TOBB EKONOMİ VE TEKNOLOJİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

TAM KAMYON YÜKÜ GÖNDERİCİ İŞ BİRLİĞİNDE KARARLI KOALİSYON SEÇİMİ

DOKTORA TEZİ

Nihat ÖNER

Endüstri Mühendisliği Anabilim Dalı

Tez Danışmanı: Dr. Öğr. Üyesi Gültekin KUYZU

26 Kasım 2021

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, alıntı yapılan kaynaklara eksiksiz atıf yapıldığını, referansların tam olarak belirtildiğini ve ayrıca bu tezin TOBB ETÜ Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırlandığını bildiririm.

Nihat ÖNER

İMZA

ÖZET

Doktora Tezi

TAM KAMYON YÜKÜ GÖNDERİCİ İŞ BİRLİĞİNDE KARARLI KOALİSYON SEÇİMİ

Nihat ÖNER

TOBB Ekonomi ve Teknoloji Üniversitesi
Fen Bilimleri Enstitüsü
Endüstri Mühendisliği Anabilim Dalı

Tez Danışmanı: Dr. Öğr. Üyesi Gültekin KUYZU

Tarih: 26 Kasım 2021

Tam kamyon yükü gönderici iş birliği ağlarında; katılımcıların seçimi, kimin kimle iş birliği yapacağının belirlenmesi, iş birliğinden elde edilen toplam maliyetin hesaplanması ve edinilen maliyetin katılımcılar arasında paylaşılması önemli problemler olarak öne çıkmaktadır. Tam kamyon yükü gönderici iş birliğinde kazanımlar maliyetlerin azaltılması şeklinde olduğu için göndericiler en düşük maliyetli iş birlikli çözüm arayışı içerisindedir. Toplam maliyetin en aza indirilmesinin yanında bu maliyetin iş birliğinde katılan firmalara dağıtılması ile her firmanın tasarruf miktarı ortaya çıkar. Dağıtılan bu maliyetin firmalar tarafından kabul edilebilir olmaması durumunda iş birliğinin dağılma riski vardır. Kararlı bir maliyet dağıtımında, firmaların koalisyondan koparak; alt koalisyon oluşturmasına izin verilmeyeceği için iş birliğinin dağılması riski söz konusu değildir. Bu nedenle, kararlı maliyet dağıtımı hesaplayan bir mekanizmaya ihtiyaç vardır. Literatürdeki çalışmalarda, toplam maliyeti en küçükleyen en iyileme problemi ve en küçük maliyetin dağıtımı birbirini izleyen iki ayrı aşama olarak ele alınmıştır. Bu çalışmada ilk olarak, problem ardışık olarak ele alındığında; kararlı bir maliyet dağıtımın

bulunması için yeter ve gerek koşulların neler oldukları tanımlanmıştır. Bu koşullar sonucunda böyle bir maliyet dağıtımının bulunmasının oldukça güç olduğu gösterilmiştir. Bu tez kapsamında, bugüne kadar iki ayrı yaklaşım olarak ele alınan; en düşük maliyetli iş birliği çözümünü hesaplayan en iyileme problemi ve maliyet dağıtım problemi birleştirilerek; kararlı maliyet dağıtımına sahip en düşük maliyetli iş birliğini hesaplayan karma tam sayılı programlama formülasyonu önerilmiştir. Önerilen bu modeli çözmek için kesin ve sezgisel yöntemler geliştirilmiştir. Ayrıca bu tez kapsamında nükleolus (çekirdekçik) temelli maliyet dağıtım yöntemleri de geliştirilmiştir. Bu yöntemlere ek olarak, kararlı maliyet dağıtımları bulmak için iki farklı yöntem de önerilmiştir. Geliştirilen bütün yöntemler rastgele üretilen örnekler üzerinde test edilerek performansları değerlendirilmiştir. Bu yöntemler aynı zamanda literatürde yer alan bazı maliyet dağıtımlarıyla da karşılaştırılmıştır. Ayrıca geliştirilen çözüm yöntemlerinin kararlı bir maliyet dağıtımı verdiği, literatürde kullanılan klasik yaklaşımdan kararlı bir maliyet dağıtımı elde edilemeyeceği ve geliştirilen maliyet dağıtım metotlarının literatürde yer alan maliyet dağıtımlarından çok daha iyi sonuç verdiği gösterilmiştir.

Anahtar Kelimeler: Karma tam sayılı programlama, Satır ve sütün üretme, İşbirlikçi oyun kuramı, Kararlılık ve kararlı maliyet dağıtımı, Dal-fiyat ve kesi yöntemi, Sezgisel yöntemler.

ABSTRACT

Doctor of Philosophy

Stable Coalition Selection in Collaborative Truckload Transportation Procurement

Nihat ONER

TOBB University of Economics and Technology
Institute of Natural and Applied Sciences
Department of Industrial Engineering

Supervisor: Asst.Prof.Dr. Gultekin KUYZU

Date: 26 November 2021

In truckload shipper collaboration, a group of shippers purchasing the services of carriers come together and try to find tours which consist of regularly scheduled shipments with minimal empty truck movements with the aim of getting better rates from the carriers in return. Finding the best set of such tours is a challenging optimization problem, the solution of which yields the set of lanes and firms to be included in the collaboration, who will collaborate with whom, and the maximum amount of savings which can be achieved. Allocation of the total calculated cost to individual lanes determines the final savings for each participant. If the allocated costs are not accepted by the participants, the collaboration will face the risk of collapse. A stable cost allocation ensures that no subcoalition is better off by breaking away from the coalition. Therefore, it is crucial to develop such cost allocation. In the literature, minimizing the total cost and allocating the minimized cost are considered as two successive but distinct phases. In this research, we first present necessary and sufficient conditions for the problem we consider to have stable cost allocation. We then show that it could not be possible to find such allocation if the distinct-phase approach is used. Because of that, we

develop mixed integer programming model which combines these distinct phases into together and yields both minimum cost and stable cost allocation. Besides we develop a branch-price and cut algorithm which combines row generation, column generation, and an upper bounding heuristic to find a stable cost allocation. In addition, we develop cost allocation methods based on nucleolus in order to find a stable cost allocation. We evaluate the performance of our solution procedures by generating instances randomly with specific rules. We show that our solution procedures provide stable cost allocation. Moreover, we also show that our proposed methods based on nucleolus yield better solutions in terms of stability. Furthermore we propose two additional cost allocation methods which aim to find a stable cost allocation.

Keywords: Mixed integer programming, Row and column generation, Cooperative game theory, Stability and stable cost allocation, Branch-price and cut algorithm, Heuristics.

TEŞEKKÜR

Doktora hayatım boyunca karşılaştığım her türlü zorlukta benim yanımda olan, bana destek veren, en moralsiz anlarımda bana yoldaş olup beni neşelendiren, doktorama olan inancımı sürekli taze tutan, kendi iş yoğunluğunu bir kenara bırakarak beni her şeyin önüne koyan, bana inanmaktan hiçbir zaman vazgeçmeyen sevgili eşime; can yoldaşıma çok ama çok teşekkür ederim. Şunu söylemek isterim ki: eğer eşimin destekleri olmasaydı belki de bu doktorayı bitiremezdim. Bu doktoranın ortaya çıkmasında benim ve hocalarım kadar eşimin de katkısı çok büyüktür. Ayrıca anneme, babama, kardeşime ve amcalarıma da çok teşekkür ederim. Bana sürekli destek oldular. Hiç yalnız bırakmadılar. Bana olan inançlarını hiçbir zaman kaybetmediler. Beni sürekli desteklediler. Bir özel teşekkürü de tez danışmanıma yapmak istiyorum. Doktoram boyunca beni yönlendiren, bana yüksek lisansımdan bu zamana kadar kendisiyle çalışma fırsatı sunan tez danışmanım Gültekin KUYZU 'ya teşekkürü bir borç bilirim. Değerli vakitlerini ve fikirlerini dokuz tez izleme komitesi boyunca benimle paylaşan beni sürekli motive eden Okan Örsan ÖZENER 'e ve Ayşegül ALTIN KAYHAN 'a hocalarıma çok teşekkür ederim. Ayrıca tez jürimde olmayı kabul eden değerli fikirlerini benimle paylaşan; Mustafa Alp ERTEM 'e, Salih TEKİN 'e, Niyazi Onur BAKIR 'a, Nilgün FESCİOĞLU ÜNVER 'e ve Çağrı KOÇ 'a çok teşekkür ederim. Doktoram sırasında kendisiyle çalışma fırsatı bulduğum, beni motive eden, değerli fikirleriyle beni yönlendiren, beni farklı alanlarda çalışmaya teşvik eden değerli hocam Hakan GÜLTEKİN 'e teşekkür ederim. Doktoram boyunca bana desteklerini esirgemeyen sevgili dostlarım Uğur GELİR 'e, Muhammed Emin DEMİRAL 'a, Beysin Burak BAYSAL 'a, Onur ÜŞENMEZ 'e, Orkun EMİRALİOĞLU 'na, Şenol ATAÇ 'a, Evten OLCAYTU 'ya, Nusret Semih ÇELİK 'e ve Şadi ELAŞKAR 'a; değerli abilerim Celalettin KELEŞ 'e, Volkan AKYÜZ 'e, Abdullah KOCAMAN 'a, Ünal PEHLİVAN 'a, Murat ÇETİN 'e ve İbrahim ERDEM 'e çok teşekkür ederim. Son olarak, ilkokuldan doktora hayatıma kadar geçen sürede üzerimde emeği olan bütün hocalarıma, bana bu imkanları sağlayan üniversiteme ve bölümüme canı gönülden teşekkür ederim.

İÇİNDEKİLER

ŞEKİL LİSTESİ

Sayfa

TABLO LİSTESİ

Sayfa

ALGORİTMA LİSTESİ

Sayfa

KISALTMALAR

AS : Alt Sınır
Ort : Ortalama
Min : Minimum
Mak : Maksimum
İS : İşlem Süresi
İM : İhlal Miktarı
KK : Kaba Kuvvet
ZL : Zaman Limiti
ÇS : Çözüm Süresi
SD : Shapley Değeri
OİO : Ortalama İhlal Oranı
MİO : Maksimum İhlal Oranı
FP : Fiyatlandırma Problemi
RK : Rota Kapsama Problemi
DFK : Dal-Fiyat ve Kesi Yöntemi
STA : Satır Türetme Alt Problemi
MTİO : Minimum Toplam İhlal Oranı
OBS : Oransal Bazlı Sezgisel Yöntem
MMİO : Minimum Maksimum İhlal Oranı
UOMD : Uzaklık Orantılı Maliyet Dağıtımı
SSTS : Satır ve Sütun Türetme Sezgiselleri
SUKRK : Sayı ve Uzunluk Kısıtlı Rota Kapsama Problemi
DG-SUKRK : Sayı ve Uzunluk Kısıtlı Rota Kapsama Probleminin Doğrusal Gevşetmesi
MD-SUKRK : Maliyet Dağıtımlı Sayı ve Uzunluk Kısıtlı Rota Kapsama Problemi
ÇKMD-SUKRK : Çok Koalisyonlu Maliyet Dağıtımlı Sayı ve Uzunluk Kısıtlı Rota Kapsama Problemi

1. GİRİŞ

Gönderici iş birliğinde, göndericiler taşımacılık hizmeti almak için taşıyıcı firmalarla bir araya gelerek grup halinde pazarlık yaparlar. Taşımacılık sektöründe, gerek gönderici firmalar gerekse de taşıyıcı firmalar kendi operasyonel faaliyetlerini nasıl daha etkin ve düşük maliyetli yapabilirim sorularını bireysel olarak cevaplamaya çalışırlar. Ama günümüzde değişen rekabet koşulları, kaynak yetersizliği, güvenlik sorunları gibi etkenler yüzünden firmalar bu geleneksel yaklaşımlar yerine farklı yaklaşımlar aramaya başlamışlardır. Bu farklı yaklaşımlardan biri de firmalar arasında iş birliği oluşturma fikridir.

Tam kamyon yükü gönderici iş birliği ağları, üçüncü ve dördüncü parti lojistik hizmeti veren firmaların öncülüğünde ortaya çıkmıştır. Bu firmalar, göndericilere iş birliği fikrini anlatmış; iş birliği yapmak isteyen gönderici firmaları bir araya getirerek iş birliğinin kurulmasını sağlamıştır. Bu firmalar özellikle sürekli kamyon hareketlerinin bulunması ve bunun sonucunda maliyetlerin azalması konusunda önemli rol oynamaktadır. A.B.D. 'de bulunan *Transplace* ve *Nistevo* ve Avrupa 'da bulunan *Schenker* ve *Celexor* firmaları bu tarz firmalara örnektir.

Tam kamyon yükü gönderici iş birliği, göndericilere olduğu kadar taşıyıcı firmalara da avantaj sağlar. İş birliği yapan göndericiler boş kamyon yükü hareketlerini azalttıkları için taşıyıcı firmaların maliyetleri de azalmaktadır. Turların düzenli tekrarlanması sebebiyle sürücü çevrim oranı azalmakta, gönderilerinin yönetimi de kolaylaşmaktadır.

Tam kamyon yükü gönderici iş birliği son yıllarda ortaya çıkmış yeni bir iş birliği türüdür. Tam kamyon yükü gönderici iş birliğinde kurumlar arasında genellikle yatay iş birliği kullanılır. Aynı tedarik zinciri içerisinde yer alan iki veya daha fazla firmanın; aynı amaca ulaşmak için kendi aralarında yaptıkları iş birliğine

yatay iş birliği denir.

Tam kamyon yükü gönderici iş birliğinde kullanılan yatay iş birliğinin amacı toplam maliyeti azaltmak olduğu için, toplam getiri miktarını en büyüklemek toplam maliyeti en küçüklemeye eş değerdir. Bu nedenle, bu çalışma kapsamında toplam maliyetin en küçüklenmesi amaçlanmıştır.

Tam kamyon yükü gönderici iş birliği, yatay iş birliğinin bir türü olan grup satın almadan farklıdır. Grup satın almada *ölçek ekonomisi* kullanılırken; tam kamyon yükü gönderici iş birliğinde *kapsam ekonomisi* kullanılır. Bu nedenle, iş birlikçilerin rotalarının sayısından çok; bu rotaların birbirlerini ne kadar iyi tamamladığı (ikame ettiği) kazanılan fayda açısından önemlidir.

Göndericiler, iş birliği yapmak istediklerinde; hangi göndericilerin iş birliğine dahil edileceği, hangi göndericilerin rotalarının arka arkaya ekleneceği ve oluşturulan rota birleştirme çözümünden doğan toplam maliyetin göndericilere veya rotalara dağıtılması konularında en iyi kararı vermek zorundadır. Gerçek hayat uygulamalarında bu kararlar iş birliği ağının koordinasyonunu üstlenen firma tarafından verilmektedir.

İş birliğine dahil edilecek göndericilerin seçimi, genelde rotalama çözümüne bırakılır. Burada, çok sayıda ve çeşitli sektörlerden düzenli gönderi rotası bir araya getirilir. Burada amaç, birbirini tamamlayan rotaları bulma olasılığını mümkün olduğu kadar yüksek tutmaktır. Sonrasında, toplam maliyeti en aza indirmek amacıyla hangi düzenli gönderi rotalarının arka arkaya eklenerek turlar oluşturacağına karar verilir. Rota birleştirme çözümünde, rotalarının hiçbiri diğer göndericilerin rotaları ile birleştirilmeyen göndericiler iş birliği dışında kaldığı varsayılacaktır. Ağdaki üye sayısı ve hesaba katılacak rota ayrıtı sayısı arttıkça değerlendirilecek muhtemel tur sayısı üstel şekilde artacaktır.

Düzenli gönderi turlarından oluşan çözümün toplam maliyetiyle firmaların başlangıçtaki bireysel maliyetlerinin toplamı arasındaki fark, iş birliğinden elde edilecek toplam maliyet kazanımını verir. Ancak, her iş birlikçi firmanın kendi maliyet kazanımının belirlenmesi için toplam maliyetin firmalara dağıtılması gerekmektedir. Maliyet dağıtımı, var olan yaklaşımlarda rota birleştirme çözümünü takip eden ayrı bir aşamadır. Tam kamyon yükü gönderici iş birliğinde cevaplanması gereken iki temel soru vardır:

1. İş birliğinden elde edilecek toplam maliyet nasıl hesaplanır?
2. İş birliğinden elde edilen toplam maliyet, göndericiler arasında en iyi şekilde nasıl paylaştırılır?

Literatürde iş birliği üzerine yapılan çalışmalarda; iş birliğinden elde edilen maliyetin en aza indirilmesi, iş birliğine katılacak iş birlikçilerin seçimi ve elde edilen maliyetin iş birlikçiler arasında paylaştırılması ayrı olarak ele alınmaktadır. Yani yukarıda verilen bu iki soru, iki ayrı aşama olarak değerlendirilmektedir. İlk olarak, bir matematiksel model yardımıyla minimum maliyet belirlenir. Daha sonra, çeşitli maliyet dağıtım yöntemleri veya matematiksel modellerle; minimum maliyet katılımcılar arasında dağıtılır.

Bu iki aşamanın birlikte değerlendirilmesi; ayrı ayrı değerlendirilmesine kıyasla daha avantajlıdır. Çok aşamalı bu yaklaşımın en büyük dezavantajı, en küçük maliyetli çözümün bir veya birkaç iş birlikçiye daha fazla avantaj sağlamasıdır. Böyle bir durumda da kararlı bir maliyet dağıtımının bulunması zorlaşacaktır. Bu aşamalar birlikte değerlendirildiğinde, *kararlı* bir maliyet dağıtımının bulunması daha kolay olacaktır. Ayrıca, *Rota Kapsama* problemlerinden doğan oyunlarda; *kararlı* ve *bütçe dengeli* maliyet dağıtımlarının bulunması için gerekli ve yeterli şartın; oyunu tanımlayan tam sayılı modelin gevşetilmiş çözümünün, tam sayılı çözüm

olmasıdır. Bununla ilgili önermeler ve bu önermelerin ispatları ileride detaylı olarak açıklanacaktır.

Tam kamyon yükü gönderici iş birliğinde altta yatan en iyileme modeli *Rota Kapsama (RK)* (*Lane Covering*) problemidir. *RK* problemi; rotaların kümesi verilmiş iken, bütün rotaları kapsayan en düşük maliyetli turların (çevrimlerin) kümesini bulma problemidir. *RK* problemine çeşitli kısıtlar eklenerek, problemin farklı türleri oluşturulabilir. Örneğin, göndericilerin iş birliği yapabileceği firma sayısı kısıtlandığı durumda ortaya çıkan *Ortak Kısıtlı Rota Kapsama* problemi bu türlerden birisidir. *RK* probleminin diğer türlerinden biri de, çevrimler içerisinde yer alan rota ayrıtları sayısı kısıtlandığında ortaya çıkan *Sayı Kısıtlı Rota Kapsama* problemidir. Bu türlerin dışında, çevrimlerin uzunlukları sınırlandığında ortaya çıkan *Uzunluk Kısıtlı Rota Kapsama* problemi de mevcuttur.

Bu tez çalışmasında, ayrık aşamalar birlikte değerlendirilerek; *kararlı* bir koalisyon yapısı ve *kararlı* bir maliyet dağıtımı bulunmaya çalışılacaktır. Geliştirilen bir karma tam sayılı programlama modeli yardımıyla, en düşük maliyetli çözüm hesaplanarak; hangi rota ayrıtlarının iş birliğine dahil edileceğine karar verilecek ve seçilen bu rota ayrıtlarına düşen maliyet belirlenerek *kararlı* bir koalisyon yapısı oluşturulacaktır.

Gönderici iş birliğinde, *kararlı* bir maliyet dağıtımının bulunması; iş birliğinin sürdürülebilmesi için oldukça önemlidir. Maliyet dağıtım aşamasında sıklıkla işbirlikçi oyun kuramına başvurulur. Bu nedenle *kararlı* bir maliyet dağıtımı geliştirmek için işbirlikçi oyun kuramında yer alan bazı kavramlar üzerine çalışılmıştır.

Bu tez kapsamında; *Sayı ve Uzunluk Kısıtlı Rota Kapsama (SUKRK)* problemine maliyet dağıtım kararları eklenerek elde edilen, *Maliyet Dağıtımlı Sayı ve Uzunluk Kısıtlı Rota Kapsama (MD-SUKRK)* problemi dikkate alınmıştır. Literatürde

yer alan kısıtları içeren, amacı *kararlı* maliyet dağıtımlı en düşük toplam maliyeti hesaplamak olan bir karma tam sayılı matematiksel model tanımlanmış ve bu model için kesin ve sezgisel çözüm yöntemleri geliştirilmiştir.

MD-SUKRK problemi için bütün olurlu çevrimleri üretmek yerine, satır ve sütun türetme yönteminden elde edilen çevrimler kullanılacaktır. Böyle bir durumda üretilecek çevrim sayısı ciddi şekilde azalacaktır. Ancak problem için olurlu bir çözüm bulmak için tek başına satır ve sütun türetme yöntemi yeterli değildir. Bu nedenle, bu yönteminin yanı sıra *dal-fiyat ve kesi (DFK) (branch-price-and-cut)* yöntemi de geliştirilmiştir. *DFK* yöntemi kesin çözüm yöntemidir ve optimallik garantisi vardır. *DFK* yöntemi, dal ve sınır yöntemi ile satır ve sütun türetme yöntemini bir araya getiren bir çözüm yöntemidir. Ayrıca, *DFK* için iyi üst sınırlar bulmak için sezgisel bir yöntem de geliştirilmiştir. *DFK* ve sezgisel yöntem birlikte kullanılarak, *MD-SUKRK* için iyi çözümler bulunmaya çalışılmıştır.

Daha sonra geliştirilen yöntemler birbirleriyle karşılaştırılmıştır. Geliştirilen yöntemler, *kararlı* bir maliyet dağıtımı verdiği için olurlu çözümler vermektedir. Bu nedenle, bu yöntemler; çözümlerin alt sınıra olan uzaklığı, yöntemler için harcanan zaman, üretilen çevrim sayısı, seçilen rota ayrıtı sayısı, seçilen çevrim sayısı, rota ayrıtlarına düşen maliyet ve rota ayrıtlarının iş birliğinden elde ettiği kazanımlar üzerinden karşılaştırılacaktır. *MD-SUKRK* problemi için optimal çözümü elde etmek oldukça güçtür. Bu nedenle geliştirilen çözüm yöntemleri bir alt sınır kullanılarak karşılaştırılacaktır. Bu problem için, alt sınırın nasıl hesaplanacağı ileride detaylı olarak açıklanacaktır.

Bunların dışında *kararlı* bir maliyet dağıtımı bulmak için çeşitli maliyet dağıtım modelleri de geliştirilmiştir. Bu modeller, literatürde yer alan maliyet dağıtım metotlarıyla karşılaştırılmıştır. Bu karşılaştırmalar; ihlal sayıları, ortalama ihlal oranları, maksimum ihlal oranları, rota ayrıtlarına düşen maliyet ve rota ayrıtlarının iş

birliğinden elde ettiği kazanımlar üzerinden yapılacaktır. Karşılaştırmalar, belirli kurallara göre rastgele üretilen iki örnek grubu (24 ve 30) kullanılarak yapılmıştır. Analizler ilk olarak 24 örnek üzerinden yapılmıştır. Daha sonra kullanılan örneklerin boyutları arttırılarak, yöntemler 30 örnek üzerinden analiz edilmiştir. İki örnek grubu arasındaki temel fark, içerdikleri düğüm ve rota ayrıtı sayısıdır. Elde edilen çözümler tablolar halinde raporlanmıştır.

2. LİTERATÜR ARAŞTIRMASI

Literatürde tam kamyon yükü gönderici iş birliğinde *kararlı* koalisyonun bulunması üzerine yapılmış sınırlı sayıda çalışma mevcuttur. Ancak bu çalışma kapsamında kullanılan yöntemler ve kavramlar için literatürde birçok çalışma mevcuttur. Bu nedenle geniş kapsamlı bir literatür çalışması yapılmıştır. Bu başlık altında literatürde yer alan benzer çalışmalardan, kullanılabilecek olası çözüm yöntemlerini içeren çalışmalardan, maliyet dağıtımı içeren çalışmalardan ve *kararlılığı* ölçen çalışmalardan bahsedilecektir.

2.1 Lojistik İş Birliği ve İşbirlikçi Oyun Kuramı

Bu başlık altında, lojistik iş birliği üzerine yapılan çalışmalardan ve işbirlikçi oyun kuramında kullanılan kavramları içeren çalışmalardan bahsedilecektir. Burada bahsedilecek bazı çalışmalarda maliyet dağıtım yöntemleri de geliştirilmiştir. Bu nedenle, bu yöntemlerden de bahsedilecektir.

le Blanc vd. (2006), perakendeci dağıtımda yer alan *Fabrika Kapısında Fiyatlandırması (FKP) (Factory Gate Pricing)* konusu üzerine çalışmışlardır. *FKP* 'de, ürünler perakendeciler tarafından; üreticinin kapısından alınarak, dağıtım merkezine taşınır. Perakendecinin tedarik zinciri; birincil dağıtım ve ikinci dağıtım olmak üzere iki kısma ayrılmıştır. *FKP* 'nin performansını değerlendirmek için yedi farklı lojistik senaryosu geliştirmişlerdir.

Cruijssen vd. (2007), yatay iş birliği hakkında fikir sahibi olabilmek için geniş kapsamlı bir anket çalışması gerçekleştirmişlerdir. Yedi hipotez oluşturmuşlardır. Birinci ve ikinci hipotez firma büyüklüğüyle ilgilidir. Üçüncü, dördüncü ve beşinci hipotez, kamyon firmasının kapsamı ile yatay iş birliği arasındaki ilişkiyle

ilgilidir. Altıncı ve yedinci hipotez ise, firmanın etkinlik seviyesiyle ilgilidir. Altıncı ve yedinci hipotezi test etmek için; yargılar, fırsatlar ve tehditler olarak iki kategoriye ayırmışlardır. Katılımcıları ise, işbirlikçi-işbirlikçi olmayan, ilgili-ilgili olmayan ve tam yetkili-kısmi yetkili olarak sınıflandırmışlardır. Daha sonra, geliştirdikleri bu hipotezleri test etmişlerdir.

Cruijssen vd. (2010), nakliye ve lojistikte yer alan yatay iş birliği üzerine bir literatür araştırması yapmışlardır. İş birliği ve ortaklık arasındaki farklılıktan bahsetmişlerdir. Yatay iş birliğini üç kategoriye ayırmışlar ve bunları tanımlamışlardır. Bu kategoriler bütünleşmenin seviyesi, merkezileşme ve faaliyet alanı ve yoğunluktur. Deniz taşımacılığında ve havacılıkta yer alan bazı örneklerden bahsetmişlerdir. Ayrıca yatay iş birliğinin faydalarından, engellerinden, tehditlerinden ve yardımcılarından bahsetmişlerdir. Yatay iş birliğinin faydaları olarak maliyet ve verimlilik sıralanabilir. Engelleri ve tehditleri olarak; ortaklar, kazanımın belirlenmesi ve dağıtılması, pazarlıklar ve koordinasyon olarak gösterilebilir. Yardımcıları olarak da; bilgi paylaşımı, ilişki ve sözleşme yönetimi ve bilgi teknolojisi gösterilmektedir.

Frisk vd. (2010), İsveç 'te yer alan sekiz orman firması arasındaki iş birliği üzerine çalışmışlardır. Toplam kazanımın veya maliyetin iş birliğinden nasıl elde edileceğine ve bu maliyetin veya kazanımın firmalar arasında nasıl dağıtılacağına odaklanmışlardır. İşbirlikçi oyun kuramında yer alan ve bazı iyi bilinen dağıtım mekanizmalarını *(Shapley Değeri, Çekirdekçik, dağıtılabilir ve dağıtılamaz maliyetlere dayalı dağıtım, gölge fiyatlara dayalı dağıtım, hacim ağırlıklarına dayalı dağıtım)* tanımlamışlardır. Ayrıca *Eşit Getiri Metodu (Equal Profit Method)* adını verdikleri yeni bir maliyet dağıtım metodu geliştirmişlerdir. Bu metot, toplam maliyeti veya kazanımı paydaşlar arasında mümkün olduğunca eşit şekilde dağıtmayı amaçlar. Yani paydaşlar arasındaki kazanım farkını en küçüklemeyi amaçlar. Se-

kiz orman işletmesiyle birlikte bir vaka çalışması gerçekleştirmişlerdir. Bu vaka çalışmasından elde ettikleri sonuçları bu metotlarla karşılaştırmışlardır.

Audy vd. (2010), Kanadalı mobilya üreticileri arasındaki nakliye koordinasyonu üzerine çalışmışlardır. Servis sağlayıcıları için beş farklı lojistik senaryosu oluşturmuşlardır. *İş birliğinden elde edilen maliyet firmalar arasında nasıl dağıtılacak?* sorusunu cevaplamaya çalışmışlardır. Bunun için bir doğrusal matematiksel model geliştirmişlerdir. Geliştirdikleri bu matematiksel model Frisk vd. (2010) 'nin geliştirdiği matematiksel modelin değiştirilmiş halidir. Bu modelle *kararlı* bir dağıtımının bulunması amaçlanmaktadır. Daha sonra, özel gereksinimlerin olması durumunda maliyet dağıtımın nasıl olacağı üzerine çalışmışlardır.

2.2 Rota Kapsama Problemi

Bu çalışma kapsamında ele alınan problemin, altında yatan en iyileme modeli *Rota Kapsama (RK) (Lane Covering)* problemidir. Bu nedenle, bu başlık altında bu problemden ve bu problemin varyantlarını içeren çalışmalardan bahsedilecektir. Ayrıca bu çalışmalarda kullanılan çözüm yöntemlerinden de bahsedilecektir.

Ergun vd. (2007a), varlıkların yeniden konumlandırma maliyetlerini en küçükleyen turların kümesini bulmaya çalışmışlardır. *RK* problemini tanımlamışlardır. *Sayı Kısıtlı Rota Kapsama (SKRK)* ve *Uzunluk Kısıtlı Rota Kapsama (UKRK)* problemi üzerine de çalışmışlardır. Bu problemlerin **NP-Zor** olduğunu göstermişlerdir. *SKRK* problemini, *Küme Kapsama (Set Covering)* problemi olarak formüle edip, bir açgözlü algoritma önermişlerdir.

Ergun vd. (2007b), kamyon hareketlerinin sürekliliğini sağlan turların bulunması amacıyla iş birliğinin kurulmasını amaçlamışlardır. *Zaman Kısıtlı Rota Kapsama (ZKRK)* problemi üzerine çalışmışlardır. Verilen bir rota kümesi için, turların top-

lam süresini en küçükleyen turların kümesini bulmaya çalışmışlardır. *ZKRK* problemi için bazı çözüm yaklaşımlarından bahsetmişlerdir. Ayrıca, sayısal örneklerle bu yaklaşımları test etmişlerdir.

