, karıncaların feromon oluşturma ve bu feromonları takip etme şekilleri gösterilmiştir. (A) Karıncalar yiyecek kaynağı ve yuva arasına feromon izi bırakırlar (B) yerleştirilen bir engel izi böler (C) karıncalar engelin etrafından dolaşarak iki farklı yol bulurlar (D) daha kısa olan yol boyunca yeni bir feromon izi oluşur (Şenaras ve İnanç, 2017).



Şekil 3.10 : Karınca kolonisi algoritması (Şenaras ve İnanç, 2017).

Biyolojik örneğe benzer şekilde, KKOA’ da yapay feromon yollarının aracılık ettiği ve yapay karıncalar adı verilen basit ajanlardan oluşan bir koloninin dolaylı iletişime dayanır (Caric ve Gold, 2008, ss. 57-71). Yapay karıncalar, gerçek mesafeler dikkate alınarak düzenlenmiş modeller üzerinde en kısa yolu araştırmaktadırlar (Şenaras, ve İnanç, 2017; Kesintürk ve Söyler, 2006). Ancak algoritma gerçek karınca davranışlarına ilave davranışlar da sergilemektedir. Yapay karıncaların hafızaları bulunmaktadır ve algoritmada eski kötü çözümlerin denenmemesi için bir tabu listesi oluşturulmaktadır. Ayrıca, yapay karıncalar yol tercihi sırasında feromon maddesinin takibinin yanı sıra, rastgele olarak yeni kaynaklar da aramaktadır (Karaboğa, 2017). Algoritmadaki feromon izleri, karıncaların çözülmekte olan probleme olasılıklı olarak çözümler oluşturmak için kullandıkları ve algoritmanın yürütülmesi sırasında arama deneyimlerini yansıtmak için uyarladıkları, dağıtılmış, sayısal bir bilgi görevi görür. Yapay karıncalar ise, yapay feromon izlerinin bir fonksiyonu olarak olasılıksal kararlar verir ve problemin girdi verilerine dayanan sezgisel rastgele bir yapı buluşsal yöntemi uygular (Caric ve Gold, 2008, ss. 57-71). Şekil 3.11 karınca kolonisi algoritmasının ARP için akış şemasını göstermektedir (Yun ve diğ, 2013).



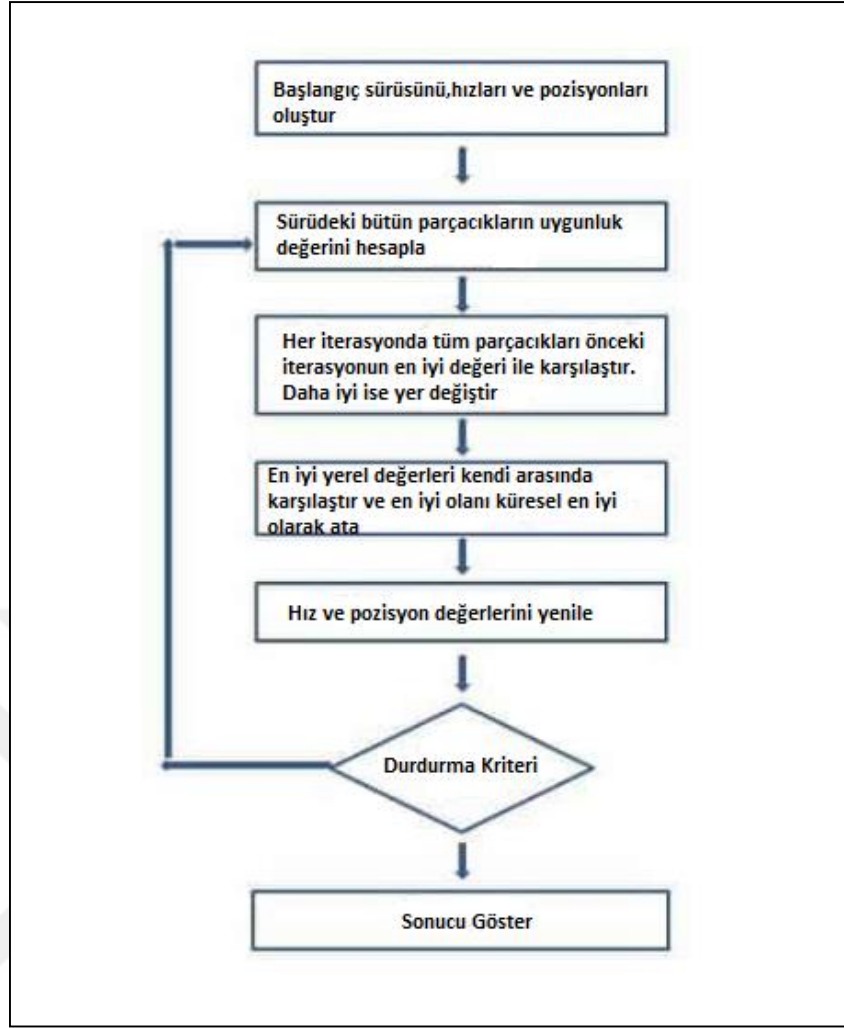
Şekil 3.11 : Karınca kolonisi algoritması akış şeması (Yun ve diğ, 2013).

Karıncı Kolonisi ilk olarak GSP çözümünde uygulanmış ve oldukça etkili çözümler bulunmuştur (Dorigo ve Gambardella, 1996). Bunun üzerine yöntem başka araştırmacılar tarafından da kullanılmaya başlanmış ve özellikle araç rotalama problemlerinin de içerisinde bulunduğu geniş bir kombinatoriyal optimizasyon problem yelpazesine uygulanmıştır (Şenaras ve İnanç, 2017).

- Parçacık sürü optimizasyonu algoritması

PSOA, Kennedy ve Eberhart (1995) tarafından doğrusal olmayan fonksiyonların optimizasyonu için geliştirilmiş meta-sezgisel bir hesaplama tekniğidir (Kennedy ve Eberhart 1995; Eberhart ve Kennedy, 1995; Eberhart, Simpson ve Dobbins 1996). Parçacık sürüsü kavramı, ilk olarak basitleştirilmiş bir sosyal sistemin simülasyonu olarak ortaya çıkmıştır. Asıl amaç, bir kuş- balık sürüsünün içgüdüsel hareketlerinin grafiksel olarak simüle edilmesidir (Shi, 2001). Kuşlar besin ararken, yeme en yakın olan kuşun peşinden gider.

PSOA, genetik algoritmaya benzer şekilde rastgele çözümler bulunduran ve sürü olarak tanımlanan bir popülasyon ile başlatılır ve nesilleri güncelleyerek optimum çözümü araştırır (Çavuşlu ve diğ, 2010). Bununla birlikte, yöntem GA'dan farklı olarak her potansiyel çözüme rastgele bir hız atar ve potansiyel çözümlerin problem uzayında dolaşması sağlanır. Daha detaylı olarak, kuş-balık gibi bazı sürülerinin davranışları ile aynı şekilde algoritmada da parçacık olarak adlandırılan muhtemel çözümler, aralarındaki bilgi aktarımı sayesinde o andaki optimum parçacığı izleyerek problem uzayında dolaşırlar (Çevik ve Koçer, 2013). Sürüdeki her bir parçacığa rastgele konum ve hız atanır ve bu şekilde diğer parçacıklardan aldığı bilgiler doğrultusunda parçacık optimum sonuca doğru yönelir. Uygunluk fonksiyonu ile her bir parçacığın uygunluk değeri hesaplanır. Bu fonksiyon parçacıkların en iyi çözüme ne kadar yaklaştıklarını ölçmeyi amaçlar (Kennedy ve Eberhart, 1995). Parçacıkların performansı konumları ile belirlenmektedir. Her iterasyonda, parçacıkların konum ve hızı iki bilgi kaynağından güncellenir. İlk bilgi kaynağı, her bir parçacığın çözümü aradığı süre boyunca çözüme en çok yaklaştığı, bireysel en iyi performans çözümüdür. Bu duruma “pbest” denmektedir. İkinci bilgi kaynağı ise, bütün sürü içinde ve arama boyunca çözüme en çok yaklaşan parçacığın en iyi performans çözümüdür ve gbest olarak tanımlanır (Shi, 2001). Bu iki bilgiye göre parçacıkların konum ve hızları her yeni nesilde güncellenir ve bu sayede popülasyonda (sürüde) bulunan parçacıklar giderek daha iyi pozisyona gelir. Bitirme kriterine ulaşıncaya kadar parçacıklar sürü halinde aramaya devam eder (Kılıçaslan ve diğ, 2019; Xue ve diğ, 2014). Şekil 3.12 parçacık sürü optimizasyonu algoritması akış şemasını göstermektedir (Öztürk ve Alkan, 2013). Diğer evrimsel algoritmalara nazaran parçacık sürü optimizasyonu algoritmasının uygulanması, kullanılan parametre sayısının az olması nedeniyle kolaydır (Çevik ve Koçer, 2013; Çavuşlu ve diğ, 2010).

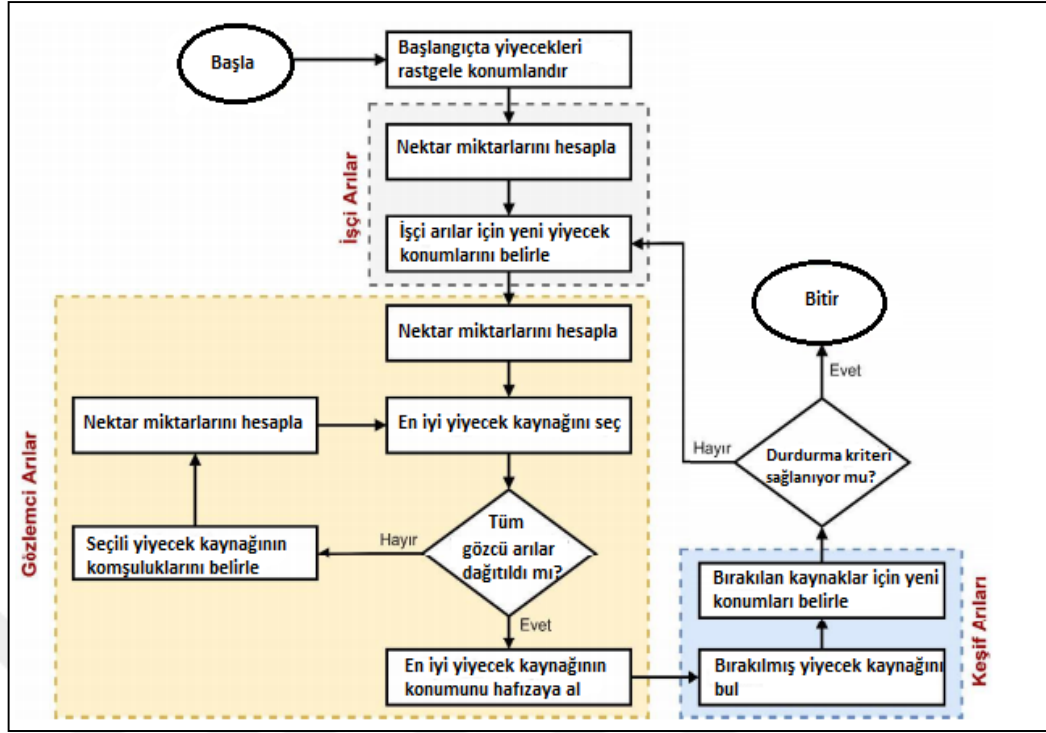


Şekil 3.12 : Parçacık sürü optimizasyonu algoritması akış şeması (Öztürk ve Alkan, 2013).

- Yapay arı kolonisi algoritması

İlk olarak, Sato ve Hagiwara (1998), arıların davranışlarını gözlemlemiş ve “Arı Sistemi” olarak isimlendirilen sezgisel bir yaklaşım geliştirmiştir. Ardından, Teodorovic ve diğ. (2005) Arı Sistemi’nin bir türevi olan “Arı Kolonisi Optimizasyonunu” geliştirmiştir (Teodorovic, ve Dell’Orco, 2005). Son olarak YAKA, 2005 yılında Karaboğa tarafından, bal arılarının yiyecek, iletişim ve arama davranışlarının benzetimine dayanarak modellenmiştir (Karaboğa, 2005). Popülasyon tabanlı meta-sezgisel yapay arı kolonisi algoritmasında üç çeşit arı bulunmaktadır. Bunlar; ziyaret edeceği besin kaynağı atanmış olan işçi arı, besin kaynağını tercihlerine göre belirleyen gözcü arı ve rastgele besin arayışındaki kâşif arıdır (Elen ve Ekmekçi, 2020).

Yöntemde bazı kabullenmeler yapılmıştır. Besin kaynağı, işçi arı ve gözcü arı sayılarına eşittir, aynı zamanda işçi arıların ve gözcü arıların sayısı da eşittir (Kaya, 2020). Algoritmanın başlangıç adımında, işçi arılara rastgele besin kaynağı atanır ve işçi arılar bu besin kaynağına gidip nektarları toplayarak kovana geri döner. Kovana dönen işçi arılar bilgi paylaşımı için arıları dansı (waggle dance) yapar. Gözcü arılara besin kaynağının kalitesi, miktarı ve konumu ile alakalı bilgi aktarımı sağlar (Grüter ve Farina, 2009). Yuvada bekleyen gözcü arılar, işçi arılardan bilgileri alır. Besin ile kovanın konumu ve besinin kalitesine göre bir kaynağı tercih eder. Yeni bulunan besin daha iyiye, gözcü arı işçi arı olur ve kendi bulduğu kaynak ile ilgili bilgileri yuvadaki gözcü arılara aktarır. Bu şekilde daha kaliteli yeni yiyecek kaynakları keşfedilir. Eğer kaynak daha iyi değilse, kovana geri gelerek gözcülük görevine devam eder. İşçi arılar besin tükenene kadar aynı kaynaktan yiyecek toplamaya devam ederler. Nektar kaynağı tükendikten sonra işçi arılar modelde kaşif arı olarak görev alır ve rastgele kaynak aramaya başlar. Yiyecek kaynağı bulan kaşif arılar yeniden işçi arı olurlar (Akay, 2009, ss. 301-302). Yiyecek arama sürecinin başlangıcında, kaşif arılar çevrede rastgele arama yaparak yiyecek aramaya başlarlar. Yiyecek kaynağı tükendikten sonra işçi arılar, kaşif arı olurlar ve rastgele yeni bir kaynak aramaya başlar (Reis ve Torun, 2015; Akay, 2009, ss. 301-302). Doğadaki uygulamasına benzer şekilde, uzaydaki sonuçlardan, optimumunu veren sonuca en uygun değeri bulmak amaçlanmaktadır. Burada, besin kaynaklarının yerleri optimizasyona ait olasılık sonuçlarını ve besin miktarları ise sonuçların kalitesine karşılık gelmektedir. Algoritma bitirme kriterine ulaşıncaya kadar yiyecek kaynakları, işçi arı, kaşif arı ve gözcü arı fazları tekrarlanır (Karaboga ve Akay, 2009). YAKA adımları Şekil 3.13'te gösterilmektedir (Elen ve Ekmekçi, 2020).



Şekil 3.13: Yapay arı kolonisi algoritması akış şeması (Elen ve Ekmekçi, 2020).



4. YÖNTEM

4.1 Problem Tanımı

Yakıt masrafları deniz taşımacılığında karşılaşılan en önemli maliyet kalemlerinden biridir. Yakıt tüketimi; gidilecek mesafe, hız, geminin yük durumu, hava durumu, makine tipi vb. parametrelerin varlığıyla değişebilmektedir. Özellikle, yakıt tüketiminin hızın kübik fonksiyonu olduğu düşünüldüğünde, hız ayarlamalarının maliyetleri düşürmede büyük önemi bulunmaktadır.

İncelenen problem, yüklerin alınması ve teslim edilmesi için gereken zaman pencerelerinin ayırıştırılarak her bir gemi tarafından hizmet edilecek kargo kümesine (kontrata) karar verilmesinden meydana gelmektedir. Bu karar, gemilerin varış zamanı, ayrılış zamanı, limanda bekleme süresi, seyir hızı gibi değişkenleri içermektedir. Problemdeki asıl amaç; toplam operasyonel maliyeti en aza indirirken, tüm kargoları ulaştırabilmektir.

Problem özellikleri ve model yaklaşımı:

- Gemilerin, belirli bir planlama süresi içinde hizmet etmesi gereken sözleşmeler kümesi bulunmaktadır. Bu küme, planlama başlangıcında bilinmektedir.
- Her bir sözleşme; tek bir sefer (yük) için, başlangıç ve varış limanları ile bu limanlarda, yüklerin alınması ve teslimi için gereken zaman pencereleri ile karakterize edilmektedir (Yükü almak için yapılan balastlı taşıma da dahil).
- Kapasite kısıtını elemine edebilmek için, tüm gemilerin tüm kontratlara hizmet götürebilmeleri planlanmıştır. Probleme yakın kapasiteli capesize (110,000-199,000 dwt) gemiler atanmıştır. Ancak, seyir hızları, yakıt tüketimi, maliyetler vb. parametreler, gemiden gemiye değişmektedir.
- Seyir hızı kontrol edilebilen bir değişkendir. Yakıt tüketimi seyir hızına bağlı olduğundan, hızın kontrol edilmesi yalnızca sefer süresini değil, aynı zamanda sefer maliyetlerini de etkilemektedir.
- Problemde her bir sözleşmeden elde edilen gelir sabit kabul edilmiştir. Bu nedenle, toplam ilgili maliyetler en aza indirilerek net kar maksimize

edilebilmektedir. Söz konusu maliyetler, genel olarak operasyonel maliyetlerdir.

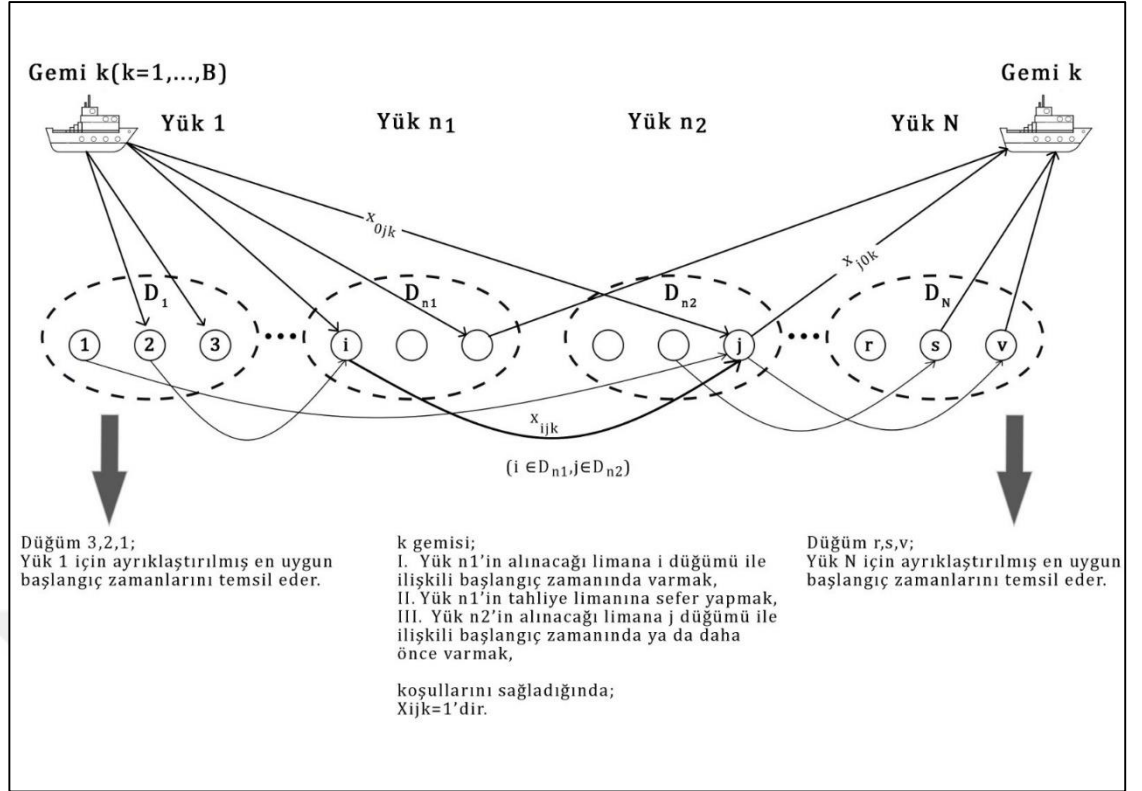
- Taşınacak olan yük çifti, aynı limanda tahliye edilip, ardından yüklenmeyecekse bir sonraki yükün alınmasına kadar gemi balastlı olarak seyretmesi gerekecektir. Bu balastlı seyirler, toplam kaçınılabilecek maliyetin önemli bir bölümünü temsil etmektedir.
- Modelde toplam kazancı arttırmak yerine, operasyonel maliyetleri düşürmeye odaklanılmıştır.

4.2 Modelin Matematiksel İfadesi

Modeli oluşturmak için kullanılan ana strateji, yüklerin alınması ve teslim edilmesi için gereken zaman pencerelerinin ayrı ayrı sınıflandırılmasıdır. İlk olarak Gatica ve Miranda (2011) tarafından kullanılan modele ait gemi ve zaman penceresi kısıtları değiştirilerek model yeniden kurgulanmıştır.

- A aralığı, yükün alınması için gereken zaman pencerelerini temsil ettiğinde, başlangıç limanına varış (hizmet başlangıcı), yalnızca ayrıklaştırılmış zaman kümesi $\bar{A} = \{a_1, a_2, \dots, a_i\}$, $a_i \in A$ ve her $i = 1, 2, \dots, m$ gibi belirli bir zaman diliminde gerçekleşebilir. Yükün teslim edileceği zaman pencereleri de benzer şekilde probleme dahil edilir.
- $i = 1, \dots, N'$ e kadar indekslenmiş olan noktalar, her yük için ağdaki ayrık ve uygulanabilir başlangıç zamanlarını temsil eder.
- Her i düğümü için, $n(i)$ bu düğümle ilişkilendirilmiş yükü temsil eder.
- Yük $n(i)$ için düğüm kümesi D_i ile gösterilir.
- Probleme dahil edilecek gemiler, $k = 1, \dots, B'$ ye kadar indekslenir.

Şekil 4.1’de tek bir gemi için sefer örneği verilmektedir. Her bir küme, tek bir yüke ait düğümleri bulundurur. Düğümler ise, belirlenmiş olan yüke hizmet vermeye başlanacak olası zamanları temsil eder. Her bir sefer, ağdaki düğümler ile ilişkilidir ve her bir düğüm arasında birden çok yay oluşabilir.



Şekil 4.1 : Bir gemi için sefer örneği.

k gemisi peş peşe n(i) ve n(j) seferlerini yaptığında, i den j' ye yay(i,j,k) oluşur. Yay (i,j,k) nın çözümün bir parçası olması için ($x_{ijk}=1$), yük n(i) ile yük n(j) uyumlu olmalı ve yükler k gemisi ile de taşınabilir olmalıdır.

k gemisi, belirlenen i zamanında ilk yükün teslim limanı n(i) den, bir sonraki yükün teslim limanı n(j) ye j zamanında (düğümünde) hizmet verebiliyorsa, yay (i,j,k) ağa dahil edilir. Yük n(j) nin alınması için, yapılan balastı seyir süreleri de toplam seyir süresine eklenmektedir.

Her yay için maliyet parametresi c_{ijk} , ilk yük n(i) nin teslim edilmesinin ardından, bir sonraki yük n(j) nin, teslimine kadar oluşan maliyeti temsil eder. Maliyetler, n(i) yükü ile alakalı limanlar arası seferleri, bekleme zamanlarını ve n(j) yükünün alımı için, limana yapılan balastlı (boş) seferi içerir.

Ağı tamamlamak için, tüm gemilerin başlangıç noktalarını temsil eden, hayali bir "0" düğümü oluşturulmuştur. Yay (0,j,k); n(j)'nin gemi tarafından hizmet verilen ilk yük olduğu anlamına gelmektedir.

Yay (j,0,k) ise, n(j) nin gemi tarafından hizmet verilen son yük olduğunu gösterir. c_{0jk} , k gemisi yük n(j) için kullanılıyorsa minimum toplam maliyeti gösterir ve k

gemisinin, başlangıç konumuna göre hesaplanır. k gemisinin, en son yükün varış limanında kalacağı kabul edilir ve c_{j0k} maliyetinde boş sefer gideri bulunmaz.