Ozener ve Ergun (2008), gönderici işbirlikçi oyunları dikkate almışlardır. Rotaları kapsamanın toplam maliyetini en küçükleyen en iyi turları bulmayı amaçlamışlardır. Rotaları kapsamanın maliyeti, orijinal rota maliyetlerinden (dolu gitme maliyeti) ve rotaların yeniden konumlandırılmasının maliyetlerinden (boş gitme maliyeti) oluşmaktadır. Maliyet hesaplandıktan sonra, hesaplanan bu maliyet oyuncular arasında dağıtılmaya çalışılmıştır. Bu çalışmada oyuncular göndericiler değil, rotalardır. İşbirlikçi oyun kuramında yer alan bazı iyi bilinen özellikleri tanımlamışlardır. Bunlar; *etkinlik, kararlılık, çapraz monotonik, eşit olanlara eşit davran* ve *yapay oyuncudur*. Bu özellikleri sağlayan bazı maliyet dağıtım mekanizmaları tanımlamışlardır. *Kararlı* bir maliyet dağıtımının, kullanılan doğrusal problemin dualinin en iyi çözümünden elde edilebileceği gösterilmiştir. Ayrıca, bazı iyi bilinen özeliklerin aynı anda sağlanmasının mümkün olmadığı gösterilmiştir. *Etkinlik* ve *kararlılık* özelliklerinde gevşetmeye gidilmiştir. En düşük borçluluk maliyet dağıtımı ve pozitif faydalı maliyet dağıtımı mekanizmaları geliştirmişlerdir.

2.3 Sütun Türetme, Dengelenmiş Sütun Türetme ve İç Nokta Dengeleme

Bu başlık altında, ele alınan problemi çözmek için sıklıkla kullanılan yöntemlerden biri olan *sütun türetme (column generation)* yönteminden bahsedilecektir. Ayrıca problemde dejenere çözümlere rastlanabileceği için; *dengelenmiş sütun türetme (Stabilized Column Generation) ve iç nokta dengeleme (Interior Point Stabilization)* konuları üzerine de araştırmalar yapılmıştır.

du Merle vd. (1999), *dengelenmiş sütun türetme yönteminin*, dejenere tam sayılı probleme nasıl uygulanacağı üzerine çalışmışlardır. *Sütun türetme yöntemi* uygulandıktan sonra parametrelerin nasıl güncelleneceği ve algoritmayı durdurma kriterlerinin neler olduğu hakkında bilgi vermişlerdir. Daha sonra, *sütun türetme yöntemini* hava yolları mürettebat eşleştirme, çok kaynaklı *Weber problemi* ve *p-ortanca* problemine uygulamışlardır.

Xu vd. (2003), gerçek hayat lojistik operasyonlarında sıklıkla karşılaşılan *Toplamalı ve Teslimatlı Araç Rotalama* problemini dikkate almışlardır. Çalıştıkları problemde çoklu taşıyıcı ve çoklu araç türleri bulunmaktadır. Her toplanması ve teslim edilmesi gereken ürünün zaman penceresi mevcuttur. Yükleme ve boşaltma işlemi belirli sıraya ve kurala uygun yapılmalıdır. Problem, *Küme Bölmeleme (Set Partitioning)* problemi olarak formüle edilmiştir. Problemin gevşetilmiş halini çözmek için *sütun türetme* prosedürünü uygulamışlardır. Ayrıca, bu problemi çözmek için *sütun türetme* bazlı sezgisel bir algoritma geliştirmişlerdir.

Lubbecke ve Desrosiers (2005), *sütun türetme* ve *Dantzing-Wolfe* ayrıştırması üzerine bir çalışma yapmıştır. Tam sayılı programlar için *sütun türetme* tekniğinden bahsetmişlerdir. *Kısıtlı ana problem (restricted master problem)* ve onun çözümü hakkında bilgi vermişlerdir. Ayrıca *fiyatlandırma problemi (pricing problem)* ve alternatif fiyatlandırma kuralları hakkında bilgi vermişlerdir. Simpleks bazlı sütun türetmenin yakınsama hızının yavaş olduğunu vurgulamışlardır. Bu durumun önüne geçmek için, *dengelenmiş sütun türetme yönteminin* nasıl uygulanacağını göstermişlerdir. Ayrıca, *dengelenmiş sütun türetme yönteminde* kullanılan parametrelerin nasıl güncelleneceği hakkında bilgi vermişlerdir.

Oukil vd. (2007), *Çok Depolu Araç Çizelgeleme (Multiple Depot Vehicle Scheduling Problem)* problemi üzerinde çalışmışlardır. Büyük boyutlu problemler için, dejenere çözümleri azaltmak için *dengelenmiş sütun türetme yöntemini* uygula-

mışlardır. Bu problemin, doğrusal gevşetilmiş halini çözmek için etkin bir yaklaşım geliştirmişlerdir.

Rousseau vd. (2007), *Zaman Pencereli Araç Rotalama (Vehicle Routing Problem with Time Window)* problemi için *sütun türetme* algoritmalarının hızını arttırmak için kullanılabilecek yöntemlerden bahsetmişlerdir. *Sütun türetme* bazlı yöntemlerde dejenere çözümlerden dolayı elde edilen çözümlerin yakınsama hızı oldukça yavaş olabilmektedir. Bu problemi çözmek için bahsedilen ilk metot, *kutu-adım (box-step)* metodudur. Bu metot uygulandığında problemin nasıl değiştirileceği açık bir şekilde gösterilmiştir. Diğer bir metot ise *iç nokta dengeleme* yöntemidir. Bu yöntem, sıfır ile bir arasında rastgele sayılar üretilerek problemin iteratif olarak tekrar çözülmesini içermektedir. Fiyatlandırmanın yapılacağı değişkeni içeren değişkenin sağ taraf değerleri, sıfır ile bir arasında rastgele üretilen sayılarla çarpılarak; problem belirli sayıda tekrar çözülür. Bu yöntem uygulanmadan önce problemin doğrusal gevşetilmiş hali optimal olarak çözülür ve problem yukarıda anlatıldığı şekilde yeniden ifade edilir. Bu yöntemin nasıl uygulanacağı ve problemin dualinde oluşan değişiklikler detaylı olarak açıklanmıştır. Ayrıca yukarıda bahsedilen iki metot birbirleriyle karşılaştırılmıştır.

Desrosiers vd. (2013), dejenere çözümleri avantaja çevirebilecek *satırca indirgenmiş sütun türetme (row-reduced column generation)* metodunu önermişlerdir. Metodun ana fikri kısıt sayısını azaltmaktır. Satırca indirgemenin avantajı, daha küçük temelde çalışma imkanı vermesidir.

Muter vd. (2013), büyük boyutlu problemler için *satır ve sütun türetme algoritması (row and column generation)* geliştirmişlerdir. Büyük boyutlu problemleri çözmek için, hem satırın hem de sütunun aynı anda üretilmesine ihtiyaç duyulabileceği vurgulanmıştır.

2.4 Maliyet Dağıtım Mekanizmaları ve Koalisyon Yapıları

Bu başlık altında literatürde yer alan maliyet dağıtım yöntemlerinden, bu yöntemlerin nasıl kullanıldığından, alternatif koalisyon yapılarından ve bu yapıların nasıl kullanıldığından bahsedilecektir.

Gothe-Lundgren vd. (1996), *Araç Rotalama Probleminde Nükleolusun (Çekirdekçik)* hesaplanması üzerine çalışmışlardır. Problem, bütün olası rotalar üretilerek *Küme Kapsama* problemi olarak formüle edilmiştir. Müşteriler kullanılarak *nükleolus* için koalisyonlar oluşturulmuştur. Bu koalisyonun maliyetini hesaplamak için matematiksel model kullanılmıştır. Modelin amacı, koalisyon içerisinde yer alan müşterileri kapsayan minimum maliyetli rotaları bulmaktır. Koalisyonlar olurlu ve olursuz olarak sınıflandırılmıştır. Daha sonra *çekirdekte* yer alan bir maliyet dağıtımının hangi koşulları sağlaması gerektiği tanımlanmıştır. Olursuz koalisyonlar için bu koşulların gereksiz (redundant) olduğu gösterilmiştir. Ayrıca eğer *çekirdek* boş küme değilse, optimalde yer alan rotalar için o rotanın maliyeti o rotanın kapsadığı müşteriler arasında dağıtılacağı gösterilmiştir. Son olarak, araç rotalama probleminin *çekirdeğinin* boş küme olmaması için gerek koşul tanımlanmıştır. Bu tanımlamalardan ve gösterimlerden sonra *nükleolusun* tanımı yapılmıştır. *Nükleolusu* çözmek için, *öncül nükleolusun (pre-nucleolus)* çözülmesi gerekir. *Öncül nükleusun* iç güdüsel olarak uç noktaların orta noktasında olduğu ve olursuz koalisyonlar için *çekirdeği* tanımlayan eşitsizliklerin *nükleolusun* hesaplanmasında gereksiz olduğu varsayımında bulunulmuştur. Ancak bu varsayımlar Chardaire (2001) tarafından hatalı bulunmuştur. Yapılan varsayımların hatalı olduğunu göstermek için bir örnek kullanılmıştır. Bu örnek için, yapılan varsayımların geçerli olmadığı gösterilmiştir. *Nükleolus*, koalisyonların bütün alt kümelerinden tanımlanır. Bu nedenle, koalisyonların bütün alt kümelerini üretmek zor olabilir. Bu durumlar için kısıt türetme yöntemi geliştirilmiştir. *Nükleolus*, kısıtlı alt küme-

lerden tanımlanmıştır. Bu şekilde model çözüldükten sonra, kısıt türetmek için bir alt model oluşturularak çözdürülmüştür. Bu model, *kararlılık* kısıtını en fazla ihlal eden ve modelde yer almayan koalisyonu bulmayı amaçlar. Bu tür koalisyon bulunarak ana modele kısıt olarak eklenir ve tekrar çözdürülür. Alt problemin amaç fonksiyonu sıfır ve sıfırdan büyük bir değer aldığında durdurulur. Ayrıca rotaları üretmek için de matematiksel model geliştirilmiştir. Bu model olurlu rotaları bulmayı amaçlar. Bu model, *Gezgin Satıcı Probleminin* özel bir formuna benzetilmiştir. Ancak bu model alt tur eleme kısıtlarını içerdiği için kısıt sayısı fazla olabilir. Bu durumda, bu sayıyı kontrol altına almak için iki farklı *Sırt Çantası Problemi (Knapsack Problem)* geliştirilmiştir. Bu problemler, rota üretmek için kullanılan model için üst sınır tanımlamaktadır.

Feillet vd. (2005), *Getirili Gezgin Satıcı (Traveling Salesman Problem with Profits)* problemi üzerine çalışmışlardır. Bu problemde, her düğümün ziyaret edilmesine gerek yoktur. Bazı düğümler dışarıda bırakılabilir. Bu problem için farklı amaç fonksiyonları önerilmiştir. Bunlar sırasıyla; toplam taşıma maliyetini en küçüklemek, toplam getiriyi en büyüklemek ve kazanç ile taşıma maliyeti arasındaki toplam farkı en küçüklemektir. Bu problemi çözmek için; kesin çözüm yöntemi ve sezgisel çözüm yönteminden bahsedilmiştir. Kesin çözüm yöntemi için, *Dal-Sınır (Branch and Bound)* yöntemi geliştirilmiştir. Bu yöntem kullanılırken bazı kısıtlar gevşetilmiştir. Sezgisel yöntemde, dört farklı komşuluk tanımlanarak; bu komşulukların nasıl kullanılacağı açıklanmıştır. Meta-sezgisel yöntem olarak ise, *Genetik Algoritmadan (Genetic Algorithm)* ve *Tabu Arama (Tabu Search)* yönteminden bahsedilmiştir. Bunların yanı sıra, nasıl iyi sınırlar elde edilebileceğinden bahsedilmiştir.

Estevez-Fernandez vd. (2009), *Getirili Araç Rotalama (Vehicle Routing Problem with Profits)* problemi üzerine çalışmışlardır. Bu problem, *Gezgin Satıcı (GS)*

probleminin farklı bir türüdür. *GS* probleminden farkı, her düğümün bir kere ziyaret edilmesine gerek yoktur. Bu problemde, her düğüm en fazla bir kere ziyaret edilebilmektedir. Problemin amacı, toplam getiriyi en büyüklemektir. Bazı düğümler, getiriyi arttırmak için dışarıda bırakılabilmektedir. Ayrıca, bu problemin her zaman *çekirdekte* yer alan bir maliyet dağıtımının olduğu gösterilmiştir.

Berger ve Bierwirth (2010), taşıyıcı iş birliğinde *İstek Atama (Request Reassignment)* problemi üzerine çalışmışlardır. Problemin amacı, toplam kazancı en büyüklemektir. Her bir istek (request) yalnızca bir taşıyıcıya atanmaktadır. Problem, karma tam sayılı model olarak formüle edilmiştir. Getiriler taşıma oranına ve uzunluklara göre hesaplanmaktadır. Bu problemi çözmek için, beş adımlı iteratif bir çözüm yöntemi geliştirilmiştir. Geliştirilen çözüm yöntemi iki algoritmadan oluşmaktadır. İlk algoritmada, amaç fonksiyonu değeri iyileşecek şekilde taşıyıcıların istekleri değerlendirilir. İkinci algoritmada, tam sayılı olarak modellenen problem çözülür. Bu algoritmadaki problemin amacı taşıyıcıların toplam kazancını en büyüklemektir. Bu problem istekler atandıktan sonra çözülmektedir.

Kimms ve Cetiner (2012), hava yolu ortaklıklarında ortaya çıkan kazanımın ortaklar arasında nasıl dağıtılacağı konusu üzerine çalışmışlardır. Problem, hangi hava yolunun toplam beklenen kazanımını en büyüklemek için, ne kadar kapasitesini ortaklığa ayırmasını konu almaktadır. Hava yolları ortaklığa ayıracakları koltuk sayılarını belirledikten sonra, beklenen toplam kazanım ortaklar arasında dağıtılmaktadır. Bu dağıtım için *nükleolus (çekirdekçik)* yöntemi kullanılmıştır. Bu yöntemin nasıl uygulanacağı üzerinde durulmuştur. *Nükleolusun* hesaplanması zaman alacağı için, bir algoritma geliştirilmiştir. Bu algoritmanın adımları ve adımların asıl probleme nasıl uyarlanacağı açıklanmıştır. *Nükleolusu* hesaplarken sezgisel yöntemin de kullanılabileceği ancak böyle bir yöntemin ne gibi sorunlara yol açacağına da değinilmiştir.

Dai ve Chen (2012), *Zaman Pencereli Taşıyıcı İş bilirliği (Carrier Collaboration under Time Window)* problemi üzerine çalışmışlardır. Problemin amacı, toplam kazancı en büyüklemektir. Problem iki aşamalı olarak çözülmüştür. İlk aşamada, en büyük toplam kazancı bulmak için karma tam sayılı model çözülmüştür. Bu aşamadan sonra, üç farklı paylaşım tekniği kullanılarak toplam kazanç oyuncular arasında adil bir şekilde dağıtılmaktadır. Bu metotlar; *Shapley Değeri, Oransal Dağıtım* ve kendilerinin geliştirdikleri paylaşım metodudur. Geliştirilen metodun amacı, *bireysel rasyonellik* ve *kararlılık* koşulu altında herhangi iki taşıyıcı arasındaki kazançlar arasındaki farkı en küçüklemektir.

Ozener vd. (2013), *Envanter Rotalama Oyunları (Inventory Routing Game)* üzerine çalışmışlardır. Bu tarz oyunların *çekirdeğinin* boş küme olabileceğini ispatlamışlardır. Bu problemden kaynaklanan maliyeti dağıtmak için çeşitli maliyet dağıtım metotları (*oran bazlı dağıtım, dualite bazlı dağıtım, Shapley Değeri* gibi) önermişlerdir. Önerilen metotların performansları, çözüm kalitesi ve çözüm süresi üzerinden test edilmiştir. Çözüm kalitesi, bir maliyet dağıtımının *kararlılık* koşulunu maksimum ne derecede ihlal ettiğini gösterir. Bu testin bütün olası alt kümelerden yapmanın oldukça güç olduğundan bahsetmişlerdir. Bu engeli aşmak için, altı adımlık bir algoritma (*Approximate Stability Assessment*) önermişlerdir. Önerilen bu algoritma *kararlılık* koşulunu değerlendirmek için kullanılır. Bütün olası alt kümeleri üretmek yerine, belirli sayıda alt kümeler üreterek; değerlendirmeler yapılmaktadır.

Vanovermeire ve Sorensen (2014), *Kapasiteli ve Zaman Pencereli Paketleme (Bundling)* problemi üzerine çalışmışlardır. Bu problemi çözmek için iki model önerilmiştir. Problem, maliyeti dağıtmak için farklı bir yaklaşım içermektedir. Önerilen iki modelin de amacı toplam operasyonel maliyetleri en küçüklemektir. İlk modelde toplam en küçük maliyet belirlenir ve bu maliyet *Shapley Değeri* kullanıla-

rak dağıtılır. İkinci modelde ise, bu iki aşama tek bir aşama olarak değerlendirilir. Zaman penceresi kısıtları için ceza maliyeti tanımlanarak bu kısıtlar gevşetilmiştir. Bu gevşetme yalnızca ilk model için geçerlidir. İkinci modelde bir gevşetme söz konusu değildir. İkinci model bütün maliyet dağıtım kararlarını içermektedir. Bu kararları modele eklemek için, *Shapley Değerine* dayalı *Shapley* kısıtları modele eklenmiştir. Son olarak tanımlanan modelleri çözmek için sezgisel yöntem geliştirilmiştir. Geliştirilen bu sezgisel yöntem dört farklı komşuluk içermektedir ve bu yöntem iteratif bir yöntemdir. Her aşamada, olurlu çözüm elde edilmeye çalışılır. Olursuz bir çözüm elde edilirse, bu çözüm olurlu çözüme dönüştürülür.

Guajardo ve Ronnqvist (2015), her bir oyuncuya dağıtılan toplam maliyeti en küçüklemek için karma tam sayılı matematiksel model geliştirmişlerdir. Bu model, *kararlılık* ve nicelik kısıtlarını içermektedir. Problemin altında yatan matematiksel model *Küme Bölmeleme* problemidir. Nicelik kısıtı, her koalisyonda belirli sayıda oyuncu olmasını sağlamaktadır. *Kararlılık, rasyonellik* ve *etkinlik* koşullarını içermektedir. *Kararlı* koalisyon yapısı, güçlü *kararlı* koalisyon yapısına dönüştürülmüştür. Bu koalisyon yapısında, farklı koalisyondaki oyuncuların kendi koalisyonlarını oluşturmasına izin verilmektedir. Bu problemi çözmek için iki aşamalı çözüm yöntemi geliştirilmiştir. İlk aşamada, olası koalisyonlara karşılık gelen maliyeti en küçükleyen model çözülür. İkinci aşamada ise, en küçüklenen maliyet kullanılarak; karma tam sayılı model çözülür. Maliyet, oyunculara ikinci aşamada dağıtılmaktadır.

Guajardo ve Jornsten (2015), iş birlikçi oyunlarda kullanılan *nükleolusun (çekirdekçik)* nasıl hesaplanabileceği ve hesaplama aşamasında ne gibi hataların yapılabileceği üzerinde durulmuştur. *Nükleolusu* hesaplayabilmek için doğrusal modelin iteratif olarak çözülmesi gerekmektedir. Her bir adımda, bir önceki adımdan elde edilen çözümler kullanılmaktadır. Herhangi bir adımda, bir önceki adımdaki opti-

mal çözümde *kararlılık* kısıtını eşitlik olarak sağlayan elemanlar ve amaç fonksiyonu değeri kullanılır. Hesaplama hatası bu aşamada yapılmaktadır. Doğrusal modelin, alternatif optimalinin olması durumunda hesaplamalarda hatalar oluşmaktadır. Bunun için doğrusal problemin optimal çözümünde eşitlik olarak sağlanan kısıtlara bakmak yerine; problemin dual problemin optimal çözümünde yer alan ve sıfırdan büyük olan değerlere bakılması önerilmiştir. *Tamamlayıcı boşluk koşulları (complementary slackness conditions)* nedeniyle, dual değerlerin sıfırdan büyük olması; dual değerlerin karşılık geldiği kısıtların eşitlik olarak sağlanmasını garanti edecektir. Bu ilişkinin nasıl kullanılacağı ve her bir adımda modelin nasıl güncelleneceği açıklanmıştır. Ayrıca bu hesabı beş farklı alanda yanlış kullanılan makaleler örnek olarak verilerek; doğru değerlerin ne olması gerektiği açıklanmıştır. Bu aşamaları izlerken karşılaşılabilecek zorluklardan ve bu zorlukların nasıl çözüleceğinden de bahsedilmiştir. İlk olarak, dual modelin alternatif optimal çözümü olması durumunda ne yapılması gerektiği açıklanmıştır. Ayrıca hesaplamalarda kullanılan hassasiyetlerden kaynaklanabilecek hataların nasıl çözüleceğinden de bahsedilmiştir.

Sun vd. (2015), *Gezgin Satıcı (Traveling Salesman Problem)* problemi üzerine çalışmışlardır. Beş adillik koşulunu ve üç maliyet dağıtım metodunu dikkate alarak; toplam en küçüklenen rota maliyetleri dağıtılmıştır. Adillik koşulları sırasıyla; *bütçe dengesi, bireysel rasyonellik, monotonik, eşit olana eşit davran ve yaklaşık kararlılıktır*. Dikkate alınan maliyet dağıtımları ise *Shapley Değeri, Oransal Dağıtım* ve *Moat* modelidir. *Moat* modeli, bütçe kararlılık koşulunu sağlamadığı vurgulanmıştır.

Kimms ve Kozeletskyi (2016), *Planlama Ufuklu İşbirlikçi Gezgin Satıcı* probleminde maliyet dağıtımı üzerine çalışmışlardır. Bu problem, çoklu depo ve tur zamanı kısıtlarını içermektedir. Problemin amacı, belirli koalisyon için toplam ula-

şım maliyetini en küçüklemektir. Bu problemde birden fazla satıcı bulunmaktadır. *Çekirdekte (core)* yer alan bir maliyet dağıtımı bulmak için bir algoritma geliştirilmiştir. Algoritma iki tane en iyileme problemini içermektedir. İlk problem belirli koalisyon kümeleri için maliyet dağıtımı veren ana problemdir. İkinci problem ise *kararlılık* koşulunu ihlal eden koalisyonları bulmayı amaçlayan, alt problemdir. Geliştirilen algoritma iteratif olarak ilerler ve *çekirdekte* yer alan bir maliyet dağıtımı bulana kadar devam eder. Alt problemde satır üretilmektedir. Elde edilen satırlar ana probleme eklenmektedir.

Defryn vd. (2016), *Seçimli Araç Rotalama Problemi (Selective Vehicle Routing Problem)* için maliyet dağıtımı üzerine çalışmışlardır. Bu maliyet dağıtımı, bir müşteri *büyük koalisyon (grand coalition)* dışında kaldığı durumda; bu müşterinin bireysel maliyetinin nasıl hesaplanacağını tanımlar.

Kimms ve Kozeletskyi (2016), *İşbirlikçi Gezgin Satıcı Problemi (Cooperative Traveling Salesman Problem)* için bir maliyet dağıtımı geliştirmişlerdir. Bu maliyet dağıtımı, problemin *çekirdeği (core)* boş küme olmadığı durumda; *çekirdekte* yer alan bir maliyet dağıtımı vermektedir. *Çekirdeğin* boş olduğu durumda ise, *en düşük çekirdek (least-core)* çözümünü vermektedir. Başlangıçta bütün koalisyonları üretmek yerine, kısıtlı sayıda koalisyondan başlanarak; gerekli durumlarda üretilmektedir. Bunun için problem; ana problem (master problem) ve alt problem (sub-problem) olarak ikiye ayrılmıştır. Koalisyonlar eğer gerekli ise alt problemden üretilip; ana probleme eklenmektedir.

De Vos ve Raa (2018), *Farklı Politikalı Envanter Rotalama Problemi (Inventory Routing Problem with Different Policies)* için; *Shapley Değeri, nükleolus, öncül nükleolus* ve diğer maliyet metotları için *kararlılık* analizi yapmışlardır.

Guajardo vd. (2018), tam kamyon yükü iş birliğinde birden fazla koalisyon durumunu içeren koalisyon yapısı problemi üzerine çalışmışlardır.

Akyol ve De Koster (2018), *Zaman Pencereli Araç Rotalama Problemi* üzerine çalışmışlardır. Problemin amacı, üç farklı şehirde yer alan üç farklı perakendeci için toplam maliyeti en küçükleyecek şekilde rotaların bulunmasıdır. Toplam en küçük maliyet bir matematiksel model yardımıyla hesaplanmaktadır. Bu maliyet hesaplandıktan sonra, her bir şehir için *tatmin skoru* hesaplanır. Bu skor hesaplandıktan sonra, *Shapley Değeri* kullanılarak; her bir şehir için en iyi zaman penceresi belirlenmiştir.

Lu ve Quadrifoglio (2019), *Araç Paylaşım Probleminde (APP) (Ridesharing Problem) nükleolusu* bulmak için bir algoritma geliştirmişlerdir. *APP*, ortak rotalara sahip müşterileri, toplam maliyeti en küçükleyecek ve araç kapasitelerini aşmayacak şekilde bir araya getirme problemidir. Kimms ve Kozeletskyi (2016) yaptığı çalışmada olduğu gibi, *nükleolusu (çekirdekçik)* bulmak için problem; ana problem (master problem) ve alt problem (sub-problem) olarak ikiye ayrılmıştır. Alt problemde üretilen kısıtlar ana probleme eklenmektedir. Algoritma bu şekilde, alt problemde kısıt bulunamayana kadar devam etmektedir.

Yang vd. (2020), *Tek Koalisyonlu İşbirlikçi Araç Rotalama Problemi* üzerine çalışmışlardır. Problem iki farklı sistem kullanılarak formüle edilmiştir. Problemin amacı toplam maliyeti en küçüklemektir. Problemi çözmek için *dal-sınır* ve *Danzing Wolfe ayrışmasını* birleştirerek; bir *dal-fiyat-sınır* algoritması geliştirmişlerdir. Toplam maliyet en küçüklendikten sonra, *nükleolus* ve *Shapley Değeri* 'ni kullanılarak bu maliyet oyuncular arasında paylaştırılmıştır.

2.5 Dal-Fiyat Algoritmaları ve Fiyatlandırma Problemleri için Dinamik Programlama

Bu başlık altında, çalışılan problem için optimal veya olurlu bir çözüm elde edebilmek için kullanılabilecek *dal-fiyat* ve *dal-sınır* yöntemini içeren çalışmalardan bahsedilecektir. Ayrıca *fiyatlandırma problemini* optimal çözebilmek için *dinamik programlama* yöntemini içeren çalışmalardan da bahsedilecektir.

Feillet vd. (2005), *Ayrıt Rotalamanın* özel bir türü olan *Kazanımlı Ayrıt Rotalama* problemi üzerine çalışmışlardır. Problemin amacı, araç kapasitesini ve uzunluk kısıtını ihlal etmeyen en büyük toplam kazancı sağlayan çevrimleri bulmaktır. Problem patika bazlı (path-based) olarak modellenmiştir. Problem çok sayıda tam sayılı değişken içerdiği için büyük ölçekli tam sayılı model olarak değerlendirilmektedir. Bu tarz problemleri çözmek için genellikle *sütun türetme* bazlı çözüm yöntemleri kullanılmaktadır. Bu problemi çözmek için *dal-fiyat (DF)* yöntemi geliştirilmiştir. Sütun türetme aşamasına başlamadan önce, başlangıç sütunları atama problemi ve iki açgözlü algoritma kullanılarak üretilmektedir. *DF* yöntemi için yeni bir dallandırma stratejisi geliştirilmiştir. Ayrıtlar üzerinden dallandırmanın mümkün olmaması durumunda, kısa dizeli ayrıtların oluşturduğu kümeler üzerinden dallandırılma yapılmaktadır.

Chabrier (2006), *Zaman Pencereli Araç Rotalama (Vehicle Routing Problem with Time Window)* problemi üzerine çalışmışlardır. Problem, *Dantzing-Wolfe* ayrışım yöntemi kullanılarak iki probleme ayrıştırılmıştır. Bu yöntem kullanılarak *sütun türetme* yöntemi geliştirilmiştir. *Fiyatlandırma problemini* çözmek için *Etiketleme Algoritması (Labeling Algorithm)* yöntemi kullanılmıştır. Bu yöntemin etkinliğini artırmak için çeşitli yöntemler tanımlanmıştır. Ayrıca bu yöntem için *baskınlık kuralı (dominance rule)* geliştirilmiştir. Bu yöntem zaman alıcı olduğu için, sezgisel yöntem ve çizge indirgeme yöntemi tanımlanmıştır.

Dell'Amico vd. (2006), *Araç Rotalama (Vehicle Routing Problem)* problemi için *sütun türetme* yöntemi geliştirmişlerdir. Problem, iki probleme ayrılmıştır. *Fiyatlandırma problemi* için *dinamik programlama* modeli geliştirilmiştir. *Sütun türetme* aşamasında üretilen sütun sayısını kontrol altına almak için iki farklı yöntem kullanılmıştır. Bunlardan ilki, belirli iterasyon sonunda değişkenlerin indirgenmiş maliyetlerine göre bazı değişkenlerin modelden çıkarılmasıdır. Diğeri ise alt sınır geliştirilmesidir. Bu problemi çözmek için iki farklı yaklaşım söz konusudur. Bunlardan ilki, problemi sezgisel olarak çözmektir. Problemi sezgisel olarak çözmek için, problemin doğrusal gevşetilmiş hali iteratif olarak çözülerek; her iterasyon sonunda en yüksek kesirli değere sahip değişken bire sabitlenmiştir. İkinci çözüm yöntemi olarak, *dal-fiyat* yöntemi geliştirilmiştir. Doğrusal gevşetilmiş modelin optimal çözümünden elde edilen çevrimlerin minimum uzunluğu belirlenerek, bu uzunluğun bir eksiği kadar düğüm üretilmektedir. Arama stratejisi olarak *en iyi öncelikli arama (best first search)* yöntemi kullanılmıştır.

Estevez-Fernandez vd. (2009), *Çok Depolu Kapasitesiz Ayrıt Rotalama* problemi üzerine çalışılmıştır. Problemin amaç fonksiyonu, taşıyıcıların iş birliğinden elde ettiği toplam faydayı en büyüklemektir. Taşıyıcılar arasındaki maliyet dağıtımının nasıl yapılacağı üzerinde durulmamıştır. Yalnızca toplam kazanımın nasıl iyileştirilebileceği üzerinde durulmuştur. Toplam fayda; elde edilen kazanımlardan, toplam maliyetin ve yan ödemelerin (side payment) çıkarılmasıyla bulunmaktadır. Problem, tam sayılı programlama modeli olarak formüle edilmiştir. Bu problemi çözmek için *dal-kesi (branch and cut)* algoritması geliştirilmiştir. Dallandırma işlemi değişkenler üzerinden değil; kısıtlar üzerinden yapılmaktadır. Modelde yer alan bir kısıt gevşetilerek, ihtiyaç duyulduğunda modele eklenmektir. Bunun nedeni, ilgili kısıtın üstel sayıda olmasıdır. Değişkenler ise sabit sayıdadır. Bu aşamada, ayrışım algoritması tanımlanmıştır.

Bode ve Irnich (2012), *Kapasiteli Ayrıt Rotalama (Capacitated Arc Routing Problem)* problemi için, bir indeksli ve iki indeksli iki farklı formülasyon geliştirmişlerdir. Bu formülasyonlar dışında, problem *Küme Kapsama (KK)* problemi olarak da formüle edilmiştir. Bir ve iki indeksli model kullanılarak problem için kesiler üretilmektedir. Üretilen kesiler, *KK* problemine eklenmektedir. Ayrıca problemin çözümü için, *dal-fiyat* yöntemi de geliştirilmiştir. Dallandırma aşaması, üç seviyeli kararlardan oluşmaktadır.

Bartolini vd. (2013), *Kapasiteli Ayrıt Rotalama* probleminin genişletilmiş hali üzerine çalışılmıştır. Problem, *Küme Bölmeleme* problemi olarak formüle edilmiştir. Problem birden fazla araç içermektedir ve bu araçların her birinin kapasiteleri vardır. Problemin formülasyonuna araç sayısı da dahil edilmiştir. Bu araç sayısı üzerinden, üç tane geçerli eşitsizlik tanımlanmıştır. Bu geçerli eşitsizlikler tanımlanarak daha iyi alt sınır elde etmek amaçlanmıştır. Burada daha iyi alt sınırdan kast edilen, olurluluğu bozmadan daha yüksek alt sınırlar elde etmektir. Problem için olurlu bir çözüm elde etmek için, *Değişken Komşuluk Arama (Adaptive Large Neighborhood Search)* sezgiseli kullanılmıştır. Problemin amaç fonksiyonu maliyeti en küçüklemek olduğu için, elde edilen olurlu çözüm problem için bir üst sınır tanımlar. Bu sezgisel, dört tane *yok etme (destroy)* ve üç tane *onarma (repair)* operatörü içermektedir. Bu sezgisel olursuzluğa izin vermektedir. Olursuz çözümler, onarma operatörleriyle olurlu hale getirilmektedir. Bu aşamalardan sonra problem kesin olarak çözülmek istenmiştir. Bunun için, *dal-fiyat* yöntemi geliştirilmiştir. Bulunan alt ve üst sınırlar, bu yönteminin etkinliğini arttırmak için kullanılmıştır. Dallandırma stratejisi olarak, üç aşamalı hiyeraşik dallandırma stratejisi kullanılmıştır. Aktif düğümün seçimi için ise, *en iyi sınır stratejisi* kullanılmıştır. Son olarak, geliştirilen yöntem çeşitli örneklerde test edilmiştir.

Vacca vd. (2013), *Rıhtım Dağıtımı ve İskele Atama (Berth Allocation and Quay Crane)* problemlerini bir araya getiren bir problem üzerine çalışmışlardır. Problemin amacı, vinç kapasitesini aşmadan ve her gemiyi çizelgeleyecek şekilde toplam getiriyi en büyüklemektir. Problem, karma tam sayılı programlama modeli olarak modellenmiştir. Problemin boyutu büyük olduğu için, *sütun türetme* bazlı çözüm yöntemine başvurulmuştur. Problemin doğrusal gevşetmesi kullanılarak; problem iki probleme ayrılmıştır. Başlangıçta kullanılan sütunlar, kısıtları sağlayacak şekilde rastgele üretilmiştir. *Fiyatlandırma problemi dinamik programlamayla* sezgisel olarak çözülmüştür. Problemi optimal olarak çözmek için, *dal-fiyat (DF)* algoritması geliştirilmiştir. Dallandırma kuralı, her bir değişken için farklılık göstermektedir. *DF* algoritması için olur bir çözüm elde etmek için iteratif bir sezgisel algoritma geliştirilmiştir. Her iterasyonda en yüksek kesirli değere sahip değişken bire sabitlenmektedir. Değişkenler bire sabitlendikten sonra ana problem tekrar çözülmektedir. Bu algoritma olurlu bir çözüm bulunduğunda durmaktadır. Bunların dışında, kullanılan tekniklerin etkinliğini nasıl arttırılabileceğinden bahsedilmiştir.

Muter vd. (2014), *Araç Rotalama* problemi için *dal-fiyat* yöntemi geliştirmişlerdir. *Fiyatlandırma problemi* iki farklı şekilde (çizge bazlı ve depo bazlı) ifade edilmiştir. Bu problem, *Kaynak Kısıtlı En Kısa Yol (Shortest Path Problem with Resource Constraint)* problemine dönüştürülerek çözülmüştür. Bu problemi çözmek için de iki aşamalı *dal-fiyat (DF)* algoritması geliştirilmiştir. *DF* bazlı yöntem için üç aşamalı dallandırma stratejisi geliştirilmiştir. Bunlar sırasıyla; araç sayısı üzerinden dallandırma, depolara atanan müşterilen üzerinden dallandırma ve ayrıtlar üzerinden dallandırmaktır. Kesirli değeri en fazla olan değişken, dallandırılacak değişken olarak seçilmiştir. Dallandırma stratejisi olarak olarak *en iyi öncelikli arama* yöntemi kullanılmıştır.

Lysgaard ve Wohlk (2014), *Toplam Kapasiteli Araç Rotalama (Cumulative Capacitated Vehicle Routing Problem)* problemi üzerine çalışmışlardır. Bu problem için, hem *Küme Bölmeleme* hem de araç akış formülasyonu geliştirilmiştir. Bu problemi çözmek için *sütun türetme* bazlı çözüm yöntemi geliştirilmiştir. *Fiyatlandırma problemi, dinamik programlama* yardımıyla çözülmüştür. Çözüm yönteminde; sütunlar üretildikten sonra *kesiler (cuts)* üretilmektedir. *Sütun türetme* bazlı çözüm yönteminin yanı sıra *dal-fiyat* algoritması da geliştirilmiştir. Dallandırma işlemi, kümeler üzerinden yapılmaktadır. Aktif düğümün seçimi için, *en iyi sınır stratejisi* kullanılmıştır.

Archetti vd. (2016), *Kazanımlı Ayrıt Rotalama (Arc Routing Problem with Profits)* probleminin farklı bir türü üzerine çalışılmıştır. Problemin amacı, toplam faydayı en büyüklemektir. Bunun için, problemde yer alan bazı müşteriler dışarıda bırakılabilmektedir. Problem, tam sayılı programlama modeli olarak formüle edilmiştir. Problemde, turlar için süre kısıtı da bulunmaktadır. Problemi çözmek için, kesin bir çözüm yöntem olan *dal-kesi (DK)* algoritması geliştirilmiştir. Problemde yer alan süre kısıtı kullanılarak, bu kısıtlar üzerinden geçerli eşitsizlikler türetilmeye çalışılmıştır. Geliştirilen algoritmanın etkinliğini artırmak için sezgisel çözüm yöntemi geliştirilmiştir. Bu çözüm yöntemi, asıl problem için olurlu bir çözüm yani bir alt sınır vermektedir. Bu sezgisel yöntem, asıl problemin doğrusal gevşetmesinin optimal çözümüyle başlar. Her bir düğüme giren tam sayılı ve kesikli akışlar belirlenerek, kaynak ve hedef düğümler belirlenir. Bu düğümler, düğümlere giren tam sayılı ve kesikli akışların sayılarına bakılarak belirlenir. Eğer bu sayılar eşit değilse, bu düğümün kaynak mı yoksa hedef mi düğüm olduğuna karar verilir. Bu düğümler belirlendikten sonra, *Minimum Maliyetli Akış (Minimum-Cost Flow Problem)* problemi çözülür. Bu sezgiselden elde edilen her çözüm olurlu olmayabilir. Bunun sebebi, turlar üzerindeki zaman kısıtıdır. *DK* yönteminde dallandırma, kısıtlar üzerinden yapılmaktadır. Bu nedenle, eşitsizlik-

ler için ayrışım algoritması tanımlanmıştır.

Hernandez vd. (2016), *Zaman Pencereli Araç Rotalama* problemi üzerine çalışmışlardır. Bu problem için iki farklı *Küme Kapsama* formülasyonu geliştirilmiştir. Her iki problem için de ayrı ayrı *fiyatlandırma problemi* tanımlamışlardır. Bu problemleri çözmek için *dinamik programlamanın* nasıl kullanılacağı ayrıntılı olarak açıklanmıştır. Ayrıca bu problemi çözmek için *dal-fiyat* algoritması geliştirilmiştir. Bunların dışında *sütun türetme* algoritmasının performansını iyileştirmek için iki metot önerilmiştir.

Lozano vd. (2016), *Kaynak Kısıtlı En Kısa Yol* problemini optimal çözmek için bir çözüm yöntemi geliştirmişlerdir. Bu algoritma, problemin negatif maliyetli çevrimleri içermesi durumunda da; problemi optimal olarak çözmektedir. Literatürde, bu problem *dinamik programlama* kullanılarak optimal çözülmektedir. Problem iki ayrı probleme ayrılmaktadır. *Fiyatlandırma problemi*, *kaynak kısıtlı en kısa yol* problemine dönüştürülmektedir. *Fiyatlandırma problemini* etkin olarak çözmek; *sütun türetme* bazlı yönteminin etkinliğini direk etkilemektedir. Her düğüm için baskılanmayan patikalar iteratif olarak belirlenmektedir.

Ozbaygin vd. (2017), *Araç Rotalama (AR)* probleminin farklı bir türü üzerine çalışılmıştır. Bu problemde, klasik *AR* probleminden farklı olarak; teslimatlar müşteri adresleri yerine belirli yerlere yapılarak; teslim edilecek ürünlerin tek bir noktaya getirilmektedir. Problem iki farklı şekilde formüle edilmiştir. Bunlar ayrıt bazlı formülasyon ve *Küme Kapsama (KK)* formülasyonudur. Problemi optimal çözmek için *dal-fiyat (DF)* yöntemi geliştirilmiştir. *KK* formülasyonu üzerinden *sütun türetme* yöntemi tanımlanmıştır. Daha sonra, *fiyatlandırma problemi* tanımlanarak, bu problemin etiket ayarlı (label setting) algoritma ile nasıl çözüleceği açıklanmıştır. Bu algoritma, bir *dinamik programlama* temellidir. Bu algoritmanın etkinliği, *baskınlık kuralına* bağlıdır. Bu problem için, bu kuralın nasıl ta-

nımlanacağı açıklanmıştır. Sütun türetmek için, kesin çözüm yöntemini (fiyatlandırma yöntemi) kullanmak zaman açısından çok maliyetlidir. Bu nedenle, sütun türetmek için öncelikle sezgisel yöntem kullanılmıştır. *Fiyatlandırma problemini* hızlı çözmek için, iki yönlü arama (bidirectional search) tekniği açıklanmıştır. *DF* yönteminde, dallandırma stratejisi olarak üç yöntem izlenmiştir. İlk olarak, 0.5 değerine en yakın değerin dallandırılmasıdır. İkinci olarak, en çok rotada geçen ayrıtın dallandırılması yöntemi izlenmiştir. Problemde zaman penceresi olduğu için en erken zamanı sağlayan ayrıtın dallandırılması stratejileri de geliştirilmiştir. Bu stratejilerde, eşitlik olması durumunda hangi değişkenin seçileceği de açıklanmıştır. Sütun türetme yöntemi kullanıldığı için, başlangıçtaki sütunların nasıl üretildiği önemlidir. Bu nedenle, bu sütunların üretilmesi için literatürde yer alan bir sezgisel yönteme başvurulmuştur. Ayrıca, *fiyatlandırma probleminin* çözümünde oluşabilecek kuyruk etkisiyle (tailing-off effect) başa çıkmak için iki yöntemden bahsedilmiştir. Bunlardan ilki, erken budama için; lagranj dual sınırlarının kullanılmasıdır. İkincisi de, erken budamadır.

Chabot vd. (2018), bir taşıma probleminde iş birliği üzerine çalışmışlardır. Talepleri rotalara ve periyotlara atamak için farklı amaçlara sahip dört iş birliği mekanizması sunmuşlardır. Bu mekanizmaları çözmek için *dal-kesi* yöntemi ve *değişken komşuluk arama* yöntemleri geliştirmişlerdir. Geliştirilen yöntemlerin performansı, farklı talepli örnekler üzerinde test edilmiştir.

2.6 Kararlılığın Ölçülmesi ve Kararlı Eşleme

Kararlı bir maliyet dağıtımın bulunması oldukça zor ve zaman alıcı olabilir. Bu nedenle kararlı bir maliyet dağıtımı bulunamadığı durumdaki ölçüm kriterlerine ihtiyaç vardır. Bu başlık altında literatürde yer alan, maliyet dağıtımının *kararlılığını* ölçen bazı kavramları içeren, çalışmalardan bahsedilecektir. Bunlardan en

önemlisi *kararlılığın bedeli (price of stability)* ve *düzensizliğin bedelidir (price of anarchy).*

Kararlılığın bedeli, sistemin olabilecek en iyi durumu ile kararlı durumu arasındaki maliyet oranıdır. *Düzensizliğin bedeli* ise, sistemin olabilecek en iyi durumu ile düzensizlik durumu arasındaki maliyet oranıdır. Bu kısımda *kararlılığın bedeli, düzensizliğin bedeli* ve *adalet (fairness)* üzerine yapılan çalışmalardan bahsedilecektir.

Bu ölçütlerin dışında *kararlılık*, *kararlı eşlemeler (stable matchings)* üzerinden de test edilebilir. *Kararlı eşlemelerin* temeli, Gale ve Shapley (1962) tarafından tanımlanan *Kararlı Evlilik Problemine (Stable Marriage Problem)* dayanmaktadır. *Kararlı Evlilik* problemi; eşit sayıda, ayrık ve sonlu sayıda iki küme üzerinden *kararlı eşlemeyi* bulma problemidir öyle ki herhangi iki oyuncu kendisine atanan eşlemeden başka bir eşlemeyi tercih edemez. *Kararlı eşlemeler*, oyuncuların tercihleri belirlenerek çalışılan her problem için de tanımlanabilir. Bu nedenle, bu başlık altında *kararlı eşleme* problemlerini dikkate alan çalışmalardan da bahsedilecektir.

Correa vd. (2007), bir ağda en büyük gecikmeli akışı; en küçükleyen problem üzerine çalışmışlardır. En büyük gecikmeyi en küçükleyen olurlu bir akışı, en küçük-en büyük akış olarak isimlendirmişlerdir. Dört farklı amaç fonksiyonu kullanmışlardır. Bunlar; en büyük gecikme, ortalama gecikme, adaletsizlik ve en büyük gecikmeyi en küçüklemektir. Ayrıca, en büyük akışlı amaç için; *düzensizliğin bedeline* sınırlar önermişlerdir.

Anshelevich vd. (2008), yönlü çizge ağ tasarım oyunlarını dikkate almışlardır. *Nash dengesi (Nash Equilibrium), kararlılığın bedeli* ve *düzensizliğin bedeli* gibi kavramları tanımlamışlardır. İyi bir *Nash dengesinin* en iyi cevap dinamiğiyle

başarılabileceğini göstermişlerdir. *Nash dengesi*, her bir oyuncunun karşısındaki oyuncunun seçeneğine karşı verdiği en iyi yanıttır. 1950 yılında John Nash tarafından ortaya atılmıştır.

Bachrach vd. (2009), oyunları *kararlı* hale getirmek için en düşük harici ödemeyi tanımlamışlardır. En düşük harici ödeme, *kararlılığın bedelini* işaret etmektedir. Oyunun *çekirdeği* boş olduğunda, oyunculara dışsal bir ödeme teklif edilerek; *kararlı* bir koalisyon bulunabileceğinden bahsedilmiştir. Ayrıca, koalisyon oyunlarının özel bir türü olan ayarlanmış koalisyon oyununu tanımlamışlardır. Ayarlanmış koalisyon oyunlarında *kararlılığın bedelini* tanımlamışlardır.

Resnick vd. (2009), *kararlılığın bedelini*; oyunları *kararlı* hale getiren minimum ödeme olarak tanımlamışlardır. Ayrıca, eşik ağ akışı oyunlarında, *kararlılığın bedeli* üzerine çalışmışlardır. *Kararlılığın bedeli* için sınır ve yaklaşım vermişlerdir.

Christodoulou vd. (2010), yönsüz ağ tasarımı için *kararlılığın bedeli* üzerine çalışmışlardır. Bu çalışma, Anshelevich vd. (2008) 'nin yaptığı çalışmanın farklı bir türüdür. Hatırlanacağı üzere, Anshelevich vd. (2008) tek yönlü ağ tasarımı için *kararlılığın bedeli* üzerine çalışmışlardır. İki ve üç oyunculu ağ tasarım oyunlarında, *kararlılığın bedeli* için sınırlar tanımlamışlardır.

Bertsimas vd. (2011), *etkinlik* ve adalet arasındaki ödünleşim üzerine çalışmışlardır. Çağrı merkezi, sağlık hizmetlerinin çizelgelenmesi, hava trafiği kontrolü ve kadavra organlarının dağıtılmasında yer alan kaynak dağıtımı üzerine örnekler vermişlerdir. Toplam faydayı en büyüklemeye çalışan fayda ölçütü tanımlamışlardır. *Adaletin bedelini* tanımlamışlardır. *Adaletin bedeli* ve *kararlılığın bedeli* için en kötü durum örnekleri vermişlerdir. Ayrıca, hava trafiği akış yönetimi üzerine bir vaka çalışması gerçekleştirmişlerdir.

Feldman ve Tamir (2012), görevlerin kaynaklara atanmasında yer alan kaynak

atama uygulamaları üzerine çalışmışlardır. Kaynak aktivasyon maliyetinin kullanıcılar arasında eşit bir şekilde dağıtıldığında, *Saf Nash Dengesinin (Pure Nash Equilibrium)* var olmayabileceğini göstermişlerdir. *Kararlılığın bedelini* ve *düzensizliğin bedelini* tanımlamışlardır. *Saf Nash Dengesine* yakınsamak için, en iyi cevap dinamiğini kullanmışlardır.

Chen ve Zhang (2012), iletişim ağı problemleri için, fiyat mekanizmalarının tasarımı üzerine çalışmışlardır. Yeniden ölçeklendirme, zayıf tutarlılık, toplanabilirlik ve pozitiflik aksiyomlarını tanımlamışlardır. Geliştirdikleri aday mekanizmalar, bu aksiyomlarla karakterize edilmektedir. Ademi merkezi sistemi tanımlamışlardır. Kullanıcıların kararlarının bilinmediği durumda, *düzensizliğin bedeli* üzerinde çalışmışlardır.

Chen ve Gurel (2012), yük dengeleme modeli ve maliyet dağıtım modeli üzerine çalışmışlardır. Bu problemler için, *kararlılığın bedelini* ve *düzensizliğin bedelini* tanımlamışlardır. *Kararlılığın bedeli* ve *düzensizliğin bedeliyle* birlikte *saf Nash dengesini* değerlendirmişlerdir.

Ito vd. (2017), bir ağ probleminde (network problem) işbirlikçi eşleme problemi üzerine çalışmışlardır. *Kararlı* bir çözüm bulabilmek için bazı yaklaşımlar önermişlerdir. Bu yaklaşımların karmaşıklığı (complexity) test edilmiştir.

Wang vd. (2018), *Araç Paylaşım Probleminde (Ride-Sharing Problem) kararlı eşleme* üzerine çalışmışlardır. Problem, *maksimum ağırlıklı iki parçalı eşleme problemi (maximum-weight bipartite matching problem)* olarak formüle edilmiştir. Problemin amacı, toplam ağırlığı en büyükleyerek araçlar ve yolcular arasında *kararlı eşlemeyi* bulmaktır. Araçlar ve yolcular arasındaki ağırlık, toplam gidilecek yoldan yapılacak tasarruflar üzerinden hesaplanmaktadır. Optimal çözüm ile *kararlılık* arasındaki ödünleşimi analiz edebilmek için, *kararlılık* kısıtını gev-

şetmişlerdir. Ayrıca yuvarlama ufkuna (rolling horizon) dayalı bir çözüm yöntemi geliştirmişlerdir.

Liu vd. (2018), işbirlikçi oyun *kararlı* olmadığında oyuncular arasında büyük koalisyonu (grand coalition) *kararlı* yapmak için yeni bir yaklaşım önermişlerdir. Bu yaklaşımın, büyük koalisyondan ayrılmak isteyen oyunculara bir ceza maliyeti; büyük koalisyonda kalmak isteyenlere ise teşvik atar. Bu kararlar için bazı fonksiyonlar tanımlamışlardır. Bu fonksiyonları hesaplayabilmek için kesme yüzeyi algoritmasına dayalı iki çözüm yöntemi geliştirmişlerdir.

Zhong ve Bai (2019), *Üç Taraflı Kararlı Eşleme Problemini (Three Sided Stable Matching Problem)* tanımlamışlardır. Bu problem için, bir *kararlı eşlemenin (stable matching)* var olduğunu kanıtlamışlardır. Ayrıca, bu *kararlı eşlemeyi* bulabilmek için bir algoritma geliştirmişlerdir.

Rasulkhani ve Chow (2019), atama oyununu (assignment game) kullanarak ağ taşıma sistemleri için bir *kararlı* fiyatlandırma bulmaya çalışmışlardır. Atama oyunları için *kararlılık* koşullarını tanımlamışlardır. *Kararlı* bir maliyet dağıtımı bulmak için tam sayılı programlama modeli geliştirmişlerdir.

Konemann vd. (2020), *Ağırlıklandırılmış İşbirlikçi Eşleme Oyunları (Weighted Cooperative Matching Games)* için *çekirdekte* ve *en düşük çekirdekte (least-core)* yer alan çözümleri bulmaya çalışmışlardır. Bu oyunlar için *nükleolus* 'un nasıl hesaplanacağını göstermişlerdir.

Peng vd. (2020), *Araç Paylaşım Probleminde* araçlar ve yolcular için *kararlı eşlemenin* bulunması üzerine çalışmışlardır. Wang vd. (2018) yaptığı gibi, problemi *maksimum ağırlıklı iki parçalı eşleme problemi* olarak formüle etmişlerdir. Bu problemde amaç, toplam maliyeti en küçüklemektir. Problemde her bir araç ve yolcu yalnızca bir yolcuya ve araca atanmaktadır. *Kararlılık* koşulunu formüle

etmek için, zaman pencerelerini ve cinsiyet kısıtlarını kullanmışlardır. Gale ve Shapley (1962) önerdiği algoritmayı temel alarak bir çözüm yöntemi geliştirmişlerdir.

2.7 Tezin Literatüre Katkısı

Yukarıda verilen alt başlıklarda, bu tez kapsamında dikkate alınan konularla ilgili detaylı bir literatür taraması verilmiştir. Bu tezin literatüre olan en büyük katkısı, tam kamyon yükü gönderici iş birliğinde *kararlı* bir maliyet dağıtımı bulmak için kullanılan iki aşamalı yaklaşıma daha iyi bir alternatif getirmesidir.

Literatürde yapılan çalışmalara bakıldığında, *kararlı* bir maliyet dağıtımı bulmak veya *kararlı* bir koalisyon yapısı oluşturmak iki aşamalı olarak ele alınmaktadır. Böyle bir durumda, bu tarz bir maliyet dağıtımının bulunması için güçlü koşulların olduğu, bu tez kapsamında ispatlanmıştır.

Kararlı maliyet dağıtımı veya koalisyon bulabilmek için bütün olurlu çevrimler üzerinden bir karma tam sayılı matematiksel model tanımlanmıştır. Önerilen bu modelde, üretilen her bir çevrime karşılık bir değişken ve bir de kısıt bulunmaktadır. Bu nedenle, problemde yer alan rota ayrıtı sayısı artması durumunda modelin makul sürelerde çözülmesi mümkün olmayacaktır. Çevrimleri hızlı bir şekilde üretmek için satır ve sütun türetme yöntemi geliştirilmiştir.

Bu yöntemde çevrimler, tam sayılı modellerle ve sezgisel yöntemlerle üretilmektedir. Sezgisel yöntemler, çevrimleri hızlı bir şekilde üretmek için geliştirilmiştir. Geliştirilen sezgiseller, bu tarz çevrimleri üretmek için kullanılan matematiksel modellere duyulan ihtiyacı azaltarak; satır ve sütun türetme aşamasını hızlandırmıştır.

Satır ve sütun türetme yöntemi tek başına önerilen matematiksel modeli optimal

olarak çözmek için yeterli değildir. Bu nedenle, problemi optimal olarak çözmek için dal-fiyat ve kesi *(DFK)* yöntemi de geliştirilmiştir. *DFK* yöntemi, satır ve sütun türetme yöntemiyle dal-sınır yöntemini bir arada kullanan kesin çözüm yöntemidir. Satır ve sütun türetme yöntemi için geliştirilen sezgiseller ve *DFK*, literatürde tanımlanan *Araç Rotalama (Vehicle Routing)* bazlı bütün problemlere kolaylıkla uyarlanarak; sütun (çevrim veya rota) türetmek ve iyi çözümler elde etmek için kullanılabilir.

Maliyet dağıtımları üzerine sıkı kısıtların konulması durumunda, çok sayıda rota ayrıtı (oyuncu) koalisyon dışında bırakılabilmektedir. Bu durumda, oluşacak koalisyon *kararlı* olmasına rağmen; içerdiği rota ayrıtı sayısı düşük olacaktır. Bu olumsuzluğun önüne geçmek için, önerilen tek koalisyonlu model çok koalisyonlu yapıya uygun hale getirilmiştir. Oluşturulan her bir koalisyon kendi içerisinde *kararlıdır*. Ayrıca, bu yapıda ortaya çıkan çözüm detaylıca incelenerek; tek koalisyonlu yapıyla karşılaştırılmıştır.

Literatürde *çekirdekte* yer alan bir maliyet dağıtımı bulmak için kullanılan *nükleolus* yöntemi temel alınarak; dört farklı maliyet dağıtım yöntemi geliştirilmiştir. *Nükleolus* tanım gereği, *kararlılık* kısıtında oluşan ihlali sözcüksel olarak (*lexicographically*) minimize etmeye çalışır. Bu kısıtta oluşacak ihlaller miktar olarak azaltılmaya çalışılmaktadır. Bu tez kapsamında, miktara alternatif olarak; oran bazlı ihlallerin minimize edilmesi dikkate alınmıştır. Yaptığımız araştırmalara göre, bu yaklaşım literatürde daha önce dikkate alınmamıştır.

Bu tezin literatüre olan katkısı aşağıda özetlendiği gibidir.

1. *SUKRK* probleminin *çekirdeğinin* boş küme olmaması için yeter ve gerek koşullar tanımlanmıştır.

2. *Kararlı* bir maliyet dağıtımı bulabilmek için *nükleolus* temelli maliyet da-

ğıtım yöntemleri geliştirilmiştir.

3. Literatürde yer alan iki aşamalı yaklaşıma alternatif olarak hem toplam maliyeti minimize eden hem de *kararlı* maliyet dağıtımı bulan *MD-SUKRK* problemi tanımlanmıştır.

4. *MD-SUKRK* probleminde kullanılan çevrimleri etkin şekilde üretebilmek için satır ve sütun türetme yöntemi geliştirilmiştir.

5. Geliştirilen satır ve sütun türetme yöntemini hızlandırmak için dört farklı sezgisel yöntem tanımlanmıştır.

6. *MD-SUKRK* problemini optimal çözmek için hem dal-sınır hem de satır ve sütun türetme yöntemini bir araya getiren dal-sınır ve kesi (*DFK*) yöntemi geliştirilmiştir.

7. *DFK* yönteminden daha iyi olurlu çözümler elde etmek için, *DFK* 'ye üst sınır tanımlayan bir sezgisel geliştirilmiştir.

8. Çok koalisyonlu yapının, kapsanan rota ayrıtlarına ve çözüm süresine olan etkisini analiz etmek için *ÇKMD-SUKRK* problemi tanımlanmıştır.

9. *MD-SUKRK* probleminde yer alan ve maliyet dağıtımlarını kısıtlayan parametrelerin çözüme olan etkisi analiz edilmiştir.

3. PROBLEM TANIMI ve FORMÜLASYONU

3.1 İşbirlikçi Oyun Kuramı

Kararlı bir maliyet dağıtımı bulmak için, işbirlikçi oyun kuramına başvurulabilir. İşbirlikçi oyun kuramı, bencil oyuncuların faydalarını artırmak için bireysel olarak hareket etmek yerine; başka oyuncularla ortak hareket ettiği durumları ele alır. İşbirlikçi oyun kuramında, iş birliğinden elde edilen ortak kazanımın veya maliyetin oyuncular arasında paylaştırılması gerekir. İş birliğinden elde edilen kazanımın veya maliyetin oyuncular arasında dağıtılmasında, işbirlikçi oyun kuramında kullanılan bazı kavramlardan yararlanılabilir.

Eğer bir maliyet dağıtımında, iş birliğinden elde edilen toplam maliyet; oyunculara dağıtılan toplam maliyete eşit ise bu maliyet dağıtımına *bütçe dengeli (budget balance)* maliyet dağıtımı denir. Eğer her oyuncuya düşen maliyet, kendi bireysel maliyetlerinden düşük veya eşit ise böyle bir maliyet dağıtımına *bireysel rasyonel (individual rationality)* maliyet dağıtımı denir. *Grup stratejisine dayanıklı (group-strategy proof)* veya *kararlı (stable)* bir maliyet dağıtımında, hiçbir oyuncu büyük koalisyondan *(grand coalition)* kopup; alt koalisyon oluşturarak mevcut faydasını arttıramaz. Bu kavram iş birliğini bir arada tutan en önemli kavramdır. İş birliğinin sağlıklı biçimde sürdürülebilmesini sağlar. *Bütçe dengeli* ve *kararlı* maliyet dağıtımlarının oluşturduğu kümeye oyunun *çekirdeği (core)* denir. *Çekirdek* bir küme tanımlar. Bir oyunun çekirdeği boş küme olabilir.

Çekirdekte tek bir tane maliyet dağıtımı olmak zorunda değildir. Birden fazla maliyet dağıtımı bulunabilir. Böyle bir durumda, bazı maliyet dağıtımları bazılarına göre daha tercih edilebilir olabilir. Schmeidler (1969) tarafından tanımlanan *nükleolus (nucleolus)* böyle maliyet dağıtımlardan biridir. *Nükleolus (Çekirdekçik)*,

bütün koalisyonlar üzerinden en düşük faydayı sözlüksel (*lexicographically*) biçimde en büyüklemeye çalışır. Eğer *çekirdek* boş değilse, *nükleolus* vardır. *Çekirdeğin* boş olması durumunda da *nükleolus* var olabilir.

İş birliği kurulduktan sonra iş birliğine dahil olmak isteyen yeni oyuncular olabilir. Yeni oyuncular iş birliğine dahil edildikten sonra, mevcut oyuncuların faydalarının olumsuz yönde etkilenmemesi gerekir. Bu durumun önüne geçmek için *çapraz monotonik* (*cross monotonic*) özelliği kullanılabilir. *Çapraz monotonik* bir maliyet dağıtımı yeni oyuncuların iş birliğine dahil edilmesi durumunda; mevcut oyuncuların faydalarının olumsuz yönde etkilenmeyeceğini garanti eder. *Çapraz monotonik* bir maliyet dağıtımı *bütçe dengesi* özelliğini sağlarsa aynı zamanda *grup stratejisine* dayanıklıdır. Bütün bu özelliklerin aynı anda sağlanması çok zor olabilir. Bu durumda bazı özelliklerde gevşetme yoluna gidilebilir.