Problemin matematiksel gösterimi aşağıdaki gibidir:

Amaç fonksiyonu:

$$\min z = \sum_{(i,j,k) \in A} c_{ijk} x_{ijk} \quad (4.1)$$

kısıtlar,

$$\sum_{i \in V(0,i,k) \in A} x_{0ik} = 1, \quad k = 1, \dots, B \quad (4.2)$$

$$\sum_{(i,j,k) \in A / j \in D_n} x_{ijk} = 1, \quad n = 1, \dots, N \quad (4.3)$$

$$\sum_{i \in V / (i,j,k) \in A} x_{ijk} = \sum_{l \in V / (j,l,k) \in A} x_{ijk}, \quad j \in V, \quad k = 1, \dots, B \quad (4.4)$$

$$x_{ijk} \in \{0,1\} \quad (i,j,k) \in A \quad (4.5)$$

$$x_{ijk} = \begin{cases} 1, & (i,j,k) \text{ yayı çözümün parçası ise, } i \text{ noktasından } j \text{ noktasına gidiliyor ise} \\ 0, & \text{diğer} \end{cases} \quad (4.6)$$

Burada;

- N Hizmet götürülecek kontrat sayısı
- V Ağdaki düğümler kümesi
- D_n Sefer n için muhtemel başlangıç zamanları
- B Kullanılacak gemi sayısı
- A Ağdaki yay kümeleri
- C_{ijk} Yay (i,j,k) için maliyet

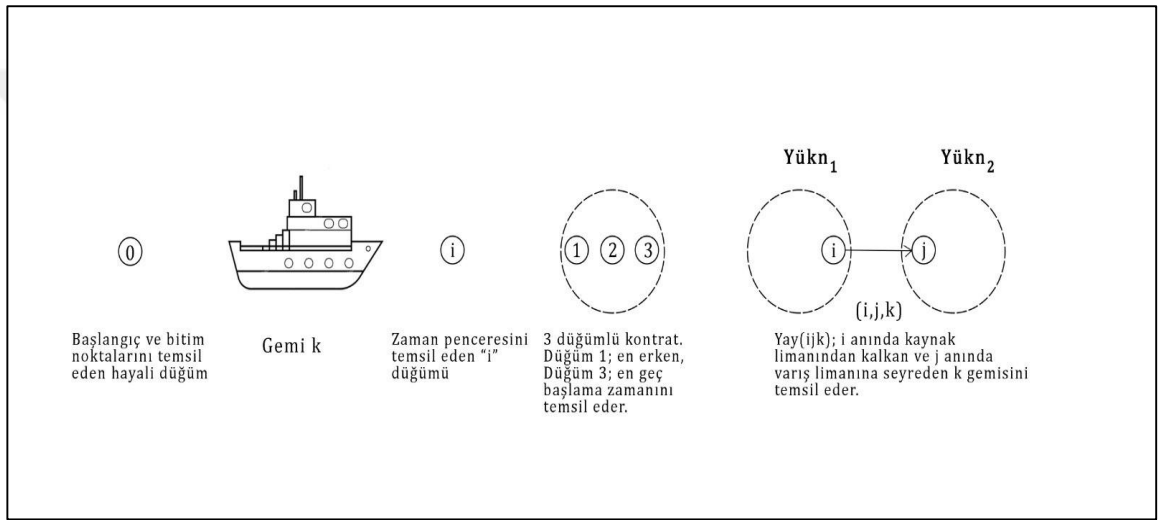
Amaç fonksiyonu 4.1; toplam sistem maliyetini temsil etmektedir.

Kısıt 4.2; her bir geminin, hizmet vereceği kargo dizisinde ve tek bir rotada kullanılmasını sağlamaktadır.

Kısıt 4.3; her n yükü için, yay giriş kümesi D_n seçilmiş olması durumunu sınırlandırmaktadır. Daha açık ifadeyle, her yükün 1 kez ve yükü almak için belirlenen zaman penceresindeki düğümlerden birinden başlayarak taşınmasını kontrol etmektedir.

Kısıt 4.4; merkezi hayali düğümler dışındaki düğümler için bir giriş yayının seçilmesi durumunda, bir çıkış yayı da seçilmesi gerektiğini ve bu durumda, her iki yayın da aynı gemi ile ilişkilendirilmesi gerektiğini belirtmektedir. Hayali düğümler için ise bu kısıt; k gemisiyle ilişkilendirilmiş bir ayrılma yayı oluşturulmuşsa, aynı gemi ile ilişkilendirilmiş bir giriş yayı oluşturulmalıdır şeklinde işleme dahil edilmektedir.

Şekil 4.2’de modelin görsel ifadesi verilmektedir. Her bir sözleşme, başlangıç ve teslim zamanlarını temsil eden 3 düğümden meydana gelir. Düğümler arasında çizilen yaylar, başlangıç ve teslim limanları arasındaki mesafe, yapılan hız ve kullanılan yakıt maliyetiyle ilişkili olarak, tek bir seferi temsil eder.



Şekil 4.2: Modelin görsel ifadesi.

4.3 Karınca Kolonisi Optimizasyonu Algoritma Modeli

4.3.1 Geçiş kuralı

Dorigo vd. tarafından geliştirilen orijinal karınca kolonisi optimizasyonunda, yukarıdaki bölümlerde detaylı olarak anlatıldığı üzere, her bir karınca kolonisi besine ulaşmak için bir rota oluşturur. Model ifadesinde ise, karıncalar hizmet götüren araçları, besin ise müşterileri temsil eder. Amaç, müşterileri ziyaret etmek için araç rotası oluşturmaktır. İki nokta arası feromon iz miktarına, noktalar arası uzaklık eklenerek bir karıncanın hangi yolu seçmesi gerektiği belirlenir. Sistemde P_{ij}^k , k karıncasının i noktasından j noktasına hareket etme olasılığını sembolize etmektedir (Dorigo ve diğ, 1996).

$$P_{ij}^k = \frac{[\tau_{ij}]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{k \in N_i^k} [\tau_{ik}]^\alpha \cdot [\eta_{ik}]^\beta} \text{ eğer } j \in N_i^k \quad (4.7)$$

Denklem 4.7’de j noktaları daha önceden tur içerisinde k karıncası tarafından ziyaret edilmemiş noktalardan oluşmaktadır. Buna göre daha büyük feromon ve görünürlük değeri toplamına sahip yol en çok olasılıkla seçilmektedir.

Burada;

$j \in N_i^k$: k. karıncanın gidebileceği tüm j noktaları

$k \in N_i^k$: i. noktada bulunan tüm karıncalar

P_{ij}^k : k. karıncanın i. noktadan j noktasına geçme olasılığı

τ_{ij} : i ve j noktaları arasındaki feromon değeri

η_{ij} : i ve j noktaları arasındaki görünürlük değeri

α : feromon katsayısı

β : görünürlük katsayısı

N : noktalar kümesi

4.3.2 Feromon yenilenmesi

Bu aşamada, çözüm uzayı incelenir. Yeme giden en iyi yolu ,en fazla sayıda karınca takip eder ve bununla doğru orantılı olarak yoldaki feromon miktarı da artar.Tüm karıncalar turlarını tamamladıktan sonra daha iyi sonuçlar için rotalarında yenileme yapılır. Yenileme, karıncaların geçtiği tüm yollardaki feromonların belirli oranda buharlaşmasıyla yapılır. Denklem 4.7’de verilen değişkeni, i ile j noktaları arasındaki hareketin ne kadar tercih edildiğini göstermektedir. Amaç, en iyi yolun hesaplanması olduğundan, problem, cazip hareket değişkeni ’nin daha büyük olacağı şekilde kurulmalıdır. değişkeni, i ve j noktaları arasındaki feromon miktarını temsil etmektedir. Her döngü sonunda daha iyi yolu bulmak için sıfırlanan değişkenin hesaplanması için aşağıdaki denklem kullanılır. Burada t döngü sayısı, ise buharlaşma parametresini ifade etmektedir.

$$\tau_{ij}^t = \rho \tau_{ij}^{t-1} + \Delta \tau_{ij} \quad (4.8)$$

1-p yoldaki feromon buharlaşma oranı olduğunda, i ve j noktaları arasına eklenen feromon miktarı aşağıdaki gibidir;

$$\Delta t_{ij} = \sum_{k=1}^m \Delta t_{ij}^k \quad (4.9)$$

Δt_{ij} , karıncaların i ve j noktaları arasında gidip gelmeleri sonucu oluşan toplam feromon katkısını göstermektedir. m karınca bulunan bir sistemde Δt_{ij} ;

$$\Delta t_{ij} = \begin{cases} \frac{Q}{L_k} & \text{eğer } k, (i, j) \text{ yolunu kullandıysa,} \\ 0 & \text{diğer durumlarda} \end{cases} \quad (4.10)$$

L_k değişkeni, karıncanın (k) çözüm turunda yola çıktığı şehre dönene kadar kat ettiği toplam mesafe ve Q sabit bir sayıdır.

4.4 Genetik Algoritma

4.4.1 Temel kavramlar

GA, evrimsel süreci taklit ederek problemlere çözüm üretmeyi amaçladığı için genetik algorithmadaki birçok kavram genetik bilimi ile benzerlik göstermektedir.

Genetik algorithmalarda gen, kromozomları oluşturan her bir çözümün tek bir özelliğini temsil eder. Algoritmanın kullanıldığı programlama yapısında, gen yapıları programı oluşturanın tanımlamasına bağlıdır (Elmas ve Yiğit, 2007).

Genler bir araya gelerek, kromozomları oluşturur. Algorithmada her kromozom problem için olası çözümleri ifade etmektedir. Problem çözümünde en önemli birimdir.

Kromozomların bir araya gelmesiyle de popülasyon oluşur. Kromozomlar olası çözümleri temsil ettiğinden, popülasyon bu çözümler topluluğunu ifade eder. Popülasyondaki kromozom sayısının problemin yapısına göre belirlenmesinin ardından, nesiller boyunca burada korunur.

Allele, genotip ve fenotip kavramları GA'da kullanılan diğer önemli kavramlardır. Allele, genin alabileceği değerlerdir. Genotip, kromozomun genetik yapısını oluşturur. Fenotip ise genotipin fiziksel açıklamasıdır.

Uygunluk değeri, kromozomların amaç fonksiyonundaki performans değeridir. Kromozomlar ile popülasyon içindeki iyi özellikler sonraki nesillere aktarılır.

Uygunluk değeri, uygunluk fonksiyonu kullanılarak hesaplanır. Uygunluk fonksiyonu, genlerin çaprazlama ve mutasyon gibi çeşitli evrimsel işlemlere tabi tutulması ile hesaplanır. Bu işlemler belli bir nesil sayısına ya da istenen uygunluk değerine ulaşıncaya kadar devam ettirilir (Chu ve Beasley, 1997). İyi olan kromozomları koruyarak daha iyiye götürmesinden dolayı, algoritmanın en önemli faktörlerinden biridir (Yücel, 2012, ss. 58-66).

4.4.2 Genetik algoritma adımları

Adım 1. Kromozomların kodlaması

Bu adım, fenotipin genotipe dönüştürülme işlemidir. GA uygulamasına başlamadan önce, veriler algoritmanın kullanabileceği şekilde kodlanır (Rothlauf, 2006). Kromozom yapısı aşağıda verilmiştir.

$$S = S_1 S_2 \dots S_j \dots S_n \quad (4.11)$$

$S_i, i = 1, 2, \dots, n$ genleri ifade eder. Her bir gen kromozom üzerinde lokus adı verilen belirli bir yerde konumlandırılır (Mitchell, 1998, ss. 15-31). i değeri genin lokusudur. Bir kromozomun uzunluğu (n) sabit tutulur (Sakawa, 2012, ss. 11-25). Sayı değerini genetik koda çevirme işlemi fonksiyon olarak düşünülebilir. Fonksiyonda fenotipler çözüm uzayında, genotipler kod uzayında yer alır. Fonksiyonunun çalışabilmesi için çözüm uzayı ve kod uzayı birebir eşleşir olmalıdır (Eren, 2002, ss. 4-18). Gerçek sayılar, tamsayılar, harfler ya da semboller kodlama için kullanılır (Sakawa, 2012, ss. 11-25).

- İkili kodlama: Çözüm uzayındaki değerler ikilik sayma sistemine dönüştürülür (Yücel, 2012, ss. 58-66). İkili kodlamada, her kromozom 0 ve/veya 1 karakter dizilerinden oluşur. İkili kodlama tekniği çaprazlama ve dönüşüm aşamalarını kolaylaştırır (Yüksel ve diğ, 2019). Çözüm uzayındaki değerlerin temsil edilebileceği bir dizi uzunluğu belirlenir. Şekil 4.3'te ikili kodlama ile temsil edilen kromozom gösterimi verilmektedir (Çakar, 2009, ss. 39-65).

KROMOZOM A	10110010110010101110
KROMOZOM B	11111110000011000001

Şekil 4.3: İkili kodlama örneği (Çakar, 2009, ss. 39-65).

- Değer kodlama: Bu kodlama türünde problemin gerçek değerleri kullanılır. Kodlanan karakter dizileri tamsayı, ondalık sayı, harf ve hatta sözcüklerden oluşabilir. Şekil 4.4'te değer kodlamayı gösteren örnek kromozomlar verilmiştir (Sivanandam ve Deepa, 2008).

KROMOZOM A	ADBHNJMKLERTY
KROMOZOM B	(1,234),(5,678),(4,678)
KROMOZOM C	(ön),(sağ),(ön),(arka)

Şekil 4.4 : Değer kodlama örneği (Sivanandam ve Deepa, 2008).

- Permutasyon kodlama: Kodların karakter dizileri bir sırayı temsil eden sayılardan oluşur. Sıralamanın önemli olduğu problemlerin çözümde tercih edilir. Şekil 4.5'te permutasyon kodlama örneği verilmiştir (Dursun, 2009, ss. 29-49).

KROMOZOM A	135792468
KROMOZOM B	123498765

Şekil 4.5: Permutasyon kodlama örneği (Dursun, 2009, ss.29-49).

Adım 2. Başlangıç popülasyonu oluşturma

Bu aşamada, algoritmada kullanılacak bireyler oluşturulur ve popülasyondaki birey (kromozom) sayısı belirlenir (Okur ve Atlas, 2020). Kromozomlar, çözüm uzayının farklı bölgelerinde olabileceğinden, popülasyonu oluşturmak için, farklı noktalarda aynı anda arama yapılır. Başlangıç popülasyonundaki birey sayısı problemi programlayan tarafından ve genellikle problemin yapısına göre rassal olarak belirlenir (Altay, 2007, ss. 10-24). Popülasyona atanan kromozom sayısı sonraki nesillere de aktarılır. Her problem tipi için en iyi çözümün başlangıç popülasyonu sayısı farklıdır. Genetik algoritmanın göstereceği performans popülasyon büyüklüğü ile doğrudan ilişkilidir. Popülasyonda çok sayıda birey olması, çözüm uzayının daha geniş bir bölümünde arama yapılmasını sağlarken, her döngüde daha çok birey

inceleneyeceğinden, çözüme ulaşma süresini uzatır (Okur ve Atlas, 2020). Popülasyon büyüklüğünün gereğinden küçük olması ise algoritmanın çalışma hızını arttırırken, tarama daha dar bir çözüm uzayında yapılacağından çaprazlana için seçenekler azalır ve buna bağlı olarak, en iyi çözüme ulaşılma olasılığı da azalacaktır (Paksoy ve Uzun, 2008).

Adım 3. Uygunluk fonksiyonunun belirlenmesi

Uygunluk fonksiyonu, kromozomların popülasyon içerisindeki değerini belirler. Daha iyi bireyler hayatta kalır ve yeniden ürer (Dursun, 2009, ss. 29-49). İyi özelliklere sahip kromozomlar bir sonraki aşama için yeni topluma (kuşağa) aktarılır. Uygunluk fonksiyonu sayesinde olurlu olmayan çözümler tespit edilerek, dikkate alınmaz (Michalewicz, 2013, ss. 8-18; Kahraman ve Özdağlar, 2004). Uygunluk fonksiyonları en iyi olan bireyler ise en iyiye yakın çözümü temsil eden kromozomlardır (Paksoy ve Uzun, 2008; Mori ve Tseng, 1997, s.137).

Adım 4. Seçim operatörü

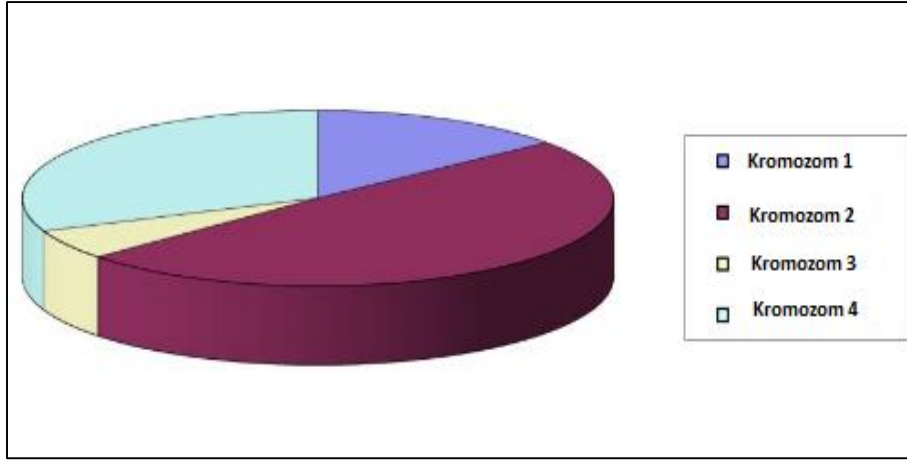
Seçme işlemi, yetenekli ve iyi olanın hayatta kalması prensibine uygun olarak çalışır (Paksoy ve Uzun, 2008). Seçimin belirli bir dengeye göre yapılması gerekir. Popülasyon büyüklüğünün belirlenmesinin ardından, seçilimin doğru olarak yapılması optimum çözüme gidebilmek için en önemli aşamalardan biridir. Seçimin çok güçlü olması, çözüme giden çeşitliliğin azalmasına, zayıf olması ise geniş bir çevrenin taranması sonucu evrimin yavaşlamasına neden olur (Okur ve Atlas, 2020; Bolat ve diğ., 2004). Seçilen iyi bir kromozom, yeni oluşturulacak toplumun bir veya daha fazla kromozomunun oluşumuna katılır. Böylece toplumdan topluma en iyiye doğru giden iyileştirilmiş kalıtsal özellikler aktarılabilir (Paksoy ve Uzun, 2008). Bir sonraki nesile Hangi kromozomların aktarılacağına tespiti farklı seçim teknikleri ile yapılır. Bu tekniklerden en çok tercih edilenleri; sıralı seçim yöntemi, turnuva yöntemi ve rulet tekerleği yöntemidir.

- Sıralı Seçim Yöntemi: Bu yöntemde bireyler uygunluk değerine göre çok olandan aza doğru sıralanır. En kötü uygunlukta olan bireye en düşük değer verilerek, uygunluk değeri ile doğru orantılı olarak atanan değer de artar (Liang ve Wang, 2020). Olasılıklar bu sıralamaya göre oluşturulur. Sonraki nesilde hangi kromozomların olacağı rassal olarak sayı ataması yapılarak belirlenir (Vural, 2005, ss. 18-31).

- Turnuva Seçim Yöntemi: İlk olarak, toplumu oluşturan kromozomlar arasından bir kısmı rastgele olarak seçilir ve eşleşme havuzunda eşleştirilirler. Rassal olarak seçilen kromozomlardan oluşan bu gruba turnuva genişliği denir (Bolat ve diğ., 2004). Eşleştirilen kromozomlar arasından uygunluk değeri yüksek olan birey yeni topluma aktarılmak üzere tutulur ve geriye kalanlar atılır (Altay, 2007, ss. 10-24; Miller ve Goldberg, 1995).

Yöntemde en iyi kromozomu tespit etmeye yönelik 2 eğilimden meydana gelen doğrusal bir sıralama oluşturulur. $[0.5 - 1]$ değerleri arasında bir p olasılığı belirlenir. En iyi 1. kromozom bir sonraki nesle p olasılıkla aktarılken, en iyi 2. kromozom bir sonraki nesle $p(1-p)$ olasılıkla ve n . kromozom bir sonraki nesle $p(1-p)^{n-1}$ olasılıkla aktarılır (Miller ve Goldberg, 1995). En iyi bireyin iki turnuvayı kazandığı kabul edilir ve iki kopyası tutulur. Ortalama bir kromozom, bir turnuva kazanır, bir tane kaybeder ve bir kopyası tutulur. En kötü bireyler turnuvayı kaybeder ve üretimleri durdurulur (Vural, 2005, ss. 18-31).

- Rulet tekerleği yöntemi: İlk olarak toplumu oluşturan tüm kromozomların uygunluk değerleri bulunur. Herbir kromozomun uygunluk değerlerinin toplanması ile, topluma ait uygunluk değeri hesaplanır. Her bir kromozomun uygunluk değeri, kromozomların toplam uygunluk değerlerine bölündüğünde, kromozomların seçilme olasılıkları bulunur (Dursun, 2009, ss. 29-49). Kromozomlar, uygunluk fonksiyonları (f) ölçüsünde rulet çarkına yerleştiriliyormuş gibi kabul edilir. Uygunluk fonksiyonu yüksek olan kromozomların rulet çarkında kapladığı hacim daha fazla olur (Şekil 4.6). Bu durumda, rulet çarkı dizi sayısı kadar çevrildiğinde uygunluk fonksiyonu yüksek olan kromozomların seçilme şansı da yükselir (Kahraman ve Özdağlar, 2004).



Şekil 4.6 : Rulet seçim yöntemi (Kahraman ve Özdağlar, 2004).

- Uygunluk fonksiyonuna göre kromozom i 'nin seçilme olasılığı $p(i)$ aşağıdaki gibi hesaplanmaktadır (Onwubolu, 2002, s.24);

$$p(i) = f_i / \sum_{j=1}^n f_j \quad (4.12)$$

Yöntemin adımları aşağıdaki gibidir (Sakawa, 2012, ss.11-25; Vural, 2005, ss. 18-31);

Adım 1: N tane kromozom içeren toplulukta t zamanındaki nesil için uygunluk değerleri $f_i (i = 1, \dots, N)$ ve tüm topluluğun toplam uygunluk değeri

$$f_{\text{toplam}} = \sum_{j=1}^N f_j \text{ hesaplanır.}$$

Adım 2: $[0,1]$ aralığında rastgele bir sayı yaratılır ve $s = \text{rand}() \times f_{\text{toplam}}$ hesaplanır.

Adım 3: $\sum_{j=1}^k f_j \geq s$ en küçük değerli k elde edilir ve k . birey $t+1$ zamandaki nesil için seçilir.

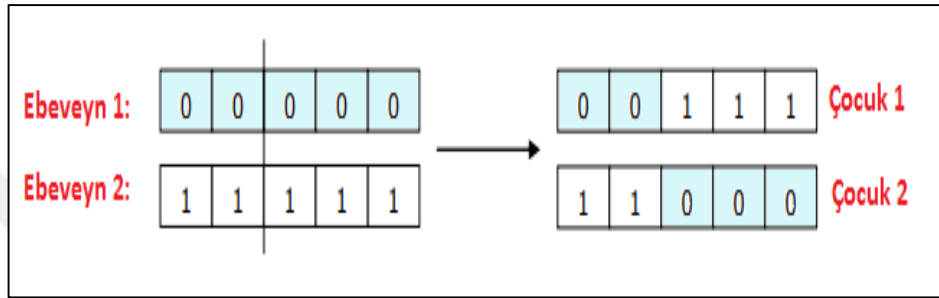
Adım 4: Seçilen birey sayısı N olana kadar 2. ve 3. Adımlar tekrarlanır.