İşbirlikçi oyun kuramında sıklıkla karşılaşılan maliyet dağıtımlarından biri de *Shapley Değeridir* (*Shapley Value*). *Shapley Değeri*, her oyuncunun alt koalisyonlara ayrı ayrı yaptığı marjinal katkısının ağırlıklı ortalamasıdır. Başka bir deyişle, iş birliğinin teker teker kurulması halinde her oyuncunun iş birliğine yaptığı ortalama katkıdır. *Eşit olanlara eşit davran (equal treatment of equals)* özelliğinde ise, her açıdan aynı olan iki oyuncuya aynı maliyet atanır. Aynı maliyete sahip olan oyuncular, aynı oyuncu olmak zorunda değildir. Eğer bir oyuncu iş birliğine dahil edildiğinde, herhangi bir maliyet artışı sağlamıyorsa; o oyuncuya *yapay oyuncu (kukla oyuncu) (dummy player)* denir.

Yukarıda verilen tanımların bir kısmı, bu çalışma kapsamında dikkate alınmıştır. Daha önce de belirtildiği gibi yukarıda tanımlanan koşulları aynı anda sağlayan bir maliyet dağıtımı bulmak oldukça güçtür. Bu nedenle, geliştirilen yöntemlerde bütün koşulların sağlanması hedeflenmemiştir. Seçilen bazı koşulların sağlanması amaçlanmıştır. Bunlar; *kararlılık, bütçe dengelilik* ve *bireysel rasyonelliktir*. *Ka-*

rarlı ve *bütçe dengeli* bir maliyet dağıtımı elde edileceği için *çekirdekte* yer alan bir maliyet dağıtımı da bulunacaktır. Bunların dışında, *Shapley Değeri* ve *nükleolus (çekirdekçik)* maliyet dağıtımları da dikkate alınmıştır.

3.2 Rota Kapsama Problemi

Tam kamyon yükü gönderici iş birliğinde altta yatan en iyileme modeli, *Rota Kapsama (RK) problemidir*. *RK* problemi, rotaların kümesi verilmiş iken bütün rota ayrıtlarını kapsayan en düşük maliyetli çevrimlerin (turların) kümesini bulma problemidir. Matematiksel bir ifadeyle, yönlü bir *Öklid* çizgesi $G = (N, A)$, N tane düğüm, A tane ayrıt ve $\rho_a f_a (a \in A)$ ayrıtların maliyetleri ile tanımlanan şebeke ve bu şebekede düzenli gönderileri temsil eden bir alt kümesini $(L \subseteq A)$ kapsayan ve toplam maliyeti en küçükleyen çevrimleri bulma problemi olarak tanımlanabilir. *RK* problemi atama problemine dönüştürülebildiği için **NP-Zor** bir problem değildir. Bilindiği gibi atama problemi *Macar Algoritmasıyla (Hungarian Algorithm)* polinom zamanda çözülebilmektedir.

RK probleminde kullanılan çevrimler üzerine kısıtlar konulduğunda, ortaya çıkan problemler **NP-Zor** problemlerdir (Ergun vd. (2007a)). Örneğin çevrimlerin uzunluğuyla ilgili bir kısıtlama yapıldığında ortaya çıkan *Uzunluk Kısıtlı Rota Kapsama* problemi, çevrimler üzerine konulan zaman penceresi kısıtlarıyla ortaya çıkan *Zaman Kısıtlı Rota Kapsama* problemi ve çevrimler içerisinde yer alan rota ayrıtları üzerine sayı kısıtı konulduğunda ortaya çıkan, *Sayı Kısıtlı Rota Kapsama* problemi **NP-Zor** problemlerdir (Ergun vd. (2007b)).

RK problemi, *Küme Bölmeleme* (*Set Partitioning*) problemi olarak formüle edilebilir. $C(L)$ olurlu çevrimleri ve L rota ayrıtlarını göstersin. s_{lc}, bir çevrimin bir rota ayrıtını kapsayıp kapsamadığını gösteren parametre ve f_c, her $c \in C(L)$ için olurlu

çevrimlerin maliyetlerini gösteren parametre olsun. x_c ise bir çevrimin seçilip seçilmediğini gösteren ikili karar değişkeni olsun. Bu durumda, *Küme Bölmeleme* formülasyonu aşağıdaki şekilde ifade edilebilir.

$$\min \sum_{c \in C(L)} f_c x_c \tag{3.1}$$

$$\text{s.t.} \sum_{c \in C(L)} s_{lc} x_c = 1 \qquad \forall l \in L \tag{3.2}$$

$$x_c \in \{0,1\} \qquad \forall c \in C(L) \tag{3.3}$$

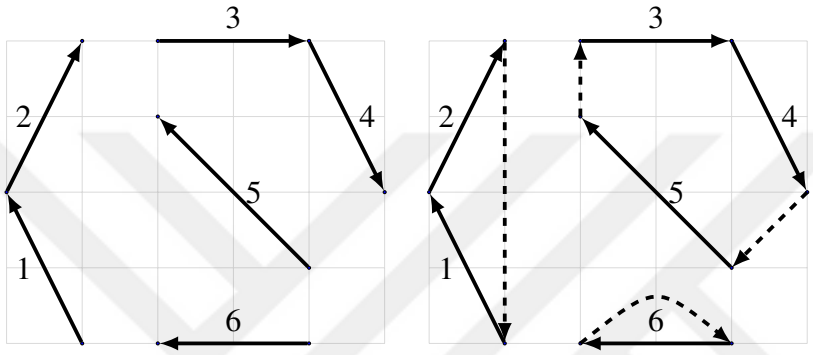
RK probleminin amacı (3.1), seçilen tüm çevrimlerin toplam maliyetlerini en küçüklemektir. Kısıt 3.2, her rota ayrıtının bir çevrim tarafından kapsanmasını sağlar.

3.3 Sayı ve Uzunluk Kısıtlı Rota Kaplama Problemi ve Problemin Çekirdeği

RK problemine gerçek hayatta kullanılan bir takım kısıtlar eklenerek problemin farklı varyantları oluşturulabilir. Oluşturulacak çevrimlerin kapsayabileceği rota ayrıtı sayısı ve uzunlukları kısıtlanarak *Sayı ve Uzunluk Kısıtlı Rota Kaplama (SUKRK)* problemi tanımlanabilir. *SUKRK* problemi matematiksel olarak ifade edilecek olursa; yönlü bir *Öklid* çizgesi $G = (N, A)$, N tane düğüm, A tane ayrıt ve $\rho_a f_a (a \in A)$ ayrıtların maliyetleri ile tanımlanan şebeke verildiğinde, bütün rota ayrıtlarını kapsayan ve toplam maliyeti en küçükleyen çevrimleri bulma problemi olarak tanımlanabilir öyle ki her çevrim aşağıdaki koşuların hepsini sağlar:

1. Her çevrim en fazla K_{max} tane rota ayrıtını kapsayabilir.
2. Her çevrimin uzunluğu en fazla L_{max} kadar olabilir.
3. Her çevrim içerisindeki düğümler en fazla bir kere ziyaret edilebilir.

Yukarıda tanımlanan kısıtlar literatürde sırasıyla, *sayı, uzunluk* ve *basit çevrim* kısıtı olarak bilinmektedir. Burada, rota ayrıtı tam dolu kamyon yükü hareketini; ayrıt kavramı ise boş kamyon yükü hareketini temsil etmektedir. Bu problemin *RK* probleminden farkı, üretilecek çevrimler üzerinde sayı ve uzunluk kısıtının olmasıdır. Çevrimler yukarıda verilen koşullara uygun olacak şekilde üretilir.



(a) Rota ayrıtlarının bir çizgede gösterimi

(b) Çevrimlerin bir çizgede gösterimi

Şekil 3.1: *SUKRK* Problemi için Örnek Problem ve Optimal Çözümü

Şekil 3.1 'de *SUKRK* problemi için 6 rota ayrıtlı bir problem ve bu problemin çözümü gösterilmektedir. Şekil 3.1a 'da görüldüğü gibi; her bir rota ayrıtı bir başlangıç (origin), bir de bitiş (destination) düğümünden oluşmaktadır. Aynı zamanda her bir rota ayrıtı, tam dolu kamyon yükü hareketini temsil etmektedir. Altta yatan yönlü çizgenin, tam çizge olduğu varsayılmıştır. *SUKRK* probleminin amacı, bütün rota ayrıtlarını kapsayan minimum maliyetli çevrimleri bulmaktır. $K_{max} = 3$, $L_{max} = 10$, birim boş ve dolu gitme maliyetinin de 1 olduğunu varsayalım. Çizge üzerinde yer alan her bir karenin de 1 birim uzunluğa karşılık geldiğini varsayalım. Bu durumda ortaya çıkan optimal çözüm, Şekil 3.1b 'de gösterildiği gibidir. Şekil 3.1b 'de kullanılan kesikli çizgiler, boş kamyon yükü hareketini temsil etmektedir. Bu tür hareketler çevrimleri tamamlamak için ihtiyaç duyulduğunda kullanılır. Şekil 3.1b 'de, üç farklı çevrim görülmektedir: $\{1,2\}$, $\{3,4,5\}$ ve $\{6\}$.

Bu çevrimlerin sırasıyla 8.48, 9.48, ve 4 birim uzunluktadır. Bu şekilde yer alan her bir çevrim, yukarıda tanımlanan üç koşulu sağlamaktadır.

İş birliğinden elde edilen toplam maliyetin dağıtılması, iş birliğinden elde edilen faydayı belirleyecektir. Maliyet dağıtım problemi, işbirlikçi oyun olarak modellenerek; *çekirdekte* yer alan bir maliyet dağıtımı bulunabilir mi bunun analizin yapılması gerekir. İlk olarak problemin *alttoplamsallık* (*subadditive*) özelliğini inceleyeceğiz. Alttoplamsallık özelliği, büyük koalisyonda (grand coalition) toplam maliyetinin; oyuncuların bireysel maliyetleri (stand-alone cost) toplamından küçük veya eşit olmasını garanti eder.

Önerme 1. *SUKRK problemi alttoplamsaldır.*

İspat. T ve S, L 'nin herhangi iki ayrık alt kümesi olsun öyle ki $T \cap S = \emptyset$. Problemin alttoplamsal olduğunu ispatlamak için, aşağıda verilen eşitsizliğin herhangi iki alt küme (T ve S) için geçerli olduğu gösterilmelidir.

$$F(T) + F(S) \geq F(T \cup S) \qquad \forall T, S \subseteq L \tag{3.4}$$

Herhangi iki ayrık alt küme, bu kümelerin birleşimi için bir ayrım (partition) tanımlar. Ayrık kümelerin optimal çözümlerinin birleşimi, bu kümelerin birleşimi için olurlu bir çözüm tanımlar. Matematiksel olarak ifade etmek gerekirse: $x_c^{T^*}$ ve $x_c^{S^*}$, T ve S kümesi üzerinden elde edilen *SUKRK* probleminin optimal çözümü olsun. Bu durumda, $F(T) = \sum_{c \in C_{T^*}(L)} f_c x_c^{T^*}$ ve $F(S) = \sum_{c \in C_{S^*}(L)} f_c x_c^{S^*}$ şeklinde ifade edilebilir. $x_c^{T^*}$ ve $x_c^{S^*}$, $T \cup S$ kümesi üzerinden elde edilen *SUKRK* problem için olurlu bir çözüm tanımlar. Bu nedenle, $\sum_{c \in C_{T^*}(L)} f_c x_c^{T^*} + \sum_{c \in C_{S^*}(L)} f_c x_c^{S^*} \geq \sum_{c \in C_{T \cup S}(L)} f_c x_c$ eşitsizliği yazılabilir. Bu eşitsizlik, herhangi iki alt küme için $F(T) + F(S) \geq F(T \cup S)$ eşitsizliğini sağlar. □

Problem alttoplamsal olduğu için, büyük koalisyon toplam maliyeti minimize

eder. Problem aynı zamanda, *dışbükey olmayan (non-convex)* olabilir. Dışbükey problemlerin (convex problems) *çekirdeği* her zaman *boş olmayan (non-empty)* kümedir (Shapley (1971)). Problemin *çekirdeği* boş küme olabileceği için, problemin oyunu *dışbükey olmayan (non-convex)* olarak adlandırılır. Bir sonraki aşamada, problemin *çekirdeğinin* boş küme olmaması için yeter ve gerek koşullar tanımlanacaktır.

$F(L)$, *SUKRK* problemi için L rota ayrıtları üzerinden elde edilen çözüm olsun. $F(L)$ 'yi adil bir şekilde dağıtılıp dağıtılamayacağını inceleyelim. Amacımız, aşağıdaki koşulları sağlayan, her bir rota ayrıtı için bir maliyet dağıtımı w_l ($l \in L$) bulmak olsun.

$$\sum_{l \in L} w_l = F(L) \tag{3.5}$$

$$\sum_{l \in S} w_l \leq F(S) \qquad \forall S \subseteq L \tag{3.6}$$

Yukarıda verilen 3.5 ve 3.6 koşulları sırasıyla, *bütçe dengesi* ve *kararlılık* koşulu olarak adlandırılır. Bu iki koşulu aşağıda verilen önerme yardımıyla sadeleştirmek mümkündür.

Önerme 2. *Koşul 3.6 yerine aşağıdaki eşitsizliği yazmak çekirdekte yer alan maliyet dağıtımların kümesini değiştirmez.*

$$\sum_{l \in L_c} w_l \leq f_c \qquad \forall c \in C(L) \tag{3.7}$$

İspat. Her çevrim, rota ayrıtlarının bir alt kümesine karşılık geldiği için ($C(L) \subseteq S \subseteq 2^L$); *Koşul 3.6* 'yı sağlayan herhangi bir maliyet dağıtımı w_l, *Koşul 3.7* 'yi de otomatik olarak sağlar. *Koşul 3.7* sağlandığı herhangi bir maliyet dağıtımını ele alalım. Rota ayrıtlarının her bir alt kümesinin maliyeti; bu alt kümenin oluştura-

bileceği olurlu çevrimler üzerinden elde edilen optimal çevrim kümesinin toplam maliyetine eşittir. Bu nedenle, dağıtılan maliyetler *Koşul 3.7* 'yi sağlıyorsa, *Koşul 3.6* 'da sağlar. □

Bu önermelerden, önemli bir sonuç çıkarılabilir. *SUKRK* probleminin *çekirdeği* ancak ve ancak 3.5 ve 3.7 koşullarını sağlayan bir maliyet dağıtımı varsa boş küme değildir.

Önerme 3. *SUKRK probleminin maliyet dağıtımlarının oluşturduğu çekirdek ancak ve ancak SUKRK probleminin doğrusal gevşetilmiş halinin optimal çözümü tam sayılı çözüm verirse boş küme değildir.*

İspat. *Koşul 3.6*, *SUKRK* problemin doğrusal gevşetilmiş problemine karşılık gelen dual problemin (*D-SUKRK*) oluşturduğu olurlu bölge olarak yorumlanabilir. Bu durumda, maliyet dağıtımı w_l, dual değişkenlere karşılık gelecektir.

$$D\text{-}SUKRK: \quad \max \sum_{l \in L} w_l \tag{3.8}$$

$$\text{s.t.} \sum_{l \in L} s_{lc} w_l \leq f_c \qquad \forall c \in C(L) \tag{3.9}$$

$$w_l \in \mathbb{R} \qquad \forall l \in L \tag{3.10}$$

Yukarıdaki doğrusal programlama modelinin en iyi amaç fonksiyonu değeri $H(L)$ ve w_l^* 'de yine yukarıdaki modelden elde edilmiş maliyet dağıtımı olsun. Bu durumda $H(L)$ ifadesini şu şekilde yazabiliriz: $H(L) = \sum_{l \in L} w_l^*$. Zayıf dualite teoremi gereğince, $H(L) \leq F(L)$ eşitsizliği geçerlidir. *D-SUKRK* bir maksimizasyon problemi olduğu için, *kararlılık* koşulunu sağlayan herhangi bir maliyet dağıtımı yani herhangi bir olur çözüm için $\sum_{l \in L} w_l \leq H(L) \leq F(L)$ eşitsizliği yazılabilir. Ayrıca $\sum_{l \in L} w_l^{SUKRK} = F(L)$ ancak ve ancak *SUKRK* problemin doğrusal gevşetilmiş halinin en iyi çözümü tam sayılı çözüm verirse geçerlidir. Dolayısıyla,

eğer *SUKRK* probleminin doğrusal gevşetilmiş halinin en iyi çözümü tam sayılı çözüm vermezse, hem *bütçe dengesi* hem de *kararlılık* koşullarını sağlayan bir maliyet dağıtımı bulunamaz. □

Önerme 4. *Çekirdekte yer alan bir maliyet dağıtımda, her bir çevrimin rota ayrıtlarına atanan toplam maliyet; o çevrimin toplam maliyetine eşittir.*

İspat. *SUKRK* probleminden elde edilen en iyi çevrimlerin kümesi $C^*(L)$ olsun. Ayrıca $w_l^{core} l \in L$ 'da çekirdekte yer alan bir maliyet dağıtımı olsun. Kısıt 3.7 'den dolayı, çevrim içerisindeki rota ayrıtlarına atanan toplam maliyet f_c 'den küçük eşit olacaktır. $\sum_{l \in L_c} w_l^{core} \leq f_c, \forall c \in C^*(L)$. Bu eşitsizlik her $c \in C^*(L)$ üzerinden toplanırsa, $\sum_{l \in L} w_l^{core} \leq \sum_{c \in C^*(L)} f_c$ eşitsizliği elde edilir. $\sum_{c \in C^*(L)} f_c = F(L) \geq 0$ ve $\sum_{l \in L} w_l^{core} = F(L)$ olduğu için, $\sum_{l \in L_c} w_l^{core} = f_c, \forall c \in C^*(L)$ olmalıdır. □

3.4 Maliyet Dağıtımlı Sayı ve Uzunluk Kısıtlı Rota Kaplama Problemi

Önermelerden de anlaşılacağı gibi, *çekirdekte* yer alan bir maliyet dağıtımı bulmak için güçlü koşullar mevcuttur. Bu nedenle, *SUKRK* problemine maliyet dağıtım mekanizmalarının eklenmesi gerekmektedir. *SUKRK* problemine, maliyet dağıtım kararları eklenmesiyle *Maliyet Dağıtımlı Sayı ve Uzunluk Kısıtlı Rota Kapsama (MD-SUKRK)* problemi ortaya çıkar. *MD-SUKRK* problemi de **NP-Zor** bir problemdir.

MD-SUKRK problemi matematiksel olarak ifade edilecek olursa; yönlü bir *Öklid* çizgesi $G = (N, A)$, N tane düğüm, A tane ayrıt ve $\rho_a f_a (a \in A)$ ayrıtların maliyetleri ve L tane rota ayrıtı ve $g_l (l \in L)$ iş birliği dışı maliyetleri ile tanımlana şebeke verildiğinde seçilen rota ayrıtları kümesini kapsayan ve toplam çevrim maliyetini ve toplam iş birliği dışı maliyeti en küçükleyen çevrimleri bulma ve *kararlılık* koşulunu sağlayan, seçilen rota ayrıtlarına düşen maliyeti bulma problemi olarak

tanımlanabilir öyle ki her çevrim aşağıdaki koşuların hepsini sağlar:

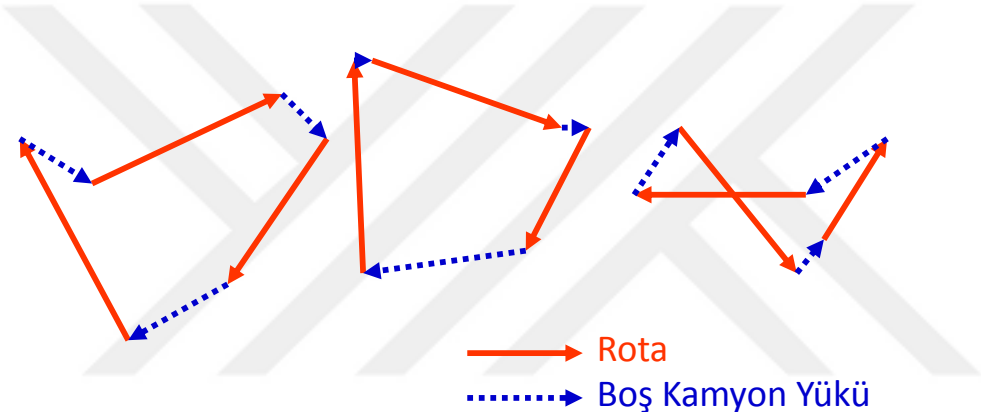
1. Her çevrim en fazla K_{max} tane rota ayrıtını kapsayabilir.
2. Her çevrimin uzunluğu en fazla L_{max} kadar olabilir.
3. Her çevrim içerisindeki düğümler en fazla bir kere ziyaret edilebilir.

MD-SUKRK problemini formüle etmek için daha fazla kümeye, parametreye, karar değişkenine ve kısıta ihtiyaç vardır. Problemde kullanılan kümeler, parametreler ve karar değişkenleri Tablo 3.1 'de tanımlanmıştır.

Tablo 3.1: *MD-SUKRK* Probleminde Kullanılan Notasyonlar

Kümeler	
N	Düğümler kümesi
A	Ayrıtlar kümesi
L	Rota ayrıtları kümesi ($L \subset A$)
$C(L)$	Bütün olurlu çevrimler kümesi
Parametreler	
θ_l	$l \in L$ rota ayrıtı için yüzde tasarrufu garanti eden katsayı ($\theta_l \in [0,1]$)
λ_l	$l \in L$ rota ayrıtı için alt sınır katsayısı ($\lambda_l \in [0,1]$)
ρ_a	$a \in A$ ayrıtı için birim boş gitme maliyeti ($\rho_a \in [0,1]$)
η_l	$l \in L$ rota ayrıtı için birim dolu gitme maliyeti ($\eta_l \in [0,1]$)
f_a	$a \in A$ ayrıtının uzunluğu
f_l	$l \in L$ rota ayrıtının uzunluğu
f_c	$c \in C(L)$ çevriminin maliyeti
g_l	$l \in L$ rota ayrıtının iş birliği dışı maliyeti ($g_l = (\eta_l + \rho_l) f_l$)
s_{lc}	1 eğer $l \in L$ rota ayrıtı $c \in C(L)$ çevrimi içerisindeyse, 0 diğer durumda
Karar Değişkenleri	
x_c	1 eğer $c \in C(L)$ çevrimi seçilmişse, 0 diğer durumda
u_l	1 eğer $l \in L$ rota ayrıtı kapsanmamışsa, 0 diğer durumda
w_l	$l \in L$ rota ayrıtına düşen maliyet

Yukarıda verilen ayrıtlar kümesi, hem gönderileri hem de boş kamyon yükü hareketini temsil etmektedir. Rota ayrıtlarının kümesi ise yalnızca gönderileri yani tam dolu kamyon yükü hareketini temsil etmektedir. Çevrimlerin kümesi, her bir rota ayrıtından başlayarak, rota ayrıtları uç uca eklenerek oluşturulur. Gerekli durumlarda düğümler arasında boş hareket ayrıtları kullanılır. Rota ayrıtları uç uca eklendiğinde bir yol (path) oluşur. Bu yolun son düğümünden ilk düğümüne bir boş hareket ayrıtı eklenerek çevrimler (cycle) oluşturulmuş olur. Bu tarz çevrimlerin nasıl oluşturulduğu Şekil 3.2 'de gösterilmektedir.



Şekil 3.2: Rota Kapsama Probleminde Kullanılan Çevrimler

Bu çalışma yapılırken, bazı varsayımlarda bulunmaya ihtiyaç duyulmuştur. Yapılan bu varsayımlar aşağıda listelendiği gibidir:

1. Altta yatan yönlü çizge tam çizgedir.
2. Düğümler, iki boyutlu bir uzayda (x, y) koordinatlarıyla tanımlanmıştır.
3. İş birliğine katılacak olan rota ayrıtları önceden belirlidir.
4. Rota ayrıtları arasındaki taşıma bir kamyonla yapılmaktadır.
5. Gönderiler ve sürücüler üzerinde herhangi bir zaman kısıtı yoktur.

6. İş birliği oluşturulduktan sonra, kurulan iş birliğine yeni bir rota ayrıtı eklenemez.

7. Her firmanın birden fazla rota ayrıtı olabilir. Ancak, her rota ayrıtı yalnızca bir firmaya aittir.

8. Düğümler arasındaki uzaklıkların hesaplanmasında *Öklid* mesafesi kullanılmıştır.

9. İki düğüm arasındaki taşıma maliyeti, düğümler arasındaki uzaklıkla doğru orantılıdır. Ayrıca, düğümler arasındaki uzaklık simetriktir ve üçgensel eşitsizliğe uygundur.

10. İş birliği maliyetinin, yalnızca çevrimlerin uzunluğuna bağlı olduğu varsayılmıştır.

11. İş birliği dışında kalan rota ayrıtına atanan maliyetin sıfır olduğu varsayılmıştır.

Yeni parametrelerin ve karar değişkeninin tanımlanmasıyla oluşan *MD-SUKRK* problemi aşağıda verilmektedir.

Matematiksel Model

$$\min \sum_{c \in C(L)} f_c x_c + \sum_{l \in L} g_l u_l \tag{3.11}$$

$$\text{s.t.} \sum_{l \in L} w_l = \sum_{c \in C(L)} f_c x_c \tag{3.12}$$

$$\sum_{l \in L} s_{lc} w_l \leq f_c \qquad \forall c \in C(L) \tag{3.13}$$

$$w_l \leq (1-\theta_l) g_l (1-u_l) \qquad \forall l \in L \tag{3.14}$$

$$w_l \geq \lambda_l f_l (1-u_l) \qquad \forall l \in L \tag{3.15}$$

$$\sum_{c \in C(L)} s_{lc} x_c = (1 - u_l) \qquad \forall l \in L \qquad (3.16)$$

$$x_c \in \{0,1\} \qquad \forall c \in C(L) \qquad (3.17)$$

$$u_l \in \{0,1\} \qquad \forall l \in L \qquad (3.18)$$

$$w_l \geq 0 \qquad \forall l \in L \qquad (3.19)$$

MD-SUKRK probleminde amaç (3.11), toplam çevrim maliyetini ve toplam iş birliği dışı maliyeti en küçüklemektir. Kısıt 3.12, *bütçe denge* kısıtıdır. Bu kısıt, dağıtılan toplam maliyetin iş birliğinden elde edilen toplam maliyete eşit olmasını sağlar. Bu kısıt, dağıtılan maliyetin eksik veya fazla olmasına izin vermez. Kısıt 3.13, *kararlılık* kısıtıdır. Bu kısıt, rota ayrıtlarına atanan toplam maliyetin; herhangi bir çevrimin maliyetinden küçük olmasını sağlar. Bu kısıt, her bir çevrim için yazıldığı için; modele eklenen her bir çevrime karşılık gelen *kararlılık* kısıtı da modele eklenmelidir. Kısıt 3.12 ve 3.13, birlikte *çekirdekte* yer alan bir maliyet dağıtımın bulunmasını sağlar. Kısıt 3.14, *bireysel rasyonellik* kısıtıdır. Bu kısıt, aynı zamanda yüzde tasarrufu da garanti eder. Başka bir değişle, eğer bir rota ayrıtı iş birliğine dahil edilmişse; o rota ayrıtına düşen maliyet kendi bireysel maliyetinin belirli bir yüzdesinden fazla olamaz. Bu kısıt, iş birliğine dahil olan rota ayrıtının en az yüzde θ_l kadar tasarruf etmesini de sağlar. Kısıt 3.15, seçilen rota ayrıtlarına sıfır maliyet atanmasını engeller. Başka bir deyişle, seçilen rota ayrıtlarına atanan maliyetler için bir alt sınır tanımlar. Seçilen rota ayrıtlarına atanan en düşük maliyet; o rota ayrıtının uzunluğu ile doğru orantılıdır. Kısıt 3.16, rota kapsama kısıtıdır. Eğer bir çevrim seçilmişse o çevrim içerisindeki rota ayrıtları bu çevrim tarafından kapsanmaktadır. Diğer kısıtlar ise tam sayı olma ve işaret kısıtıdır.

Yukarıdaki formülasyonlarda, rota ayrıtı sayısı arttıkça satır ve sütun sayısı üssel olarak artar. Polinom sayıda rota ayrıtı olduğu için; toplam sütun ve satır sayısını,

olurlu çevrimlerin sayısı belirler. Olurlu çevrim sayısını da, rota ayrıtı sayısı belirler. Bu nedenle, rota ayrıtı sayısının artması, sütun ve satır sayısını üssel olarak arttırır. En kötü durumda, rota ayrıtları kümesi; boş olmayan her bir alt kümesine karşılık gelen bir olurlu çevrim olur ve toplam $2^{|L|} - 1$ tane sütun oluşur.

Yukarıda verilen formülasyon, *patika bazlı (path-based)* bir formülasyondur. Dolayısıyla bütün olurlu çevrimlerin üretilmesi gerekmektedir. Problemin çözümü için üretilecek çevrimler belirli kısıtlar altında üretilmelidir. Bunlardan ilki, basit çevrim kısıtıdır. Basit çevrim kısıtı, her bir düğümün yalnızca bir kere ziyaret edilmesini sağlar. Basit olmayan bir çevrim, toplam uzunlukları aynı olacak şekilde birden fazla basit çevrime bölünebilir. Dolayısıyla, basit olmayan çevrimler en iyi çözümde yer almayacaktır. Çevrimlerin kapsayabileceği rota ayrıtı sayısı da sınırlıdır. Çevrimler, en fazla belirli bir sayıda rota ayrıtını kapsayacak şekilde üretilirler. Üretilen çevrimlerin uzunlukları için de bir sınır tanımlanmıştır. Yalnızca uzunlukları belirli bir değerden küçük olan çevrimler üretilir. Bu durumda, r_l ve t_a sırasıyla tam kamyon yükü ve boş kamyon yükü hareketini temsil etmek üzere; K_{max} bir çevrimin kapsayabileceği maksimum rota ayrıtı sayı ve L_{max} bir çevrimin maksimum uzunluğu olmak üzere ortaya çıkan sayı kısıtlı (3.20) ve uzunluk kısıtlı (3.21) çevrimler matematiksel olarak aşağıda verilen şekilde ifade edilebilir.

$$\sum_{l \in L_c} r_l \leq K_{max} \qquad \forall c \in C(L) \qquad (3.20)$$

$$\sum_{l \in L_c} f_l r_l + \sum_{a \in A_c} f_a t_a \leq L_{max} \qquad \forall c \in C(L) \qquad (3.21)$$

Düğümleri farklı sıralarda ziyaret eden aynı çevrimlerin üretilmesi mümkündür. Üretilen aynı çevrimlerden yalnızca bir tanesi dikkate alınır. Düğümleri farklı sıralarda ziyaret eden farklı çevrimlerin üretilmesi de mümkündür. Bu gibi durumlarda, bu çevrimler arasından en düşük maliyete sahip çevrim dikkate alınır.

Çünkü, en düşük maliyete sahip çevrim en iyi çözümde yer alabilir. Ayrıca bir çevrimin dolu gitme maliyetinin o çevrimin toplam maliyetine oranı 0.5 'den küçük olan çevrimler dikkate alınmayacaktır. Çünkü bu tip çevrimler de en iyi çözümde yer almayacaktır. Olurlu çevrim oluştururken bu gibi kısıtlar koymak, üretilecek çevrimlerin sayısını azaltırken bu kurallara uyan olurlu çevrimleri bulmak zorlaşacaktır.

3.5 Çok Koalisyonlu Maliyet Dağıtımlı Sayı ve Uzunluk Kısıtlı Rota Kaplama Problemi

Tek koalisyonlu yapıda *kararlı* bir maliyet dağıtımı bulabilmek için birçok rota ayrıtı koalisyon dışında bırakılabilir. Bu nedenle, bu durumun önüne geçebilmek için; model çok koalisyonlu yapı için uygun hale getirilmiştir. Çok koalisyonlu bir yapıda izin verilen koalisyon sayısına bağlı olarak; kapsanan rota sayısı yüksek olacaktır.

MD-SUKRK problemi gibi, bu problem de **NP-Zor** bir problemdir. Çünkü, izin verilen koalisyon sayısı bir olduğu durumda, problem *MD-SUKRK* problemine dönüşmektedir. *MD-SUKRK* problemi baz alınarak çok koalisyonlu bir model tasarlanmıştır. Tasarlanan bu model *Çok Koalisyonlu Maliyet Dağıtımlı Sayı ve Uzunluk Kısıtlı Rota Kapsama (ÇKMD-SUKRK)* olarak adlandırılmıştır. Bu modeli formüle etmek için, karar değişkenlerinin yeniden tanımlanması gerekir. *MD-SUKRK* probleminde çevrimler üzerine uygulanan kısıtlar bu problem için de geçerlidir. T koalisyon kümesi olmak üzere *ÇKMD-SUKRK* probleminde kullanılan karar değişkenleri Tablo 3.2 'de tanımlanmıştır.

Tablo 3.2: *ÇKMD-SUKRK* Probleminde Kullanılan Karar Değişkenleri

Karar Değişkenleri	
x_{ct}	1 eğer $c \in C(L)$ çevrimi $t \in T$ koalisyonuna atanmışsa, 0 diğer durumda
z_l	1 eğer $l \in L$ rota ayrıtı kapsanmışsa, 0 diğer durumda
u_{lt}	$l \in L$ rota ayrıtı $t \in T$ koalisyonuna atanmışsa, 0 diğer durumda
w_{lt}	$l \in L$ rota ayrıtına $t \in T$ koalisyonunda düşen maliyet

Yeni parametrelerin ve karar değişkeninin tanımlanmasıyla oluşan *ÇKMD-SUKRK* problemi aşağıda verilmektedir.

Matematiksel Model

$$\min \sum_{c \in C(L)} \sum_{t \in T} f_c x_{ct} + \sum_{l \in L} g_l (1 - z_l) \tag{3.22}$$

$$\text{s.t.} \sum_{l \in L} \sum_{t \in T} w_{lt} = \sum_{c \in C(L)} \sum_{t \in T} f_c x_{ct} \tag{3.23}$$

$$\sum_{l \in L} s_{lc} w_{lt} \leq f_c \qquad \forall c \in C(L), t \in T \tag{3.24}$$

$$w_{lt} \leq (1 - \theta_l) g_l u_{lt} \qquad \forall l \in L, t \in T \tag{3.25}$$

$$w_{lt} \geq \lambda_l f_l u_{lt} \qquad \forall l \in L, t \in T \tag{3.26}$$

$$\sum_{c \in C(L)} s_{lc} x_{ct} = u_{lt} \qquad \forall l \in L, t \in T \tag{3.27}$$

$$\sum_{t \in T} u_{lt} \leq 1 \qquad \forall l \in L \tag{3.28}$$

$$\sum_{t \in T} u_{lt} = z_l \qquad \forall l \in L \tag{3.29}$$

$$z_l \in \{0,1\} \qquad \forall l \in L \tag{3.30}$$

$$u_{lt} \in \{0,1\} \qquad \forall l \in L, t \in T \tag{3.31}$$

$$w_{lt} \geq 0 \qquad \forall l \in L, t \in T \tag{3.32}$$

$$x_{ct} \in \{0,1\} \qquad \forall c \in C(L), t \in T \tag{3.33}$$

ÇKMD-SUKRK amaç (3.22), *MD-SUKRK* probleminde olduğu gibi; toplam çev-

rim maliyetini ve toplam iş birliği dışı maliyeti en küçüklemektir. *ÇKMD-SUKRK* 'de yer alan kısıt 3.23 - 3.27, *MD-SUKRK* 'de yer alan kısıtlarla benzerdir. Bu modeldeki fark, bu kısıtların her koalisyon için de geçerli olmasıdır. Kısıt 3.23, *bütçe denge* kısıtıdır. Bu kısıt, dağıtılan toplam maliyetin iş birliğinden elde edilen toplam maliyete eşit olmasını sağlar. Kısıt 3.24, *kararlılık* kısıtıdır. Kısıt 3.25, *bireysel rasyonellik* kısıtıdır. Bu kısıt aynı zamanda yüzde tasarrufu da garanti eder. Kısıt 3.26, seçilen rota ayrıtlarına atanan maliyetler için bir alt sınır verir. Kısıt 3.27, rota kapsama kısıtıdır. Eğer bir çevrim bir koalisyon için seçilmişse, o çevrim içerisindeki rota ayrıtları bu çevrim tarafından kapsanmaktadır. Kısıt 3.28, her bir rota ayrıtının en fazla bir koalisyona atanmasını sağlar. Kısıt 3.29, eğer bir rota ayrıtı herhangi bir koalisyona atanmış ise; bu rota ayrıtının kapsanmasını sağlar. Kısıt 3.30 - 3.33, ise tam sayı olma ve işaret kısıtıdır.

3.6 *MD-SUKRK* Problemi için Alt Sınır

MD-SUKRK probleminde, optimal çözümün elde edilmesi oldukça zordur. Özellikle rota ayrıtlarının sayısının artması, üretilecek olurlu çevrimlerin sayısını üssel olarak arttıracaktır. Dolayısıyla modeldeki satır ve sütun sayısı yani değişken ve kısıt sayısı da üssel biçimde artacaktır. Bu nedenle, geliştirilen çözüm yönteminden elde edilen sonuçların; optimal çözümle karşılaştırılması mümkün olmayacaktır.

Alt sınır elde etmek için basit bir atama probleminin çözülmesi yeterlidir. Bu atama problemi, rota ayrıtları arasındaki toplam boş gitme maliyetini minimize ederek; her rota ayrıtını bir rota ayrıtına atar. Alt sınır elde etmek için, bu problemin amaç fonksiyonu değerine rota ayrıtlarının toplam dolu gitme maliyeti eklenir. Bu problem, aslında *RK* probleminin atama problemine dönüştürülmüş halidir. Atama problemine çevirebilmek için gerekli parametre ve karar değişkeni

Tablo 3.3 'de tanımlanmaktadır.

Tablo 3.3: Alt Sınır için Kullanılan Parametre ve Karar Değişkeni

Parametre	
c_{kl}	1 $k \in L$ rota ayrıtı ile $l \in L-\{k\}$ rota ayrıtı arasındaki boş gitmenin maliyeti
Karar Değişkeni	
x_{kl}	1 eğer $k \in L$ rota ayrıtı, $l \in L-\{l\}$ rota ayrıtına atanmışsa, 0 diğer durumda

Yukarıda tanımlanan matematiksel model aşağıda verildiği gibidir.

$$\min \sum_{k \in L} \sum_{l \in L-\{k\}} c_{kl} x_{kl} \tag{3.34}$$

$$\text{s.t.} \sum_{l \in L} x_{lk} = 1 \qquad \forall k \in L-\{l\} \tag{3.35}$$

$$\sum_{k \in L} x_{lk} = 1 \qquad \forall l \in L-\{k\} \tag{3.36}$$

$$x_{kl} \in \{0,1\} \qquad \forall k \in L, l \in L-\{k\} \tag{3.37}$$

Problemin amacı (3.34), toplam boş gitme maliyetini en küçüklemektir. Kısıt 3.35 ve 3.36 atama kısıtlarıdır. Her bir rota ayrıtının tam olarak bir rota ayrıtına atanmasını sağlar. Kısıt 3.37 ise tam sayı olma kısıtıdır.

x^*_{kl} 'lar bu problemden elde edilen optimal çözüm olmak üzere; alt sınır (AS) aşağıdaki şekilde hesaplanır.

$$AS = \sum_{k \in L} \sum_{l \in L-\{k\}} c_{kl} x^*_{kl} + \sum_{l \in L} \eta_l f_l \tag{3.38}$$

4. ÇÖZÜM YÖNTEMİ

MD-SUKRK problemini çözmek için akla gelen ilk yöntem, bütün olurlu çevrimleri üreterek; problemi tam sayılı olarak ticari bir çözücü kullanarak çözmektir. Bu yöntemde, bir rota ayrıtından başlayarak; rota ayrıtları uç uça eklenir. Gerekli durumlarda, düğümler arasında ayrıtlar kullanılarak olurlu çevrimler oluşturulur. Bir çevrimin olurlu olabilmesi için bir önceki bölümde bahsedilen bütün kısıtların sağlaması gereklidir. Bu yolla elde edilen bütün olurlu çevrimler modele eklenir.

Bu şekilde elde edilen çevrimlerin sayısı, rota ayrıtı sayısı arttıkça üssel olarak artacaktır. Örneğin 20 rota ayrıtı içeren bir örnek için oluşturulacak çevrim sayısı en kötü durumda 2^{20} olacaktır. Bu da $1,048,575$ adet çevrimin üretilmesi anlamına gelir. Rota ayrıtı sayısının bir artması durumunda bu sayı iki katına çıkacaktır.

Çevrimlerin sayısının üssel bir şekilde artması demek satır ve sütun sayısının da üssel bir şekilde artması demektir. Çünkü üretilecek her bir çevrim, bir değişkene (x_c) ve bir kısıta (3.13) karşılık gelir. Burada satır sayısından kast edilen kısıt sayısı, sütun sayısından kast edilen ise değişken sayısıdır. Dolayısıyla bu problemin optimal çözümünü makul sürelerde bulmak bazı örnekler için mümkün olmayacaktır. Ayrıca bazı örnekler için, makul sürelerde olurlu bir çözüm elde etmek bile zor olacaktır.

Bu gibi çok sayıda karar değişkeni ve kısıt içeren matematiksel modelleri çözmek için sütun ve satır türetme bazlı yöntemler kullanılır. Bu problemde sütun türetme bazlı bir çözüm yöntemi kullanmak tek başına yeterli olmayacaktır. Sütun türetme bazlı çözüm yönteminin yanında, satır türetme bazlı bir yönteme de ihtiyaç vardır. Bu nedenle satır türetme bazlı çözüm yöntemi de geliştirilmiştir.

Bu çözüm yöntemini kullanmak, bütün olurlu çevrimleri üretmek için harcanacak

zamanı ciddi şekilde azaltacaktır. Ancak bu çözüm yöntemini tek başına kullanmak, *MD-SUKRK* problemi için olurlu veya optimal çözüm almak için yeterli değildir. Çünkü bu yöntem yalnızca problemin doğrusal gevşetilmiş hali için optimal çözüm verir. Dolayısıyla tam sayılı çözümler vermeyecektir. Optimal veya tam sayılı (olurlu) çözümler elde edebilmek için çeşitli sezgisel yöntemlere veya dal-sınır (branch and bound) bazlı çözüm yöntemlerine ihtiyaç vardır.

4.1 Sütun Türetme Çözüm Yöntemi

Sütun türetmede, karar değişkenlerine karşılık gelen bütün sütunlar yerine; sütunların sınırlı sayıdaki alt kümelerini modele dahil ederek, modelin en iyi çözümünün bulunması hedeflenir. Sütun türetme, matematiksel modelin sınırlı sayıda sütun için çözülmesi ile başlar.

Elde edilen en iyi çözüm kullanılarak, modele henüz dahil olmayan ancak temele girebilecek yani negatif indirgenmiş maliyetli bir sütun bulunarak modele eklenir. Daha sonra pivotlama işlemi yapılır. Negatif indirgenmiş maliyetli bir değişken bulunamadığında çözüm yöntemi durdurulur. Elde edilen son çözüm, modelin doğrusal gevşetmesi için en iyi çözümdür.

Sütun türetme, sınırlı sayıda sütun üzerinden çözülür. Sınırlı sayıda sütun üzerinden çözülen orjinal problem *sınırlandırılmış problem* (*restricted problem*) denir. Negatif indirgenmiş maliyetli yeni sütunlar bulunamadığında, negatif indirgenmiş maliyetli yeni sütunlar bulmayı amaçlayan ikinci bir en iyileme modeli çözülür. Bu ikinci en iyileme modeline *fiyatlandırma problemi* (*pricing problem*) denir.

Bu yöntem için problemin doğrusal gevşetilmiş haline ihtiyaç vardır. Problemin doğrusal gevşetilmiş hali kullanarak indirgenmiş maliyetler hesaplanacaktır. Bu nedenle modelde yer alan tam sayılı değişkenlerin gevşetilmesi gerekmektedir.

Bu değişkenlerin gevşetilmiş halleri aşağıda verildiği gibidir.

$$0 \leq x_c \qquad \forall c \in C(L) \tag{4.1}$$

$$0 \leq u_l \leq 1 \qquad \forall l \in L \tag{4.2}$$

x_c ikili değişkenini $x_c \geq 0$ şeklinde doğrusallaştırmakta herhangi bir sakınca yoktur. Çünkü kapsama kısıtından (3.16) ve problemin en küçükleme problemi olmasından dolayı, x_c değişkeni birden büyük değer almayacaktır. Modelin bu şekilde değiştirilmesi, indirgenmiş maliyet hesabını kolaylaştıracaktır.

4.1.1 Fiyatlandırma Problemi

Fiyatlandırma probleminin formülasyonu, türetilecek sütunların (çevrimlerin) indirgenmiş maliyetlerine göre oluşturulur. *MD-SUKRK* probleminde, rota ayrıtlarının seçilip seçilmediğine karar veren değişkenlere (u_l) karşılık gelen sütunlar sınırlı sayıdadır ve bunlar sabit tutulur. Çevrimlerin seçilip seçilmediğine karşılık gelen değişkenler (x_c) üzerinden fiyatlandırma yapılır. Bir çevrim $(c \in C(L))$ için indirgenmiş maliyet matematiksel modelin kısıtlarına karşılık gelen dual değişkenler üzerinden tanımlanır.

μ ve π_l $(l \in L)$ sırasıyla ana modeldeki x_c değişkenini içeren, 3.12 ve 3.16 kısıtlarına karşılık gelen dual değişkenler olsunlar. Bu durumda, her bir çevrim için $(c \in C(L))$ indirgenmiş maliyet $\overline{f_c}$ aşağıdaki gibi hesaplanır.

$$\overline{f_c} = f_c + \mu f_c - \sum_{l \in L} s_{lc} \pi_l \tag{4.3}$$

Fiyatlandırma problemi genel olarak aşağıdaki gibi modellenebilir. Aşağıdaki modelden de anlaşılabileceği gibi amaç, bütün kısıtları sağlayan en düşük indirgen-

miş maliyetli olurlu çevrimleri bulma problemidir.

$$\begin{aligned} min \quad & \overline{f_c} \\ s.t. \quad & c \in C(L) \end{aligned}$$

Fiyatlandırma problemini daha açık ifade edebilmek için yeni karar değişkenlerine ihtiyaç vardır. Yeni karar değişkenleri Tablo 4.1 'de tanımlandığı gibidir.

Tablo 4.1: Fiyatlandırma Probleminde Kullanılan Karar Değişkenleri

Karar Değişkenleri	
r_l	1 eğer $l \in L$ rota ayrıtı, tam dolu kamyon hareketi için seçilmişse 0 diğer durumda
t_a	1 eğer $a \in A$ ayrıtı boş kamyon hareketi için seçilmişse 0 diğer durumda

Matematiksel Model

$$\min \sum_{l \in L}((1+\mu)\eta_l f_l - \pi_l)r_l + \sum_{a \in A}(1+\mu)\rho_a f_a t_a \tag{4.4}$$

$$\text{s.t.} \sum_{(m,n)\in L} r_{(m,n)} + \sum_{(m,n)\in A} t_{(m,n)} - \sum_{(n,m)\in L} r_{(n,m)} - \sum_{(n,m)\in A} t_{(n,m)} = 0 \qquad \forall n \in N \tag{4.5}$$

$$\sum_{(n,m)\in L} r_{(n,m)} + \sum_{(n,m)\in A} t_{(n,m)} \leq 1 \qquad \forall n \in N \tag{4.6}$$

$$\sum_{(n,m)\in A} t_{(n,m)} + \sum_{(k,n)\in A} t_{(k,n)} \leq 1 \qquad \forall n \in N \tag{4.7}$$

$$\sum_{l \in L} r_l \leq K_{max} \tag{4.8}$$

$$\sum_{l \in L} f_l r_l + \sum_{a \in A} f_a t_a \leq L_{max} \tag{4.9}$$

$$r_l \in \{0,1\} \qquad \forall l \in L \tag{4.10}$$

$$t_a \in \{0,1\} \qquad \forall a \in A \tag{4.11}$$

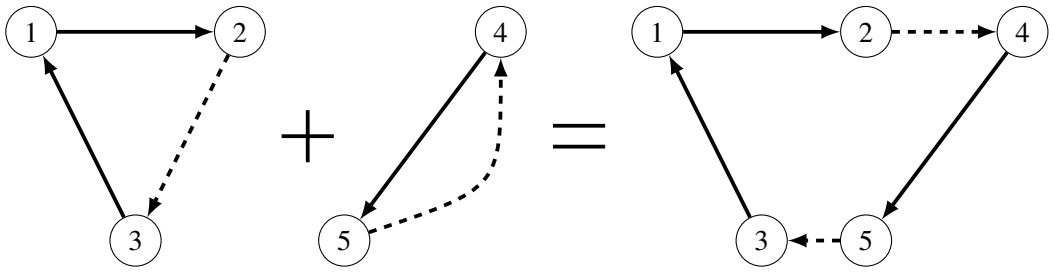
Fiyatlandırma probleminin amacı (4.4), negatif indirgenmiş maliyetli çevrimlerin

bulunmasıdır. Kısıt 4.5, akış denge kısıtı olarak düşünülebilir. Bu kısıt, bir düğüme giren ve çıkan toplam ayrıt sayısının eşit olmasını sağlar. Kısıt 4.6, basit çevrim kısıtıdır. Bu kısıt, her düğümün yalnızca bir defa ziyaret edilmesini sağlar. Kısıt 4.7, art arda iki boş kamyon yükü hareketinin seçilmesini engeller. Kısıt 4.8, oluşturulacak çevrimin kapsayabileceği rota ayrıtı sayısını kısıtlar. Kısıt 4.9, oluşturulacak çevrimin uzunluğunu kısıtlar. Geri kalan kısıtlar ise tam sayı olma kısıtıdır.

Yukarıda verilen fiyatlandırma problemi bir tam sayılı programlama modelidir. Dolayısıyla bu problemin optimal çözümünü elde etmek zaman alacaktır. Bu nedenle, dört farklı sezgisel yöntem geliştirilmiştir. Negatif indirgenmiş çevrimler, bu sezgisel yöntemler kullanılarak aranacaktır. Fiyatlandırma problemi, yalnızca bu sezgisel yöntemlerden negatif indirgenmiş maliyetli çevrimler bulunamadığında çözülecektir. Bu stratejiyle amaçlanan, fiyatlandırma problemine ihtiyaç duymadan hızlı bir şekilde temele girmeye aday olurlu çevrimleri üretmektir.

4.1.2 Birinci Sezgisel Yöntem: Sona Ekleme

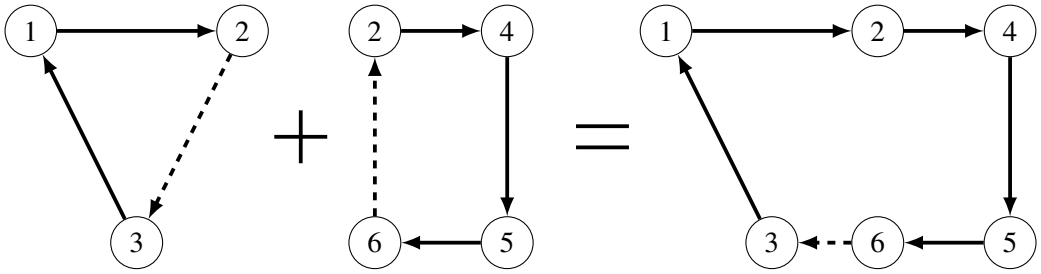
Bu sezgiselde, sınırlandırılmış problemin optimal çözümünden elde edilen ve temelde yer alan sütunlara karşılık gelen çevrimlere, rota ayrıtları yani tek boyutlu çevrimler (tek bir rota ayrıtından oluşan) sırayla eklenerek; negatif indirgenmiş maliyetli çevrimler türetilmeye çalışılır. Bu yöntemin zaman karmaşıklığı (time complexity) $O(|L| \times |C_b(L)|)$ mertebesindedir. Burada, $C_b(L)$ temelde yer alan çevrimlerin kümesini ifade etmektedir. Temelde yer alan bir çevrime bir rota ayrıtının nasıl eklenebileceği Şekil 4.1 'de gösterilmektedir.



Şekil 4.1: Temelde Yer Alan Bir Çevrime Bir Rota Ayrıtının Eklenmesi

4.1.3 İkinci Sezgisel Yöntem: Birleştirme

Bu sezgiselde, sınırlandırılmış problemin optimal çözümünden elde edilen ve temelde bulunan sütunlara karşılık gelen çevrimler birbirleriyle birleştirilir. Başka bir deyişle sınırlandırılmış problemin optimal çözümde yer alan çevrimler birleştirilerek negatif indirgenmiş maliyetli çevrimler türetilmeye çalışılır. Bu yöntemin zaman karmaşıklığı $O(|C_b(L)|^2)$ mertebesindedir. İki ayrık çevrimin nasıl birleştirilebileceği Şekil 4.2 'de gösterilmektedir.



Şekil 4.2: İki Ayrık Çevrimin Birleştirilmesi

4.1.4 Üçüncü Sezgisel Yöntem: Çapraz Birleştirme

Bu sezgiselde, sınırlandırılmış problemin optimal çözümünden elde edilen ve temelde bulunan sütunlara karşılık gelen çevrimler ile temelde olmayan ve tek boyutlu olamayan çevrimler sırayla birleştirilerek; negatif indirgenmiş maliyetli çev-

rimler türetilmeye çalışılır. Bu yöntemin zaman karmaşıklığı $O(|C_b(L)| \times |C_{nb}(L)|)$ mertebesindedir. Burada, $C_{nb}(L)$ temelde yer almayan çevrimlerin kümesini ifade etmektedir.

4.1.5 Dördüncü Sezgisel Yöntem: Araya Ekleme

Bu yöntem, birinci sezgisel yöntemin genel halidir. Bu sezgiselde, tek boyutlu çevrimler; her bir çevrimin bütün olası konumlarına eklenerek; negatif indirgenmiş maliyetli çevrimler türetilmeye çalışılır. *Araya Ekleme* sezgiseli, diğer yöntemlere kıyasla daha geniş komşuluk tanımlar. Bu nedenle, diğerlerine kıyasla daha uzun sürecektir. Sütun türetmenin etkinliğini arttırmak amacıyla bu yöntem en son olarak denenecektir. Bu yöntemin zaman karmaşıklığı $O(K_{max} \times |L| \times |C(L)|)$ mertebesindedir.

Yukarıda tanımlanan sezgiseller, verildikleri sıraya göre teker teker denenir. Herhangi bir sezgiselden negatif indirgenmiş maliyetli bir çevrim üretilirse, diğerleri kullanılmaz. Eğer birden fazla böyle çevrim üretilirse, içlerinden en negatif indirgenmiş maliyete sahip çevrim modele eklenir.

4.1.6 Yeni Çevrimlerin İndirgenmiş Maliyetlerinin Hesaplanması

İki ayrık çevrimin birleşmesinden oluşan yeni çevrimin indirgenmiş maliyeti 4.3 'de verilen formülle hesaplanamaz. Bu formülün yeniden ifade edilmesi gerekir. Yeni formülasyonu elde etmek için aşağıdaki adımlar izlenir. Sınırlandırılmış problemin optimal çözümünde yer alan iki ayrık c ve d çevrimlerinin indirgenmiş maliyetleri sıfır olduğu için aşağıdaki eşitlikler yazılabilir.

$$0 = (1+\mu) f_c - \sum_{l \in L_c} \pi_l \tag{4.12}$$

$$0 = (1+\mu) f_d - \sum_{l \in L_d} \pi_l \tag{4.13}$$

$$\sum_{l \in L_c} \pi_l = (1+\mu) f_c \tag{4.14}$$

$$\sum_{l \in L_d} \pi_l = (1+\mu) f_d \tag{4.15}$$

Temelde yer alan iki ayrık çevrimin birleştirilmesinden elde edilen yeni çevrimin indirgenmiş maliyeti ($\overline{f_e}$) aşağıdaki gibi hesaplanır.

$$\overline{f_e} = (1+\mu) f_e - \sum_{l \in L_c} \pi_l - \sum_{l \in L_d} \pi_l \tag{4.16}$$

Bu iki çevrimin indirgenmiş maliyetlerinin sıfır olmasından faydalanarak yukarıdaki eşitlik yeniden düzenlendiğinde aşağıdaki ifade karşımıza çıkar.

$$\overline{f_e} = f_e - f_c - f_d + \mu \left(f_e - f_c - f_d \right) \tag{4.17}$$

Yukarıda verilen indirgenmiş maliyet hesabı, temelde yer alan iki çevrim çiftinin birleştirildiği durum için geçerlidir. Temelde yer alan bir çevrim ile temelde yer almayan bir çevrim birleştirildiğinde yeni çevrimin indirgenmiş maliyeti yukarıdaki gibi hesaplanamaz. Bu durumda, temelde yer alan çevrimin indirgenmiş maliyeti sıfırdır. Ancak temelde yer almayan çevrimin indirgenmiş maliyeti sıfırdan farklı olabilir. Temelde yer alan c çevrimi ve temelde yer almayan d çevrimi için aşağıdaki eşitlikler yazılabilir.

$$0 = (1+\mu) f_c - \sum_{l \in L_c} \pi_l \tag{4.18}$$

$$\overline{f_d} = (1+\mu) f_d - \sum_{l \in L_d} \pi_l \tag{4.19}$$

$$\sum_{l \in L_c} \pi_l = (1+\mu) f_c \tag{4.20}$$

$$\sum_{l \in L_d} \pi_l = -\overline{f_d} + (1+\mu) f_d \tag{4.21}$$

Yukarıdaki adımlar burada da uygulanırsa, bu çevrim çiftlerinin birleştirilmesinden elde edilen yeni çevrimin indirgenmiş maliyeti ($\overline{f_e}$) aşağıdaki gibi hesaplanır.

$$\overline{f_e} = f_e - f_c - f_d + \mu\,(f_e - f_c - f_d) + \overline{f_d} \tag{4.22}$$

Yukarıda verilen aşamalar sırasıyla temelde yer almayan iki çevrimin birleştirilmesi içinde yapılabilir. Temelde yer almayan iki çevrim çifti sırasıyla *c* ve *d* olsun. Bu durumda, bu iki çevrimin birleştirilmesiyle oluşan *e* çevriminin indirgenmiş maliyeti aşağıda verildiği gibi hesaplanır.

$$\overline{f_e} = f_e - f_c - f_d + \mu\,(f_e - f_c - f_d) + \overline{f_c} + \overline{f_d} \tag{4.23}$$

Yukarıda tanımlanan eşitlikler, kullanılan sezgiselin komşuluk tanımına göre kullanılır. Eğer temelde yer alan iki çevrim çifti birleştirilmiş ise, ortaya çıkan yeni çevrim için indirgenmiş maliyet hesabında *Eşitlik 4.17* kullanılır. Eğer temelde yer alan ve temelde yer almayan çevrim çiftleri birleştirilirse *Eşitlik 4.22* kullanılır. Eğer temelde yer almayan iki çevrim çifti birleştirilmiş ise, ortaya çıkan yeni çevrim için indirgenmiş maliyet hesabında *Eşitlik 4.23* kullanılır.

4.2 Satır Türetme Çözüm Yöntemi

Model başlangıçta sınırlı sayıda çevrim ile başladığı için, bütün *kararlılık* kısıtları modele dahil edilmemektedir. Bu kısıtlar, tıpkı sütun türetmede olduğu gibi ihtiyaç duyulduğunda modele eklenmektedir. *MD-SUKRK* problemin doğrusal gevşetilmiş modelinden elde edilen en iyi maliyet dağıtımları $\tilde{w}_l$ olsun. Bu maliyet dağıtımlarıyla ($\tilde{w}_l$), *kararlılık* kısıtını ihlal eden ve modelde yer almayan bir çev-

rimin olup olmadığı araştırılır.

Burada dikkat edilemesi gereken nokta; mevcut çevrimler için bu kısıt ihlal edilmese bile, modelde yer almayan yani daha önce üretilmemiş bir çevrim için de bu kısıt ihlal edilebilir. Böyle bir durumda, bu tarz çevrimlerin bulunması gerekir. Bu tarz çevrimler iki şekilde bulabilir. Akla gelen ilk yöntem bir tam sayılı programlama modeliyle, en fazla ihlal edilen çevrimin bulunmasıdır. Ancak bunun her seferinde yapılması her zaman mümkün değildir. Bu nedenle, bu tarz çevrimler sezgisel olarak bulunmaya çalışılır. İhtiyaç duyulması halinde tam sayılı modele başvurulur.

4.2.1 Satır Türetme İşlemi için Sezgisel Yöntemler

Bir önceki bölümde, sütun türetmek için tanımlanan dört sezgisel yöntem; satır türetmek için de kullanılabilir. Ancak bu sezgisellerde değişiklik yapılması gerekmektedir. Sütun türetmede amaçlanan, negatif indirgenmiş maliyetli çevrimler üretmekken; satır üretmekte amaç, *kararlılık* kısıtını ihlal eden çevrimleri (satırları) üretmektir.

Yukarıdaki sezgiseller kullanılarak elde edilen çevrimlerin, *kararlılık* kısıtını ihlal edip etmediğine bakılır. Yalnızca bu kısıtı ihlal eden çevrimler dikkate alınır. Modele ise, en yüksek ihlale sahip çevrim ve ona ait satır eklenir. İhlal miktarı, 4.24 'de verilen eşitliğe göre hesaplanır.

$$\sum_{l \in L_c} \tilde{w}_l - f_c \tag{4.24}$$

4.2.2 Satır Türetme Alt Problem

Sezgisel yöntemlerden *kararlılık* kısıtı ihlal eden bir çevrim bulunamadığı durumlarda, bu tarz çevrimleri bulmak için kesin çözüm yöntemine ihtiyaç vardır. Bunun için tam sayılı bir matematiksel model geliştirilmiştir. Geliştirilen bu modele *Satır Türetme Alt (STA)* problemi olarak adlandırılmıştır. *STA* problemi, fiyatlandırma problemine benzemektedir. Geliştirilen matematiksel yöntem aşağıda verilmektedir.

$$\min \sum_{l \in L} (\eta_l f_l - \tilde{w}_l) r_l + \sum_{a \in A} \rho_a f_a t_a \tag{4.25}$$
$$\text{s.t. (4.5)-(4.11)}$$

STA probleminin amacı (4.25), *kararlılık* kısıtını en fazla ihlal eden çevrimi bulmaktır. Geri kalan kısıtlar, fiyatlandırma probleminde kullanılan kısıtlarla aynıdır. Problemin amaç fonksiyonu negatif olduğu sürece, bu problemden istenilen çevrimler üretilebilecektir. Amaç fonksiyonu değerinin alabileceği maksimum değer sıfırdır. Amaç fonksiyonunun sıfır değerini alması, mevcut sütunlarla ve satırlarla elde edilen maliyet dağıtımıyla *kararlılık* kısıtını ihlal eden bir çevrimin üretilmeyeceği anlamına gelir.