Adım 5. Çaprazlama operatörü

Çaprazlama adımı, GA'nın en önemli aşamalarından biridir. Çaprazlama sürecinde seçilmiş iki bireye ait özellikler rastlantısal olarak çocuğa aktarılır ve iki bireye ait dizilerden yeni bir nesil meydana getirilir. Seçim işlemi ardından iyi özelliklere sahip bireylerden meydana gelen topluluğun çaprazlanmasıyla üstün ve iyi genlerin aktarımı sağlanır (Vural, 2005, ss. 18-31). Amaç, önceki topluluğa göre daha iyi bireylerden oluşan yeni ve daha gelişmiş bir popülasyon oluşturmaktır. Optimum çözümü olabildiğince erken bulabilmek için farklı çaprazlama teknikleri kullanılmaktadır

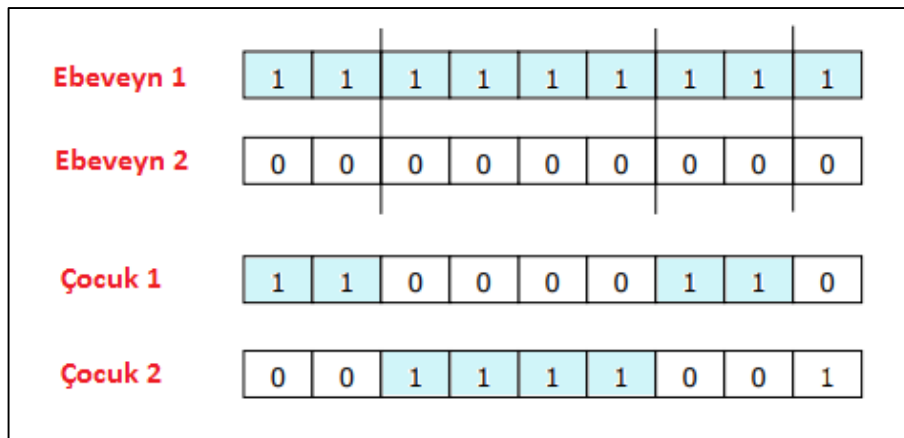
(Umbarkar ve Sheth, 2015). Bu tekniklerden en temel olanları ve en çok kullanılanları; tek-noktalı çaprazlama, çok noktalı çaprazlama, karşılaştırılmalı çaprazlama, tekdüze çaprazlama ve sıralı çaprazlamadır.

- Tek-noktalı çaprazlama: Bu teknik çaprazlama yöntemlerinin en temeli ve öncüsüdür. Ebeveyn dizileri üzerinde, toplam birey uzunluğundan daha kısa olan tek bir değişim noktası belirlenir. Ebeveynler, çocuk yapmak için belirlenen tek bir noktadan bölünür ve bölünen genler çapraz şekilde birleşir ve Şekil 4.7’de gösterilen şekilde yeni iki çocuk oluşur (Özalp ve Alp, 2020).



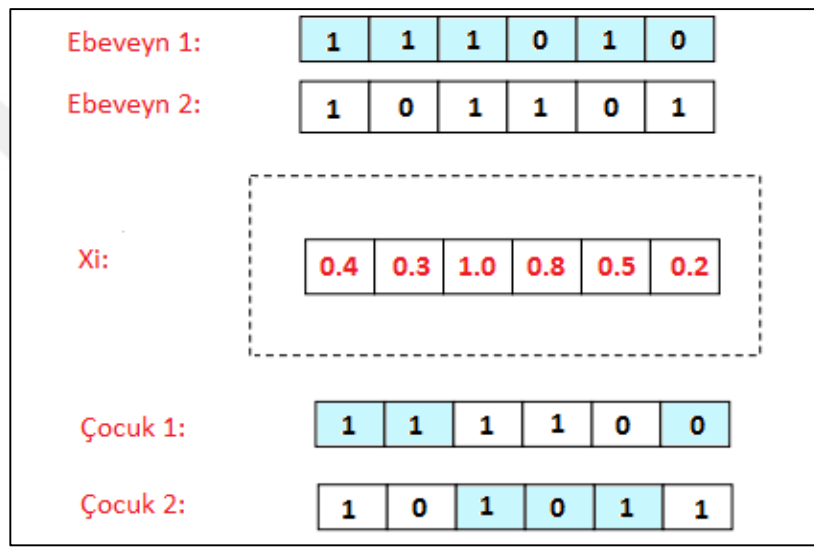
Şekil 4.7: Tek-Nokta çaprazlama (Özalp ve Alp, 2020).

- Çok noktalı çaprazlama: Çok noktalı çaprazlama tekniği, tek noktalı çaprazlama tekniğine benzerdir ve daha gelişmiş şeklidir (Onwubolu, 2002, s.24). Birden fazla çaprazlama noktası rassal olarak seçilir. Kromozomlar değişim noktalarından çaprazlanır ve yeni nesiller oluşturulur. Kromozom uzunluğuna göre çaprazlama noktası sayısı belirlenir. Şekil 4.8’de üç noktalı çaprazlama örneği gösterilmektedir (Vural, 2005, ss. 18-31).



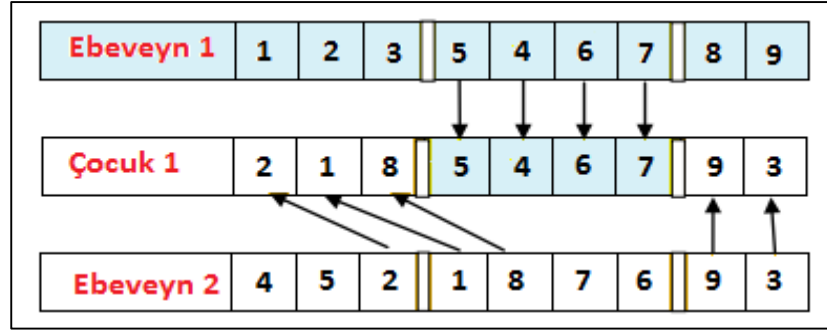
Şekil 4.8 : Üç Noktalı çaprazlama (Vural, 2005, ss.18-31).

- Tekdüze (Uniform) çaprazlama: Bu teknik GA'nın iyileştirme gücünü artırmak için geliştirilmiştir. Bu yöntemde bir çaprazlama noktası yerine ailedeki genlerin sırasıyla yavrulara kopyalanması söz konusudur. Kopyalama işlemi sırasında her bir gen $X_i \in [0,1] (i=1,...,n)$ olasılığıyla ebeveynlerden birinden gelir. Örnek olarak; $X_i < 0.5$ olduğunda, i. gen birinci aileden birinci çocuğa ikinci aileden ikinci çocuğa, $X_i \geq 0.5$ olduğunda ise gen birinci aileden ikinci çocuğa, ikinci aileden birinci çocuğa aktarılır (Sakawa, 2012, ss. 11-25; Nissen ve Biethahn, 1995). Şekil 4.9' da uniform çaprazlama örneği gösterilmektedir.



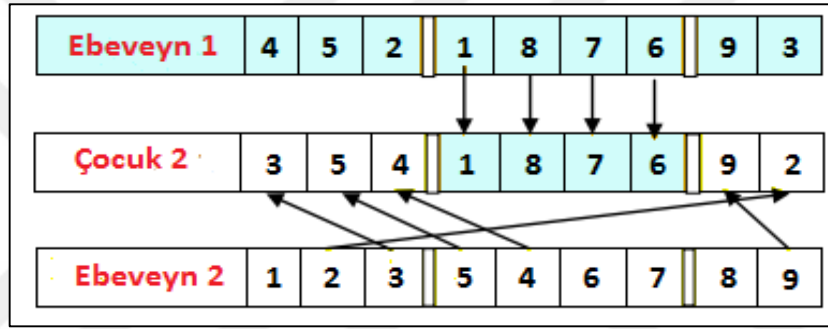
Şekil 4.9: Uniform çaprazlama.

- Sıralı çaprazlama: Genler, kromozomlarda bir sıralamaya göre dizilir. Sıralı çaprazlama tekniğinde, birinci ebeveyninden bir kısım gen çocuk kromozoma kopyalanır, kalanlar ise ikinci ebeveynde göründükleri sıraya göre çocuklara aktarılır. Bu yöntemde, rassal olarak seçilen iki kromozom arasında rastgele iki kesme noktası seçilir. Ebeveyn 1 üzerinde bulunan iki kesme noktasından (5-4-6-7) numaralı genler, Şekil 4.10'da gösterildiği gibi çocuk düğümlere kopyalanır (Varun ve Panneerselvam, 2017). Ardından, genlerin geri kalanı, ikinci kesim noktasından başlayarak tüm pozisyonlar dolana kadar Ebeveyn 2'de göründükleri sırayla kopyalanır. Kromozomlarda gen tekrarı oluşması durumunda, bu genler yerine sıra ile soldan sağa doğru kromozomda tekrarlanmayan genler yazılır (Cheng ve diğ, 1999).



Şekil 4.10 : İlk çocuklar için sıralı çaprazlama (Varun ve Panneerselvam, 2017).

Çocuk 2'yi oluşturmak için, Şekil 4.11'de gösterildiği gibi Ebeveyn 1 ve Ebeveyn 2'nin yerleri değiştirilerek aynı prosedür izlenir (Varun ve Panneerselvam, 2017).



Şekil 4.11 : İkinci çocuklar için sıralı çaprazlama (Varun ve Panneerselvam, 2017).

- Karıştırmalı Çaprazlama: Çaprazlama için rastgele iki kromozom seçilir ve her iki kromozoma ait genler rastgele bir şekilde karıştırılır. Bir nokta çaprazlama konumu olarak belirlenir ve tek-noktalı çaprazlama yöntemi uygulanarak, iki çocuk kromozom oluşturmak için her iki ebeveyn bütünleştirilir (Özalp ve Alp, 2020).
- Şekil 4.12'de görüldüğü gibi, ilk aşamada ebeveynlerin genleri ikisinde de eşit olacak şekilde karıştırılır. Şekil 4.13'te görüldüğü gibi, ikinci adımda tek nokta çaprazlama gerçekleştirilir.

Ebeveyn 1	1	2	3	4	5	6	7	8	9
Ebeveyn 2	4	1	2	8	7	6	9	3	5
Ebeveyn 1	3	1	5	2	4	9	8	6	7
Ebeveyn 2	2	1	3	5	7	4	9	8	6

Şekil 4.12: Karıştırmalı çaprazlama 1. adım.

Ebeveyn 1	3	1	5	2	4	9	8	6	7
Ebeveyn 2	2	1	3	5	7	4	9	8	6
Çocuk 1	3	1	5	2	7	4	9	8	6
Çocuk 2	2	1	3	5	4	9	8	6	7

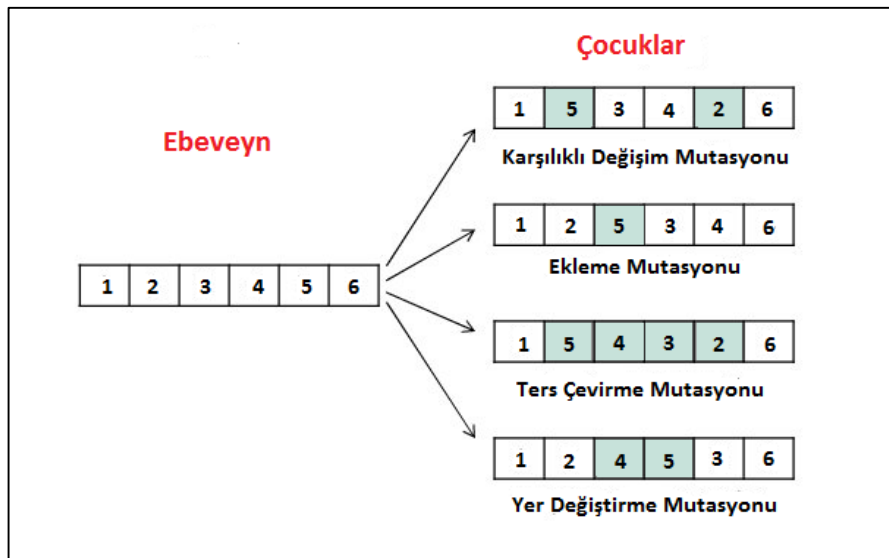
Şekil 4.13: Karıştırmalı çaprazlama 2. adım.

Adım 6. Mutasyon operatörü

Mutasyon operatörü, çaprazlamanın ardından toplulukta çeşitliliği arttırmak ve korumak amacıyla gerçekleştirilen bir diğer adımdır (Çolak, 2010). Mutasyon adımı, çözümün rastlantısal olarak bozulması ve farklı bir noktaya gitmesi ile komşularının da araştırılması gerekir ve bu sayede çözüm yerel optimumdan, global optimuma taşınır (Ho ve diğ, 2008). Bu aşamada, gen potansiyellerinin değişikliğe uğratılması ile mevcut bireylerden yeni bireyler türetilir (Emel ve Taşkın 2002). Mutasyon, diğer adımlarda gözden kaçırılmış veya ulaşılammış olası en iyi sonuçlara ulaşmak için rastgele bir seçenek sunar ve böylece önceki aşamalarda kaybedilen faydalı genler kurtarılabilir (Penev, 2005; Smith ve diğ, 1996). Mutasyon olasılığına bağlı olarak çocuk kromozoma ait genlerin değerleri rastlantısal olarak değişir, başka bir ifadeyle bozulmaya uğrar (Chen ve diğ, 1995).

Sürekli olarak yeni nesil üretimi neticesinde belirli bir iterasyonun ardından birbirini tekrarlayan kromozomlar oluşabilir ve farklı kromozom üretimi durabilir. Bu durağanlığın önüne geçebilmek amacıyla ve nesildeki birey çeşitliliğini makul düzeyde arttırmak için bireylerden bazıları mutasyona uğratılır (Dursun, 2009, ss. 29-49; Burke ve Kendall, 2005). Mutasyon tüm kromozomlarda meydana gelmez. Üretilen kromozomlar, belirli bir olasılıkla mutasyona uğrar. Bu olasılığa mutasyon olasılığı veya mutasyon oranı denir. Mutasyon olasılığı, problemin başında belirlenir ve aynı doğal yaşamda olduğu gibi çok düşük bir olasılıkla gerçekleşir (Nissen ve Biethahn, 1995). Kromozomları mutasyona uğratabilmek için farklı teknikler kullanılabilir. Mutasyon operatörü olarak, problemin yapısına göre en uygun olan aşağıdakilerden biri seçilir (Şekil 4.14) (Liu, 2014, ss. 29-31).

- Karşılıklı değişim mutasyonu: Kromozom üzerinde rastgele iki pozisyon seçilir ve bu pozisyonlarda bulunan iki gen karşılıklı olarak değiştirilir.
- Ekleme mutasyonu: Eklemede, kromozomdan bir gen rastgele olarak çıkarılır ve yine rassal olarak seçilen bir pozisyona yerleştirilir
- Ters çevirme mutasyonu: Bu teknikte kromozom üzerinde rastgele iki konum seçilir ve iki konum arasındaki genler ters çevrilir.
- Yer değiştirme mutasyonu: Kromozomda rastgele bir alt dizi seçilir, seçilen alt dizide bulunan genler kromozomdan çıkarılır ve sonra bunları kromozomdaki rastgele bir yere yerleştirilir.



Şekil 4.14 : Mutasyon yöntemleri (Liu, 2014, ss. 29-31).

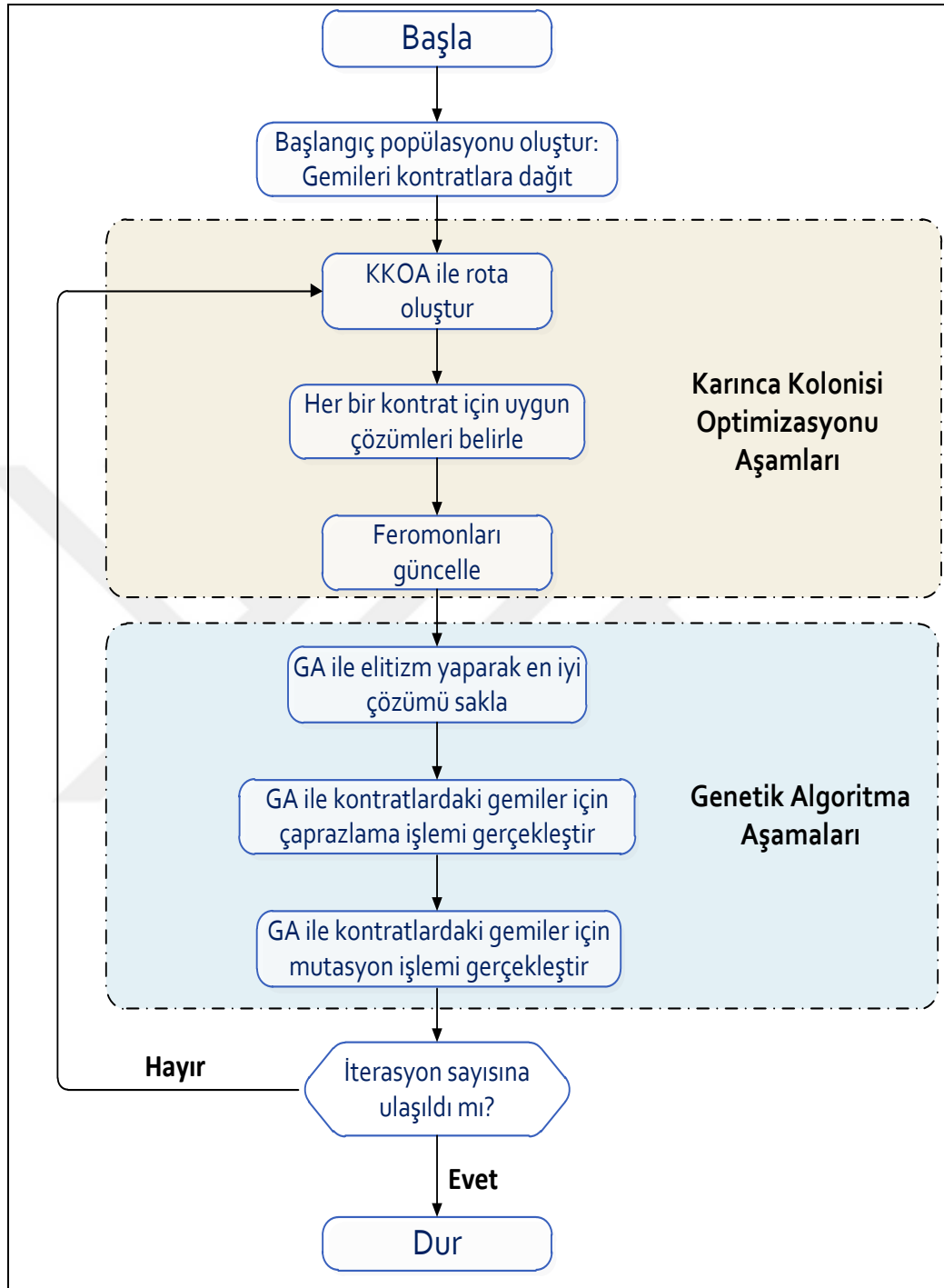
Adım 7. Elitizm

En iyi uygunluk fonksiyonuna sahip kromozomun, çaprazlama, mutasyon gibi operatörler neticesinde sonraki jenerasyona aktarımın gerçekleşmeyebilir. Bu durumun önüne geçebilmek için, toplumda uygunluk fonksiyonu en iyi olan (elit) birey, yeni kuşaktaki herhangi bir kromozomla değiştirilir ve hiçbir işlem uygulanmadan bir sonraki kuşağa aktarılır. Böylece bir sonraki kuşakta hesaplanacak en iyi birey, bir önceki kuşakta bulunan en iyi bireyden kötü olamaz (Haldurai ve diğ, 2016; Dursun, 2009, ss. 29-49).

Adım 8. Bitirme Koşulu

Algoritmada, en iyi kromozom değeri belirlenmiş olan bitirme koşulunu sağlıyorsa algoritma sona erdirilir. GA'da bitirme koşulu, problemin yapısına göre belirlenir. İterasyon sayısı, değişiklik olmayan jenerasyon sayısı gibi parametreler verilerek, algoritmanın nerede duracağı kontrol edilir (Bolat ve diğ, 2004).

4.5 Hibrit Yöntem



Şekil 4.15: Hibrit yöntem işlem adımları.

Şekil 4.15'te, kullanılan yönteme ait işlem adımları genel çerçeve ile verilmiştir. Aşağıda tüm adımlar detaylı olarak açıklanacaktır.

Adım 1: Başlangıç popülasyonu oluştur

Çalışmada kuru dökme yük taşımacılığı yapan ve tramp olarak çalışan gemi filosuna ait rota ve sefer çizelgelemesi yapılmıştır. Planlama aşamasında, taşıma sözleşmelerinin, gemiler arasında en az maliyete denk gelecek ve sözleşme zamanları aşılmayacak şekilde paylaştırılması amaçlanmıştır. Aşağıdaki yöntem adımları, 15 kontrat ve 5 gemiden meydana gelen örnek çözüm uzayı için oluşturulmuştur. Şekil 4.16'da, tek bir (k1) kromozomu üzerinden gösterildiği üzere, başlangıç aşamasında 15 kontratı temsil eden (c1,c2,.....,c15) ve 15 genden meydana gelen kromozomlar oluşturulur.

	c1	c2	c3	c4	c5	c6	c7	c8	c9	c10	c11	c12	c13	c14	c15
k1:															

Şekil 4.16 : Kontrat sayılarını temsil eden kromozom örneği.

Ardından, kromozom üzerinde 5 gemi, her bir kontrata hizmet götürecek şekilde ve rassal olarak, değer kodlama yöntemi ile atanır (Şekil 4.17).

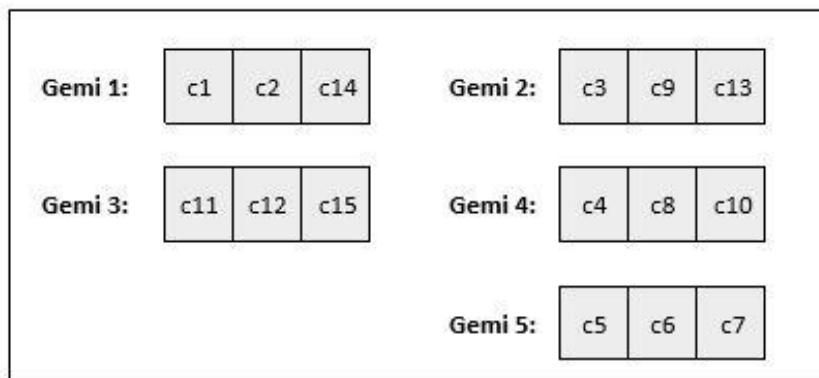
	c1	c2	c3	c4	c5	c6	c7	c8	c9	c10	c11	c12	c13	c14	c15
k1:	1	1	2	4	5	5	5	4	2	4	3	3	2	1	3

Şekil 4.17 : Başlangıç kromozomu oluşturma.

Ayrıca bu aşamada koku değeri Δt_{ij} : 1/14 olacak şekilde her rotaya eşit olarak atanır.

Adım 2: Rota oluştur

Bu adımda, KKOA'na atanan koku değerleri ile 5 gemi için 15 kontrata denk gelecek şekilde rota oluşturulur (Şekil 4.18).



Şekil 4.18 : Rota oluşturma.

Karınca sisteminde, k karıncasının i noktasından j noktasına hareket etme olasılığı P_{ij}^k değeri hesaplanırken kullanılan alfa ve beta parametreleri 1 olarak alınmıştır.

Adım 3: Her bir kontrat için uygun çözümleri belirle

KKOA, karıncaların bıraktıkları koku yoğunluğu sonucunda oluşturulan rotaların maliyet değerleri $f(x)$ bu adımda hesaplanır. Çizelge 4.1’de k1 kromozomu için belirlenmiş rotanın maliyet hesaplaması verilmektedir. Problemde 100 kromozomdan meydana gelen bir popülasyon oluşturulmuş ve her bir kromozom için maliyetler hesaplanarak uygun çözüm oluşturulur.



Çizelge 4.1: Tek kromozom için maliyet hesabı.