Fiyatlandırma probleminde olduğu gibi *STA* problemi de bir tam sayılı programlama modelidir. Dolayısıyla bu problemin optimal çözümünü elde etmek zaman alacaktır. Bu nedenle, sezgisel yöntemler kullanılmıştır.

4.3 Satır ve Sütun Türetme Çözüm Yöntemi

Geliştirilen satır ve sütun türetme bazlı çözüm yöntemi yedi adımdan oluşmaktadır. İlk olarak, bütün olası çevrimleri üretmek yerine; bir boyutlu ve iki boyutlu

(bir ve iki rota ayrıtı kapsayan) olurlu çevrimler üretilir. Böylelikle, bütün olası çevrimleri üretmek için harcanan süre ciddi şekilde azaltılmış olur. Daha sonra, bu çevrimler, $C(L)$ kümesine eklenir ve bu $C(L)$ kümesinde bulunan çevrimler kullanılarak *MD-SUKRK* probleminin gevşetilmiş hali çözülür.

Çözüm yöntemi ilk olarak, *kararlılık* kısıtını (3.13) ihlal eden çevrimleri bulmayı amaçlar. Bunun için yukarıda tanımlanan dört sezgisel yöntem tanımlandıkları sıraya göre kullanılır. Bir sezgisellerden herhangi bir tanesinden kısıtı ihlal eden bir çevrim bulunursa, diğer sezgiseller kullanılmaz. Eğer birden fazla çevrim bulunursa, bu çevrimler arasından *kararlılık* kısıtını en fazla ihlal eden çevrim modele hem satır hem de sütun olarak eklenir. Yani bu çevrime karşılık gelen hem değişken hem de kısıt modele eklenmiş olur. Eğer sezgisel yöntemlerden bir çevrim üretilemez ise *STA* problemi optimal olarak çözülür. Buradan elde edilen çevrim hem satır hem de sütun olarak modele eklenir ve problemin doğrusal gevşetilmiş hali tekrar çözdürülür.

Eğer *kararlılık* kısıtını ihlal eden bir çevrim bulanamaz ise, sütun türetmek için geliştirilen sezgisel yöntemler kullanılarak sütun türetmeye çalışılır. Bu yöntemlerden herhangi bir tanesi negatif indirgenmiş maliyetli bir çevrim üretirse diğer yöntemler kullanılmaz. Üretilen çevrimler arasından indirgenmiş maliyeti en negatif olan çevrim modele hem değişken hem de kısıt olarak eklenir. Ekleme işlemi, *Simpleks* tablosundaki pivotlama işleminde olduğu gibi; en negatif indirgenmiş maliyetli çevrimin temele girmesidir. Eğer sezgisel yöntemlerden bir çevrim üretilemez ise fiyatlandırma problemi optimal olarak çözülür. Eğer bu problemden negatif indirgenmiş maliyetli bir çevrim üretilirse modele hem değişken hem de kısıt olarak eklenir ve problemin doğrusal gevşetilmiş hali tekrar çözdürülür.

STA ve fiyatlandırma problemini optimal olarak çözmek yerine, çözüm süresini kısaltmak için zaman kısıtı veya ilk elde edilen olurlu çözümü almak gibi çe-

şitli teknikler kullanılabilir. Bu durumda, bu problemler için harcanan süre azalacakken; bu problemlere duyulan ihtiyaç artacaktır. Bu tarz tekniklere literatürde sıklıkla karşılaşılır. Ancak geliştirilen çözüm yönteminde bu tarz teknikler kullanılmamıştır.

Modele eklenen her çevrim veya kısıttan sonra, problemin doğrusal gevşetilmiş hali tekrar çözülmektedir. Modele eklenen her bir çevrim $C(L)$ kümesine eklenmektedir. Satır ve sütun türetme yöntemleri arasındaki geçiş şu şekilde yapılmaktadır: Eğer satır türetme yönteminden bir çevrim üretilirse modelin gevşetilmiş hali tekrar çözülür ve tekrar satır türetmeye çalışılır. Eğer satır türetilemezse, sütun türetmeye çalışılır. Aynı işlemler sütun türetme içinde geçerlidir. Eğer sütun türetme için kullanılan yöntemlerden sütun türetilebilir ise; bu sütun modele eklenerek, modelin doğrusal gevşetilmiş hali tekrar çözülür ve satır türetme işlemi tekrar denenir.

Satır ve sütun türetme aşamasında üretilen çevrimler modele, hem ilgili karar değişkeni (x_c) hem de ilgili kısıt (3.13) tanımlanarak eklenir. Eğer geliştirilen çözüm yöntemlerinden ard arda hem satır hem de sütun üretilemezse algoritma durdurulur. Bu aşamadan sonra problemin tam sayılı olarak çözdürülmesi yada bir sezgisel yöntem geliştirilmesi gerekmektedir. Problemin optimal olarak çözülmesi için, dal ve fiyat yöntemi geliştirilmesi gerekmektedir.

Geliştirilen yöntemde satır türetme işlemine öncelik verilmektedir. Alternatif olarak, sütun türetme yöntemine de öncelik verilebilir. Bu durumda çözüm algoritmasında, başlangıç çevrimleri üretildikten sonra, sütun türetme adımları izlenmelidir. Sütun türetme başarılı olamadığı durumda satır türetme aşamaları izlenmelidir. Geliştirilen çözüm yönteminin adımları ve sözde kodu (pseudo code) aşağıda verildiği gibidir.

Adım 0: Bir boyutlu ve iki boyutlu çevrimleri üret ve $C(L)$ kümesine ekle.

Adım 1: $C(L)$ kümesindeki çevrimleri kullanarak; *MD-SUKRK* probleminin doğrusal gevşetilmiş problemi çöz.

Adım 2: *Kararlılık* koşulunu ihlal eden çevrimleri bulmak için satır türetme için geliştirilen dört yöntemi çalıştır.

Adım 3: Eğer böyle bir çevrim üretilemezse *STA* problemini optimal çöz.

Adım 4: Eğer *STA* probleminden böyle bir çevrim üretilemezse sütun türetme için geliştirilen dört yöntemi çalıştır.

Adım 5: Eğer bu yöntemlerden negatif indirgenmiş maliyetli çevrim üretilemezse fiyatlandırma problemini optimal olarak çöz.

Adım 6: Eğer fiyatlandırma probleminden böyle bir çevrim üretilemezse **Adım 1** 'e geri dön.

Adım 7: Eğer hem satır hem de sütun türetilemezse çözüm yöntemini durdur.

Algoritma 4.1 Satır ve Sütun Türetme Çözüm Yöntemi

$C(L) \leftarrow$ Bir boyutlu ve iki boyutlu çevrimler
repeat
  $c_{new} \leftarrow \emptyset$
  MD-SUKRK probleminin gevşetilmiş halini $C(L)$ üzerinden çöz
  $C_b(L) \leftarrow$ Temelde yer alan çevrimler
  $C_{nb}(L) \leftarrow$ Temelde yer alamayan çevrimler
  $c_{new} \leftarrow SatırTüretmeYöntemi(C_b(L), C_{nb}(L), L)$
  if $c_{new} = \emptyset$ **then**
    $c_{new} \leftarrow SütunTüretmeYöntemi(C_b(L), C_{nb}(L), L)$
  if $c_{new} \neq \emptyset$ **then**
    $C(L) \leftarrow C(L) \cup \{c_{new}\}$
    MD-SUKRK$\leftarrow \sum_{l \in L_{c_{new}}} w_l \leq f_{c_{new}}$
    MD-SUKRK$\leftarrow x_{c_{new}}$
until $c_{new} = \emptyset$

4.4 DAL-FİYAT ve KESİ YÖNTEMİ

MD-SUKRK problemi için olurlu bir çözüm elde etmek veya optimal çözümü bulmak için tek başına satır ve sütun türetme çözüm yöntemini kullanmak yeterli değildir. Bunun için başka yöntemlere ihtiyaç vardır. Bu problemi optimal olarak çözmek için kullanılabilecek yöntemlerden bir tanesi *dal ve fiyat (branch and price)* yöntemidir. Bu yöntem, *dal ve sınır (branch and bound)* ve sütun türetme yöntemini bir araya getiren bir çözüm yöntemidir. Bu yöntem, yeterli zaman verildiğinde bütün olası kombinasyonları akıllıca değerlendirerek optimal çözüme ulaşmaktadır. Bu algoritmada kullanılan mantık, dinamik programlamada kullanılan mantığa benzemektedir. Bu yöntemde kötüye gideceği bilinen çözümleri önceden elemek için, sınırlar kullanılır. Bu sınırlar, olurlu çözümlerden elde edilir ve problem için bir üst sınır tanımlar.

Problemi çözmek için aynı zamanda satır türetme yöntemi de kullanıldığı için geliştirilen çözüm yöntemi *dal-fiyat ve kesi (DFK)* olarak adlandırılmıştır. *DFK*

yöntemi kullanılmaya; satır ve sütun türetme bazlı çözüm yöntemi durduğunda başlanır. Satır ve sütun türetme bazlı çözüm yöntemi durduğunda, son doğrusal gevşetilmiş çözüm *kök düğüm* (*root node*) olarak kullanılır. *Mevcut en iyi çözüm* (*incumbent solution*) olarak da bir sonraki başlık altında açıklanan sezgisel yöntemden elde edilen en iyi çözüm kullanılır. Bu sezgiselin amacı, *DFK* için iyi sınırlar elde ederek; elde edilecek çözümlerin kalitesini arttırmaktır.

Kök düğüm çözüldükten sonra, hangi karar değişkenlerinin kesirli değerler aldığı belirlenir. Bu kesirli değer alan değişkenler arasından, hangi değişkenin dallandırılacağı belirlenir. Bu dallandırma işleminin nasıl yapıldığı ileride detaylı olarak açıklanacaktır.

Dallandırılacak değişken belirlendikten sonra; bu değişkenin bulunduğu düğümün; *çocuk düğümleri* (*child node*), bu çocuk düğümlerin *ailesi* (*parent*) ve onların *kimlikleri* (*ID*) ve bu düğümlede yer alan değişkenlerin alacağı değerler belirlenir. Tam sayılı değişkenler, iki farklı değer aldıkları için bir çocuk düğümün değeri sıfıra, diğer çocuk düğümün değeri ise bire eşitlenir. Sol çocuk düğümde bulunan dallandırılmış değişken bire, sağ çocuk düğümde bulunan dallandırılmış değişken sıfıra sabitlenir. Daha sonra bu çocuk düğümler; aktif düğüm ve düğümler listesine eklenir.

Bir sonraki aşamada, ana problemde yer alan değişkenlerin üst ve alt sınırlarının yeniden ayarlanması ve sabitlenmesi gerekir. İlk önce, ana problemde yer alan değişkenlerin üst ve alt sınırları sıfır veya bire eşitlenir. Daha sonra, *aktif düğümden* başlayarak *kök düğüme* kadar; bu düğümlerde yer alan değişkenler ana modelde gerekli değerlere sabitlenir.

Daha sonra, satır ve sütun türetme yöntemi tekrar kullanılır ve bu çözüm yöntemi için tanımlanan adımlar tekrar izlenir. Satır ve sütun türetme yöntemi durduğunda

kesme koşulları (*termination rules*) kontrol edilir. Daha sonra, kesirli değerlere sahip değişkenler ve değerleri belirlenir. Kesirli değerlere sahip değişkenler belirlendikten sonra dallandırılacak değişken belirlenir. Bu aşamadan sonra, yukarıdaki aşamalar durma koşulları sağlanana kadar tekrar edilir. Durma koşullarının neler olduğu ileride detaylı olarak açıklanacaktır.

4.4.1 Dallandırılacak Değişkenin Seçimi

Dallandırılacak değişkenin seçimi *DFK* için önemli bir problemdir. Çünkü belirlenecek strateji direk olarak algoritmanın performansını etkiler. Modelde iki farklı tam sayılı değişken vardır. Bunlar u_l ve x_c değişkenleridir. Bu değişkenler arasından, dallandırılacak değişkeni seçmek için bir hiyerarşi belirlenmiştir.

Öncelikli olarak u_l değişkenleri dallandırılmaya çalışılır. Bunun temel sebebi, bu değişkenlerin modelde sabit sayıda olmasıdır. Bu değişkenler arasından ise bire en yakın olan u_l değişkeni, dallandırılacak değişken olarak seçilir. u_l değişkenin bire sabitlenmesi, bu değişkenin temsil ettiği rota ayrıtının kapsanmayacağı anlamına gelir. Böylelikle bu rota ayrıtını içeren çevrimler de çözümde yer alamazlar. Başka bir ifadeyle, bu rota ayrıtını içeren çevrimlere karşılık gelen x_c değişkenleri kapsama kısıtından dolayı sıfır değerini alır. Böylelikle modelde ciddi bir değişken elemesi yapılmış olur.

Eğer kesirli değer almış herhangi bir u_l değişkeni yoksa, bu durumda x_c değişkenleri üzerinden dallandırma yapılır. Tıpkı u_l değişkenlerinde olduğu gibi, bu değişkenler arasından ise bire en yakın olan x_c değişkeni, dallandırılacak değişken olarak seçilir. x_c değişkeninin bire sabitlenmesi, kapsama kısıtından dolayı bu çevrim içerisinde yer alan rota ayrıtlarına karşılık gelen u_l değişkenlerinin sıfıra sabitlenmesine neden olur. Ayrıca, yine kapsama kısıtı sayesinde bu rota ayrıtla-

rını içeren diğer x_c değişkenleri sıfır değerini alır. Çünkü kapsama kısıtı her bir rota ayrıtının yalnızca bir çevrim tarafından kapsanmasını garanti eder.

Bu stratejiler geliştirilirken amaç, modelin boyutunu mümkün olduğunca küçültmektir. Yukarıda belirtildiği gibi, bir değişkenin seçimi ve belirli bir değere sabitlenmesi modelde yer alan birçok değişkenin değerini de belirler. Bu nedenle, stratejiler buna uygun olarak geliştirilmiştir ve amaç modeli küçülterek çözüm süresini kısaltmaktır.

4.4.2 Aktif Düğümün Seçimi

Aktif düğümün seçimi aşaması, dallandırılacak değişkenin hangi değer üzerinden dallandırılacağını belirler. Bunun için literatürde kullanılan birçok strateji vardır. Bu stratejiler, arama stratejileri veya arama kuralları (*searching rules*) olarak bilinir. Geliştirilen algoritmada, *derinlik öncelikli arama* (*depth-first search*) stratejisi kullanılmıştır. Bu stratejiyle, *aktif düğümler* listesine eklenen son düğüm, *aktif düğüm* olarak seçilir.

4.4.3 Kesme Koşulları

DFK yönteminde üç farklı kesme koşulu kullanılır. İlk kesme koşulu olarak problemin olurluluğuna bakılır. Eğer problem olursuzsa, olursuzluğu sağlayan düğüm; *aktif düğümler listesinden* çıkartılır. Bu işleme *olursuzlukla budama (terminated by infeasibility)* denir.

İkinci kesme işlemi, gevşetilmiş modelin amaç fonksiyonu değeri ile mevcut en iyi çözümün karşılaştırmasıyla yapılır. Eğer modelin amaç fonksiyonu, mevcut en iyi çözümünden büyük ise ilgili düğüm *aktif düğümler listesinden* çıkartılır. Bu

kesme koşulu *sınırla budama (terminated by bound)* olarak adlandırılır.

Üçüncü kesme işleminde ise, problemin gevşetilmiş çözümünde; kesirli değer olup olmadığına bakılır. Eğer problemin gevşetilmiş çözümünde kesirli değerlere sahip karar değişkenleri yoksa tam sayılı çözüm elde edilmiş olur ve ilgili düğüm *aktif düğümler listesinden* çıkarılır (*terminated by solving*). Bu aşamadan sonra *mevcut en iyi çözümün* güncellenip güncellenmeyeceğine bakılır. Eğer elde edilen çözüm, *mevcut en iyi çözümden* daha düşük ise; *mevcut en iyi çözüm* güncellenir. Eğer değilse, olduğu gibi bırakılır.

4.4.4 Durdurma Koşulları

DFK yöntemini durdurmak için kullanılan iki kural vardır. Bunlardan ilki, aktif düğümler listesinin boş olmasıdır. Eğer aktif düğümler listesi boş ise; optimal çözüm elde edilmiş olur. *DFK* yöntemini durdurmak için kullanılan diğer bir kural ise zaman kısıtıdır. Eğer *DFK* yöntemi için harcanan süre belirli bir değeri aşarsa yöntem durdurulur. Şimdiye kadar elde edilmiş en iyi tam sayılı çözümler ve amaç fonksiyonu değeri, son çözüm olarak alınır.

4.4.5 Çözüm Yönteminin Adımları

DFK çözüm yöntemi kullanılmadan önce, ilk olarak *satır ve sütun türetme* çözüm yöntemi kullanılır. Bunun için bütün olası çevrimleri üretmek yerine, sınırlı sayıda (bir ve iki boyutlu) çevrimler üretilir. Sınırlı sayıda üretilen çevrimler üzerinden *MD-SUKRK* probleminin doğrusal gevşetmesi *satır ve sütun türetme* çözüm yöntemi kullanılarak; optimal olarak çözülür. Bu algoritma durduğu anda, geliştirilen *Oran Bazlı Sezgisel* kullanılarak *DFK* için bir üst sınır elde edilir. Üst sınır elde edildikten sonra, *MD-SUKRK* probleminin doğrusal gevşetmesinin optimal çözü-

münde yer alan kesirli değerler belirlenir.

Kesirli değer alan değişkenler yukarıda tanımlanan kurala göre seçilerek; bu değişkenin çocukları oluşturulur. Oluşturulan çocuklar *aktif düğümler* listesine eklenir. Bu liste içerisinde yer alan düğüm, yukarıda tanımlanan kurala göre seçilir. Bu düğüme karşılık gelen değişken, dallandırıldığı değere sabitlenir. Değişken sabitlendikten sonra, *satır ve sütun türetme* yöntemi yeniden kullanılarak; *MD-SUKRK* probleminin doğrusal gevşetmesi, sabitlenen değer üzerinden optimal olarak çözülmeye çalışılır. *Satır ve sütun türetme* yöntemi durduktan sonra, kesme (budama) koşulları kontrol edilir ve kesirli değer alan değişkenlerin belirlenmesi aşamasına geri dönülür.. Yukarıdaki adımlar belirli sayıda tekrarlandıktan sonra, *Oran Bazlı Sezgisel* tekrar kullanılarak mevcut en iyi çözüm iyileştirilmeye çalışılır. Bu adımlar, durma koşulları sağlanana kadar devam ettirilir. *DFK* yönteminin adımları ve sözde kodu (pseudo code) aşağıda verildiği gibidir.

Adım 0: Satır ve sütun türetme çözüm yöntemini kullanarak *MD-SUKRK* probleminin gevşetilmiş halini çöz.

Adım 1: *Oran Bazlı Sezgisel* yöntemini kullanarak *DFK* için bir üst sınır elde et.

Adım 2: Kesme koşullarını kontrol et.

Adım 3: Kesirli değer alan değişkenleri belirle.

Adım 4: Bu kesirli değerler arasından dallandırılacak değişkeni belirle.

Adım 5: Dallandırılacak değişkenin çocuklarını oluştur.

Adım 6: Oluşturulan çocukları, *"aktif düğümler"* listesine ekle.

Adım 7: *"Aktif düğümler"* listesini kullanarak, aktif düğümü belirle.

Adım 8: *MD-SUKRK* probleminde yer alan karar değişkenlerinin alt ve üst sınır-

larını yeniden ayarla ve sabitle.

Adım 9: İterasyon sayısını kontrol et. Eğer iterasyon sayısı belirli bir değere ulaştıysa; iterasyon sayısını sıfırla ve *Oran Bazlı Sezgisel* yöntemini mevcut çevrimler kümesi üzerinden tekrar çöz.

Adım 10: Satır ve sütun türetme çözüm yöntemini yeniden kullanarak *MD-SUKRK* probleminin gevşetilmiş halini çöz.

Adım 11: Durma koşullarını kontrol et. Eğer durma koşulları sağlanıyorsa, **Adım 12** 'e git. Sağlanmıyorsa, **Adım 2** 'e geri dön.

Adım 12: *DFK* yöntemini durdur.

Algoritma 4.2 *DFK* Algoritması

1: *Algoritma 4.1*
2: *Algoritma 4.3*
3: **repeat**
4: *MD-SUKRK* probleminin gevşetilmiş halini *Algoritma 4.1* ile çöz
5: **if** *MD-SUKRK* olursuz ise **then**
6: *olursuzlukla buda*
7: $T \leftarrow T - \{ACN\}$
8: **if** $minMaliyet < yeniMaliyet$ **then**
9: *sınırla buda*
10: $T \leftarrow T - \{ACN\}$
11: $F \leftarrow \emptyset$
12: **if** $\exists u_l : 0 < u_l < 1$ **then**
13: $F \leftarrow F \bigcup \{u_l\}$
14: **if** $\exists x_c : 0 < x_c < 1$ **then**
15: $F \leftarrow F \bigcup \{x_c\}$
16: **if** $|F| = 0$ **then**
17: *çözüm ile buda*
18: **if** $minMaliyet > yeniMaliyet$ **then**
19: $minMaliyet \leftarrow yeniMaliyet$
20: $T \leftarrow T - \{ACN\}$
21: 30.Satıra git
22: **if** $\exists\ u_l \in F$ **then**
23: $\chi = u_l^i : \min_i \{[1 - u_l^*]\}$
24: **else if** $\exists\ x_c \in F$ **then**
25: $\chi = x_c^i : \min_i \{[1 - x_c^*]\}$
26: $CHLD_1 \leftarrow \chi = 0$
27: $CHLD_2 \leftarrow \chi = 1$
28: $T \leftarrow T \cup \{CHLD_1\}$
29: $T \leftarrow T \cup \{CHLD_2\}$
30: $ACN \leftarrow T_{|T|}$
31: $harcananZaman \uparrow$
32: $iteration \uparrow$
33: **if** $iteration = iterLimit$ **then**
34: *Algoritma 4.3* 'ye git
35: $iteration \leftarrow 0$
36: **until** $T \neq \emptyset$ *veya* $harcananZaman > zamanLimit$

4.5 Oran Bazlı Sezgisel Yöntem

Bu başlık altında, *DFK* algoritmasının performansını arttırmak için geliştirilen sezgisel yöntemden bahsedilecektir. Bu sezgisel yöntemle amaç, *DFK* için iyi sınırlar bularak algoritmanın etkinliğini arttırmaktır. İyi sınırlar, dallarda oluşturulacak çözümleri hızlı bir şekilde budayarak optimal çözüme ulaşmayı kolaylaştırabilir. Bu nedenle, bu problem için olurlu üst sınırlar tanımlayan sezgisel bir yöntem geliştirilmiştir. Geliştirilen bu sezgisel yöntem *Oran Bazlı Sezgisel (OBS)* olarak adlandırılmıştır. Bir çevrimin, toplam dolu gitme maliyeti ile toplam maliyet arasındaki oranın yüksek olması bu çevrimin iyi çözümlerde yer alma şansını yükseltmektedir. Çünkü bu tür çevrimlerde bu oran ne kadar yüksek ise boş gitme maliyeti o kadar düşüktür. Dolayısıyla, toplam maliyet düşünüldüğünde bu tür çevrimlerin maliyeti kendi aralarında paylaşılabilmektedir.

Bu nedenle, bu oran dikkate alınarak bir sezgisel yöntem geliştirilmiştir. Bu sezgisel yöntem, satır ve sütun türetme yöntemi durduğunda veya *DFK* algoritması belirli bir iterasyon sayısına ulaştığında kullanılır. *OBS* yöntemi, *DFK* algoritması için birden fazla defa kullanılabilir. *DFK* algoritmasında, optimal çözüme ulaşmak için çevrimler üretildiği için; çevrim kümesinin ($C(L)$) boyutu artmaktadır. *OBS* yöntemi ise bu küme üzerinden arama yaptığı için, arama uzayı genişleyecektir. Bu nedenle, daha iyi çözümlere ulaşma şansı daha da artacaktır. Satır ve sütun türetme bazlı çözüm yönteminden ve *DFK* yönteminden elde edilen çevrimler bir kümede ($C(L)$) toplanılarak; bu kümede yer alan her bir çevrim için, *Eşitlik 4.26* ile tanımlanan oran hesaplanır. Bu oran, her bir çevrimin seçilme olasılığını ifade etmektedir.

$$\hat{r}_c = \frac{\sum_{l_c \in L_c} \eta_{l_c} f_{l_c}}{f_c} \qquad \forall c \in C(L) \tag{4.26}$$

Bu oran her bir çevrim için hesaplandıktan sonra, çevrimler yukarıda verilen orana

göre büyükten küçüğe doğru sıralanır. Çevrimler, olasılıklarına göre çözümde yer alıp almadığına karar verilir. Eğer bir çevrim çözüm için aday olarak seçilirse, kapsama kısıtının ihlal edilip edilmediğine bakılır. Eğer bu kısıt ihlal edilmez ise, çevrim olurlu çözüme aday çevrimler kümesine ($\hat{C}(L)$) eklenir. Daha sonra; mevcut çevrimler kümesi ($C(L)$), olurlu çözüme aday çevrimler kümesi ($\hat{C}(L)$) ve kapsanan rota ayrıtları kümesi ($\hat{L}$) üzerinden olurlu bir maliyet dağıtımı bulunmaya çalışılır. Bunun için *Model 4.27* 'yi çözmek yeterlidir. Eğer bu model mevcut durum için olursuz ise; aday çevrimler kümesine ($\hat{C}(L)$) son eklenen çevrim, bu kümeden çıkarılır ve seçim aşamasına dönülür. Eğer olurlu ise hesaplanan maliyet dağıtımları üzerinden *STA* problemi çözülür. Bu problemi çözmekteki amaç, mevcut maliyet dağıtımının *kararlı* bir maliyet dağıtımı olup olmadığını kontrol etmektir. Eğer *STA* problemden bir çevrim elde edilirse; bu çevrim, *Model 4.27* 'e eklenerek tekrar çözülür. Bu işlemler, *Model 4.27* olursuz olana kadar veya *STA* probleminden bir çevrim elde edilmeyene kadar devam edilir.

$$\min 0 \tag{4.27}$$

$$\text{s.t.} \sum_{l \in \hat{L}} w_l = \sum_{c \in \hat{C}(L)} f_c \tag{4.28}$$

$$\sum_{l \in L_c} w_l \leq f_c \qquad \forall c \in C(L) \tag{4.29}$$

$$w_l \leq (1 - \theta_l) g_l \qquad \forall l \in \hat{L} \tag{4.30}$$

$$w_l \geq \lambda_l f_l \qquad \forall l \in \hat{L} \tag{4.31}$$

$$w_l \geq 0 \qquad \forall l \in \hat{L} \tag{4.32}$$

$$w_l = 0 \qquad \forall l \in L \setminus \hat{L} \tag{4.33}$$

Eğer maliyet dağıtımı *kararlı* ise yani *Model 4.27* olurlu ve *STA* probleminden bir çevrim elde edilmez ise seçim işlemine geri dönülür. Eğer değil ise, olurlu çözüme aday çevrimler kümesine ($\hat{C}(L)$) son eklenen çevrim bu kümeden çıkarılır

ve seçim işlemine geri dönülür.

Yukarıdaki işlemler bütün çevrimler dikkate alınana kadar devam eder. Bütün çevrimler dikkate alındıktan sonra elde edilen, olurlu çözüme aday çevrimler kümesi ($\hat{C}(L)$), *MD-SUKRK* problemi için olurlu bir çözüm tanımlar. Çünkü bu işlemler sonucunda, *MD-SUKRK* probleminde yer alan hiçbir kısıt ihlal edilmez. Bütün çevrimler dikkate alındıktan sonra amaç fonksiyonu değeri hesaplanır ve bu aşamada en iyi amaç fonksiyonu değeri (*best objective function value*) ve bu değeri veren çevrimler hafızada tutulur. *MD-SUKRK* problemi bir en küçükleme problemi olduğu için en iyi çözüm en düşük amaç fonksiyonu değeri demektir.

Amaç fonksiyonu değeri hesaplandıktan sonra ve güncellendikten sonra olurlu çözüme aday çevrimler kümesi ($\hat{C}(L)$) ve kapsanan rota ayrıtları kümesi ($\hat{L}$) kümesi boşaltılarak seçim aşamasına geri dönülür ve yukarıda verilen aşamalar tekrarlanır. Ancak olurlu çözüme aday çevrimler kümesi ($\hat{C}(L)$) boşaltılmadan önce, bu küme içerisinde yer alan çevrimlerin hesaplanan oranları ($\hat{r}_c$) 0.2 azaltılır. Bu azaltma işleminin amacı, çözüm uzayını genişleterek farklı çözümlere ulaşmaktır. Azaltma işlemi bir çevrim için birden fazla defa yapılabilir. Dolayısıyla bu azaltma işleminden sonra eğer bir çevrimin oranı ($\hat{r}_c$) 0.5 değerinin altına inerse, bu oran tekrar hesaplanır. Bu sezgisel yöntem belli bir zaman limitini aştıktan sonra durdurulur ve hafızdaki en iyi çözüm, sezgiselden elde edilen çözüm olarak alınır. Bu sezgiselin adımları ve sözde kodu (pseudo code) aşağıdaki gibidir.

Adım 1: $C(L)$ kümesi içerisinde yer alan her bir çevrim için $\hat{r}_c$ değerini *Eşitlik 4.26* 'e göre hesapla.

Adım 2: Çevrimleri $\hat{r}_c$ değerine göre büyükten küçüğe doğru sırala.

Adım 3: Çevrimleri $\hat{r}_c$ olasılığına göre seç ve seçilen çevrimleri $\hat{C}(L)$ kümesine ekle.

Adım 4: $\hat{C}(L)$ kümesinde yer alan çevrimler için olurluluk koşullarını kontrol et.

Adım 5: Eğer bir ihlal varsa, son eklen çevrimi $\hat{C}(L)$ kümesinden çıkar.

Adım 6: Eğer bir ihlal yoksa, $C(L)$, $\hat{C}(L)$ ve $\hat{L}$ kümelerini kullanarak *Model 4.27* çöz.

Adım 7: Eğer *Model 4.27* olursuz ise son eklen çevrimi $\hat{C}(L)$ kümesinden çıkar ve *Adım 3* 'e geri dön.

Adım 8: Eğer *Model 4.27* olurlu ise *STA* problemini çöz.

Adım 9: Eğer *STA* probleminden bir çevrim üretilebiliyorsa, bu çevrimi *Model 4.27* 'e ekle ve tekrar çöz.

Adım 10: Eğer *STA* probleminden bir çevrim üretilemezse, *Adım 3* 'e geri dön.

Adım 11: Yukarıdaki adımları bütün çevrimler dikkate alınana kadar devam ettir.

Adım 12: Eğer bütün çevrimler değerlendirilmişse, $\hat{C}(L)$ kümesi içerisinde yer alan çevrimlerin $\hat{r}_c$ değerlerini 0.2 azalt ve $\hat{C}(L)$ ve $\hat{L}$ kümelerini boşalt.