Gemi 1:

C1			
node	1	2	3
0	1313865	1317735	1367735
C2			
node	4	5	6
C1	1	1615870	1570360
		1623360	
C14			
node	40	41	41
C2	5	1565195	1540683
		1590683	
node			
C41	0		1819120

Route: 
Cost: 6244027,5 \$

Gemi 2:

C3			
node	7	8	9
0	1881850	1805155	1855155
C9			
node	25	26	27
C3	8	2408990	2308730
		2358730	
C13			
node	37	38	39
C9	26	1549057	1501890
		1551890	
node			
C13	0		1414159

Route: 
Cost: 7029933,75 \$

Gemi 3:

C11			
node	31	32	33
0	1419336	1413014	1463014
C12			
node	34	35	36
C11	32	1140730	1155065
		1205065	
C15			
node	43	44	45
C12	34	1301180	1289250
		1342250	
node			
C15	0		1559560

Route: 
Cost: 5402553,5\$

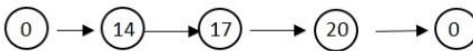
Gemi 4:

C4			
node	10	11	12
0	1803930	1749640	1799640
C8			
node	22	23	24
C4	11	2014320	1908390
		1958390	
C10			
node	28	29	30
C8	23	1654500	1612810
		1662810	
node			
C10	0		2088660

Route: 
Cost: 7359500\$

Gemi 5:

C5			
node	13	14	15
0	2173140	2087825	2137825
C6			
node	16	17	18
C5	14	1687890	1667875
		1717875	
C7			
node	19	20	21
C6	17	1499590	1488250
		1538250	
node			
C7	0		1945265

Route: 
Cost: 7189215\$

Adım 4: Feromonları güncelle

Bu aşamada en az maliyetli rotalara denklem 4.9'da verilen formüle göre koku eklemesi yapılarak, $f(x)$ fonksiyonları güncellenir. Koku uçuşma değeri $pl_{ij} = 0,9$ olarak belirlenmiştir.

Adım 5: En iyi çözümü sakla

Elitizm aşamasında, 100 kromozom arasında en düşük maliyeti veren 10 kromozoma ait rotalar saklanır ve direkt olarak bir sonraki aşamaya aktarılarak çaprazlama ve mutasyon aşamalarında gözden kaçmaları ya da bozulmaya uğramaları engellenerek problemin çözüm kalitesi artırılır.

Adım 6: Çaprazlama işlemini gerçekleştir

Çaprazlama operatörüne geçilirken, kromozom seçimi yukarıda kromozom seçimi için anlatılan tekniklerden rulet tekerleği yöntemi kullanılarak gerçekleştirilmiştir. Yöntemde, elitizm aşamasının ardından geride kalan daha az iyi olan kromozomlara ait $f(x)$ maliyetleri, $1/f(x)$ cinsinden büyükten küçüğe doğru sıralanır ve denklem (4.12) kullanılarak, kromozomların seçilim olasılığı hesaplanır. Bu şekilde en düşük maliyetli kromozomun rulet tekerleğinde kapladığı alan artırılarak, bu oranlara göre en iyi genler belirlenir ve çaprazlama operatörüne aktarılır.

Çaprazlama aşamasında, rulet tekerleği seçiminde en kötü değerleri veren iki gen alınarak, bu genlerden itibaren sıralı olarak çaprazlama işlemi gerçekleştirilir ve çocuk kromozomlar üretilir. Böylece, kötü sonuç veren ebeveynler ölererek, çözümden çıkar ve sonraki nesillere aktarılamaz. Çaprazlama yöntemi olarak tek noktalı çaprazlama kullanılmıştır (Şekil 4.19).

	c1	c2	c3	c4	c5	c6	c7	c8	c9	c10	c11	c12	c13	c14	c15
k1:	1	1	2	4	5	5	5	4	2	4	3	3	2	1	3
k9:	2	2	4	1	3	3	3	1	2	1	5	5	4	4	5
ç1:	1	1	2	4	5	5	5	4	2	1	5	5	4	4	5
ç2:	2	2	4	1	3	3	3	1	2	4	3	3	2	1	3

Şekil 4.19 : Çaprazlama işlemi.

Adım 7: Mutasyon işlemini gerçekleştir

%10 en iyi kromozomların dışında kalan bireyler, çaprazlama operatörünün ardından çeşitliliği arttırmak için mutasyon işlemine sokulur. Yer değiştirme mutasyon tekniği kullanılarak, seçilen gen değiştirilir ve en iyi çözümlerin evrimleştirilmesi amaçlanır (Şekil 4.20).

	c1	c2	c3	c4	c5	c6	c7	c8	c9	c10	c11	c12	c13	c14	c15
ç1:	1	1	2	4	5	5	5	4	2	4	3	3	2	1	3
	1	1	5	2	4	5	5	4	2	4	3	3	2	1	3

Şekil 4.20: Mutasyon işlemi.

Adım 8: İterasyon sayısına ulaşıldı mı?

Elitizm, çaprazlama ve mutasyon işlemlerin sonucunda yine 100 kromozomlu yeni bir popülasyon oluşur. Belirlenen iterasyon sayısına gelene kadar, rota oluşturma aşamasından itibaren tüm adımlar tekrarlanarak yeni popülasyonlar oluşturulmaya devam edilir.

5. UYGULAMA

5.1 Data Seti Oluřturulması

Dökme yük taşımacılığı yapan ve tramp olarak çalışan gemi filosunu rota ve sefer çizelgelemesi yapılmıştır. Planlama yapılırken, gemi ve kontrat sayıları arttırılarak, farklı problem hacimleri yaratılmıştır. Taşıma sözleşmelerinin, gemiler arasında arasında en az maliyete denk gelecek ve sözleşme zamanları aşılmayacak şekilde paylaştırılması amaçlanmıştır. Yapılacak olan sözleşme kümeleri bitiş tarihleri öncelikli olarak sıralanmış ve gemilerin, sözleşmelerin son hizmet verilme tarihlerini aşmadan tüm yükleri taşıyabilmesi sağlanmıştır. Her bir taşıma sözleşmesine hizmet götürülebilecek zaman aralığı 20 gün kabul edilmiş ve bu süre içerisinde uygun olmayan geminin yükü alması engellenmiştir.

Data seti oluşturulurken ilk önce, sözleşmenin içerisinde bulunan tüm limanlar arası mesafeler hesaplanmış ve gemilerin bir kontrattan diğerine geçişinin takibi için, başlangıç düğümleri de dahil edilerek, kontrat sayıları kadar kontrattan kontrata geçiş matrisleri oluşturulmuştur. Matris oluşturulurken, gemilerin yük almak için yaptıkları balastlı seferler ve limanlarda beklenecek süreler göz önünde bulundurulmuştur. Matrisin hizmet süreleri göz önüne alınarak belirlenmesinin ardından, kontratlardaki her bir rota bacağı için mesafe ve gemilerin hızlarına göre değişen zaman hesaplaması yapılmıştır.

Problemde, kapasite kısıtı göz ardı edilmiştir. Bu nedenle, gemilerin yaklaşık DWT'lere (110.000-199.000) sahip olduğu ve tüm kontratlara hizmet götürebilecekleri kabul edilmiştir. Maliyet ve varış zamanları açısından heterojenlik yaratmak için gemilere ait farklı hız aralıkları ve yakıt tüketimleri belirlenmiştir. Yakıt harcamaları gemilerin boş ya da dolu sefer yapmalarına göre değişkenlik göstermektedir. Gemilerin erken gitmeleri gereken limanlara seyirlerinde hızları 13 kts, diğer durumlarda ise 12 kts ekonomik hız olarak belirlenmiştir.

Sonraki aşamada ise, her bir sözleşmeye 3 zaman penceresi eklenerek, ayrıklaştırma seviyelerinin çözümün optimallığı üzerindeki etkisi hesaplanmıştır. Geçiş yapılan her bir sözleşme için, sözleşmenin bitebileceği 3 tarih ve diğer sözleşmenin başlayabileceği 3 tarih 3x3 farklı varış zamanı opsiyonu ile oluşturulmuştur. Tüm bu zamanlar belirlenirken yine bir sözleşmeden diğerine geçme imkanı olup olmaması durumu göz önüne alınmıştır. Probleme çözüm kolaylığı sağlaması amacıyla teslim sürelerinin kısıtlayıcı olmadığı varsayılmış, yükten yüke ve yükten gemiye uyumsuzluklar ihmal edilmiştir.

Zaman aralıkları ve kısıtlara göre gemilerin hizmet götüremeyecekleri sözleşmelerin belirlenmesinin ardından, her bir zaman penceresi için, gemilerin hızlarına, yakıt tüketimlerine ve limanlardaki bekleme sürelerine göre değişen maliyet tabloları oluşturulmuştur. Çizelge 5.1’de, 15 kontratlık bir problem uzayında, bir gemi için yakıt tüketimi, bekleme zamanları, liman masraflarının da dahil olduğu sefer maliyetleri gösterilmiştir. Maliyet hesabı yapılırken, limanlar arası mesafelerin zaman cinsinden değeri, gemilerin belirtilen hız ve durumdaki (boş/ dolu) günlük yakıt harcamaları kullanılmıştır. Gemilerin çalıştıkları bölgeler göz ardı edilerek, ton başına IFO380 fiyatı 540\$ olarak hesaplamaya dahil edilmiştir.

Sonuç olarak, veri oluşturma safhasında, başlangıç noktası, yükleme ve tahliye limanları, mesafeler, seyir hızları, hıza bağlı yakıt tüketimleri ve ilgili liman maliyetleri ile alakalı gerçeğe yakın veriler derlenmiştir. Deneysel faktörler olarak; sözleşme sayısı, ayrıklaştırma seviyesi, gemi sayısı ve hızın değişimi kullanılmıştır.

Çizelge 5.1: Bir gemi için maliyet tablosu.

Cargo Type	C0	C1			C2			C3			C4			C5		
Time nodes	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
C0	M	1313865	1317735	1367735	1460556	1429364	1479364	1885850	1807155	1857155	1799930	1747640	1797640	2173140	2087825	2137825
C1	1	M	M	M	1615870	1570360	1623360	1769490	1720055	1773055	1915230	1838740	1891740	2246320	2161105	2214105
	2	M	M	M	1605870	1565360	1618360	1759490	1718055	1768055	1905230	1836740	1886740	2236320	2159105	2209105
	3	M	M	M	1620870	1622870	1625360	1774490	1776490	1775055	1920230	1922230	1893740	2251320	2253320	2216105
C2	4	M	M	M	M	M	M	M	M	M	1788870	1731820	1784820	2119960	2036365	2089365
	5	M	M	M	M	M	M	M	M	M	1778870	1729820	1779820	2109960	2034365	2084365
	6	M	M	M	M	M	M	M	M	M	1793870	1795870	1786820	2124960	2126960	2091365
C3	7	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	8	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	9	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
C4	10	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	11	M	M	M	M	M	M	M	M	M	M	M	M	1508160	1486125	1539125
	12	M	M	M	M	M	M	M	M	M	M	M	M	1498160	1484125	1534125
C5	13	M	M	M	M	M	M	M	M	M	M	M	M	1513160	1515160	1541125
	14	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	15	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
C6	16	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	17	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	18	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
C7	19	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	20	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	21	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
C8	22	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	23	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	24	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
C9	25	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	26	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	27	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
C10	28	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	29	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	30	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
C11	31	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	32	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	33	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
C12	34	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	35	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	36	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
C13	37	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	38	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	39	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
C14	40	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	41	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	42	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
C15	43	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	44	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
	45	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M

Çizelge 5.1(devamı): Bir gemi için maliyet tablosu.

Cargo Type	C6				C7				C8				C9				C10			
	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30					
C0	Time nodes																			
C1	0	2447110	2360675	2410675	2006090	1945265	1694420	1639090	1689090	1674830	1635750	1685750	2163050	2086660	2136660					
	1	2414990	2327035	2380035	2121390	2036365	1620180	1551990	1604990	1812250	1760310	1813310	2086810	2017380	2070380					
	2	2404990	2325035	2375035	2111390	2034365	1610180	1549990	1599990	1802250	1758310	1808310	2078810	2015380	2065380					
C2	3	2419990	2421990	2382035	2126390	2128390	2091365	1625180	1627180	1606990	1817250	1819250	1815310	2093810	2095810	2072380				
	4	2393930	2309215	2362215	1995030	1929445	1408520	1358150	1411150	1980730	1920690	1973690	2257290	2159940	2212940					
	5	2383930	2307215	2357215	1985030	1927445	1398520	1356150	1406150	1970730	1918690	1968690	2247290	2157940	2207940					
C3	6	2398930	2400930	2364215	2000030	2002030	1984445	1413520	1413150	1985730	1987730	1975690	2262290	2264290	2214940					
	7	2014850	1970635	2023635	1826550	1786885	1939885	1915020	1819290	1872290	2422990	2312730	2365730	2425770	2320320	2373320				
	8	2004850	1968635	2018635	1818550	1784885	1905020	1817290	1867290	2412990	2310730	2360730	2415770	2318320	2368320					
C4	9	2019850	2021850	2025635	1831550	1833550	1920020	1922020	1874290	2427990	2429990	2367730	2430770	2432770	2375320					
	10	1867430	1828075	1881075	1637010	1608685	1661685	2020320	1908390	1961390	2359810	2241450	2294450	2446830	2355960	2408960				
	11	1857430	1826075	1876075	1627010	1606685	1656685	2010320	1906390	1956390	2349810	2239450	2289450	2436830	2353960	2403960				
C5	12	1872430	1874430	1883075	1642010	1644010	1663685	2025320	2027320	1963390	2364810	2366810	2296450	2451830	2453830	2410960				
	13	1697890	1669875	1722875	1721250	1679965	1732965	2104560	1991490	2050490	2444050	2330550	2320470	2231220	2284220					
	14	1687890	1667875	1717875	1711250	1677965	1727965	2094560	1995490	2045490	2434050	2328550	2310470	2229220	2279220					
C6	15	1702890	1704890	1724875	1728250	1734965	2109560	2111560	2052490	2052490	2449050	2451050	2385550	2325470	2327470	2286220				
	16	M	M	M	1509590	1490250	2042010	1953820	2006820	2039580	1951390	2004390	2165940	2076130	2129130					
	17	M	M	M	1499590	1488250	2032010	1951820	2001820	2029580	1949390	1999390	2155940	2074130	2124130					
C7	18	M	M	M	1514590	1516590	2047010	2049010	2008820	2044580	2046580	2006390	2170940	2172940	2131130					
	19	M	M	M	M	M	M	M	M	M	M	M	2123620	2022670	2075670					
	20	M	M	M	M	M	M	M	M	M	M	M	M	2113820	2020670	2070670				
C8	21	M	M	M	M	M	M	M	M	M	M	M	M	2128820	2130820	2077670				
	22	M	M	M	M	M	M	M	M	M	M	M	M	1660500	1612810	1665810				
	23	M	M	M	M	M	M	M	M	M	M	M	M	1650500	1610810	1660810				
C9	24	M	M	M	M	M	M	M	M	M	M	M	M	1665500	1667500	1667810				
	25	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
	26	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
C10	27	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
	28	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
	29	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
C11	30	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
	31	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
	32	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
C12	33	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
	34	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
	35	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
C13	36	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
	37	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
	38	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
C14	39	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
	40	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
	41	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
C15	42	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
	43	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
	44	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				
	45	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M				

Çizelge 5.1(devamı): Bir gemi için maliyet tablosu.

Cargo Type	C11					C12					C13					C14					C15				
	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45										
C0	0	1416336	1411514	1461514	1587808	1577005	1627005	1433980	1416159	1466159	1874145	1819120	1869120	1569850	1558060	1608060									
	1	1571650	1552510	1605510	1491725	1490580	1358957	1329565	1382565	1778845	1732020	1785020	1495610	1486780	1541780										
C1	2	1561650	1550510	1600510	1481725	1488580	1348957	1327565	1377565	1768845	1730020	1780020	1485610	1486780	1536780										
	3	1576650	1578650	1607510	1496725	1498725	1363957	1363957	1363957	1783845	1785845	1787020	1500610	1502610	1543780										
C2	4	1740130	1695070	1748070	1723385	1686600	1739600	1147297	1135725	1188725	1575195	1542683	1595683	1706210	1666980										
	5	1730130	1693070	1743070	1713385	1684600	1734600	1137297	1133725	1183725	1565195	1540683	1590683	1696210	1664980										
C3	6	1745130	1747130	1750070	1728385	1730385	1741600	1152297	1154297	1580195	1582195	1597683	1711210	1713210	1721980										
	7	2182390	2104930	2157930	2018225	1953900	2006900	1653797	1596865	2094745	2071740	2070140	2043170	1969920	2022920										
C4	8	2172390	2102930	2152930	2008225	1951900	2001900	1643797	1594865	2084745	2071540	2065140	2033170	1967920	2017920										
	9	2187390	2189390	2159930	2023225	2025225	2008900	1658797	1660797	2099745	2101745	2072140	2048170	2050170	2024920										
C5	10	2098150	2033650	2088650	2228825	2149920	2029920	1759097	1685965	2178985	2106240	2159240	2232710	2148120	2201120										
	11	2088150	2031650	2081650	2218825	2147920	2197920	1749097	1683965	2168985	2104240	2154240	2222710	2146120	2196120										
C6	12	2103150	2105150	2088650	2233825	2235825	2024920	1764097	1766097	2183985	2185985	2161240	2233710	2239710	2203120										
	13	2203450	2122750	2175750	2060345	2007360	2060360	1843337	1775065	2284285	2195340	2248340	2085290	2023380	2076380										
C7	14	2193450	2120750	2170750	2050345	2005360	2055360	1833337	1773065	2274285	2193340	2243340	2075290	2021380	2071380										
	15	2208450	2210450	2177750	2065345	2067345	2062360	1848337	1850337	2289285	2291285	2250340	2090290	2092290	2078380										
C8	16	1804680	1766710	1819710	1800590	1743625	1796625	1626237	1571770	1924560	1852570	1905570	1931040	1873630	1926630										
	17	1794680	1764710	1814710	1790590	1741625	1791625	1616237	1575170	1914560	1850570	1900570	1921040	1871630	1921630										
C9	18	1809680	1811680	1821710	1805590	1807590	1798625	1631237	1632327	1925560	1931560	1907570	1936040	1938040	1928630										
	19	2036340	1962730	2015730	1716350	1654525	1707525	1647297	1594990	1945620	1870590	1923390	1888920	1820170	1873170										
C10	20	2026340	1960730	2010730	1708350	1652525	1709250	1637297	1592990	1935620	1868390	1918390	1878920	1818170	1868170										
	21	2041340	2043340	2017730	1721350	1723350	1709525	1652297	1654297	1950620	1952620	1925390	1893920	1895920	1875170										
C11	22	1551960	1517230	1570230	1210910	1209025	1262025	1499877	1452430	1798200	1727630	1780830	1383480	1374670	1427670										
	23	1541960	1515230	1565230	1200910	1207025	1257025	1499877	1450430	1786200	1725830	1775830	1373480	1372670	1422670										
C12	24	1556960	1558960	1572230	1215910	1217910	1264025	1504877	1506877	1803200	1805200	1782830	1388480	1390480	1429670										
	25	1636200	1588510	1641510	1295150	1280305	1333305	1558890	1505890	1861380	1799110	1852110	1467720	1445950	1498950										
C13	26	1626200	1586510	1636510	1285150	1278305	1328305	1553057	1503890	1851380	1797110	1847110	1457720	1443950	1493950										
	27	1641200	1643200	1643510	1300150	1302150	1335305	1568057	1570057	1866380	1868380	1854110	1472720	1474720	1500950										
C14	28	1298180	1287750	1340750	1295150	1280305	1436697	1398970	1451970	1735020	1674370	1727370	1425600	1410310	1463310										
	29	1288180	1285750	1335750	1285150	1278305	1426697	1396970	1446970	1725020	1672370	1722370	1415600	1408310	1458310										
C15	30	1303180	1305180	1342750	1300150	1302150	1335305	1441697	1443697	1740020	1742020	1729370	1430600	1432600	1465310										
	31	M	M	M	1147730	1155565	1208585	1394577	1386577	1692900	1638730	1691730	1290816	1280402	1333402										
C16	32	M	M	M	1137730	1153565	1203585	1384577	1434577	1682900	1636730	1686730	1280816	1278402	1328402										
	33	M	M	M	1152730	1154730	1210565	1399577	1401577	1697900	1699900	1693730	1295816	1297816	1335402										
C17	34	M	M	M	M	M	M	1394577	1363330	1416330	1692900	1638730	1298180	1287750	1340750										
	35	M	M	M	M	M	M	1384577	1361330	1411330	1682900	1636730	1288180	1285750	1335750										
C18	36	M	M	M	M	M	M	1399577	1401577	1418330	1697900	1699900	1303180	1305180	1342750										
	37	M	M	M	M	M	M	M	M	1439120	1409250	1462250	1551960	1517230	1570230										
C19	38	M	M	M	M	M	M	M	M	1429120	1407250	1451960	1515230	1565230											
	39	M	M	M	M	M	M	M	M	1444120	1446120	1464250	1556960	1572230											
C20	40	M	M	M	M	M	M	M	M	M	M	M	1290395	1279511											
	41	M	M	M	M	M	M	M	M	M	M	M	1280395	1277511											
C21	42	M	M	M	M	M	M	M	M	M	M	M	1295395	1297395											
	43	M	M	M	M	M	M	M	M	M	M	M	M	M											
C22	44	M	M	M	M	M	M	M	M	M	M	M	M	M											
	45	M	M	M	M	M	M	M	M	M	M	M	M	M											

5.2 Uygulama Çıktıları

5.2.1 Doğrusal Programlama

Gerçek gemi rotalama sisteminin, gemi rotalama ve çizelgeleme problemine aktarılmasında, problemin yapısına uygun olarak karma tam sayılı doğrusal programlama modeli kullanılmıştır. Karma tamsayılı doğrusal programlamada, bazı değişkenler tamsayılı değerler alır ve problem kısıtları doğrusaldır (Kiremitci ve Özçakar, 2011). Modelleme dilleri, problemi ve probleme ait çözüm modelinin matematiksel ifadesini bilgisayar ortamına alarak, bir çözücü için uygun duruma getirir (Kallrath, 2004).

Çalışmada, problemin doğrusal çözümünde gurobi yazılımı kullanılmıştır. Gurobi, doğrusal programlama ve karma tamsayılı programlama problemlerinin çözümü için kullanılan, kullanıcı dostu bir ara yüze sahip olan, modern çözücülerden biridir. Doğrusal programlama çözümünde, primal ve dual simpleks yöntemlerini kullanır. Karma tamsayılı doğrusal programlama modelleri için kesme düzlemi ve çözüm sezgisellerini barındıran yöntemleri birlikte uygulayabilir.

Doğrusal ve meta-sezgisel hibrit çözüm yöntemlerinin, zaman pencereli gemi rotalama probleminde verdiği sonuçları değerlendirmek amacıyla, 10 farklı problem hacmi oluşturulmuş ve problemler her iki yöntemle de çözülmüştür. Doğrusal programlamada, data setlerinin matris üzerinde hesaplanmasının ardından, maliyetlerin Python programlama dilinde kullanılmasını amacıyla, problem hacimlerine ait hesaplamalar “python tuple list” haline getirilmiş ve yazılımın içinde kullanılmıştır. Açıklayıcı olması amacıyla, tüm örneklerde 5 gemi, hizmet götürülmesi gereken 15 kontrat ve 3 zaman pencereli problem hacmine ait veriler kullanılmıştır. Ek A1’ de tuple list örneği verilmiştir. Modelin gurobi çözücüsü üzerinde gösterimi Şekil 5.1’de verilmiştir.

```

1  import gurobipy as gp
2  from gurobipy import *
3
4  def mycallback(model, where):
5      if where == GRB.Callback.MIPSOL:
6          # MIP solution callback
7          nodecnt = model.cbGet(GRB.Callback.MIPSOL_NODCNT)
8          obj = model.cbGet(GRB.Callback.MIPSOL_OBJ)
9          solcnt = model.cbGet(GRB.Callback.MIPSOL_SOLCNT)
10         y = model.cbGetSolution(model._vars)
11         print('**** New solution at node %d, obj %g, sol %d,' % (nodecnt, obj, solcnt))
12
13     cargoes = 15
14     M = 9999999999999999999
15     D1 = [1 for i in range(1, 4)] # cargo 1
16     D2 = [1 for i in range(4, 7)] # cargo 2
17     D3 = [1 for i in range(7, 10)] # cargo 3
18     D4 = [1 for i in range(10, 13)] # cargo 4
19     D5 = [1 for i in range(13, 16)] # cargo 5
20     D6 = [1 for i in range(16, 19)]
21     D7 = [1 for i in range(19, 22)]
22     D8 = [1 for i in range(22, 25)]
23     D9 = [1 for i in range(25, 28)]
24     D10 = [1 for i in range(28, 31)]
25     D11 = [1 for i in range(31, 34)]
26     D12 = [1 for i in range(34, 37)]
27     D13 = [1 for i in range(37, 40)]
28     D14 = [1 for i in range(40, 43)]
29     D15 = [1 for i in range(43, 46)]
30
31     K = ['ship1', "ship2", "ship3", "ship4", "ship5"] # set of ships
32
33     D = [D1, D2, D3, D4, D5, D6, D7, D8, D9, D10, D11, D12, D13, D14, D15]
34
35     V = [0] + D1 + D2 + D3 + D4 + D5 + D6 + D7 + D8 + D9 + D10 + D11 + D12 + D13 + D14 + D15 # {0,...50}
36
37     A = [(i, j, k) for i in V for j in V for k in K] # arcs for each ship
38
39     c = {...}
40
41     # costs for each arc
42
43     f = Model('VRP Network Flow')
44
45     x = f.addVars(A, vtype=GRB.BINARY, name='x')
46
47     f.modelSense = GRB.MINIMIZE
48
49     f.addConstrs(sum(x[i, j, k] for j in V if j != 0) == 1 for k in K)
50
51     f.addConstrs(sum(x[i, j, k] for i, j, k in A if j in D[n] if i not in D[n]) == 1 for n in range(cargoes))
52
53     f.addConstrs((sum(x[i, j, k] for i in V if i != j if i not in A)) == (sum(x[j, i, k] for i in V if j != i if i not in A)) for j in V for k in K)
54
55     f.setObjective(sum(x[i, j, k] * c[i, j, k] for i, j, k in A))
56
57     gp.setParam('Heuristics', 0)
58
59     f._vars = f.getVars()
60
61     f.optimize(mycallback)
62
63     f.write("Disct_Time_Network.lp")
64
65     print('')
66     print('Optimization complete')
67     if f.SolCount == 0:
68         print('No solution found, optimization status = %d' % f.Status)
69     else:
70         print('Solution found, objective = ')
71         print(x.prod(c).getValue())
72         for v in f.getVars():
73             if v.X != 0.0:
74                 print('%s %g' % (v.VarName, v.X))

```

Şekil 5.1: Bir örnek için doğrusal programlama ekranı.

15 gemi, 5 kontrat için elde edilen hesaplama sonuçları EK A2’de verilmiştir. Rotaların ifade edilmesinde kullanılan değerler, kontratların zaman pencerelerini vermektedir. Örneğin ilk kontrat c1: 1,2,3 zaman pencerelerinden meydana gelmektedir. Bu sonuçlara göre, 5 geminin 15 kontrata belirlenen zaman aralıklarında optimum maliyetle yapacağı seyir,

Gemi1 için; 0-5-23-29-32-0 rotası, Gemi 2 için; 0-11-14-17-20-38-41-44-0 rotası, Gemi 3 için; 0-26-35-0, Gemi 4 için; 0-8-0 Gemi 5 için; 0-1-0 rotasıdır. Ve hesaplanan toplam maliyet; 30.042.889,2 \$’dır. Bu problem hacmi için hesaplanan ortalama çözüm süresi 0,065saniyedir.

5.2.2 Hibrit Çözüm

Genetik Algortima ve Karınca Kolonisi Optimizasyonu Algoritmasının beraber kullanıldığı hibrit yöntemin çözüm kalitesini karşılaştırmak amacıyla, oluşturulan problem hacimleri hibrit yöntemle de çözülmüştür. Model, hibrit yöntemde, ” Visual Studio” ara yüzü kullanılarak, C# programlama dili ile kodlanmıştır. Şekil 5.2’de, 15 gemi ve 5 kontrat için hazırlanmış problem hacmine ait çözüm ekranı paylaşılmıştır. Ekranın sol tarafında modele girilen veriler, sağ tarafında ise sonuçlar verilmektedir. İterasyon adımları sonuna kadar devam etmektedir. Programa ait hibrit kodları Ek-B1’de verilmektedir.

Hesaplama sonucunda görülmüştür ki, 15x5x3 lük problem hacmi için hibrit yöntemde hesaplanan maliyet değeri 30.042.889,2 \$ ve ortalama çözüm süresi 0,275 saniyedir.

Evaporation: 0,9

Alpha: 1

Beta: 1

Population: 100

Selection: Roulette Wheel Sel

Crossover: Single point

CrossoverProb: 0,4

Mutation: Displacement

MutationProb: 0,6

Vessel: 5

Iteration: 100

Customer: 15

Start

-----1. Iteration Results -----
 1. Vessel Route: 0, 3, 2, 5, 6, 7, 9.
 Total Distance: 9655
 2. Vessel Route: 1, 4, 8.
 Total Distance: 9655
 -----2. Iteration Results -----
 1. Vessel Route: 0, 3, 2, 5, 6, 7, 9.
 Total Distance: 9655
 2. Vessel Route: 1, 4, 8.
 Total Distance: 9655
 -----3. Iteration Results -----
 1. Vessel Route: 0, 3, 2, 5, 6, 7, 9.
 Total Distance: 9655
 2. Vessel Route: 1, 4, 8.
 Total Distance: 9655
 -----4. Iteration Results -----
 1. Vessel Route: 0, 3, 2, 5, 6, 7, 9.
 Total Distance: 9655
 2. Vessel Route: 1, 4, 8.
 Total Distance: 9655
 -----5. Iteration Results -----
 1. Vessel Route: 0, 3, 2, 5, 6, 7, 9.
 Total Distance: 9655
 2. Vessel Route: 1, 4, 8.
 Total Distance: 9655
 -----6. Iteration Results -----
 1. Vessel Route: 0, 3, 2, 5, 6, 7, 9.
 Total Distance: 9655
 2. Vessel Route: 1, 4, 8.
 Total Distance: 9655
 -----7. Iteration Results -----
 1. Vessel Route: 0, 3, 2, 5, 6, 7, 9.
 Total Distance: 9655
 2. Vessel Route: 1, 4, 8.
 Total Distance: 9655
 -----8. Iteration Results -----
 1. Vessel Route: 0, 3, 2, 5, 6, 7, 9.
 Total Distance: 9655
 2. Vessel Route: 1, 4, 8.
 Total Distance: 9655

Şekil 5.2 : Bir örnek için hibrit çözüm ekranı.

5.2.3 Bulgular

Problem hacimleri örnek 1’den, örnek 10’a doğru artan sırada gösterilmiştir. Şekil 5.3’te yöntemlerin 10 problem hacmi için ürettikleri en iyi sonuçlar, ortalama çözüm süreleri ve karşılaştırmaları verilmiştir. Modelin her iki yöntem ile çözümü sonucu elde edilen değerler aşağıdaki gibidir:

Maliyet karşılaştırılması yapıldığında, her iki yöntemin de hemen hemen aynı sonuçları verdiği görülmüştür. Gurobi çözümünün, örnek problem 4’te 0,027 ve örnek problem 6’da 0,031 oranında daha az maliyetli çözümler sağladığı görülmüştür.

Çözüm yöntemlerinin en iyi sonuçları vermeleri kadar, uygun çözümleri makul çözüm süreleri içerisinde vermeleri de önemlidir. Süre açısından bakıldığında, ilk 7 problem örneği için gurobi çözümlerinin daha kısa zamanda çözüm sağladığı görülmüştür. Sırasıyla, örnek 8, örnek 9 ve örnek 10 için hibrit çözümün verdiği zaman farkları; 0,33,2,0 ve 1,5’tir.

Çözülen problemin NP-hard problem tipi olduğu göz önünde bulundurulduğunda, modelin gurobi ile çözümü küçük problem hacimlerinde en iyi çözümü vermektedir. Ancak, problem hacmi büyüdükçe gurobi doğrusal programlama ile çözüm üretmekte

zorlanmaya başlamaktadır. Büyük hacimli problemlerin çözümünde etkin değerler oluşturmamaktadır. Kullanılan meta-sezgisel hibrit yöntem ise, küçük problem hacimlerinde daha uzun süreli çözüm zamanları verse de problem hacmi büyüdükçe daha kısa sürede ve doğru çözümler vermeye başlamaktadır. Yöntemi kullanışlı kılan, gemi rotalama problemi gibi büyük hacimli problemlerin çözümünde genetik algoritma ve karınca kolonisi algoritması meta-sezgisel yöntemlerini bir arada kullanarak, çözümün doğruluğunu arttırması ve klasik yöntemlerin çözüm üretemediği kadar büyük hacimlerdeki problemlerde dahi makul çözüm süreleri ile optimuma en yakın çözüm değerlerini vermesidir.

	Gurobi		Hibrit Çözüm		Karşılaştırma	
	En İyi Çözüm	Ortalama Süre	En İyi Çözüm	Ortalama Süre	En İyi Çözüm	Ortalama Süre
Örnek 1	25.110.952	0.04	25.110.952	0.2	0.000	0.8
Örnek 2	25.875.750.55	0.05	25.875.750.55	0.15	0.000	0.667
Örnek 3	27.850.230.10	0.06	27.850.230.10	0.25	0.000	0.76
Örnek 4	29.150.922.15	0.06	29.970.955.18	0.25	0.027	0.76
Örnek 5	30.042.889.20	0.065	30.042.889.20	0.275	0.000	0.764
Örnek 6	32.155.125.05	0.07	33.180.140.05	0.3	0.031	0.767
Örnek 7	35.250.145.10	0.09	35.250.145.10	0.325	0.000	0.723
Örnek 8	37.125.100.50	0.2	37.125.100.50	0.15	0.000	-0.333
Örnek 9	40.412.750.25	0.3	40.412.750.25	0.1	0.000	-2.0
Örnek 10	45.128.235.35	0.5	45.128.235.35	0.2	0.000	-1.5

Şekil 5.3 : Gurobi ve hibrit yöntemleri için karşılaştırma tablosu.

6. SONUÇ VE ÖNERİLER

Ticaretin küreselleşmesi, teknolojinin gelişmesi ve şirketlerin ortak pazara girme çabaları lojistik yönetiminin önemini artırmıştır. Lojistik yönetimi, ürünlerin üretildikleri noktalardan tüketim noktalarına aktarılmasını sağlamaktadır. Herhangi bir noktada üretilmiş bulunan bir ürüne değer kazandıran fonksiyon, onu tüketileceği yere ulaştıran taşıma işlemidir. Taşımacılık fonksiyonu, lojistik yönetiminin en maliyetli aşamasıdır, bu aşamada azaltılan her maliyet kalemi şirketlere kar olarak geri dönmektedir. Uluslararası ticaret söz konusu olduğunda ilk olarak akla deniz yolu taşımacılığı gelmektedir. Büyük hacimli yüklerin, uygun maliyetle ulaşımını sağlaması, sanayileşmenin de artmasıyla özellikle hammadde ve yarı mamul taşınması hususunda deniz yoluna olan talebi her geçen gün arttırmaktadır.

Deniz yolu taşımacılığı en çok tercih edilen taşımacılık tipi olmasına rağmen, içerisinde birçok parametre bulundurmaktadır; navlun piyasasındaki değişimler, sektörün kendine özgü finansman yapısı, gemilerin uluslararası hizmet vermesi gibi bu parametreler denizcilik sektörünü hassas ve rekabetçi bir konuma getirmektedir. Rekabetin çok büyük, stabiliteğin ise bir o kadar az olduğu bu sektörde firmalar filolarını en iyi şekilde yöneterek karlarını arttırmayı hedeflemektedir. Maliyetlerin azaltılması ve belirsizliklerle başa çıkılabilmesi aşamasında, denizcilik sektöründe optimizasyon konusu giderek önem kazanmaktadır.

Deniz taşımacılığı, yüklerin taşınma şekli bakımından liner ve tramp taşımacılık olarak ikiye ayrılmaktadır. Liner taşımacılık adından da anlaşılacağı gibi belirli bir hat üzerinde yapılan ve gemilerin gidecekleri rotaların, hizmet edecekleri kontratların, limanlara varış ve o limanlardan ayrılma zamanlarının önceden belirli olduğu taşımacılık şeklidir. Konteyner ve Ro-Ro gemilerince tercih edilen bu taşıma şeklinde, gemiler tam yüklü olsa da olmasa da söz konusu seferi yapmakla yükümlüdürler. Tramp taşıma şeklinde ise gemilerin takip ettikleri belirli bir rota sıralaması ve çizelgeleme bulunmamaktadır. Bu gemiler, imzalanan sözleşmelere göre belirlenen yükleri taşır. Dökme yük taşıyan kuru yük gemileri ve tankerler bu taşıma sınıfında

bulunurlar. Bu alanda yükler, sözleşmelerle belirlendiğinden firmalar arasında söz konusu yüklerin alınması için büyük bir rekabet bulunmaktadır. Tramp taşımacılık yapan şirketlerdeki gemiler, taşıma kapasiteleri, yakıt tüketimleri, hızları gibi faktörler bakımından birbirinden farklılık göstermektedir. Tramp taşımacılık şeklinde bulunan bu değişkenlikler, rekabet ve belirsizlikler bu çalışma şeklinin optimizasyonunu dikkat çekici hale getirmektedir.

Denizcilik sektöründe şirketlerin karşı karşıya bulundukları birçok maliyet alanı bulunmaktadır. Özellikle bu çalışmaya konu olan ve doğrudan operasyonel kararlarla ilgili olan maliyetler; gemilerin günlük çalışmaları ile ilgili olan, yapılan seferin özelliklerine (mesafe, seyir hızı, rota vb.), yakıt tüketimine, liman ücretlerine ve kanal geçiş ücretlerine göre değişkenlik gösteren maliyetlerdir.

Çalışmanın çıkış amacı, tramp şeklinde çalışan gemilerin rotalama ve çizelgelemelerinin optimize edilmesi ve bu optimizasyon ile söz konusu operasyonel maliyetlerin azaltılmasıdır. Bu amaçla, formülasyonda karma tam sayılı doğrusal programlama modeli kullanılmıştır.

Tezde, kuru dökme yük taşıyıcılığı yapan, yakıt tüketimleri, hızları, gibi parametreler bakımından birbirlerinden farklı olan ancak kapasite kısıtını probleme dahil etmemek için taşıma kapasiteleri birbirine çok yakın cape-size gemi hacmi oluşturulmuştur. Farklı çözümler elde edebilmek için, gemi sayıları ve kontrat sayıları değiştirilerek 10 farklı problem boyutu oluşturulmuştur. Ana amaç, tüm gemilerin minimum maliyetle söz konusu tüm kontratlara hizmet götürülmesini sağlamaktır. Ayrıca, deney faktörü olarak zaman pencereleri de probleme dahil edilmiştir. Yüklerin teslim edilebileceği ve teslim alınacağı zamanlar, 3 ayrı zaman penceresine ayrılmıştır. Bu zaman pencereleri söz konusu sözleşmelere hizmet edilecek zaman aralığını temsil etmektedir. Her bir sözleşmenin birbiri ile geçişinde ortaya çıkacak maliyetler, bekleme zamanları, liman masrafları ve yakıt masrafları dikkate alınarak hesaplandıktan sonra, maliyetler data seti oluşturma aşamasında eklenmiştir.

Farklı boyutlarda oluşturulan problemlerin çözümünde ilk olarak Gurobi sürüm 9.1.2 çözücü kullanılmıştır. Daha sonra aynı problemler, KKO ve GA meta-sezgisel yaklaşımlarının bir arada kullanıldığı hibrit meta-sezgisel yaklaşım ile çözülmüştür. Karşılaştırma sonuçları göstermiştir ki, geliştirilen hibrit yöntem %80 oranında en iyi sonuçları Gurobi çözümü ile aynı değerde vermektedir. Ortalama süre olarak

karşılaştırma yapıldığında ise, hibrit meta-sezgisel yöntem küçük problem hacimlerinde Gurobi çözümüne göre ortalama daha uzun sürelerde çözüm vermektedir. Ancak, problem hacimleri büyüdükçe aradaki fark kapanarak, meta-sezgisel yöntemin daha iyi ortalama çözüm süreleri verdiği tespit edilmiştir. Bu karşılaştırma ile geliştirilen yöntemin, büyük problem hacimlerinde optimuma en yakın değerleri makul çözüm sürelerinde ürettiği görülmüştür.

Deniz taşımacılığının komplike ve belirsiz yapısı göz önünde bulundurulduğunda, çözümünde meta-sezgisel yöntemlerin giderek önem kazanacağı ve denizcilik firmaları tarafından rekabet aracı olarak kullanılabileceği göz önünde bulundurulmalıdır.

İleride yapılacak çalışmalarda, rotalama aşamasında kullanılan KKOA yerine, kuş sürülerinin hareketlerinden yola çıkarak oluşturulan Parçacık Sürü Optimizasyonu algoritması (PSOA) ya da arıların besin arama davranışlarını kullanan Yapay Arı Kolonisi Algoritması (YAKA) Genetik Algoritma ile beraber kullanılabilir. GA'da kullanılan çaprazlama ve mutasyon operatörleri yerine çeşitliliği arttıracak farklı operatörler çözüme dahil edilerek, çözümün iyileştirilmesinde kullanılabilir.



KAYNAKLAR

- Erkin, U. C.** (1995). Altı Prelüd [V. Erman]. *Ulvi Cemal Erkin: Complete works for piano solo* [CD]. Avusturya: Hungaroton Classic. (1994)
- Acar, Z.A. ve Çakmak E.** (2018). Depolama ve Depo Yönetimi, Nobel Yayıncılık, Ankara
- Ahmed, Z. E., Saeed, R. A., Mukherjee, A., & Ghorpade, S. N.** (2020). Energy optimization in low-power wide area networks by using heuristic techniques. *LPWAN Technologies for IoT and M2M Applications*, 199-223
- Aksakal, B.** (2014). Bir firmanın zaman pencereli belirli talepli araç rotalama Probleminin genetik algoritma kullanılarak çözülmesi. Yayımlanmamış Yüksek Lisans Tezi. İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul
- Alakaş, H. M., Kızıldaş, Ş., Tamer, E. R. E. N., & ÖZCAN, E.** (2018). Sıfır atık projesi kapsamında atıkların toplanması: kırkkale ilinde homojen çok araçlı araç rotalama uygulaması. *Harran Üniversitesi Mühendislik Dergisi*, 3(3), 190-196
- Almoustafa, S., Goldengorin, B., Tso, M., & Mladenovic, N.** (2009). Two new exact methods for asymmetric distance-constrained vehicle routing problem. *Proceedings of SYM-OP-IS*, 297-300.
- Altabeeb, A. M., Mohsen, A. M., & Ghallab, A.** (2019). An improved hybrid firefly algorithm for capacitated vehicle routing problem. *Applied Soft Computing*, 84, 105728
- Altay, A.** (2007). Genetik algoritma ve bir uygulama (Doctoral dissertation, Fen Bilimleri Enstitüsü) (ss. 10-24)
- Alvarenga, G. B., & Mateus, G. R.** (2004, December). Hierarchical tournament selection genetic algorithm for the vehicle routing problem with time windows. In *Fourth International Conference on Hybrid Intelligent Systems (HIS'04)* (pp. 410-415). IEEE.
- Alvarenga, G. B., Mateus, G. R., & De Tomi, G.** (2007). A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows. *Computers & Operations Research*, 34(6), 1561-1584.
- Applegate, D., Bixby, R., Chvatal, V., Cook, W.,** (1998). "On the Solution of Traveling Salesman Problems", *Documenta Mathematica*, 645.

- Araque, J.R., Kudva, G., Morin, T.L., Pekny, J.F.,** (1994). A branch-and-cut algorithm for vehicle routing problems. *Annals of Operations Research* 50, 37-59.
- Akay, B.,** (2009). Nümerik Optimizasyon Problemlerinde Yapay Arı Kolonisi(Artificial Bee Colony) Algoritmasının Performans Analizi. Erciyes Üniversitesi Fen Bilimleri Enstitüsü, Doktora Tezi, 301s, Kayseri
- Ahmmed, A., Rana, M., A. Haque, A. A. M. M., & Al Mamun, M** (2008) A Multiple Ant Colony System for Dynamic Vehicle Routing Problem with Time Window.
- Baker, B. M., & Ayeche, M. A.** (2003). A genetic algorithm for the vehicle routing problem. *Computers & Operations Research*, 30(5), 787-800.
- Başkaya, Z., & Öztürk, B. A.** (2005). Tamsayılı programlamada dal kesme yöntemi ve bir ekmek fabrikasında oluşturulan araç rotalama problemine uygulanması. *Uludağ Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, 24(1), 101-114.
- Beasley, D., Bull, D. R., & Martin, R. R.** (1993). An overview of genetic algorithms: Part 1, fundamentals. *University computing*, 15(2), 56-69.
- Bellmore, M., Nemhauser G.L.,** (1968). The Travelling Salesman Problem: A Survey. *Operations Research*, Vol. 16, No. 3 , pp. 538-558.
- Berger, J., & Barkaoui, M.** (2004). A parallel hybrid genetic algorithm for the vehicle routing problem with time windows. *Computers & operations research*, 31(12), 2037-2053.
- Bock, F.** (1958). “An algorithm for solving 'travelling salesman' and related network optimization problems”, 14. National meeting of the Operations Research Society of America, 897,
- Bodin, L. ve Golden, B.** (1981). Classification in vehicle routing and scheduling. *NETWORKS*, 11(2):97–108.
- Bolat, B., Erol, K., & Imrak, C.** (2004). Genetic algorithms in engineering applications and the Function of operators. *Sigma*, 4.
- Bouchra, B., Btissam, D., & Mohammad, C.** (2016, May). A hybrid genetic algorithm for the static and dynamic vehicle routing problem with soft time windows. In 2016 3rd International Conference on Logistics Operations Management (GOL) (pp. 1-9). IEEE.
- Böttner, C. U.** (2007). Weather routing for ships in degraded conditions. In *International Symposium on Maritime Safety, Security and Environmental Protection*.
- Braekers, K., Ramaekers, K. ve Van Nieuwenhuyse, I.,** (2016) The vehicle routing problem: State of the art classification and review. *Computers and Industrial Engineering*, V.99, s.300–313
- Burke, E. K. and Kendall, G.,** (2005). *Search Methodologies*, Springer, USA.
- Carić, T., Galić, A., Fosin, J., Gold, H., & Reinholz, A.** (2008). A modelling and optimization framework for real-world vehicle routing problems (s. 142). *IntechOpen*

- Caric, T., & Gold, H.** (2008). Vehicle routing problem.(ss.57-71)
- Chelouah, R., & Siarry, P.** (2000). Tabu search applied to global optimization. *European journal of operational research*, 123(2), 256-270
- Chen, C.L. and Vempati, V., S. and Aljaber, N.,** (1995). An application of genetic algorithms for flow shop problems, *European Journal of Operational Research*, 80, 389-396.
- Chen, T. K., Hay, L. L., & Ke, O.** (2001). Hybrid genetic algorithms in solving vehicle routing problems with time window constraints. *Asia-Pacific Journal of Operational Research*, 18(1), 121
- Chen, B., Qu, R., Bai, R., & Ishibuchi, H.** (2016, February). A Variable Neighbourhood Search Algorithm with Compound Neighbourhoods for VRPTW. In *ICORES* (pp. 25-35).
- Chen, P., Golden, B., Wang, X., & Wasil, E.** (2017). A novel approach to solve the split delivery vehicle routing problem. *International Transactions in Operational Research*, 24(1-2), 27-41.
- Cheng, R., Gen, M., & Tsujimura, Y.** (1999). A tutorial survey of job-shop scheduling problems using genetic algorithms, part II: hybrid genetic search strategies. *Computers & Industrial Engineering*, 36(2), 343-364.
- Cheng, C. B., & Wang, K. P.** (2009). Solving a vehicle routing problem with time windows by a decomposition technique and a genetic algorithm. *Expert Systems with Applications*, 36(4), 7758-7763.
- Christiansen, M., Fagerholt, K., & Ronen, D.** (2004). Ship routing and scheduling: Status and perspectives. *Transportation science*, 38(1), 1-18.
- Christofides, N., Mingozzi, A., & Toth, P.** (1979). Loading problems. N. Christofides and al., editors, *Combinatorial Optimization*, 339-369.
- Chu, P. C., & Beasley, J. E.** (1997). A genetic algorithm for the generalised assignment problem. *Computers & Operations Research*, 24(1), 17-23.
- Cibej, U., & Mihelič, J.** (2017). Adaptation and evaluation of the simplex algorithm for a data-flow architecture. In *Advances in computers* (Vol. 106, pp. 63-105). Elsevier.
- Clarke, G., & Wright, J. W.** (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations research*, 12(4), 568-581
- Clarkson Research Study.** (2004). *The Tramp Shipping Market* (s. 24).
- 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(5), 512-522.
- Cordeau, J. F., Laporte, G., Savelsbergh, M. W., & Vigo, D.** (2007). Vehicle routing. *Handbooks in operations research and management science*, 14, 367-428.
- Croes, G. A.** (1958). A method for solving traveling-salesman problems. *Operations research*, 6(6), 791-812.