Adım 13: Eğer bir çevrim için $\hat{r}_c$ değeri 0.5 değerinin altına inerse, $\hat{r}_c$ değerini tekrar hesapla.

Adım 14: Algoritma belirli bir zaman limitine ulaştığında durdur.

Algoritma 4.3 Oran Bazlı Sezgisel Yöntem

1: **for** i to $|C(L)|$ **do**
2: *Eşitlik 4.26* 'i kullanarak $\hat{r}_c$ hesapla
3: $\hat{r}_c$ $\forall c \in C(L)$ değerlerini büyükten küçüğe doğru sırala
4: **repeat**
5: $\hat{C}(L) \leftarrow \emptyset$
6: $\hat{L} \leftarrow \emptyset$
7: **for** i to $|C(L)|$ **do**
8: **if** $\{l_{c_i}\} \notin \hat{L}$ **then**
9: $rastgele \in [0,1]$ üret
10: **if** $rastgele \leq \hat{r}_c$ **then**
11: $\hat{C}(L) \leftarrow \hat{C}(L) \bigcup \{c_i\}$
12: $\hat{L} \leftarrow \hat{L} \bigcup \{l_{c_i}\}$
13: **if** $\hat{C}(L)$ kümesi olurlu ise **then**
14: *Model 4.27* 'yi $\hat{C}(L), C(L), \hat{L}, L$ üzerinden çöz
15: **if** *Model 4.27* olurlu ise **then**
16: *STA* 'yı çöz
17: **if** *STA* 'dan bir çevrim (c_{new}) üretilebiliyorsa **then**
18: *Model 4.27* $\leftarrow \sum_{l \in L_{c_{new}}} w_l \leq f_{c_{new}}$
19: 14.Satıra git
20: **else**
21: $toplamMaliyet = 0$
22: **for** $\{l\} \in L \setminus \hat{L}$ **do**
23: $toplamMaliyet \leftarrow toplamMaliyet + g_l$
24: **for** $\{c\} \in \hat{C}(L)$ **do**
25: $toplamMaliyet \leftarrow toplamMaliyet + f_c$
26: $\hat{r}_c \leftarrow \hat{r}_c - 0.2$
27: **if** $\hat{r}_c \leq 0.5$ **then**
28: $\hat{r}_c$ 'yi yeniden hesapla
29: **if** $minMaliyet < toplamMaliyet$ **then**
30: $minMaliyet \leftarrow toplamMaliyet$
31: **else**
32: $\hat{C}(L) \leftarrow \hat{C}(L) - \{c_i\}$
33: $\hat{L} \leftarrow \hat{L} - \{l_i\}$
34: **else**
35: $\hat{C}(L) \leftarrow \hat{C}(L) - \{c_i\}$
36: $\hat{L} \leftarrow \hat{L} - \{l_i\}$
37: $harcananZaman \uparrow$
38: **until** $harcananZaman > zamanLimit$

5. MALİYET DAĞITIM YÖNTEMLERİ

Bu başlık altında, literatürde kullanılan maliyet dağıtımlarından ve geliştirilen maliyet dağıtım yöntemlerinden bahsedilecektir. Bu maliyet dağıtımları, *SUKRK* problemi çözüldükten sonra kullanılacaktır. Bu yaklaşım, literatürde kullanılan iki aşamalı klasik yaklaşımdır. Bu yaklaşım, önerilen yöntem ile karşılaştırılacaktır. Karşılaştırmalar, *kararlılık* koşulu üzerinden yapılacaktır. Başka bir ifadeyle, bu maliyet dağıtımlarından elde edilen çözümler ile *MD-SUKRK* problemi için geliştirilen yöntemlerden elde edilen çözümler birbirleriyle karşılaştırılacaktır. Ayrıca geliştirilen maliyet dağıtım metotları ile literatürde yer alan maliyet dağıtım metotları da bir birleriyle karşılaştırılacaktır.

Bunun için öncelikle *SUKRK* problemi çözülecek, daha sonra elde edilen çözüm kullanılarak aşağıda tanımlanan maliyet dağıtımları kullanılarak bir maliyet dağıtımı elde edilmeye çalışılacaktır. Maliyet dağıtım yöntemlerinde, *SUKRK* probleminden elde edilen en düşük maliyetli çözüm ($F(L)$) kullanılacaktır.

$$F(L) = \sum_{c \in \hat{C}(L)} f_c \tag{5.1}$$

Geliştirilen maliyet dağıtım yöntemleri, farklı problemlere de kolayca uyarlanabilir. Ancak ilk önce toplam maliyeti en küçükleyen bir matematiksel modele ihtiyaç vardır. Bu modelden elde edilen en küçük toplam maliyet bulunduktan sonra geliştirilen modeller kolaylıkla uygulanabilir.

5.1 Alternatif Maliyet Dağıtım Yöntemleri

Bu başlık altında literatürde kullanılan üç farklı maliyet dağıtım yönteminden bahsedilecektir. Bu maliyet dağıtım yöntemleri geliştirilen yöntemlerle karşılaştırıla-

caktır.

5.1.1 Uzaklık Orantılı Maliyet Dağıtımı

Maliyet dağıtım aşamasında akla gelen ilk yöntem, kaynak tüketimine dayalı; maliyetlerin dağıtılmasıdır. İşbirlikçi turlarda, kaynak tüketimi kat edilen mesafeye bağlıdır. Kat edilen mesafeler, tam kamyon yükü hareketlerinden ve boş kamyon yükü hareketlerinden oluşur. Bu nedenle, seçilen her bir çevrimin maliyeti; rota ayrıtlarına mesafeleri oranında dağıtılabilir. Formal bir ifadeyle, bir rota ayrıtına düşen maliyet *Eşitlik 5.2* 'de verildiği gibi hesaplanır.

$$w_l^{c,prop} = \left(\frac{f_l}{\sum_{l_c \in L_c} f_{l_c}} \right) f_c \tag{5.2}$$

Eşitlik 5.2 'de tanımlanan maliyet dağıtımı, *bütçe dengesi* koşulunu sağlamaktadır.

5.1.2 Shapley Değeri

Maliyet dağıtımında kullanılabilecek diğer bir yöntem de *Shapley Değeridir (SD)*. *SD*, her oyuncunun alt koalisyonlara ayrı ayrı yaptığı marjinal katkısının ağırlıklı ortalamasıdır. Başka bir deyişle *SD*, iş birliğinin teker teker kurulması halinde her oyuncunun iş birliğine yaptığı ortalama katkıdır. Matematiksel bir ifadeyle, bir rota ayrıtının *SD* aşağıdaki gibi hesaplanır.

$$w_l^{shap} = \sum_{S \subseteq L : l \in S} \left(\frac{(\mid S \mid -1)!(\mid L \mid - \mid S \mid)!}{\mid L \mid !} (F(S) - F(S - \{l\})) \right) \tag{5.3}$$

Eşitlik 5.3 kullanarak *SD* 'ni hesaplamak kolay değildir. Çünkü gerekli bütün alt kümeleri üretmek oldukça zaman alıcıdır. Bu nedenle, *SD* sınırlı sayıda alt küme kullanılarak yaklaşık olarak hesaplanacaktır. Yalnızca $l \in L$ rota ayrıtını kapsa-

yan çevrimleri oluşturan rota ayrıtlarına ait alt kümeler dikkate alınacaktır. Bu durumda, *SD* aşağıdaki gibi hesaplanır.

$$w_l^{c,shap} = \sum_{S \subseteq L_c : l \in S} \left(\frac{(\mid S \mid -1)!(\mid L_c \mid - \mid S \mid)!}{\mid L_c \mid !} (F(S) - F(S - \{l\})) \right) \tag{5.4}$$

Yukarıdaki eşitlikten elde edilen maliyet dağıtımı, *bütçe dengeli* bir maliyet dağıtımı olmayabilir. Bu durumda, maliyet dağıtımı $w_l^{c,shap}$ toplamları, çevrimlerin maliyetlerine eşit olacak şekilde ölçeklenecektir. Bu ölçekleme oransal olarak yapılacaktır.

5.1.3 Nükleolus (Çekirdekçik) Maliyet Dağıtımı

Nükleolus veya Çekirdekçik, bütün koalisyonlar üzerinden en düşük faydayı sözlüksel (*lexicographically*) biçimde en büyüklemeye çalışır. *Nükleolus* çözümü tektir (unique). Bu nedenle alternatif optimal çözümlerden sakınmak gerekir.

Nükleolus 'u hesaplayabilmek için doğrusal modelin iteratif olarak çözülmesi gerekmektedir. Herhangi bir adımda, bir önceki adımdaki optimal çözümde *kararlılık* kısıtını eşitlik olarak sağlayan elemanlar ve amaç fonksiyonu değeri kullanılır.

S, koalisyon kümesi olsun $S \subseteq L$. $F(S)$, S koalisyonunun maliyeti olsun. Rota ayrıtına düşen maliyet $w_l^{nuc}, l \in L$ olsun. ε ise, *kararlılık* kısıtının ihlal miktarını gösteren karar değişkeni olmak üzere *nükleolus* aşağıdaki gibi tanımlanır.

$$\min \varepsilon \tag{5.5}$$

$$\text{s.t.} \sum_{l \in L} w_l^{nuc} = F(L) \tag{5.6}$$

$$\sum_{l \in S} w_l^{nuc} \leq F(S) + \varepsilon \qquad \forall S \subseteq L \tag{5.7}$$

$$w_l^{nuc} \in \mathbb{R} \qquad \forall l \in L \tag{5.8}$$

$$\varepsilon \in \mathbb{R} \tag{5.9}$$

Bu modelin amacı, en büyük ihlali en küçüklemektir. Kısıt 5.6, *bütçe dengesi* kısıtıdır. Kısıt 5.7, *kararlılık* kısıtıdır. Bu kısıt bütün rota ayrıtlarının alt kümeleri üzerinden tanımlanır. Başka bir deyişle, kısıt 5.7 bütün koalisyonlar üzerinden tanımlanır. Ayrıca bu modelde yer alan atama karar değişkeni (w_l^{nuc}) ve ihlal karar değişkeni (ε) bütün reel sayılar üzerinden tanımlıdır. Yukarıda tanımlanan model literatürde *öncül nükleolus (pre-nucleolus)* olarak adlandırılır.

Model 5.5, yalnızca bir kere kullanılır. İlk aşamada çözüldükten sonra, ileri ki aşamalarda hep model 5.15 kullanılacaktır. Model 5.5 çözüldükten sonra model 5.15 'nin oluşturulması gerekir. Bunun için bu modelin optimal çözümü kullanılır. Optimal çözüm kullanılarak *kararlılık* kısıtı (5.17) yeniden düzenlenir

Doğrusal modelin, alternatif optimal çözümünün olması durumunda hesaplamalarda hatalar oluşabileceğinden *"Literatür Araştırması"* kısmında bahsedilmiştir (Guajardo ve Jornsten (2015)). Bu nedenle, doğrusal problemin optimal çözümünde, *kararlılık* kısıtını eşitlik olarak sağlanan kısıtlara bakmak yerine; bu kısıtlara karşılık gelen dual değişkenlerin optimal çözümüne bakılacaktır.

Eğer bu kısıtlara karşılık gelen dual değişkenler sıfırdan küçük ise bu kısıtlar eşitlik olarak sağlanacaktır. Çünkü *tamamlayıcı boşluk koşulları (complementary slackness conditions)* nedeniyle, dual değerlerin sıfırdan küçük olması (sıfırdan farklı olması); dual değerlerin karşılık geldiği kısıtların eşitlik olarak sağlanmasını garanti edecektir (Guajardo ve Jornsten (2015)).

υ_S *kararlılık* kısıtına (5.7) karşılık gelen dual değişken olsun. Bu durumda *tamam-*

layıcı boşluk koşulları aşağıdaki gibi yazılır.

$$\upsilon_S[F(S)+\varepsilon-\sum_{l\in S} w_l^{nuc}]=0 \qquad \forall S\subseteq L \qquad (5.10)$$

$$\upsilon_S=0 \Longrightarrow [F(S)+\varepsilon-\sum_{l\in S} w_l^{nuc}]\geq 0 \qquad \forall S\subseteq L \qquad (5.11)$$

$$\upsilon_S<0 \Longrightarrow [F(S)+\varepsilon-\sum_{l\in S} w_l^{nuc}]=0 \qquad \forall S\subseteq L \qquad (5.12)$$

$$[F(S)+\varepsilon-\sum_{l\in S} w_l^{nuc}]=0 \Longrightarrow \upsilon_S\leq 0 \qquad \forall S\subseteq L \qquad (5.13)$$

$$[F(S)+\varepsilon-\sum_{l\in S} w_l^{nuc}]>0 \Longrightarrow \upsilon_S=0 \qquad \forall S\subseteq L \qquad (5.14)$$

Alternatif optimalden uzak durmak için koşul 5.12 kullanılacaktır. Dolayısıyla model 5.5 çözdürüldükten sonra υ_S değerleri kontrol edilecek ve sıfırdan küçük olan değerler dikkate alınacaktır. υ_S dual değişkeni sıfır veya sıfırdan küçük değer alır. Çünkü problem bir en küçükleme problemidir ve bu değişkene karşılık gelen kısıtlar küçük eşit kısıtlarıdır.

Bu durumda $\hat{S}_i$ kümesinin tanımlanması gerekir. $\hat{S}_i$ kümesi, i. iterasyonda *kararlılık* kısıtını eşitlik olarak sağlayan çevrimlerin kümesidir. Matematiksel bir ifadeyle, $\hat{S}_i=\{S|\upsilon_S<0, S\subseteq L\}$ şeklinde tanımlanır. Bu durumda ortaya çıkan yeni model aşağıdaki şekilde tanımlanır.

$$\min \varepsilon_k \qquad (5.15)$$

$$\text{s.t.} \sum_{l\in L} w_l^{nuc}=F(L) \qquad (5.16)$$

$$\sum_{l\in S} w_l^{nuc}\leq F(S)+\varepsilon_k \qquad \forall S\subseteq L: S\notin \bigcup_{i=1}^{k-1}\hat{S}_i \qquad (5.17)$$

$$\sum_{l\in S} w_l^{nuc}=F(S)+\varepsilon_i \qquad \forall S\in \bigcup_{i=1}^{k-1}\hat{S}_i \qquad (5.18)$$

$$w_l^{nuc}\in\mathbb{R} \qquad \forall l\in L \qquad (5.19)$$

$$\varepsilon \in \mathbb{R} \tag{5.20}$$

Bu modelin amacı, k. iterasyon için en büyük ihlali en küçüklemektir. Kısıt 5.16, bir önceki modelde yer alan kısıt 5.6 ile aynıdır. Bu kısıtta herhangi bir değişiklik yapılmaz. Kısıt 5.17, geçmiş iterasyonlarda bu kısıtı eşitlik olarak sağlamayan kısıtları içeren *kararlılık* kısıtıdır. Bu kısıtın bir önceki modeldeki kısıt 5.7 'den farkı, kısıtlı sayıda küme üzerinden yazılıyor olmasıdır. Kısıt 5.18, geçmiş iterasyonlarda *kararlılık* kısıtı 5.17 'yi eşitlik olarak sağlayan kısıtları temsil etmektedir. Bu kısıtta yer alan ε_i ifadesi bir karar değişkeni değil, bir parametredir. ε_i i. iterasyondaki ε değerini ifade etmektedir. Geriye kalan kısıtlar, model 5.5 ile aynıdır.

Nükleolus 'un hesaplanabilmesi için ikinci modelin durma koşulu sağlanana kadar iteratif olarak çözdürülmesi gerekir. Durma koşulu, modelin alternatif optimal çözümünün olmamasıdır. Yani doğrusal modelin tek bir optimal çözümü (*unique solution*) olduğu durumda iterasyonlar durdurulur. Özellikle büyük boyutlu örneklerde, makul sürelerde ilk durma koşulunu sağlamak mümkün olmayabilir. Bu nedenle, ilk durma koşuluna alternatif olarak zaman limiti de kullanılabilir.

5.2 Geliştirilen Maliyet Dağıtım Yöntemleri

Literatürde kullanılan maliyet dağıtım yöntemlerine ek olarak, *kararlı* maliyet dağıtımları bulmak için çeşitli maliyet dağıtım yöntemleri geliştirilmiştir. Bu başlık altında geliştirilen altı farklı maliyet dağıtım yönteminden bahsedilecektir.

5.2.1 Minimum Maksimum İhlal Oranı

Uzaklık orantılı maliyet dağıtımı ve *Shapley Değeri*, *kararlılık* koşulunu dikkate almamaktadır. Bu nedenle, *kararlılık* koşulu yüksek oranlarda ihlal edilebilir. Yük-

sek oranlarda ihlallerden sakınmak için yeni bir dağıtım mekanizmasına ihtiyaç vardır. Bu metot, en yüksek ihlali en küçüklemeyi amaçlayan doğrusal programlama modeline dayanmaktadır.

Bu modelin amaç fonksiyonu değerinin sıfır olması, *çekirdekte* yer alan bir maliyet dağıtımının bulunabileceğini garanti eder. Çünkü böyle bir durumda hem *kararlı* hem de *bütçe dengeli* bir maliyet dağıtımı bulunmuş olur. Eğer amaç fonksiyonu sıfırdan farklı ise, *çekirdekte* yer alan bir maliyet dağıtımının bulunamayacağını garanti eder.

Rota ayrıtına düşen maliyet $w_l^{mm}, l \in L$ olsun. ε ise, *kararlılık* kısıtını ihlal eden en yüksek oranı gösteren karar değişkeni olmak üzere *Minimum Maksimum İhlal Oranı (MMİO)* aşağıdaki gibi tanımlanır.

$$\min \varepsilon \tag{5.21}$$

$$\text{s.t.} \sum_{l \in L} w_l^{mm} = F(L) \tag{5.22}$$

$$\sum_{l \in L_c} w_l^{mm} \leq f_c(1+\varepsilon) \qquad \forall c \in C(L) \tag{5.23}$$

$$w_l^{mm} \leq (1-\theta_l) g_l \qquad \forall l \in L \tag{5.24}$$

$$w_l^{mm} \geq \lambda_l f_l \qquad \forall l \in L \tag{5.25}$$

$$w_l^{mm} \geq 0 \qquad \forall l \in L \tag{5.26}$$

$$\varepsilon \geq 0 \tag{5.27}$$

Problemin amacı (5.21), *kararlılık* kısıtını oransal olarak en fazla ihlali en küçüklemektir. Kısıt 5.22, seçilen rota ayrıtlarına atanan toplam maliyetin, çevrimin maliyetine eşit olmasını sağlar. Kısıt 5.23, *kararlılık* koşulunu gevşetmektedir. Ne oranda gevşetileceği model tarafından karar verilecektir.

5.2.2 Minimum Toplam İhlal Oranı

MMİO modeli en büyük ihlali en küçüklemeyi amaçlamaktadır. Bu model için ihlal edilen çevrim sayısı fazla olabilir. Bu nedenle, yeni bir model geliştirilmiştir. Geliştirilen bu model, toplam ihlali en küçüklemeyi amaçlayan doğrusal programlama modelidir. Bu modelin amaç fonksiyonu değerinin sıfır olması, *çekirdekte* yer alan bir maliyet dağıtımının bulunabileceğini garanti eder. Çünkü böyle bir durumda hem *kararlı* hem de *bütçe dengeli* bir maliyet dağıtımı bulunmuş olur. Eğer amaç fonksiyonu sıfırdan farklı ise, *çekirdekte* yer alan bir maliyet dağıtımının bulunamayacağını garanti eder.

Rota ayrıtına düşen maliyet $w_l^{mt}, l \in L$ olsun. ε_c ise, her bir *kararlılık* kısıtının ihlal oranını gösteren karar değişkeni olmak üzere *Minimum Toplam İhlal Oranı (MTİO)* aşağıdaki gibi tanımlanır.

$$\min \sum_{c \in C} \varepsilon_c \tag{5.28}$$

$$\text{s.t.} \sum_{l \in L} w_l^{mt} = F(L) \tag{5.29}$$

$$\sum_{l \in L_c} w_l^{mt} \leq f_c(1+\varepsilon_c) \qquad \forall c \in C(L) \tag{5.30}$$

$$w_l^{mt} \leq (1-\theta_l) g_l \qquad \forall l \in L \tag{5.31}$$

$$w_l^{mt} \geq \lambda_l f_l \qquad \forall l \in L \tag{5.32}$$

$$w_l^{mt} \geq 0 \qquad \forall l \in L \tag{5.33}$$

$$\varepsilon_c \geq 0 \qquad \forall c \in C(L) \tag{5.34}$$

Problemin amacı (5.28), *kararlılık* kısıtını oransal olarak toplam ihlali en küçüklemektir. Kısıt 5.29, *MMİO* modelinde de olduğu gibi seçilen rota ayrıtlarına atanan toplam maliyetin, çevrimin maliyetine eşit olmasını sağlar. Kısıt 5.30, *kararlılık*

koşulunu gevşetmektedir. Ne oranda gevşetileceği model tarafından karar verilecektir. *MMİO* 'dan farklı olarak bu kısıta karşılık gelen birden fazla ε karar değişkeni vardır. *MMİO* 'da tek bir tane ε karar değişkeni varken bu modelde $|C(L)|$ kadar ε karar değişkeni vardır.

5.2.3 s-Nükleolus (s-Çekirdekçik) Maliyet Dağıtımı

Bölüm 5.1.3 'de tanımlanan *nükleolus*, bütün koalisyonlar üzerinden tanımlanır. L kümesindeki artış, üretilecek koalisyon sayısını üssel olarak arttıracaktır. Bu nedenle, rota ayrıtı sayısı belirli bir sayıyı aştığında; bütün koalisyonları makul sürelerde üretmek mümkün olmayacaktır.

Bütün koalisyonları, makul sürelerde üretmenin mümkün olmadığı durumlar için farklı bir model geliştirilmiştir. Geliştirilen bu modele, *s-Nükleolus* adı verilmiştir. Bu model, bütün koalisyonlar yerine; üretilen çevrimler üzerinden tanımlanmıştır. Üretilen her bir olurlu çevrim, olurlu bir koalisyona karşılık gelmektedir.

Rota ayrıtına düşen maliyet $w_l^{snuc}, l \in L$ olsun. ε ise, *kararlılık* kısıtının ihlal miktarını gösteren karar değişkeni olmak üzere *s-Nükleolus* aşağıdaki gibi tanımlanır.

$$\min \varepsilon \tag{5.35}$$

$$\text{s.t.} \sum_{l \in \hat{L}} w_l^{snuc} = F(L) \tag{5.36}$$

$$\sum_{l \in L_c} w_l^{snuc} \leq f_c + \varepsilon \qquad \forall c \in C(L) \tag{5.37}$$

$$w_l^{snuc} \in \mathbb{R} \qquad \forall l \in L \tag{5.38}$$

$$\varepsilon \in \mathbb{R} \tag{5.39}$$

Bu modelin amacı, en büyük ihlali en küçüklemektir. Kısıt 5.36, *bütçe dengesi*

kısıtıdır. Kısıt 5.37, *kararlılık* kısıtıdır. Bu kısıt, bütün $C(L)$ kümesi üzerinden tanımlanır. Yani, seçilen çevrim ve rota ayrıtlarını dikkate almaz. Bu model bireysel rasyonellik ve maliyetlere alt sınır atanması kısıtını da dikkate almaz. Ayrıca bu modelde yer alan atama karar değişkeni (w_l^{snuc}) ve ihlal karar değişkeni (ε) bütün reel sayılar üzerinden tanımlıdır.

Model 5.35 yalnızca bir kere kullanılır. İlk aşamada çözüldükten sonra, ileri ki aşamalarda hep model 5.45 kullanılacaktır. Model 5.35 çözüldükten sonra model 5.45 'nin oluşturulması gerekir. Bunun için bu modelin optimal çözümü kullanılır. Optimal çözüm kullanılarak *kararlılık* kısıtı 5.37 yeniden düzenlenir.

Nükleolus 'da olduğu gibi, bu yöntem için de alternatif optimal çözümlerden sakınmak gerekir. Bu nedenle, *nükleolus* için tanımlanan *tamamlayıcı boşluk koşullarının* bu model içinde tanımlanması gerekir.

υ_c *kararlılık* kısıtına (5.37) karşılık gelen dual değişken olsun. Bu durumda *tamamlayıcı boşluk koşulları* aşağıdaki gibi yazılır.

$$\upsilon_c[f_c + \varepsilon - \sum_{l \in L_c} w_l^{snuc}] = 0 \qquad \forall c \in C(L) \qquad (5.40)$$

$$\upsilon_c = 0 \Longrightarrow [f_c + \varepsilon - \sum_{l \in L_c} w_l^{snuc}] \geq 0 \qquad \forall c \in C(L) \qquad (5.41)$$

$$\upsilon_c < 0 \Longrightarrow [f_c + \varepsilon - \sum_{l \in L_c} w_l^{snuc}] = 0 \qquad \forall c \in C(L) \qquad (5.42)$$

$$[f_c + \varepsilon - \sum_{l \in L_c} w_l^{snuc}] = 0 \Longrightarrow \upsilon_c \leq 0 \qquad \forall c \in C(L) \qquad (5.43)$$

$$[f_c + \varepsilon - \sum_{l \in L_c} w_l^{snuc}] > 0 \Longrightarrow \upsilon_c = 0 \qquad \forall c \in C(L) \qquad (5.44)$$

Alternatif optimalden uzak durmak için koşul 5.42 kullanılacaktır. Dolayısıyla model 5.35 çözdürüldükten sonra υ_c değerleri kontrol edilecek ve sıfırdan küçük olan değerler dikkate alınacaktır. υ_c dual değişkeni sıfır veya sıfırdan küçük değer

alır. Çünkü problem bir en küçükleme problemidir ve bu değişkene karşılık gelen kısıtlar küçük eşit kısıtlarıdır. Bu durumda, $C_i(L)$ kümesinin tanımlanması gerekir. $C_i(L)$ kümesi, i. iterasyonda *kararlılık* kısıtını eşitlik olarak sağlayan çevrimlerin kümesidir. Matematiksel bir ifadeyle, $\hat{C}_i(L) = \{c | \upsilon_c < 0, c \in C(L)\}$ şeklinde tanımlanır. Bu durumda ortaya çıkan yeni model aşağıdaki şekilde tanımlanır.

$$\min \varepsilon_k \tag{5.45}$$

$$\text{s.t.} \sum_{l \in \hat{L}} w_l^{snuc} = F(L) \tag{5.46}$$

$$\sum_{l \in L_c} w_l^{snuc} \leq f_c + \varepsilon_k \qquad \forall c \in C(L) : c \notin \bigcup_{i=1}^{k-1} C_i(L) \tag{5.47}$$

$$\sum_{l \in L_c} w_l^{snuc} = f_c + \varepsilon_i \qquad \forall c \in \bigcup_{i=1}^{k-1} C_i(L) \tag{5.48}$$

$$w_l^{snuc} \in \mathbb{R} \qquad \forall l \in L \tag{5.49}$$

$$\varepsilon_k \in \mathbb{R} \tag{5.50}$$

Bu modelin amacı, k. iterasyon için en büyük ihlali en küçüklemektir. Kısıt 5.46, bir önceki modelde yer alan kısıt 5.36 ile aynıdır. Bu kısıtta her hangi bir değişiklik yapılmaz. Kısıt 5.47, geçmiş iterasyonlarda bu kısıtı eşitlik olarak sağlamayan kısıtları içeren kararlılık kısıtıdır. Bu kısıtın bir önceki modeldeki kısıt 5.37 'den farkı kısıtlı sayıda küme üzerinden yazılıyor olmasıdır. Kısıt 5.48, geçmiş iterasyonlarda *kararlılık* kısıtı 5.47 'yi eşitlik olarak sağlayan kısıtları temsil etmektedir. Bu kısıtta yer alan ε_i ifadesi bir karar değişkeni değil, bir parametredir. ε_i i. iterasyondaki ε değerini ifade etmektedir. Geriye kalan kısıtlar, model 5.35 ile aynıdır.

s-Nükleolus 'un hesaplanabilmesi için ikinci modelin durma koşulları sağlanana kadar iteratif olarak çözdürülmesi gerekir. Modelin alternatif optimal çözümünün olmaması durumunda, *s-Nükleolus* elde edilmiş olur. Yani doğrusal modelin

tek bir optimal çözümü (*unique solution*) olduğu durumda iterasyonlar durdurulur. Model, bazı örnekler için bu durma koşulunu sağlamayabilir. Özellikle büyük boyutlu örnekler için makul sürelerde bu durma koşulunu sağlamak mümkün olmayabilir. Bu nedenle, modelin iteratif çözüme bir zaman limiti konulmuştur. Modelin iteratif çözümüne harcanan zaman, belirli bir miktardan büyük ise çözüm durdurulur. Burada dikkat edilmesi gereken nokta, bu durumda elde edilen çözümün; gerçek optimal çözümü yansıtmamasıdır.

5.2.4 %s-Nükleolus (%s-Çekirdekçik) Maliyet Dağıtımı

s-Nükleolus, ihlal oranını miktar olarak en küçüklemeye çalışır. Böyle bir durumda eğer *kararlı* bir maliyet dağıtımı bulunamazsa ihlal oranları yüksek olabilir. Bu nedenle, modelin amaç fonksiyonu oran olarak güncellenerek yeni bir model önerilmiştir. Bunun için yalnızca kısıt 5.37 'nın, aşağıda verildiği şekilde (5.51) değiştirilmesi yeterlidir. Kısıt 5.37 yerine kısıt 5.51 eklendiğinde, model maksimum ihlali oran olarak minimize etmeye çalışacaktır.

$$\sum_{l \in L_c} w_l^{snuc} \leq f_c(1+\varepsilon) \qquad \forall c \in C(L) \tag{5.51}$$

Yukarıda tanımlanan kısıt modele eklendikten sonra, *s-Nükleolus* için tanımlanan çözüm adımları bu model için de uygulanacaktır. Gerekli güncellemeler de yukarıda tanımlandığı gibi yapılacaktır. Gerekli güncellemeler yapıldıktan sonra Kısıt 5.47 ve 5.48 yerine kısıt 5.52 ve 5.53 kullanılacaktır.

$$\sum_{l \in L_c} w_l^{snuc} \leq f_c(1+\varepsilon_k) \qquad \forall c \in C(L) : c \notin \bigcup_{i=1}^{k-1} C_i(L) \tag{5.52}$$

$$\sum_{l \in L_c} w_l^{snuc} = f_c(1+\varepsilon_i) \qquad \forall c \in \bigcup_{i=1}^{k-1} C_i(L) \tag{5.53}$$

5.2.5 q-Nükleolus (q-Çekirdekçik) Maliyet Dağıtımı

Bir önceki başlıkta tanımlanan, *Nükleolus* ve *s-Nükleolus* modeli, modelde yer alan bazı özellikleri dikkate almamaktadır. Bu nedenle, *s-Nükleolus* modeli temel alınarak yeni bir model geliştirilmiştir. Bu model, *q-Nükleolus* olarak isimlendirilmiştir. *s-Nükleolus* modelinde kullanılan aşamalar bu model için de geçerlidir. Bu modelde de, *kararlılık* kısıtı aynı şekilde kontrol edilerek güncellenir ve iteratif olarak tekrar çözülür. Ancak bu model, dikkate alınan diğer kısıtları da içermektedir. Rota ayrıtına düşen maliyet $w_l^{qnuc}, l \in L$ olsun. Bu durumda, *q-Nükleolus*, model 5.54 'de verildiği gibi tanımlanacaktır.

$$\min \varepsilon \tag{5.54}$$

$$\text{s.t.} \sum_{l \in L} w_l^{qnuc} = F(L) \tag{5.55}$$

$$\sum_{l \in L_c} w_l^{qnuc} \leq f_c + \varepsilon \qquad \forall c \in C(L) \tag{5.56}$$

$$w_l^{qnuc} \leq g_l \qquad \forall l \in L \tag{5.57}$$

$$w_l^{qnuc} \geq \lambda_l f_l \qquad \forall l \in L \tag{5.58}$$

$$w_l^{qnuc} \geq 0 \qquad \forall l \in L \tag{5.59}$$

$$\varepsilon \geq 0 \tag{5.60}$$

Bu modelin amacı, en büyük ihlali en küçüklemektir. Kısıt 5.55, *bütçe dengesi* kısıtıdır. Kısıt 5.56, *kararlılık* kısıtının gevşetilmiş halidir. Bu kısıt, bütün $C(L)$ kümesi üzerinden tanımlanır. Kısıt 5.57, *bireysel rasyonellik* kısıtıdır. Bu modelde, rota ayrıtlarına belirli oranda yüzde tasarruf garantisi verilmemektedir. Yalnızca, bir rota ayrıtına atanan maliyet; kendi bireysel maliyetinden daha fazla olmayacağının garantisi verilmektedir. Kısıt 5.58, seçilen rota ayrıtlarına atanan maliyet için bir alt sınır tanımlar. Kısıt 5.59 ve 5.60 işaret kısıttır. Ayrıca bu modelde yer

alan atama karar değişkeni (w_l^{qnuc}) ve ihlal karar değişkeni (ε) bütün pozitif reel sayılar üzerinden tanımlıdır.