- Çağatan, T.** (2003). Araç rotalama problemlerine genetik algoritma yaklaşımı ve bir uygulama. Yayınlanmamış Yüksek Lisans Tezi. Uludağ Üniversitesi Sosyal Bilimler Enstitüsü, Bursa.
- Çakar, K.** (2009). Genetik Algoritmalar Yardımıyla Acil Servis İstasyonu Yerleşiminin Optimizasyonu. Atatürk Üniversitesi Sosyal Bilimler Doktora Tezi, Erzurum, (ss.39-65).
- Çalışkan, K.** (2011). Karınca kolonisi optimizasyonu ile araç rotalama probleminin maliyetlerinin kümeleme tekniği ile iyileştirilmesi. Yayınlanmamış Yüksek Lisans Tezi. TOBB Ekonomi ve Teknoloji Üniversitesi Fen Bilimleri Enstitüsü,
- Çam, Ö.N.** (2018). Aylak zamanı en küçükleme amaçlı araç rotalama probleminin genetik algoritma ile çözümü. Yayınlanmamış Doktora Lisans Tezi. Uludağ Üniversitesi Sosyal Bilimler Enstitüsü, Bursa.
- Çavuşlu, M. A., Karakuzu, C., & Şahin, S.** (2010). Parçacık Sürü Optimizasyonu Algoritması ile Yapay Sinir Ağı Eğitiminin FPGA Üzerinde Donanımsal Gerçeklenmesi. Politeknik Dergisi, 13(2), 83-92.
- Çetin, M.** (2007). Gezgin satıcı örnek problemlerinin optimum sonuçlarının grid aracılığı ile hesaplanması (ss.9-15) .(Master's thesis, Balıkesir Üniversitesi Fen Bilimleri Enstitüsü)
- Çevik, K. K., & Koçer, H. E.** (2013). Parçacık sürü optimizasyonu ile yapay sinir ağları eğitime dayalı bir esnek hesaplama uygulaması. Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü Dergisi, 17(2), 39-45.
- Çeyrekoğlu, S.** (2017). Araç rotalama problemine genetik algoritma yaklaşımı ve örnek bir uygulama. Yayınlanmamış Yüksek Lisans Tezi. İstanbul Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.
- Çiçekdeş, B.** (2011). Heterojen kapasiteli gerçek hayat araç rotalama problemi için genetik algoritma yaklaşımı. Yayınlanmamış Yüksek Lisans Tezi. Dokuz Eylül Üniversitesi Fen Bilimleri Enstitüsü, İzmir.
- Çolak, S.** (2010). Genetik algoritmalar yardımı ile gezgin satıcı probleminin çözümü üzerine bir uygulama. Çukurova Üniversitesi Sosyal Bilimler Enstitüsü Dergisi, 19(3), 423-438.
- Dantzig, G., Fulkerson, R., & Johnson, S.** (1954). Solution of a large-scale traveling-salesman problem. Journal of the operations research society of America, 2(4), 393-410.
- Davis, Lawrence** (1987.) Genetic Algorithms and Simulated Annealing, Morgan Kaufmann Publishers.
- Dikmen, H., Elbir, A , Ekşi, Z , Çelik, F .** (2014). Gezgin Satıcı Probleminin Karınca Kolonisi ve Genetik Algoritmalarla Eniyilemesi ve Karşılaştırılması. Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü Dergisi , 18 (1) , 8-13.
- Donati Alberto, V., Gambardella Luca, M., Rizzoli Andrea, E., Casagrande, N., & Montemanni, R.** (2002). Time dependent vehicle routing problem with an ant colony system.

- Dorigo, M., Maniezzo, V., and Coloni, A.,** (1991) “Positive Feedback as a Search Strategy”, Technical Report 91-016, Italy.
- Dorigo, M., & Gambardella, L. M.** (1996, September). A study of some properties of Ant-Q. In International Conference on Parallel Problem Solving from Nature (pp. 656-665). Springer, Berlin, Heidelberg.
- Dorigo, M., Maniezzo, V., & Coloni, A.** (1996). Ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 26(1), 29-41.
- Dorigo, M., & Gambardella, L. M.** (1997). Ant colonies for the travelling salesman problem. biosystems, 43(2), 73-81.
- Dréo, J., Pétrowski, A., Siarry, P., & Taillard, E.** (2006). Metaheuristics for hard optimization: methods and case studies. Springer Science & Business Media.(ss. 47-55)
- Dursun, P.** (2009). Zaman Pencere Arac Rotalama Problemi’nin Genetik Algoritma İle Modellenmesi (Doctoral dissertation, Fen Bilimleri Enstitüsü). (ss.29-49)
- Düzakın, E., & Demircioğlu, M.** (2009). Araç rotalama problemleri ve çözüm yöntemleri. Çukurova Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi, 13(1).
- Eberhart, R., & Kennedy, J.** (1995, November). Particle swarm optimization. In Proceedings of the IEEE international conference on neural networks (Vol. 4, pp. 1942-1948). Citeseer.
- Eberhart, R., Simpson, P., & Dobbins, R.** (1996). Computational intelligence PC tools. Academic Press Professional, Inc..
- Eiselt, H. A., & Sandblom, C. L.** (2013). Integer programming and network models. Springer Science & Business Media, (ss. 229-260).
- Ekizler, H.** (2011). Araç rotalama probleminin çözümünde karınca kolonisi optimizasyonu algoritmasının kullanılması. İstanbul Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.
- Elen A. & Ekmekçi D.,** (2020) Popülasyon Tabanlı Meta-sezgisel Optimizasyon Teknikleri, Geleceğin Dünyasında Bilimsel ve Mesleki Çalışmalar, Ekin Basım Yayın Dağıtım. ISBN: 978-625-7983-95-2, ss. 1-20
- Elmas, C., & Yigit, T.** (2007). Genetic algorithm based on-line tuning of a PI controller for a switched reluctance motor drive. Electric power components and systems, 35(6), 675-691.
- Emel, G. G., & Taşkın, Ç.** (2002). Genetik Algoritmalar ve Uygulama Alanları. Uludağ Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi, 21(1), 129-152.
- Eren, H.** (2002). Akış tipi çizelgeleme problemlerinin genetik algoritma (GA) ile çözüm performansının artırılmasında deney tasarımı uygulaması (Doctoral dissertation, Fen Bilimleri Enstitüsü), (ss.4-18)

- Ermutaf, E.G.** (2019). Çok amaçlı konteyner yükleme ve araç rotalama problemlerinin çözümü için genetik algoritma yaklaşımı: porselen sektöründe karar destek sistemi önerisi. Yayımlanmamış Yüksek Lisans Tezi. Dumlupınar Üniversitesi / Fen Bilimleri Enstitüsü.
- Erol, S. & Dursun, A.** (2015). Deniz Yolu Taşımacılığında Finansal Riskler ve Riskten Korunma. Dokuz Eylül Üniversitesi Denizcilik Fakültesi Dergisi, 7(2), 172-201.
- Fagerholt, K.** (2004). A computer-based decision support system for vessel fleet scheduling-experience and future research. Decision Support Systems, 37(1), 35-47.
- Faust, O. S., Mehli, C. G., Hanne, T., & Dornberger, R.** (2020, March). A Genetic Algorithm for Optimizing Parameters for Ant Colony Optimization Solving Capacitated Vehicle Routing Problems. In Proceedings of the 2020 4th International Conference on Intelligent Systems, Metaheuristics & Swarm Intelligence (pp. 52-58).
- Foster, B. A., & Ryan, D. M.** (1976). An integer programming approach to the vehicle scheduling problem. Journal of the Operational Research Society, 27(2), 367-384.
- Gambardella, L. M., Taillard, É., & Agazzi, G.** (1999). Macs-vrptw: A multiple colony system for vehicle routing problems with time windows. In New ideas in optimization.
- Gandomi, A. H., Yang, X. S., Talatahari, S., & Alavi, A. H. (Eds.).** (2013a). Metaheuristic applications in structures and infrastructures. Newnes, Chapter 16: Genetical Algorithms (ss.389-417)
- Gandomi, A. H., Yang, X. S., Talatahari, S., & Alavi, A. H.** (2013b). Metaheuristic algorithms in modeling and optimization. Metaheuristic applications in structures and infrastructures, 1-24
- Gao, S., Zhang, Z., & Cao, C.** (2010). A Novel Ant Colony Genetic Hybrid Algorithm. JSW, 5(11), 1179-1186.
- Gatica, R. A., Miranda, P. A.** (2011). «Special Issue on Latin-American Research: A Time Based Discretization Approach for Ship Routing and Scheduling with Variable Speed,» Networks and Spatial Economics, cilt 11, pp. 465-485,
- Geiger, M.** (2003). A computational study of genetic crossover operators for multi-objective vehicle routing problem with soft time windows. In Multi-Criteria-und Fuzzy-Systeme in Theorie und Praxis (pp. 191-207). Deutscher Universitätsverlag.
- Gendreau, M., J.Y. Potvin, O. Br'aysy, G. Hasle, A. Løkketangen.** (2008). Metaheuristics for the vehicle routing problem and its extensions: A categorized bibliography. B. Golden, S. Raghavan, E. Wasil, eds., The Vehicle Routing Problem: Latest Advances and New Challenges. Springer, New York, 143-169
- Ghoseiri, K., & Ghannadpour, S. F.** (2010). Multi-objective vehicle routing problem with time windows using goal programming and genetic algorithm. Applied Soft Computing, 10(4), 1096-1107.

- Gillett, B.E., Miller, L.R.,** (1971). A Heuristic Algorithm For the Vehicle Dispatch Problem. *Operation Research*, vol. 22: 340-349
- Glover F.** (1986), Future Paths for Integer Programming and Links to Artificial Intelligence *Computers and Operations Research* 13, pp. 533-549.
- Glover F.** (1989). Tabu search—part I *ORSA J. Comput.*, 1 (3) (ss. 190-206).
- Goldberg, David E.** (1988). *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley.
- Gökçe, E.İ.** (2004). Araç rotalama problemleri için yeni bir karınca kolonisi sistemi yaklaşımı. Yayımlanmamış Yüksek Lisans Tezi. Sabancı Üniversitesi Mühendislik ve Fen Bilimleri Enstitüsü.
- Göçken, T., Yaktubay, M., & Kiliç, F.** (2017, September). Improvement of a genetic algorithm approach for the solution of vehicle routing problem with time windows. In *2017 International Artificial Intelligence and Data Processing Symposium (IDAP)* (pp. 1-8). IEEE.
- Gökçen, T., Yaktubay, M., & Kilic, F.** (2018). Zaman pencereli araç rotalama problemi çözümü için çok amaçlı genetik algoritma yaklaşımı. *Gazi Üniversitesi Fen Bilimleri Dergisi Part C: Tasarım ve Teknoloji*, 6(4), 774-786.
- Göksal, F.P.** (2010). Eşzamanlı topla-dağıt araç rotalama problemi için sezgisel yaklaşımlar: Genetik algoritma ve kuş sürüsü eniyileme. Yayımlanmamış Yüksek Lisans Tezi. Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara.
- Göksal, F.P.** (2017). Çok amaçlı eşzamanlı topla-dağıt araç rotalama problemi: Hedef programlama ve genetik algoritmalara dayalı çözüm yaklaşımları. Yayımlanmamış Doktora Lisans Tezi.
- Grüter, C., & Farina, W. M.** (2009). The honeybee waggle dance: can we follow the steps. *Trends in Ecology & Evolution*, 24(5), 242-247.
- Gutin, G., & Punnen, A. P.** (2002). *The Travelling Salesman Problem: Applications, Formulations and Variations*. The Traveling Salesman Problem and Its Variations, 12.
- Gülcü, Ş., Mahi, M., Baykan, Ö. K., & Kodaz, H.** (2018). A parallel cooperative hybrid method based on ant colony optimization and 3-Opt algorithm for solving traveling salesman problem. *Soft Computing*, 22(5), 1669-1685.
- Gülsün, B., Tuzkaya, G., & Bildik, E.** (2008). Reverse logistics network design: a simulated annealing approach. *Journal of Engineering and Natural Sciences*, 26(1), 68-80.
- Günay, N.S.** (2013). Genetik algoritma ile araç rotalama problemlerinin çözümü için görsel rotalama yazılımı geliştirme. Yayımlanmamış Yüksek Lisans Tezi. İstanbul Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.
- Gürbüz, Ö.** (2015). Tabu arama algoritmasının kuyruk problemine uygulanması, Master's thesis, Fen Bilimleri Enstitüsü (ss.12-18).

- Haldurai, L., Madhubala, T., & Rajalakshmi, R.** (2016). A study on genetic algorithm and its applications. *International Journal of Computer Sciences and Engineering*, 4(10), 139.
- Halim, S., & Yoanita, L.** (2015, December). Adjusted clustering Clarke-Wright Saving Algorithm for two depots-N vehicles. In *2015 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)* (pp. 270-274). IEEE.
- Hertrich, C., Hungerländer, P., & Truden, C.** (2019). Sweep Algorithms for the Capacitated Vehicle Routing Problem with Structured Time Windows. In *Operations Research Proceedings 2018* (pp. 127-133). Springer, Cham
- Hinnenthal J. ve Günther C.** (2010). Robust pareto-optimum routing of ships utilising deterministic and ensemble weather forecasts. *Ships and Offshore Structures*, 5: 105-114.
- Ho, W. K., Ang, J. C., & Lim, A.** (2001). A hybrid search alogrithm for the vehicle routing problem with time windows. *International journal on artificial intelligence tools*, 10(03), 431-449.
- Ho, W., Ho, G. T. S., Ji, P. and Lau, H. C. W.,** (2008). A hybrid genetic algorithm for the multi-depot vehicle routing problem, *Engineering Applications of Artificial Intelligence*, Vol. 21, Issue 4, pp. 548-557
- Hoffman, J.** (2010) «Shipping Out Of The Economic Crisis,» *The Brown Journal of World Affairs*, cilt 2, no. 16, pp. 121-1301
- Holland John, H.** (1975). *Adaptation in natural and artificial systems*. Ann Arbor: University of Michigan Press.
- Hwang, H. S.** (2002). An improved model for vehicle routing problem with time constraint based on genetic algorithm. *Computers & Industrial Engineering*, 42(2-4), 361-369.
- Ibrahim, M. F., Nurhakiki, F. R., Utama, D. M., & Rizaki, A. A.** (2021, February). Optimised Genetic Algorithm Crossover and Mutation Stage for Vehicle Routing Problem Pick-Up and Delivery with Time Windows. In *IOP Conference Series: Materials Science and Engineering* (Vol. 1071, No. 1, p. 012025). IOP Publishing.
- İnan T.,** (2019). Yüzen vasıtalar için seyir planlaması yapabilen bir zeki sistemin geliştirilmesi. *Doktora Tezi*, Marmara Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.
- İnan, T, ve Baba, A.** (2021). Ticari Gemiler İçin Seyir Süresi ve Yakıt Tüketiminin Azaltılması Amaçlı, Hava ve Deniz Şartlarına Göre Rota Optimizasyonu Sistemi (Ege Denizi Örneği). *Politeknik Dergisi*, 1-1.
- İşçi, Ö & Korukoğlu, S.** (2003). Genetik algoritma yaklaşımı ve yöneylem araştırmasında bir uygulama. *Yönetim ve Ekonomi: Celal Bayar Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, 10(2), 191-208.
- Jin, F., & Fu, Y.** (2010, August). Hybrid genetic algorithm for vehicle routing problem with time windows. In *2010 International Conference on Management and Service Science* (pp. 1-4). IEEE.

- Johnson, D. S., Aragon, C. R., McGeoch, L. A., & Schevon, C.** (1989). Optimization by simulated annealing: An experimental evaluation; part I, graph partitioning. *Operations research*, 37(6), 865-892.
- Johnson, D. S., & McGeoch, L. A.** (1997). The traveling salesman problem: A case study in local optimization. *Local search in combinatorial optimization*, 1(1), 215-310.
- Jung, S., & Haghani, A.** (2001). Genetic algorithm for the time-dependent vehicle routing problem. *Transportation Research Record*, 1771(1), 164-171
- Jung, S., & Moon, B. R.** (2002, July). A Hybrid Genetic Algorithm For The Vehicle Routing Problem With Time Windows. In *GECCO* (pp. 1309-1316).
- Jünger, M., Reinelt, G., & Rinaldi, G.** (1995). The traveling salesman problem. *Handbooks in operations research and management science*, 7, 225-330.
- Kadaba, N.** (1990). Xroute: A knowledge-based routing system using neural networks and genetic algorithms (Doctoral dissertation, North Dakota State University).
- Kahraman, A. M. & Özdağlar, D.** (2004). SU DAĞITIM SİSTEMLERİNİN GENETİK ALGORİTMA İLE OPTİMİZASYONU. *Dokuz Eylül Üniversitesi Mühendislik Fakültesi Fen ve Mühendislik Dergisi*, 6(3), 1-18.
- Kallrath, J.** (2004). *Modeling languages in mathematical optimization*. Kluwer Academic Publishers, USA.
- Kang, M.H., Choi, H., Kim, H., & Park, B.** (2011). Development of a maritime transportation planning support system for car carriers based on genetic algorithm. *Applied Intelligence*, 36, 585-604
- Karaboğa, D.** (2004). *Yapay Zeka Optimizasyon Algoritmaları*, Atlas Yayın Dağıtım, 1. Basım, İstanbul.
- Karaboga, D.** (2005). An idea based on honey bee swarm for numerical optimization (Vol. 200, pp. 1-10). Technical report-tr06, Erciyes university, engineering faculty, computer engineering department.
- Karaboga, D., & Akay, B.** (2009). A comparative study of artificial bee colony algorithm. *Applied mathematics and computation*, 214(1), 108-132.
- Karaboğa, D.** (2017). *Yapay zeka optimizasyon algoritmaları*. Nobel Akademi Yayıncılık.
- Kaya, B.** (2020) *Yapay Arı Kolonisi Algoritması ile Yapılan Geliştirmeler ve Sonuçları*. *Verimlilik Dergisi*, (1), 99-115.
- Kaya, O.** (2010). Kombinatorial Optimizasyon Probleminin Bir Sınıfın Genetik Algoritmalar ile Çözümü Üzerine (ss 7-13). *Ege Üniversitesi Fen Bilimleri Enstitüsü, Matematik Ana Bilim Dalı*
- Kaya, C.** (2017). Eş zamanlı topla dağıt araç rotalama problemi için karınca koloni sistemi ile güçlendirilmiş değişken komşuluk arama algoritması. *Yayımlanmamış Yüksek Lisans Tezi*. Pamukkale Üniversitesi Fen Bilimleri Enstitüsü, Denizli

- Keçeci, B.** (2008). Önce dağıt sonra topla araç rotalama problem için tamsayılı karar modelleri (ss10-12). (Master's thesis, Başkent Üniversitesi Fen Bilimleri Enstitüsü).
- Kemer, B.** (2010). Araç rotalama problemlerine genetik algoritma yaklaşımı: Bir gıda dağıtım firması uygulaması. Yayınlanmamış Yüksek Lisans Tezi. Karadeniz Teknik Üniversitesi Sosyal Bilimler Enstitüsü
- Kennedy, J., & Eberhart, R.** (1995, November). Particle swarm optimization. In Proceedings of ICNN'95-international conference on neural networks (Vol. 4, pp. 1942-1948). IEEE.
- Kepaptsoglou K., Fountas G. ve Karlaftis M.,** (2015). Weather impact on containership routing in closed seas: a chance-constraint optimization approach. Transportation Research Part C: Emerging Technologies, 55: 139-155.
- Keskintürk, T., & Söyler, H.** (2006). Global karınca kolonisi optimizasyonu. Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi, 21(4), 689-698.
- Keskintürk, T.** (2009). Araç rotalama problemlerinin global karınca koloni optimizasyonu ile çözümü. Yayınlanmamış Doktora Tezi. İstanbul Üniversitesi Sosyal Bilimler Enstitüsü, İstanbul.
- Keskintürk, T., Topuk, N., & Özyeşil, O.** (2015). Araç Rotalama Problemleri ve Çözüm Yöntemleri. İşletme Bilimi Dergisi, 3(2), 77-107.
- Keskintürk T., Kiremitçi, B., & Kiremitçi, S.** (2016). 2-Opt Algoritması ve Başlangıç Çözümünün Algoritma Sonuçları Üzerindeki Etkisi. Endüstri Mühendisliği, 27(3), 2-12
- Khanra, A., Pal, T., Maiti, M. K., & Maiti, M.** (2019). Multi-objective four dimensional imprecise TSP solved with a hybrid multi-objective ant colony optimization-genetic algorithm with diversity. Journal of Intelligent & Fuzzy Systems, 36(1), 47-65.
- Khoo, T. S., Mohammad, B. B., Wong, V. H., Tay, Y. H., & Nair, M.** (2020). A Two-Phase Distributed Ruin-and-Recreate Genetic Algorithm for Solving the Vehicle Routing Problem with Time Windows. IEEE Access, 8, 169851-169871.
- Khoo, T. S., & Mohammad, B. B.** (2021). The parallelization of a two-phase distributed hybrid ruin-and-recreate genetic algorithm for solving multi-objective vehicle routing problem with time windows. Expert Systems with Applications, 168, 114408.
- Kılıç, S.** (2008). Bulanık karar ortamında karınca kolonisi optimizasyonu yöntemiyle araç rotalama. Yayınlanmamış Doktora Tezi. İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü
- Kılıçarslan, S., Adem. K., & Cömert, O.** (2019). Parçacık Sürü Optimizasyonu Kullanılarak Boyutu Azaltılmış Mikrodizi Verileri Üzerinde Makine Öğrenmesi Yöntemleri ile Prostat Kanseri Teşhisi. Düzce Üniversitesi Bilim ve Teknoloji Dergisi, 7(1), 769-777.
- Kiremitci, S. & Özçakar, N.** (2011). Denizyolu Yük Tasımacılığında Rotalama ve Çizelgeleme, (ss. 14-25). İstanbul Üniversitesi, Sosyal Bilimler Enstitüsü (Doktora Tezi)

- Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P.** (1983). Optimization by simulated annealing. *science*, 220(4598), 671-680.
- Koç, Ç. & Karaoğlu, İ.** (2014). Zaman Bağımlı Araç Rotalama Problemi İçin Bir Matematiksel Model. *Journal of the Faculty of Engineering & Architecture of Gazi University*, 29(3).
- Korst, J., Aarts, E., & Lenstra, J. K.** (1997). Scheduling periodic tasks with slack. *INFORMS Journal on Computing*, 9(4), 351-362.
- Korte, B. H., Vygen, J., Korte, B., & Vygen, J.** (2011). *Combinatorial optimization* (Vol. 1). Berlin: Springer.
- Kosif, B., & Ekmekçi, İ.** (2012). Araç rotalama sistemleri ve tasarruf algoritması uygulaması
- Kosmas, O. T., & Vlachos, D. S.** (2008). Operational Optimal Ship Routing Using a Hybrid Parallel Genetic Algorithm. *arXiv preprint arXiv:0811.2166*.
- Kosmas, O. T., Vlachos, D. S. and Simos, T. E.** (2008). Obstacle Bypassing in Optimal Ship Routing Using Simulated Annealing. *Proceedings of International Electronic Conference on Computer Science*, Vol. 1060, pp. 79-82.
- Kumar, V. S., Thansekhar, M. R., Saravanan, R., & Amali, S. M. J.** (2014). Solving multi-objective vehicle routing problem with time windows by FAGA. *Procedia Engineering*, 97, 2176-2185.
- Kumar, S. N., & Panneerselvam, R.** (2015). A time-dependent vehicle routing problem with time windows for e-commerce supplier site pickups using genetic algorithm. *intelligent information management*, 7(04), 181.
- Kumar, S. N., & Panneerselvam, R.** (2016). Development of an efficient genetic algorithm for the time dependent vehicle routing problem with time windows. *American Journal of Operations Research*, 7(1), 1-25.
- Kurt, M.** (2008). Çoklu depolu araç rotalama problemleri için bir karınca kolonisi optimizasyonu algoritmasının tasarımı ve uygulaması. Yayımlanmamış Yüksek Lisans Tezi. Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.
- Kuzu, S, Önay, O, Şen, U, Tunçer, M, Yıldırım, B, ve Keskindürk, T.** (2014). Gezgin Satıcı Problemlerinin Metasezgiseller ile Çözümü . *İstanbul Üniversitesi İşletme Fakültesi Dergisi*, 43 (1), 1-27.
- Laporte, G.** (1992). The vehicle routing problem: An overview of exact and approximate algorithms. *European journal of operational research*, 59(3), 345-358.
- Lee H., Aydin N., Choi Y. Lekhavat S. ve Irani Z.,** (2018). A decision support system for vessel speed decision in maritime logistics using weather archive big data. *Computers & Operations Research*, 98: 330-342
- Li, P. Hong-Bo Wang, and Dong-Qing He** (2018). Ship weather routing based on improved ant colony optimization algorithm. *IEEE Industrial Cyber-Physical Systems (ICPS)*

- Li, M., Lu, J., & Ma, W.** (2019). A hybrid multi-objective genetic algorithm for bi-objective time window assignment vehicle routing problem. *Promet-Traffic&Transportation*, 31(5), 513-525.
- Liang, Y., & Wang, L.** (2020). Applying genetic algorithm and ant colony optimization algorithm into marine investigation path planning model. *Soft Computing*, 24(11), 8199-8210.
- Liu, C. Y.** (2013). An improved adaptive genetic algorithm for the multi-depot vehicle routing problem with time window. *Journal of Networks*, 8(5), 1035.
- Liu, C.** (2014). Multi-robot task allocation for inspection problems with cooperative tasks using hybrid genetic algorithms (Vol. 4). kassel university press GmbH. (ss.29-31)
- Liu, W. Y., Lin, C. C., Chiu, C. R., Tsao, Y. S., & Wang, Q.** (2014). Minimizing the carbon footprint for the time-dependent heterogeneous-fleet vehicle routing problem with alternative paths. *Sustainability*, 6(7), 4658-4684
- Luan, J., Yao, Z., Zhao, F., and Xin Song, X.** (2019). A novel method to solve supplier selection problem: Hybrid algorithm of genetic algorithm and ant colony optimization, *Mathematics and Computers in Simulation*, Volume 156, 294-309. <https://doi.org/10.1016/j.matcom.2018.08.011>.
- Maki A., Akimoto Y., Nagata Y., Kobayashi S.,** (2011). A new weather-routing system that accounts for ship stability based on a real-coded genetic algorithm. *Journal of Marine Science and Technology* , 16: 311- 322.
- Malani, S., Sreenidhi, T., & Prasanna, G. S.** (January 2011). A Simulated Annealing based approach for solving SplitSearch. Conference: ORSI
- Mavrovouniotis, M., & Yang, S.** (2012, June). Ant colony optimization with memory-based immigrants for the dynamic vehicle routing problem. In *2012 IEEE Congress on Evolutionary Computation* (pp. 1-8). IEEE.
- Mourkousis, G., Protonotarios, M., & Varvarigou, T.** (2003). Application of genetic algorithms to a large-scale multiple-constraint vehicle routing problem. *International Journal of Computational Intelligence and Applications*, 3(01), 1-21.
- Michalewicz, Z.** (2013). Genetic algorithms+ data structures= evolution programs. Springer Science & Business Media (ss.8-18).
- Mijwel, M. M.** (2015). Heuristic Algorithms. *Ann. Discret. Math*, 53, 221-242
- Miller, B. L., & Goldberg, D. E.** (1995). Genetic algorithms, tournament selection, and the effects of noise. *Complex systems*, 9(3), 193-212.
- Mitchell, M.** (1998). An introduction to genetic algorithms. MIT press (ss.15-31).
- Mitchell, J. E., Floudas, C. A., & Pardalos, P. M.** (2009). Integer Programming: Branch and Cut Algorithms. *Encyclopedia of optimization*, 2, 519-525.
- Moghaddam, B. F., Ruiz, R., & Sadjadi, S. J.** (2012). Vehicle routing problem with uncertain demands: An advanced particle swarm algorithm. *Computers & Industrial Engineering*, 62(1), 306-317.

- Mori, M., & Tseng, C. C.** (1997). A genetic algorithm for multi-mode resource constrained project scheduling problem. *European Journal of Operational Research*, 100(1), 134-141(s. 137).
- Mutlu, M. Ç.** (2010). A branch and bound algorithm for resource leveling problem (Master's thesis ss 39-45)
- Nissen, V., & Biethahn, J.** (1995). An introduction to evolutionary algorithms. In *Evolutionary algorithms in management applications* (pp. 3-43). Springer, Berlin, Heidelberg.
- Nowicki, E., & Smutnicki, C.** (1996). A fast tabu search algorithm for the job shop problem. *Management science*, 42(6), 797-813.
- Nguyen, P. K., Crainic, T. G., & Toulouse, M.** (2011). A hybrid genetic algorithm for the periodic vehicle routing problem with time windows (pp. 2011-25). CIRRELT.
- Nguyen, P. K., Crainic, T. G., & Toulouse, M.** (2014). A hybrid generational genetic algorithm for the periodic vehicle routing problem with time windows. *Journal of Heuristics*, 20(4), 383-416.
- Okur, E.** (2019). Genetik algoritma ile araç rotalama probleminin çözümü ve bir uygulama denemesi. Yayımlanmamış Yüksek Lisans Tezi. Anadolu Üniversitesi Sosyal Bilimler Enstitüsü, Sakarya.
- Okur, E. & Atlas, Ü. M.** (2020). Araç Rotalama Probleminin Genetik Algoritma ile Çözümü. *Anadolu University Journal of Social Sciences*, 20(3).
- Ombuki, B., Ross, B. J., & Hanshar, F.** (2006). Multi-objective genetic algorithms for vehicle routing problem with time windows. *Applied Intelligence*, 24(1), 17-30.
- Onwubolu, G. C.** (2002). *Emerging Optimization Techniques in Production Planning & Control*. World Scientific. s. 24
- Önder, E.** (2011). Araç rotalama problemlerinin parçacık sürü ve genetik algoritma ile optimizasyonu. Yayımlanmamış Doktora Lisans Tezi. İstanbul Üniversitesi Sosyal Bilimler Enstitüsü, İstanbul.
- Önsel,Ş.** (2021)<https://web.itu.edu.tr/~kabak/dersler/EM302/pdf/TSPcozum.pdf>
- Özalp, M., & Alp, S.** (2020). Uygun Dağıtım Rotası Belirlenmesi Probleminde Hibrit Sezgisel Bir Yöntem Önerisi: Bir Kargo Firması Örneği. *Akıllı Ulaşım Sistemleri ve Uygulamaları Dergisi*, 3(1), 59-70.
- Özbakır L. & Tapkan, P.,Z.** (2010). Kombinatorial Optimizasyon Problemlerinde Arı Sistemi Davranışı. *Erciyes Üniversitesi Bilimsel Araştırma Projeleri Koordinasyon Birimi*, (ss.29-33).
- Özder, U.** (2009). Kobilerde otomasyona geçil sürecinde tamsayılı doğrusal programlama yöntemini kullanarak karar verme (Doctoral dissertation, Doktora Tezi, Atatürk Üniversitesi, Sosyal Bilimler Enstitüsü, Erzurum. Erişim Adresi: <https://www.atauni.edu.tr/yuklemeler/10ee3484dd0bb2b93d5647e3ac1a757d.pdf>) (ss. 34-39).
- Özsağlam, M. Y. & Çunkaş, M.** (2008). Optimizasyon problemlerinin çözümü için parçacık sürü optimizasyonu algoritması. *Politeknik Dergisi*, 11(4), 299-305

- Öztürk, A., & Alkan, S.** (2013). Parçacık Sürü Optimizasyonu ile Fotovoltaik Sistemlerde Kullanılan DC-DC Dönüştürücünün Kontrolü. *İleri Teknoloji Bilimleri Dergisi*, 2(3), 110-120.
- Pacheco M.B., ve Soares C.G.,** (2007). Ship weather routing based on seakeeping performance, *International Conference on Marine Structures*, Glasgow, 71-78.
- Padhy C., Sen D., and Bhaskaran P.K.,** (2007). Application of wave model for weather routing of ships in the north indian. *Natural Hazards*, 44: 373-385.
- Paksoy, S. & Uzun, A.** (2008). Genetik algoritma ile kaynak kısıtlı proje çizelgeleme. *Çukurova Üniversitesi Sosyal Bilimler Enstitüsü Dergisi*, 17(2), 345-362.
- Pan, C., Zhang, Z., Sun, W., Shi, J., & Wang, H.** (2021). Development of ship weather routing system with higher accuracy using SPSS and an improved genetic algorithm. *Journal of Marine Science and Technology*, 1-16.
- Pankratz, G.** (2005). Dynamic vehicle routing by means of a genetic algorithm. *International Journal of Physical Distribution & Logistics Management*.
- Penev, M. K. V. S. S.** (2005). Genetic operators crossover and mutation in solving the TSP problem. In *International Conference on Computer Systems and Technologies*.
- Pierre, D. M., & Zakaria, N.** (2017). Stochastic partially optimized cyclic shift crossover for multi-objective genetic algorithms for the vehicle routing problem with time-windows. *Applied Soft Computing*, 52, 863-876.
- Prins, C.** (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & operations research*, 31(12), 1985-2002.
- Rabbani, M., Pourreza, P., Farrokhi-Asl, H., & Nouri, N.** (2018). A hybrid genetic algorithm for multi-depot vehicle routing problem with considering time window repair and pick-up. *Journal of Modelling in Management*.
- Reis, S. & Torun, A. T.** (2015). Yapay arı koloni algoritmasının tarım alanlarının sınıflandırılmasında kullanılabilirliğinin irdelenmesi . *Türkiye Ulusal Fotogrametri ve Uzaktan Algılama Birliği 8. Teknik Sempozyumu (TUFUAB 2015)*
- Remer, B., & Malikopoulos, A. A.** (2019, July). The multi-objective dynamic traveling salesman problem: last mile delivery with unmanned aerial vehicles assistance. In *2019 American Control Conference (ACC)* (pp. 5304-5309). IEEE.
- Renaud, J., Boctor, F. F., & Laporte, G.** (1996). An improved petal heuristic for the vehicle routeing problem. *Journal of the operational Research Society*, 47(2), 329-336
- Ronen, D.** (1983). Cargo ships routing and scheduling: Survey of models and problems. *European Journal of Operational Research*, 12(2), 119-126.
- Ropke, S.,** (2005). Heuristic and exact algorithms for vehicle routing problems.

- Rothlauf, F.** (2006). Representations for genetic and evolutionary algorithms. In *Representations for Genetic and Evolutionary Algorithms* (pp. 9-32). Springer, Berlin, Heidelberg.
- Ruiz, E., Soto-Mendoza, V., Barbosa, A. E. R., & Reyes, R.** (2019). Solving the open vehicle routing problem with capacity and distance constraints with a biased random key genetic algorithm. *Computers & Industrial Engineering*, 133, 207-219.
- Salhi, S., & Sari, M.** (1997). A multi-level composite heuristic for the multi-depot vehicle fleet mix problem. *European Journal of Operational Research*, 103(1), 95-112.
- Sallabi, O. M., & Elhaddad, Y.** (2009). An improved genetic algorithm to solve the traveling salesman problem (ss. 389-417).
- Sakawa, M.** (2012). *Genetic algorithms and fuzzy multiobjective optimization* (Vol. 14). Springer Science & Business Media (ss.11-25).
- Sato, T., & Hagiwara, M.** (1998). Bee system: finding solution by a concentrated search. *IEEJ Transactions on Electronics, Information and Systems*, 118(5), 721-726.
- Savelsbergh, M. W.** (1992). The vehicle routing problem with time windows: Minimizing route duration. *ORSA journal on computing*, 4(2), 146-154.
- Sbai, I., Limam, O., & Krichen, S.** (2020). An effective genetic algorithm for solving the capacitated vehicle routing problem with two-dimensional loading constraint. *International Journal of Computational Intelligence Studies*, 9(1-2), 85-106.
- Schneider, M., Stenger, A., & Goeke, D.** (2014). The electric vehicle-routing problem with time windows and recharging stations. *Transportation science*, 48(4), 500-520.
- Shi, Y.** (2001, May). Particle swarm optimization: developments, applications and resources. In *Proceedings of the 2001 congress on evolutionary computation* (IEEE Cat. No. 01TH8546) (Vol. 1, pp. 81-86). IEEE.
- Shi, Y., Boudouh, T., & Grunder, O.** (2017). A hybrid genetic algorithm for a home health care routing problem with time window and fuzzy demand. *Expert Systems with Applications*, 72, 160-176.
- Silva, C. A. Faria, J. M. Abrantes, P.J. M. C. Sousa, M. Surico and D. Naso,** (2005). Concrete Delivery using a combination of GA and ACO," *Proceedings of the 44th IEEE Conference on Decision and Control*, , pp. 7633-7638, doi: 10.1109/CDC.2005.1583394.
- Sivanandam, S. N., & Deepa, S. N.** (2008). Genetic algorithms. In *Introduction to genetic algorithms* (pp. 15-37). Springer, Berlin, Heidelberg.
- Smith, V. J. R., Osman, I. H., Reeves, C. R. and Smith, G. D.,** 1996: *Modern Heuristic Search Methods*, John Wiley and Sons Ltd.
- Solomon, M. M.** (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations research*, 35(2):254–265.

- Sui, H., Niu, W.** (2016). Branch-pipe-routing approach for ships using improved genetic algorithm. *Front. Mech. Eng.* 11, 316–323
<https://doi.org/10.1007/s11465-016-0384-z>
- Şahin, Y., & Eroğlu, A.** (2014). Kapasite kısıtlı araç rotalama problemi için metasezgisel yöntemler: bilimsel yazın taraması. *Süleyman Demirel Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, 19(4), 337-355..
- Şavli, Ş.** (2016). Sezgisel Eniyilemede Deney Tasarımı ve Bir Uygulama. *Süleyman Demirel Üniversitesi, Fen Bilimleri Üniversitesi, Endüstri Mühendisliği*.(ss.13-20)
- Şeker, Ş.** (2007). Araç rotalama problemleri ve zaman pencereli stokastik araç rotalama problemine genetik algoritma yaklaşımı. *Yayımlanmamış Yüksek Lisans Tezi. Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.*
- Şen, T.** (2014). Kümeleme ve genetik algoritma destekli yaklaşımlarla kapasite kısıtlı araç rotalama probleminin çözümü: perakende zincirinde uygulanması. *Yayımlanmamış Yüksek Lisans Tezi. Sakarya Üniversitesi Fen Bilimleri Enstitüsü, Sakarya*
- Şenaras, A. E., & İnanç, Ş.** (2017). GSP Çözümü İçin Karınca Kolonisi Optimizasyonu. *Sosyal Bilimler Metinleri*, 2017(2), 58-67.
- Tan, K. C., Lee, T. H., Ou, K., & Lee, L. H.** (2001, May). A messy genetic algorithm for the vehicle routing problem with time window constraints. In *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No. 01TH8546) (Vol. 1, pp. 679-686). IEEE.*
- Tanaka M. ve Kobayashi K.,** (2019). A route generation algorithm for an optimal fuel routing problem between two single ports. *International Transactions in Operational Research*, 26: 529-550.
- Tansini, L., Urquhart, M. E., & Viera, O.** (2001). Comparing assignment algorithms for the Multi-Depot VRP. *Reportes Técnicos* 01-08.
- Taşkıran, D.** (2006). Zaman kısıtlı araç rotalama problemine farklı bir karınca kolonisi sistemi yaklaşımı. *Yayımlanmamış Yüksek Lisans Tezi. Sabancı Üniversitesi Mühendislik ve Fen Bilimleri Enstitüsü,*
- Teodorovic, D., & Dell’Orco, M.** (2005). Bee colony optimization—a cooperative learning approach to complex transportation problems. *Advanced OR and AI methods in transportation*, 51, 60
- Thangiah, S. R., Nygard, K. E., & Juell, P. L.** (1991, January). Gideon: A genetic algorithm system for vehicle routing with time windows. In *Proceedings The Seventh IEEE Conference on Artificial Intelligence Application (pp. 322-323). IEEE Computer Society.*
- Thangiah, S. R.** (2019). A hybrid genetic algorithm, simulated annealing and tabu search heuristic for vehicle routing problems with time windows. In *Practical handbook of genetic algorithms (pp. 347-384). CRC Press.*

- Tonbul, E. (2016).** Gerçek hayat maliyetlerini göz önüne alan açık uçlu araç rotalama problemi için bir melez genetik algoritma önerisi. Yayınlanmamış Yüksek Lisans Tezi. Anadolu Üniversitesi Fen Bilimleri Enstitüsü, Eskişehir.
- Tong, Z., Ning, L., & Debao, S. (2004, June).** Genetic algorithm for vehicle routing problem with time window with uncertain vehicle number. In Fifth World Congress on Intelligent Control and Automation (IEEE Cat. No. 04EX788) (Vol. 4, pp. 2846-2849). IEEE.
- Toth, P., & Vigo, D. (Eds.). (2002).** The vehicle routing problem. Society for Industrial and Applied Mathematics
- Tümsekçalı, E. (2020).** Heterojen ürün çeşitli ve heterojen araç filolu yeşil araç rotalama problemi için gıda sektöründe genetik algoritma uygulaması. Yayınlanmamış Yüksek Lisans Tezi. Gazi Üniversitesi Sosyal Bilimler Enstitüsü, Ankara.
- Türe, I. (2021).** <https://www.7deniz.net/haber-2020-nin-genel-degerlendirmesi-ve-gelecek-vizyonu-36705.html>
- Trivedi, S. M. (2010).** An analysis of financial performance of state road transport corporation in Gujarat (Doctoral dissertation, Saurashtra University)
- Tsou, M.-C., Kao, S.-L., Su, C.-M. (2010).** Decision Support for Genetic Algorithms for Ship Collision Avoidance Route Planning. The Journal of Navigation, Vol. 63, pp. 167-182.
- Tsou M.C. ve Cheng H.C., (2013).** An ant colony algorithm for efficient ship routing. Polish Maritime Research, 20: 28-38.
- Umbarkar, A. J., & Sheth, P. D. (2015).** Crossover operators in genetic algorithms: a review. ICTACT journal on soft computing, 6(1).
- UNCTAD (2020).** Review of the Maritime Transport 2020, United Nations; UNCTAD, Newyork, Geneva
- Ursani, Z., Essam, D., Cornforth, D., & Stocker, R. (2011).** Localized genetic algorithm for vehicle routing problem with time windows. Applied Soft Computing, 11(8), 5375-5390.
- Uyumaz, D. (2017).** Kapasite kısıtlı araç rotalama probleminin paralel genetik algoritma ile çözümü. Yayınlanmamış Yüksek Lisans Tezi. Karabük Üniversitesi Fen Bilimleri Enstitüsü, Zonguldak.
- Varun Kumar, S. G., & Panneerselvam, R. (2017).** A study of crossover operators for genetic algorithms to solve VRP and its variants and new sinusoidal motion crossover operator. International Journal of Computational Intelligence Research, 13(7), 1717-1733.
- Vettor R. ve Soares C., (2016).** Development of a ship weather routing system. Ocean Engineering, 123: 1-14.
- Vidal, T., Crainic, T. G., Gendreau, M., & Prins, C. (2013).** A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. Computers & operations research, 40(1), 475-489.

- Vural, A.V.** (2003). Eş zamanlı dağıtım ve toplama içeren kapasiteli araç rotalama problemi için genetik algoritma bazlı sezgisel yaklaşım. Yayınlanmamış Yüksek Lisans Tezi. Sabancı Üniversitesi Mühendislik ve Fen Bilimleri Enstitüsü
- Vural, M.** (2005). Genetik algoritma yöntemi ile toplu üretim planlama (Doctoral dissertation, Fen Bilimleri Enstitüsü), ss. 18-31.
- Wang, X. P., Xu, C. L., & Hu, X. P.** (2008a, September). Genetic algorithm for vehicle routing problem with time windows and a limited number of vehicles. In 2008 International Conference on Management Science and Engineering 15th Annual Conference Proceedings (pp. 128-133). IEEE.
- Wang, W., Wang, Z., & Qiao, F.** (2008b, December). An improved genetic algorithm for vehicle routing problem with time-window. In 2008 International Symposium on computer science and computational technology (Vol. 1, pp. 189-194). IEEE.
- Wang, S. & Meng, Q.** (2012a). Sailing speed optimization for container ships in a liner shipping network. *Transportation Research Part E: Logistics and Transportation Review*, 48(3), 701-714.
- Wang, H. F., & Chen, Y. Y.** (2012b). A genetic algorithm for the simultaneous delivery and pickup problems with time window. *Computers & Industrial Engineering*, 62(1), 84-95
- Xu, J., Goncalves, G., & Hsu, T.** (2008, June). Genetic algorithm for the vehicle routing problem with time windows and fuzzy demand. In 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence) (pp. 4125-4129). IEEE.
- Xu, S. H., Liu, J. P., Zhang, F. H., Wang, L., & Sun, L. J.** (2015). A combination of genetic algorithm and particle swarm optimization for vehicle routing problem with time windows. *Sensors*, 15(9), 21033-21053.
- Xue, B., Zhang, M., & Browne, W. N.** (2014). Particle swarm optimisation for feature selection in classification: Novel initialisation and updating mechanisms. *Applied soft computing*, 18, 261-276.
- Yaktubay, M.** (2018). Araç rotalama probleminde genetik algoritma tabanlı çözüm yaklaşımı. Yayınlanmamış Yüksek Lisans Tezi. Adana Bilim ve Teknoloji Üniversitesi Fen Bilimleri Enstitüsü, Adana.
- Yang, X.** (2008). Introduction to mathematical optimization. From linear programming to metaheuristics.
- Yang, X. S.** (2020). Nature-inspired optimization algorithms. Academic Press, Chapter:6: Genetical Algorithms (ss.90-100).
- Yao, B., Hu, P. Zhang, M. Tian, X.** (2014). Improved Ant Colony Optimization for Seafood Product Delivery Routing Problem. *Promet - Traffic&Transportation*, Vol. 26 No. 1, 1-10.
- Yıldırım, U.M.** (2008). Zaman-bağımsız ve zaman-bağımlı zaman kısıtlı araç rotalama probleminde bir karınca kolonisi yaklaşımı. Yayınlanmamış Yüksek Lisans Tezi. Sabancı Üniversitesi Mühendislik ve Fen Bilimleri Enstitüsü

- Yıldırım, T.** (2014). Simetrik gezgin satıcı problemi için yeni bir meta-sezgisel: kör fare algoritması (Master's thesis, Pamukkale Üniversitesi Fen Bilimleri Enstitüsü).(ss 21-37)
- Yıldırım B & Çebi S.** (2019) Kapasite Kısıtlı Araç Rotalama Problemi İçin Sezgisel Yöntemler: E-Ticaret Tedarikçilerine Yönelik Bir Uygulama. Uluslararası İktisadi ve İdari İncelemeler Dergisi, 185-206
- Yılmaz, Ş.** (2008). Çok depolu araç rotalama probleminin karınca kolonisi optimizasyonu ile modellenmesi ve bir çözüm önerisi (s.21).
- Yu, B., Yang, Z. Z., & Yao, B.** (2009). An improved ant colony optimization for vehicle routing problem. European journal of operational research, 196(1), 171-176.
- Yun, H. Y., Jeong, S. J., & Kim, K. S.** (2013). Advanced harmony search with ant colony optimization for solving the traveling salesman problem. Journal of applied mathematics, 2013.
- Yücel, O.** (2012). Ballistic design optimization of three-dimensional grains using genetic algorithms (Master's thesis) (ss. 58-66).
- Yücel, A.** (2016). Mesafe kısıtlı çok yönlü kümelenmiş açık araç rotalama probleminin genetik algoritma ile çözümü ve bir uygulama. Yayımlanmamış Doktora Lisans Tezi. Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara.
- Yüksel, H., Çalışkan, F., & Dayık, M.** (2019). Genetik Algoritmaların Tasarım Sürecinde Kullanılması. Teknik Bilimler Dergisi, 6(2).
- Zhang, T., Wang, S., Tian, W., & Zhang, Y.** (2006, October). ACO-VRPTWRV: A new algorithm for the vehicle routing problems with time windows and re-used vehicles based on ant colony optimization. In Sixth International Conference on Intelligent Systems Design and Applications (Vol. 1, pp. 390-395). IEEE.
- Zhang, G., Wang, H., Zhao, W. et al.** (2021). Application of Improved Multi-Objective Ant Colony Optimization Algorithm in Ship Weather Routing. J. Ocean Univ. China 20, 45–55. <https://doi.org/10.1007/s11802-021-4436-6>
- Zhao, F., Yao, Z., Luan, J., & Song, X.** (2016). A novel fused optimization algorithm of genetic algorithm and ant colony optimization. Mathematical Problems in Engineering, 2016.
- Zhao, T., Pan, X., & He, Q.** (2017, April). Application of dynamic ant colony algorithm in route planning for UAV. In 2017 Seventh International Conference on Information Science and Technology (ICIST) (pp. 433-437). IEEE.
- Zhao, W., Wang, Y., Zhang, Z., Wang, H.** (2021). Multicriteria Ship Route Planning Method Based on Improved Particle Swarm Optimization–Genetic Algorithm. J. Mar. Sci. Eng. 2021, 9, 357. <https://doi.org/10.3390/jmse9040357>

- Zhen, L., Ma, C., Wang, K., Xiao, L., & Zhang, W.** (2020). Multi-depot multi-trip vehicle routing problem with time windows and release dates. *Transportation Research Part E: Logistics and Transportation Review*, 135, 101866.
- Zhou, K., Wan, W., Chen, X., Shao, Z., & Biegler, L. T.** (2013). A parallel method with hybrid algorithms for mixed integer nonlinear programming. In *Computer Aided Chemical Engineering* (Vol. 32, pp. 271-276). Elsevier
- Zhu, K. Q.** (2003, November). A diversity-controlling adaptive genetic algorithm for the vehicle routing problem with time windows. In *Proceedings. 15th IEEE International Conference on Tools with Artificial Intelligence* (pp. 176-183). IEEE.



EKLER

EK A: Gurobi Verileri

Ek A1: Tuple list örneği

EK B: Hibrit Yöntem Verileri

Ek B1: Hibrit yöntem problem kodları



EK A

c = {(0, 0, 'ship1'): M,
(0, 1, 'ship1'): 1313865,
(0, 2, 'ship1'): 1317735,
(0, 3, 'ship1'): 1367735,
(0, 4, 'ship1'): 1460556,
(0, 5, 'ship1'): 1429364,
(0, 6, 'ship1'): 1479364,
(0, 7, 'ship1'): 1885850,
(0, 8, 'ship1'): 1807155,
(0, 9, 'ship1'): 1857155,
(0, 10, 'ship1'): 1799930,
(0, 11, 'ship1'): 1747640,
(0, 12, 'ship1'): 1797640,
(0, 13, 'ship1'): 2173140,
(0, 14, 'ship1'): 2087825,
(0, 15, 'ship1'): 2137825,
(1, 4, 'ship1'): 1615870,
(1, 5, 'ship1'): 1570360,
(1, 6, 'ship1'): 1623360,
(1, 7, 'ship1'): 1769490,
(1, 8, 'ship1'): 1720055,
(1, 9, 'ship1'): 1773055,
(1, 10, 'ship1'): 1915230,
(1, 11, 'ship1'): 1838740,
(1, 12, 'ship1'): 1891740,
(1, 13, 'ship1'): 2246320,
(1, 14, 'ship1'): 2161105,
(1, 15, 'ship1'): 2214105,
(1, 16, 'ship1'): 2414990,
(1, 17, 'ship1'): 2327035,
(1, 18, 'ship1'): 2380035,
(1, 19, 'ship1'): 2121390,
(1, 20, 'ship1'): 2036365,
(1, 21, 'ship1'): 2089365,
(1, 22, 'ship1'): 1620180,
(1, 23, 'ship1'): 1551990,
(1, 24, 'ship1'): 1604990,
(1, 25, 'ship1'): 1812250,
(1, 26, 'ship1'): 1760310,
(1, 27, 'ship1'): 1813310,
(1, 28, 'ship1'): 2088810,
(1, 29, 'ship1'): 2017380,
(1, 30, 'ship1'): 2070380,
(1, 31, 'ship1'): 1571650,
(1, 32, 'ship1'): 1552510,
(1, 33, 'ship1'): 1605510,
(1, 34, 'ship1'): 1491724.5,
(1, 35, 'ship1'): 1490580,

(1, 36, 'ship1'): 1543580,
(1, 37, 'ship1'): 1358957,
(1, 38, 'ship1'): 1329565,
(1, 39, 'ship1'): 1382565,
(1, 40, 'ship1'): 1778845,
(1, 41, 'ship1'): 1732020,
(1, 42, 'ship1'): 1785020,
(1, 43, 'ship1'): 1495610,
(1, 44, 'ship1'): 1488780,
(1, 45, 'ship1'): 1541780
(2, 0, 'ship1'): 1317735,
(2, 1, 'ship1'): M,
(2, 2, 'ship1'): M,
(2, 3, 'ship1'): M,
(2, 4, 'ship1'): 1605870,
(2, 5, 'ship1'): 1568360,
(2, 6, 'ship1'): 1618360,
(2, 7, 'ship1'): 1759490,
(2, 8, 'ship1'): 1718055,
(2, 9, 'ship1'): 1768055,
(2, 10, 'ship1'): 1905230,
(2, 11, 'ship1'): 1836740,
(2, 12, 'ship1'): 1886740,
(2, 13, 'ship1'): 2236320,
(2, 14, 'ship1'): 2159105,
(2, 15, 'ship1'): 2209105,
(2, 16, 'ship1'): 2404990,
(2, 17, 'ship1'): 2325035,
(2, 18, 'ship1'): 2375035,
(2, 19, 'ship1'): 2111390,
(2, 20, 'ship1'): 2034365,
(2, 21, 'ship1'): 2084365,
(2, 22, 'ship1'): 1610180,
(2, 23, 'ship1'): 1549990,
(2, 24, 'ship1'): 1599990,
(2, 25, 'ship1'): 1802250,
(2, 26, 'ship1'): 1758310,
(2, 27, 'ship1'): 1808310,
(2, 28, 'ship1'): 2078810,
(2, 29, 'ship1'): 2015380,
(2, 30, 'ship1'): 2065380,
(2, 31, 'ship1'): 1561650,
(2, 32, 'ship1'): 1550510,
(2, 33, 'ship1'): 1600510,
(4, 1, 'ship5'): M,
(4, 2, 'ship5'): M,
(4, 3, 'ship5'): M,
(4, 4, 'ship5'): M,
(4, 5, 'ship5'): M,
(4, 6, 'ship5'): M,

(4, 7, 'ship5'): M,
(4, 8, 'ship5'): M,
(4, 9, 'ship5'): M,
(4, 10, 'ship5'): 1788870,
(4, 11, 'ship5'): 1731820,
(4, 12, 'ship5'): 1784820,
(4, 13, 'ship5'): 2119960,
(4, 14, 'ship5'): 2036365,
(4, 15, 'ship5'): 2089365,
(4, 16, 'ship5'): 2393930,
(4, 17, 'ship5'): 2309215,
(4, 18, 'ship5'): 2362215,
(4, 19, 'ship5'): 1995030,
(4, 20, 'ship5'): 1929445,
(4, 21, 'ship5'): 1982445,
(4, 22, 'ship5'): 1408520,
(4, 23, 'ship5'): 1358150,
(4, 24, 'ship5'): 1411150,
(4, 25, 'ship5'): 1980730,
(4, 26, 'ship5'): 1920690,
(4, 27, 'ship5'): 1973690,
(4, 28, 'ship5'): 2257290,
(4, 29, 'ship5'): 2159940,
(4, 30, 'ship5'): 2212940,
(4, 31, 'ship5'): 1740130,
(4, 32, 'ship5'): 1695070,
(4, 33, 'ship5'): 1748070,
(4, 34, 'ship5'): 1723384.5,
(4, 35, 'ship5'): 1686600,
(4, 36, 'ship5'): 1739600,
(4, 37, 'ship5'): 1147297,
(4, 38, 'ship5'): 1135725,
(4, 39, 'ship5'): 1188725,
(4, 40, 'ship5'): 1575194.8,
(4, 41, 'ship5'): 1542682.5,
(4, 42, 'ship5'): 1595682.5,
(4, 43, 'ship5'): 1706210,
(4, 44, 'ship5'): 1666980,
(4, 45, 'ship5'): 1719980,

Ek A1

Gurobi Optimizer version 9.1.2 build v9.1.2rc0 (win64)

Thread count: 8 physical cores, 8 logical processors, using up to 8 threads

Optimize a model with 250 rows, 10580 columns and 30600 nonzeros

Model fingerprint: 0x5db7960b

Variable types: 0 continuous, 10580 integer (10580 binary)

Coefficient statistics:

Matrix range [1e+00, 1e+00]

Objective range [1e+06, 1e+16]

Bounds range [1e+00, 1e+00]

RHS range [1e+00, 1e+00]

Warning: Model contains large objective coefficients

Consider reformulating model or setting NumericFocus parameter to avoid numerical issues.

Presolve removed 0 rows and 230 columns

Presolved: 250 rows, 10350 columns, 30375 nonzeros

Variable types: 0 continuous, 10350 integer (10350 binary)

	Nodes		Current Node		Objective Bounds		Work			
	Expl	Unexpl	Obj	Depth	IntInf	Incumbent	BestBd	Gap	It/Node	Time

*	0	0		0		3.004289e+07	3.0043e+07	0.00%	-	0s
---	---	---	--	---	--	--------------	------------	-------	---	----

Explored 0 nodes (463 simplex iterations) in 0.05 seconds

Thread count was 8 (of 8 available processors)

Solution count 1: 3.00429e+07

Optimal solution found (tolerance 1.00e-04)

Best objective 3.004288920000e+07, best bound 3.004288920000e+07, gap 0.0000%

Optimization complete

Solution found, objective = **30042889.2**

Routes:

x [0,5, ship1] 1

x [5,23, ship1] 1

x [23,29, ship1] 1

x [29,32, ship1] 1

x [32,0, ship1] 1

x [0,11, ship2] 1

x [11,14, ship2] 1

x [14,17, ship2] 1

x [17,20, ship2] 1

x [20,38, ship2] 1

x [38,41, ship2] 1

x[41,44,ship2] 1
x [44,0, ship2] 1

x [0,26, ship3] 1
x [26,35, ship3] 1
x [35,0, ship3] 1

x [0,8, ship4] 1
x [8,0, ship4] 1

x [0,1, ship5] 1
x [1,0, ship5] 1



EK B

Ek B1

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Collections;

namespace Suendam
{

    public partial class Form1 : Form
    {
        private int m_population, m_customer, iteration, acotype, functiontype,
m_vessel;
        public double m_evaporation, m_alpha, m_beta, m_crossoverprob,
m_mutationprob;

        public static readonly Random RastgeleDeger = new Random();
        public double[,] m_distance, m_dromon, m_demand, m_vesseltypecoefficient,
m_vesselfirstjob;
        public string yazdirma;
        public Genant[] genant;
        public Genant bestgenant;

        public Form1()
        {
            InitializeComponent();
            textBoxCustomer.Text = "10";
            textBoxPopulation.Text = "10";
            textBoxBuharlasma.Text = "0,9";
            textBoxBeta.Text = "1";
            textBoxAlpha.Text = "1";
            textBoxiterasyon.Text = "100";
            textBoxVessel.Text = "2";
            textBoxCrossoverProb.Text = "0,4";
            textBoxMutationProb.Text = "0,6";
```

```

//ComboBox seçimi
comboBoxCrossover.SelectedIndex = 0;
comboBoxMutation.SelectedIndex = 0;
comboBoxSelection.SelectedIndex = 0;

}

private void Form1_Load(object sender, EventArgs e)
{

}

private void button1_Click(object sender, EventArgs e)
{
    m_population = Convert.ToInt32(textBoxPopulation.Text);
    m_customer = Convert.ToInt32(textBoxCustomer.Text);
    m_alpha = Convert.ToDouble(textBoxAlpha.Text);
    m_beta = Convert.ToDouble(textBoxBeta.Text);
    m_evaporation = Convert.ToDouble(textBoxBuharlasma.Text);
    iteration = Convert.ToInt32(textBoxiterasyon.Text);
    m_vessel = Convert.ToInt32(textBoxVessel.Text);
    m_crossoverprob = Convert.ToDouble(textBoxCrossoverProb.Text);
    m_mutationprob = Convert.ToDouble(textBoxMutationProb.Text);

    m_distance = new double [,]      {{ 100, 3, 4, 5, 6, 7, 8, 9, 10.0, 10 },
                                      { 100, 3, 4, 5, 6, 7, 8, 9, 10.0, 10 },
                                      { 100, 100, 4, 5, 6, 7, 8, 9, 10.0, 10 },
                                      { 100, 100, 4, 5, 6, 7, 8, 9, 10.0, 10 },
                                      { 100, 100, 100, 100, 6, 7, 8, 9, 10, 11 },
                                      { 100, 100, 100, 100, 8, 9, 10, 9, 10, 11 },
                                      { 100, 100, 100, 100, 100, 100, 8, 9, 10, 11 },
                                      { 100, 100, 100, 100, 100, 100, 8, 9, 10, 11 },
                                      { 100, 100, 100, 100, 100, 100, 100, 10, 11 },
                                      { 100, 100, 100, 100, 100, 100, 100, 10, 11 }
                                      };

    m_vesseltypecoefficient = new double[,] {{ 110, 120, 140, 150, 160, 170,
180, 190, 100, 120 },
                                             { 105, 125, 125, 175, 165, 180, 185, 175, 110, 125
},
                                             };

    m_vesselfirstjob = new double[,]      {{ 110, 120, 140, 150, 160, 170, 180,
190, 100, 120 },
                                             { 105, 125, 125, 175, 165, 180, 185, 175, 110, 125
},
                                             };

```

```

        getstartgeneticant();

        richTextBoxSonuc.Text = yazdirma;

    }

    private void getstartgeneticant()
    {
        // İlk atama yapılıyor.
        genant = new Genant[m_population];
        bestgenant = new Genant(m_customer, m_vessel);
        bestnullrunaway(bestgenant, m_vessel);
        bestgenant.totalcost = 1111111111111111;

        m_dromon = new double[m_customer, m_customer];

        for (int i = 0; i < m_population; i++)
        {
            genant[i] = new Genant(m_customer, m_vessel);
            genant[i].firstassignment(genant[i]);
        }

        pheromon(m_dromon);

        //İterasyon başlıyor....
        for (int ite = 0; ite < iteration; ite++)
        {
            for (int g = 0; g < m_population; g++)
            {
                genant[g].vesselrouting(genant,g,m_dromon);
                genant[g].vesseltotalcost(genant, g, m_vesseltypecoefficient,
m_distance,m_vesselfirstjob);
                if (genant[g].totalcost < bestgenant.totalcost)
                {
                    bestgenrecord(bestgenant, genant, g);
                }
            }
        }

        geneticprocess(genant, m_mutationprob, m_crossoverprob, m_population);

        for (int fero1 = 0; fero1 < m_customer; fero1++)
        {
            for (int fero2 = 0; fero2 < m_customer; fero2++)
            {
                m_dromon[fero1, fero2] *= m_evaporation;
            }
        }
    }
}

```

```

    }
}

allphremonupdates(genant, m_dromon);

yazdirma += yazdirmaphase(bestgenant, ite);
}

}

private string yazdirmaphase(Genant bestgenant, int ite)
{
    string yazdir;
    if (iteration == 0)
    {
        yazdir = "----- 1. Iteration Results ----- " + "\n";

        for (int i = 0; i < m_vessel; i++)
        {
            yazdir += i + 1 + ". Vessel Route: ";

            for (int r = 0; r < bestgenant.gemi[i].routequencevessel.Count; r++)
            {
                yazdir += bestgenant.gemi[i].routequencevessel[r] + ", ";
            }
            yazdir += "\n";

        }
        yazdir += "Total Distance: " + bestgenant.totalcost + "\n";
    }
    else
    {
        yazdir = "-----" + (ite + 1) + ". Iteration Results ----- " + "\n";

        for (int i = 0; i < m_vessel; i++)
        {
            yazdir += i + 1 + ". Vessel Route: ";

            for (int r = 0; r < bestgenant.gemi[i].routequencevessel.Count; r++)
            {
                yazdir += bestgenant.gemi[i].routequencevessel[r] + ", ";
            }
            yazdir += "\n";

            yazdir += "Total Distance: " + bestgenant.totalcost + "\n";
        }
    }
}

```

```

    }
}
return yazdir;
}

private void allphremonupdates(Genant[] genant, double[,] m_dromon)
{
    for (int pop = 0; pop < m_population; pop++)
    {
        for (int tr = 0; tr < m_vessel; tr++)
        {
            for (int i = 1; i < genant[pop].gemi[tr].routequencevessel.Count-1;
i++)
            {
                int ilk = Convert.ToInt32(genant[pop].gemi[tr].routequencevessel[i
- 1]);
                int son =
Convert.ToInt32(genant[pop].gemi[tr].routequencevessel[i]);
                m_dromon[ilk, son] += 1 / genant[pop].totalcost;
            }
        }
    }
}

```

```

private void bestgenrecord(Genant bestgenant, Genant[] genant, int g)
{
    bestnullrunaway(bestgenant, m_vessel);
    for (int p = 0; p < m_customer; p++)
    {
        bestgenant.gene[p] = genant[g].gene[p];
    }
    for (int v = 0; v < m_vessel; v++)
    {
        for (int c = 0; c < genant[g].gemi[v].routequencevessel.Count; c++)
        {
            bestgenant.gemi[v].routequencevessel.Add(Convert.ToInt32(
genant[g].gemi[v].routequencevessel[c]));
        }
    }
    bestgenant.totalcost = genant[g].totalcost;
}

```

```

private void bestnullrunaway(Genant bestgenant, int m_vessel)
{
    for (int i = 0; i < m_vessel; i++)

```

```

    {
        bestgenant.gemi[i].routessequencevessel = new ArrayList();
        bestgenant.gemi[i].restroutes = new ArrayList();
    }
}

private void geneticprocess(Genant[] genant, double mutationprob, double
crossoverprob, int population)
{
    int nummutation = Convert.ToInt32(Math.Round(population *
mutationprob));
    int numcrossover = Convert.ToInt32(Math.Round(population *
mutationprob));
    double allcost = 0;
    double[] selectionprob = new double[m_population];

    for (int k = 0; k < population; k++)
    {
        for (int j = k+1; j < population; j++)
        {
            if (genant[k].totalcost > genant[j].totalcost)
            {
                double totalcost1 = 0;
                int[] contemporary = new int[m_customer];

                for (int t = 0; t < m_customer; t++)
                    contemporary[t] = genant[k].gene[t];
                totalcost1 = genant[k].totalcost;

                for (int c = 0; c < m_customer; c++)
                    genant[k].gene[c] = genant[j].gene[c];
                genant[k].totalcost = genant[j].totalcost;

                for (int d = 0; d < m_customer; d++)
                    genant[j].gene[d] = contemporary[d];
                genant[j].totalcost = totalcost1;
            }
        }
    }

    for (int i = 0; i < m_population; i++)
        allcost += 1 / genant[i].totalcost;

    for (int k = 0; k < m_population; k++)
        selectionprob[k] = (1 / genant[k].totalcost) / allcost;

    int[,] offspring = new int[(2 * numcrossover), m_customer];

    int[] pick = new int[2];

```

```

//Crossover sayısı kadar çaprazlama olacak
for (int t = 0; t < numcrossover; t++)
{
    //Genler kopyalanacak
    int[] copygene1 = new int[m_customer];
    int[] copygene2 = new int[m_customer];

    for (int l = 0; l < 2; l++)
    {
        double offset = 0;
        double rdn = RastgeleDeger.NextDouble();
        for (int j = 0; j < m_population; j++)
        {
            offset += selectionprob[j];
            if (rdn < offset)
            {
                pick[l] = j;
                break; //Seçilen genlerin belirlenmesi
            }
        }
    }
    //Kopyalanan genler

    for (int f1 = 0; f1 < m_customer; f1++)
    {
        copygene1[f1] = genant[pick[0]].gene[f1];
        copygene2[f1] = genant[pick[1]].gene[f1];
    }
    //
    int randomlocation = 0;
    randomlocation = RastgeleDeger.Next(m_customer);

    for (int f2 = randomlocation; f2 < m_customer; f2++)
    {
        copygene2[f2] = genant[pick[0]].gene[f2];
        copygene1[f2] = genant[pick[1]].gene[f2];
    }

    for (int f3 = 0; f3 < m_customer; f3++)
    {
        offspring[2 * t, f3] = copygene1[f3];
        offspring[2 * t + 1, f3] = copygene2[f3];
    }
}

```

```

for (int t1 = 0; t1 < numcrossover; t1++)
{
    for (int f4 = 0; f4 < m_customer; f4++)
    {
        genant[t1].gene[f4] = offspring[t1, f4];
        genant[t1 + 1].gene[f4] = offspring[t1 + 1, f4];

    }
}

// buraya mutasyon girilecek
for (int t2 = 0; t2 < nummutation; t2++)
{
    int randomcustomer = 0;
    randomcustomer = RastgeleDeger.Next(0, m_population - 2);
    int randomgene = 0;
    randomgene = RastgeleDeger.Next(m_customer);

    double randomnum = RastgeleDeger.NextDouble();

    int atama = -1;

    do
    {
        atama = RastgeleDeger.Next(m_vessel);
    } while (genant[randomcustomer].gene[randomgene] != atama);

    genant[randomcustomer].gene[randomgene] = atama;

}

}

private void pheromon(double[,] m_dromon)
{
    for (int i = 0; i < m_customer; i++)
    {
        for (int j = 0; j < m_customer; j++)
        {
            m_dromon[i, j] = 1;
        }
    }
}
}
}

```

```

using System;
using System.Collections.Generic;

```

```

using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Collections;

namespace Suendam
{
    public class Genant
    {
        public int customer, vessel;
        public int[] gene;
        public vessel[] gemi;
        public double totalcost;
        public static readonly Random RastgeleDeger = new Random();

        public Genant(int cust, int ves)
        {
            // TODO: Complete member initialization
            this.customer = cust;
            this.vessel = ves;
            gene = new int[customer];
            gemi = new vessel[vessel];

            for (int i = 0; i < vessel; i++)
            {
                gemi[i] = new vessel();
            }
        }

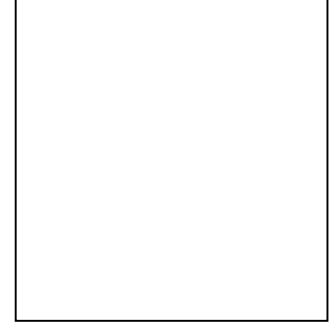
        internal void firstassignment(Genant genant)
        {
            for (int k = 0; k < customer; k++)
            {
                genant.gene[k] = Convert.ToInt32(RastgeleDeger.Next(vessel));
            }
        }

        internal void vesselrouting(Genant[] genant, int g, double[,] dromon)
        {
            for (int i = 0; i < vessel; i++)

```



ÖZGEÇMİŞ



Ad-Soyad : Seher Suendam Arıcı

ÖĞRENİM DURUMU:

- **Lisans** : 2016, Piri Reis Üniversitesi, Denizcilik Fakültesi, Deniz Ulaştırma İşletme Mühendisliği
- **Lisans** : 2021, Anadolu Üniversitesi, İşletme Fakültesi, Uluslararası Ticaret ve Lojistik Yönetimi
- **Tezsiz Yüksek Lisans**: 2019, Piri Reis Üniversitesi, Sosyal Bilimler Enstitüsü, Lojistik ve Tedarik Zinciri Yönetimi

MESLEKİ DENEYİM VE ÖDÜLLER:

- 2016-2018 yılları arasında kimyasal yük taşıyan gemilerde uzakyol vardiya zabiti olarak çalıştı.
- 2019 yılından itibaren, İstanbul Teknik Üniversitesi, Denizcilik Fakültesinde Araştırma Görevlisi olarak görev almakta.

DİĞER YAYINLAR, SUNUMLAR VE PATENTLER:

- **Arıcı, S. S., Akyuz, E., & Arslan, O.** (2020). Application of fuzzy bow-tie risk analysis to maritime transportation: The case of ship collision during the STS operation. *Ocean Engineering*, 217, 107960.