Alternatif optimalden uzak durmak için koşul 5.42 kullanılacaktır. Dolayısıyla model 5.54 çözdürüldükten sonra υ_c değerleri kontrol edilecek ve sıfırdan küçük olan değerler dikkate alınacaktır. υ_c dual değişkeni sıfır veya sıfırdan küçük değer alır. Çünkü problem bir en küçükleme problemidir ve bu değişkene karşılık gelen kısıtlar küçük eşit kısıtlarıdır. Bu durumda $C_i(L)$ kümesinin tanımlanması gerekir. $C_i(L)$ kümesi, i. iterasyonda *kararlılık* kısıtını eşitlik olarak sağlayan çevrimlerin kümesidir. Matematiksel bir ifadeyle, $\hat{C}_i(L) = \{c | \upsilon_c < 0, c \in C(L)\}$ şeklinde tanımlanır. Bu durumda ortaya çıkan yeni model, model 5.61 'de verildiği gibidir.

$$\min \varepsilon_k \tag{5.61}$$

$$\text{s.t.} \sum_{l \in L} w_l^{qnuc} = F(L) \tag{5.62}$$

$$\sum_{l \in L_c} w_l^{qnuc} \leq f_c + \varepsilon_k \qquad \forall c \in C(L) : c \notin \bigcup_{i=1}^{k-1} C_i(L) \tag{5.63}$$

$$\sum_{l \in L_c} w_l^{qnuc} = f_c + \varepsilon_i \qquad \forall c \in \bigcup_{i=1}^{k-1} C_i(L) \tag{5.64}$$

$$w_l^{qnuc} \leq g_l \qquad \forall l \in L \tag{5.65}$$

$$w_l^{qnuc} \geq \lambda_l f_l \qquad \forall l \in L \tag{5.66}$$

$$w_l^{qnuc} \geq 0 \qquad \forall l \in L \tag{5.67}$$

$$\varepsilon_k \geq 0 \tag{5.68}$$

Bu modelin amacı, k. iterasyon için en büyük ihlali en küçüklemektir. Kısıt 5.62, bir önceki modelde yer alan kısıt 5.55 ile aynıdır. Bu kısıtta her hangi bir değişiklik yapılmaz. Kısıt 5.63, geçmiş iterasyonlarda bu kısıtı eşitlik olarak sağlamayan kısıtları içeren *kararlılık* kısıtıdır. Bu kısıtın bir önceki modeldeki kısıt 5.56 'den

farkı kısıtlı sayıda küme üzerinden yazılıyor olmasıdır. Kısıt 5.64, geçmiş iterasyonlarda *kararlılık* kısıtı 5.63 'yi eşitlik olarak sağlayan kısıtları temsil etmektedir. Bu kısıtta yer alan ε_i ifadesi bir karar değişkeni değil, bir parametredir. ε_i i. iterasyondaki ε değerini ifade etmektedir. Geriye kalan kısıtlar, model 5.54 ile aynıdır.

Nükleolus yönteminde tanımlanan durma koşulları *q-Nükleolus* yöntemi için de geçerlidir. *q-Nükleolus* 'un hesaplanabilmesi için ikinci modelin durma koşulları sağlanana kadar iteratif olarak çözdürülmesi gerekir. Modelin alternatif optimal çözümünün olmaması durumunda iterasyonlar sonlandırılır. Yani doğrusal modelin tek bir optimal çözümü (*unique solution*) olduğu durumda iterasyonlar durdurulur. Diğer durma koşulu olarak zaman limiti kullanılır. Model, büyük boyutlu örnekler için makul sürelerde optimal çözümü vermeyebilir. Bu nedenle, modelin iteratif olarak çözümü için bir zaman limiti tanımlanmıştır. *Nükleolus* modellerini çözmek için kullanılan sözde kod (psuedo code) aşağıda verilmektedir.

Algoritma 5.1 s-Nükleolus ve q-Nükleolus

repeat
  $Modeli$ çöz
  Kararlılık kısıtına karşılık gelen dual değişkenleri (υ_c) hesapla
  $C_i(L) \leftarrow \emptyset$
  for $c \in C(L)$ **do**
    if $\upsilon_c < 0$ **then**
      $C_i(L) \leftarrow C_i(L) \bigcup \{c\}$
  for $c \in C_i(L)$ **do**
    $Model \leftarrow Model - \sum_{l \in L_c} w_l \leq f_c + \varepsilon_i$
    $Model \leftarrow Model + \sum_{l \in L_c} w_l = f_c + \varepsilon_i^*$
  $i \uparrow$
  $harcananZaman \uparrow$
until $C_i(L) = \emptyset$ *veya* $harcananZaman > zamanLimit$

5.2.6 %q-Nükleolus (%q-Çekirdekçik) Maliyet Dağıtımı

s-Nükleolus 'da olduğu gibi, *q-Nükleolus* 'da da ihlal oranını miktar olarak en küçüklemeye çalışır. Böyle bir durumda eğer *kararlı* bir maliyet dağıtımı bulunamazsa ihlal oranları yüksek olabilir. Bu nedenle, modelin amaç fonksiyonu oran olarak güncellenerek yeni bir model önerilmiştir. Bunun için yalnızca kısıt 5.56 'nın, aşağıda verildiği şekilde (5.69) değiştirilmesi yeterlidir.

Kısıt 5.56 yerine kısıt 5.69 eklendiğinde, model maksimum ihlali oran olarak minimize etmeye çalışacaktır.

$$\sum_{l \in L_c} w_l^{qnuc} \leq f_c(1+\varepsilon) \qquad \forall c \in C(L) \tag{5.69}$$

Yukarıda tanımlanan kısıt modele eklendikten sonra, *q-Nükleolus* için tanımlanan çözdürme adımları bu model için de uygulanacaktır. Gerekli güncellemeler de yukarıda tanımlandığı gibi yapılacaktır. Gerekli güncellemeler yapıldıktan sonra Kısıt 5.63 ve 5.64 yerine kısıt 5.70 ve 5.71 kullanılacaktır.

$$\sum_{l \in L_c} w_l^{qnuc} \leq f_c(1+\varepsilon_k) \qquad \forall c \in C(L) : c \notin \bigcup_{i=1}^{k-1} C_i(L) \tag{5.70}$$

$$\sum_{l \in L_c} w_l^{qnuc} = f_c(1+\varepsilon_i) \qquad \forall c \in \bigcup_{i=1}^{k-1} C_i(L) \tag{5.71}$$

%Nükleolus modellerini çözmek için kullanılan sözde kod (psuedo code) aşağıda verilmektedir.

Algoritma 5.2 %s-Nükleolus ve %q-Nükleolus

repeat
  Modeli çöz
  Kararlılık kısıtına karşılık gelen dual değişkenleri (υ_c) hesapla
  $C_i(L) \leftarrow \emptyset$
  for $c \in C(L)$ **do**
    if $\upsilon_c < 0$ **then**
      $C_i(L) \leftarrow C_i(L) \bigcup \{c\}$
  for $c \in C_i$ **do**
    $Model \leftarrow Model - \sum_{l \in L_c} w_l \leq f_c(1+\varepsilon_i)$
    $Model \leftarrow Model + \sum_{l \in L_c} w_l = f_c(1+\varepsilon_i^*)$
  $i \uparrow$
  harcananZaman $\uparrow$
until $C_i(L) = \emptyset$ *veya harcananZaman* $>$ *zamanLimit*

6. KARARLILIK KOŞULUNUN DEĞERLENDİRİLMESİ

Kararlı bir maliyet dağıtımının bulunabilmesi için güçlü koşulların olduğu ve bu koşulları sağlamanın zor olduğu daha önceki bölümlerde gösterilmiştir. Böyle bir durumda önerilen yöntemlerin ve karşılaştırma için kullanılan yöntemlerin nasıl değerlendirileceği ciddi bir problemdir. Bu yöntemler, *kararlılık* koşulunu farklı derecelerde sağlayabilirler veya hiç sağlamayabilirler. Bu nedenle, farklı ölçüm metrikleri kullanılarak değerlendirilecektir.

Maliyet dağıtım yöntemleri; *kararlılık* kısıtı ihlal sayısı, ortalama ve en yüksek oranda ihlal miktarı kullanılarak karşılaştırılacaktır. Bu karşılaştırma, yalnızca geliştirilen maliyet dağıtım yöntemleri için kullanılacaktır. Çünkü *MD-SUKRK* problemi için geliştirilen yöntemler *kararlı* bir maliyet dağıtımı vermektedir. Bu nedenle, bu miktar ve sayılar bu yöntemler için sıfırdır. Bir çevrim için ihlal oranı, *Eşitlik 6.1* 'de verildiği gibi hesaplanmaktadır.

$$PSV(c) = \max\{0, \frac{\sum_{l \in L_c} w_l - f_c}{f_c}\} \qquad \forall c \in C(L) \qquad (6.1)$$

Bu karşılaştırma metriği çevrimler üzerinden tanımlanmıştır. Dolayısıyla aynı çevrim kümesi üzerinden değerlendirilmesi gerekir. Bunun için farklı yöntemler izlenebilir. Bunlardan ilki, bütün olurlu çevrimleri üreterek bu metriğin değerlendirilmesidir. Bu yöntem ancak rota ayrıtı belirli bir düzeydeyse kullanılabilir. Bütün olurlu çevrimleri üretmenin mümkün olmadığı durumda ise kısıtlı sayıda rastgele çevrimler üreterek değerlendirme yapılabilir. Bu çevrimlerin olurlu olmasına da gerek yoktur.

Diğer bir yöntem ise *STA* 'yı kullanmaktır. Bu problem mevcut maliyet dağıtımları üzerinden çözdürülerek, yukarıda tanımlanan karşılaştırma kriterleri hesaplanabi-

lir. Ancak alt problemi çözdükten sonra bu modelden elde edilen çevrim, bir daha üretilmemesi için bu modele kısıt olarak eklenmesi gerekir. Çünkü bu model her aşamada en yüksek ihlali veren çevrimi hesaplar. Bu durumda eklenmesi gereken kısıt, aşağıda 6.2 'de tanımlanan eşitsizliktir.

$$\sum_{l \in L_c} r_l + \sum_{a \in A_c} t_a \leq |N_c| - 1 \tag{6.2}$$

Yukarıda tanımlanan N_c kümesi, çevrim içerisinde yer alan tekrar etmeyen düğümler kümesi; L_c kümesi, çevrim içerisinde yer alan rota ayrıtı kümesi ve A_c kümesi ise çevrim içerisinde yer alan ayrıt (boş kamyon yükü hareketi) kümesini göstermektedir. Bu yöntemi kullanarak, ihlal sayısı ve miktarları hesaplanabilir. Ancak bu problemi her seferinden çözmek zaman alıcı olacaktır. Böyle bir durumda ise zaman limiti tanımlanabilir.

Bu çalışma kapsamında *SUKRK* problemi çözdürülürken bütün olurlu çevrimler üretilmiştir. Dolayısıyla yapılan analizler, üretilen bütün olurlu çevrimler üzerinden yapılacaktır. Yukarıda belirtilen yaklaşımlar, bütün çevrimlerin üretilmesinin mümkün olmadığı durumda veya alternatif olarak kullanılabilir.

Bunun dışında maliyet dağıtım yöntemleri, ortalama dolu gitme başına düşen maliyet, en yüksek dolu gitme başına düşen maliyet ve ortalama kazanç miktarı kriterleri kullanılarak da değerlendirilecektir. Bir rota ayrıtı için dolu gitme başına düşen maliyet *Eşitlik 6.3* 'de verildiği gibi hesaplanmaktadır. Bir rota ayrıtının iş birliğinden elde ettiği fayda (kazanım veya tasarruf) ise *Eşitlik 6.4* 'de verildiği gibi hesaplanmaktadır.

$$Rt(l) = \frac{w_l}{f_l} \tag{6.3}$$

$$Sv(l) = \left(\frac{g_l - w_l}{g_l} \right) \times \%100 \tag{6.4}$$

7. DENEYSEL ÇALIŞMALAR

Geliştirilen çözüm yöntemleri ve matematiksel modeller, belirli kurallara göre rastgele üretilmiş 24 ve 30 örnekle test edilmiştir. Bu örnekler şu parametreler ile üretilmiştir: nokta sayısı, her kümedeki kesir sayısı, her kümedeki nokta sayısı ve rota ayrıtı sayısı. Kümeler noktaların coğrafik yoğunluklarını temsil etmektedir. Nokta sayısı, kümedeki noktaların kesri ve her kümedeki nokta sayısı belirlendikten sonra, küme sayısı ve kümenin yarıçapı belirlenir.

Kümeler, noktaların coğrafik yoğunluklarını temsil ederek tanımlanır. Tedarikçi, fabrika ve dağıtım merkezi ve müşteriler tarafından temsil edilen üç ayrı sınıftaki noktalar kümesine ayrılan bir tedarik zinciri yapısı oluşturulmuştur. Her bir sınıfa ait olan noktaların kesri ve herhangi iki sınıf içerisindeki noktalar arasındaki rotaların kesri verilmektedir. Bu tür örnekler, tedarik zinciri örnekleri olarak adlandırılmaktadır.

Kümelerin merkezler, 1.800×1.800 metrekarelik bir alan içerisinden tek biçimli (*uniform*) olarak belirlenir. Küme içerisindeki noktalar, koordinatlar ve kümenin yarıçapı kullanılarak belirlenir. Rota ayrıtları ise, tam bir serimin ayrıtları arasından rastgele seçilerek belirlenir. Bütün rota ayrıtları, yalnızca bir başlangıç (*origin*) bir de bitiş (*destination*) düğümünden oluşur. Bu noktalar aynı küme içerisinde değildir. Düğümler ise büyük şehirleri temsil edecek şekilde kümelenmiş halde üretilmiştir.

Üretilen her örnekte; nokta sayısı, küme içerisindeki noktaların kesri, küme içerisindeki nokta sayısı ve rota ayrıtlarının sayısı çeşitlilik göstermektedir. Örnekler; 100 ve 150 düğümlü, 0.5 ile 0.8 arasında değişen nokta kesirli ve her kümede ortalama 20 düğüm olacak şekilde üretilmiştir.

Bu tez kapsamında çalışılan bütün matematiksel modeller ve geliştirilen bütün çözüm yöntemleri *Java* programlama diliyle kodlanmıştır ve *2 Adet 2.00 GHz Intel Xeon CPU E5-2650 işlemcili, 128 GB Ram* kapasiteli iş istasyonu üzerinde koşturulmuştur. Matematiksel modeller, *Cplex OPL 12.9* ticari çözdürücüsüyle çözdürülerek tablolaştırılmıştır. Tablolarda kullanılan ifadeler ve kısaltmalar aşağıda verildiği gibidir.

Ort	: Ortalama
Min	: Minimum
Mak	: Maksimum
İM	: İhlal Miktarı
İS	: İşlem Süresi
KK	: Kaba Kuvvet
ZL	: Zaman Limiti
ÇS	: Çözüm Süresi
OİO	: Ortalama İhlal Oranı
$\|\hat{C}(L)\|$	: Seçilen Çevrim Sayısı
MİO	: Maksimum İhlal Oranı
$\|\hat{L}\|$	: Kapsanan Rota Ayrıtı Sayısı
MTİO	: Minimum Toplam İhlal Oranı
DFK	: Dal-Fiyat ve Kesi Algoritması
MMİO	: Minimum Maksimum İhlal Oranı
UOMD	: Uzaklık Orantılı Maliyet Dağıtımı
SUKRK	: Sayı ve Uzunluk Kısıtlı Rota Kapsama Problemi
MD-SUKRK	: Maliyet Dağıtımlı Sayı ve Uzunluk Kısıtlı Rota Kapsama Problemi
ÇKMD-SUKRK	: Çok Koalisyonlu Maliyet Dağıtımlı Sayı ve Uzunluk Kısıtlı Rota Kapsama Problemi

MD-SUKRK problemi için elde edilen çözümü tanımlayan fonksiyon $F(C(L),L)$ ve x_c^* 'ler ise bu problemden elde edilen optimal çözüm olsun. *AS* ise *Bölüm 3.6 'da* alt sınır için tanımlanan problemin amaç fonksiyonu değeri ve x_{kl}^* 'ler ise bu problemden elde edilen optimal çözümler olsun. Bu takdirde, *LCP Gap* değeri aşağıda verildiği şekilde hesaplanır.

$$F(C(L),L) = \sum_{c \in C(L)} f_c x_c^* + \sum_{l \in L} g_l u_l^* \tag{7.1}$$

$$AS = \sum_{k \in L} \sum_{l \in L - \{k\}} c_{kl} x_{kl}^* + \sum_{l \in L} \eta_l f_l \tag{7.2}$$

$$LCP\ Gap = \left[\frac{F(C(L),L) - AS}{AS} \right] \times \%100 \tag{7.3}$$

Yöntemlerde ve modellerde kullanılan parametreler aşağıda verildiği gibidir.

θ_l	$= 0.05$	$\forall l \in L$
λ_l	$= 0.20$	$\forall l \in L$
η_l	$= 1.00$	$\forall l \in L$
ρ_a	$= 0.80$	$\forall a \in A$
K_{max}	$= 4$	
L_{max}	$= 3{,}850$	
$Zaman\ Limiti\ (sn)$	$= 7{,}200$	

Çözdürülen örneklerin boyutlarıyla ilgili bilgiler Tablo 7.1 'de verilmektedir. Tabloda yer alan bilgiler sırasıyla; örneklerin içerdikleri düğüm sayısı ($|N|$), rota ayrıtı sayısı ($|L|$) ve kümelerde yer alan noktaların yoğunluklarını (CPR) ifade etmektedir.

Tablo 7.1: Çözdürülen Örnekler Hakkında Genel Bilgiler

Örnek	$\|N\|$	$\|L\|$	CPR
1-3	100	100	0.5
4-6	100	100	0.8
7-9	100	200	0.5
10-12	100	200	0.8
13-15	100	100	0.5
16-18	100	100	0.8
19-21	100	200	0.5
22-24	100	200	0.8

7.1 *SUKRK* Probleminin Optimal Olarak Çözülmesi

Önerilen yaklaşım ile iki aşamalı (klasik) yaklaşımı karşılaştırmak için ilk olarak *SUKRK* probleminin optimal olarak çözülmesi gerekmektedir. Daha sonra, bu problemden elde edilen optimal çözüm kullanılarak; ortaya çıkan minimum maliyet, çeşitli yöntemler kullanılarak rota ayrıtlarına dağıtılmaya çalışılacaktır.

Bu nedenle, *SUKRK* problemi bir ticari programla çözdürülerek; Tablo 7.2 'de raporlanmıştır. Tabloda verilen çözümler iki saatlik zaman limiti altında ve bütün olası çevrimler üretilerek alınmıştır. Bu çözüm yöntemi literatürde *kaba kuvvet (brute force)* olarak adlandırılmaktadır. *Kaba kuvvet (KK)* yöntemiyle, çözdürülen 24 örneğin 23 tanesinde optimal çözüm elde edilebilmiştir. Optimal çözüm elde edilememiş örnek (11.örnek) için, bütün olası çevrimleri üretmek; kullanılan iş istasyonu için mümkün olmamıştır. Bu tablonun ikinci sütununda, elde edilen amaç fonksiyonu değerinin alt sınıra olan yüzde uzaklığı (*LCP Gap*) gösterilmektedir. Bu değer bütün örnekler için oldukça düşüktür. Bunun temel sebebi, kullanılan sayı (K_{max}) ve uzunluk (L_{max}) kısıtının yeterince kısıtlayıcı olmamasıdır. Bu

kısıtların gereksiz (redundant) olduğu örnekler için, *SUKRK* problemi, *RK* problemine dönüşmektedir. Problemi çözmek için harcanan sürelere bakıldığında, 21 örnek için çok kısa sürelerde çözüm alındığı görülmektedir. Kapsanan rota sayısına ($|\hat{L}|$) ve oranına ($|\hat{L}|/|L|$) bakıldığında, %95 'in üzerinde rota ayrıtının kapsanabildiği görülmektedir. Kapsanan rota ayrıtı sayısı hesaplanırken, tek boyutlu çevrimlerin kapsadığı rota ayrıtları dışarıda bırakılmıştır. Üretilen çevrim sayısı ($|C(L)|$), problem bütün olası çevrimler üzerinden tanımlandığı için oldukça yüksektir. Beklenildiği gibi, örneklerde yer alan rota ayrıtı sayısı arttığında üretilen çevrim sayısı da üssel oranda artmıştır. Yirminci örnek için 9,959,783 tane olurlu çevrim üretilmiştir.

Tablo 7.2: *SUKRK* Problemi için *KK* Yönteminden Elde Edilen Çözümler

Örnek	LCP Gap	Cplex Gap	Çöz. Süresi (sn)	$\lvert\hat{L}\rvert$	$\lvert\hat{L}\rvert/\lvert L\rvert$	$\lvert\hat{C}(L)\rvert$	$\lvert C(L)\rvert$
1	1.0	0.0	6.7	96	0.96	36	86,399
2	0.4	0.0	27.4	96	0.96	34	301,241
3	2.4	0.0	15.9	98	0.98	33	166,085
4	0.4	0.0	8.0	97	0.97	36	85,105
5	0.1	0.0	23.8	99	0.99	31	332,606
6	0.1	0.0	15.9	98	0.98	32	272,819
7	0.8	0.0	107.2	195	0.98	73	608,667
8	1.0	0.0	65.0	193	0.97	70	467,757
9	0.4	0.0	196.2	197	0.99	68	865,391
10	0.6	0.0	2,700.4	191	0.96	65	3,760,950
11	-	-	-	-	-	-	-
12	0.0	0.0	1,804.1	197	0.99	66	3,689,916
13	0.8	0.0	13.9	98	0.98	35	190,167
14	0.2	0.0	6.8	98	0.98	38	98,371
15	1.1	0.0	6.2	99	0.99	35	71,943
16	0.1	0.0	12.0	95	0.95	32	188,298
17	0.3	0.0	5.7	96	0.96	38	80,298
18	0.1	0.0	44.8	97	0.97	32	532,870
19	1.5	0.0	104.0	193	0.97	72	714,321
20	0.0	0.0	624.6	196	0.98	62	9,959,783
21	0.8	0.0	159.6	195	0.98	74	1,183,586
22	0.3	0.0	79.6	196	0.98	75	697,036
23	0.3	0.0	112.8	196	0.98	73	807,603
24	0.3	0.0	109.2	195	0.98	69	1,027,822

7.2 *MD-SUKRK* Problemi için *DFK* 'nin ve *KK* 'nın Karşılaştırılması

MD-SUKRK problemi için geliştirilen *DFK* yönteminden elde edilen çözümlerin kalitesini analiz etmek için, *KK* 'dan elde edilen çözümlerle karşılaştırılmıştır. Karşılaştırmalar; *LCP Gap*, çözüm süresi, kapsanan rota ayrıtı sayısı, seçilen ve üretilen çevrim sayıları üzerinden yapılacaktır. *LCP Gap*, yöntemden elde edilen amaç fonksiyonu değerinin; alt sınıra olan uzaklığını göstermektedir. *LCP Gap* ne kadar düşükse, elde edilen çözüm o kadar iyidir. Çözüm süresi her iki yöntem için de iki saat ile kısıtlıdır. En kötü durumda, yöntemler bu sınıra takılarak durmaktadır. Kapsanan rota ayrıtı sayısı, çözüm kalitesini gösteren diğer bir faktördür. Ne kadar fazla rota ayrıtı kapsanırsa, koalisyon dışında kalan rota ayrıtı sayısı o kadar az olacaktır. Üretilecek çevrim sayısı, *DFK* yöntemi için oldukça önemlidir. *KK* yönteminde, bütün olurlu çevrimlerin üretilmesi gerektiği için; üretilecek çevrim sayısının oldukça yüksek olması beklenir. *DFK* yönteminde ise, üretme aşamasını akıllı bir şekilde yaptığı için, bu sayının oldukça düşük olması beklenir.

Tablo 7.3 'de *MD-SUKRK* problemi için *KK* ve *DFK* yönteminden elde edilen *LCP Gap* ve *Çözüm Süresi (sn)* değerleri verilmektedir. *KK* yöntemi kullanarak yalnızca bir örnek için optimal çözüm elde edilebilmiştir. Geriye kalan 22 örnek için iki saatlik zaman limiti sonunda elde edilen çözümler verilmektedir. Bir örnek (11.örnek) için ise, herhangi bir çözüm alınamamıştır. Zaman limitinden ötürü *KK* yönteminden iyi çözümler elde edilememiştir. Problem, *DFK* ile çözüldüğünde ise; oldukça iyi *LCP Gap* değerleri elde edildiği görünmektedir. 23 örnek için *DFK*, *KK* daha iyi çözümler vermiştir. *DFK* bütün örnekler için zaman limitine ulaşıp durmuştur. Dolayısıyla, hiçbir örnek için optimal çözüm elde edilememiştir. Ancak buna rağmen, elde edilen *LCP Gap* değerleri oldukça iyidir. Problemin doğası gereği, *kararlı* bir çözümün bu boyuttaki örnekler için kısa sürelerde elde edilmesi oldukça güçtür. 20 örnek için *DFK* 'den elde edilen *LCP Gap* değer-

leri %8.2 'nin altındadır. En yüksek *LCP Gap* değeri 11.örnek için elde edilmiştir (%28.2). Bu örnek için *Kaba Kuvvet* yönteminden herhangi bir çözüm elde edilememiştir. Bunun temel sebebi, üretilecek olası çevrimlerin sayısının oldukça fazla olmasıdır.

Tablo 7.4 'de *MD-SUKRK* problemi için *KK* ve *DFK* yönteminden elde edilen kapsanan rota ayrıtı sayısı ve oranı, seçilen çevrim ve üretilen çevrim sayıları verilmektedir. *KK*, çözdürülen 24 örneğin 15 tanesi için en kötü olurlu çözümü iyileştirememiştir. En kötü olurlu çözüm bütün rota ayrıtlarını dışarıda bırakılarak alınan çözümdür. Bu nedenle, 15 örnek için kapsanan rota ayrıtı sayısı sıfırdır. Geriye kalan örneklerden yalnızca dört tanesi için rota ayrıtlarının %80 'den fazlası kapsanabilmiştir. Üretilen çevrim sayılarına bakıldığında ise, bu sayının beklenildiği gibi yüksek olduğu görünmektedir. *DFK* için elde edilen çözümlere bakıldığında, kapsanan rota ayırtı oranının bir örnek hariç %60 'ın üzerinde olduğu görülmektedir. Bu oranın yüksek olduğu her örnekte *LCP-Gap* değerleri beklenildiği gibi oldukça düşüktür. Problem, *kararlılık* kısıtını ve diğer maliyet dağıtım kısıtlarından dolayı bazı rota ayrıtlarını koalisyon dışında bırakmaktadır. Bu nedenle, *DFK* bazı örnekler için yüksek rota ayrıtını kapsamışken, bazı örneklerde daha düşük sayıda rota ayrıtı kapsayabilmiştir. Diğer taraftan, *DFK* ile problem çözülürken maksimum 20,519 tane çevrim üretilmiştir. Bu sayı, *KK* için beş milyonun üzerindedir. Tablo 7.3 ve 7.4 birlikte değerlendirildiğinde; geliştirilen yöntemin makul sürelerde, çok az sayıda çevrim üreterek oldukça iyi çözümler verdiği söylenebilir.

Tablo 7.3: *MD-SUKRK* Problemi için Elde Edilen Çözümlerin Karşılaştırılması-1

	KK			*DFK*	
Örnek	LCP Gap	Cplex Gap	Çöz. Süresi (sn)	LCP Gap	Çöz. Süresi (sn)
1	1.0*	0.0*	89.6*	3.1	*ZL*
2	40.0	28.3	*ZL*	11.8	*ZL*
3	4.3	1.9	*ZL*	2.5	*ZL*
4	33.4	24.8	*ZL*	1.4	*ZL*
5	60.2	37.5	*ZL*	1.6	*ZL*
6	55.8	35.7	*ZL*	15.2	*ZL*
7	48.9	32.3	*ZL*	1.8	*ZL*
8	50.6	32.9	*ZL*	3.2	*ZL*
9	55.1	35.3	*ZL*	4.3	*ZL*
10	48.8	98.9	*ZL*	5.9	*ZL*
11	-	-	-	28.2	*ZL*
12	55.2	100.0	*ZL*	7.5	*ZL*
13	30.9	23.0	*ZL*	3.3	*ZL*
14	23.7	19.0	*ZL*	1.8	*ZL*
15	36.5	25.9	*ZL*	10.8	*ZL*
16	1.4	1.3	*ZL*	2.1	*ZL*
17	0.8	0.5	*ZL*	2.2	*ZL*
18	55.0	35.4	*ZL*	6.2	*ZL*
19	48.9	31.8	*ZL*	2.7	*ZL*
20	43.2	100.0	*ZL*	5.7	*ZL*
21	39.6	27.8	*ZL*	2.9	*ZL*
22	58.5	36.7	*ZL*	8.1	*ZL*
23	55.0	35.3	*ZL*	3.6	*ZL*
24	48.7	32.5	*ZL*	7.8	*ZL*

*: *Optimal Çözüm*

Tablo 7.4: *MD-SUKRK* Problemi için Elde Edilen Çözümlerin Karşılaştırılması-2

	KK				*DFK*			
Örnek	$\|\hat{L}\|$	$\|\hat{L}\|/\|L\|$	$\|\hat{C}(L)\|$	$\|C(L)\|$	$\|\hat{L}\|$	$\|\hat{L}\|/\|L\|$	$\|\hat{C}(L)\|$	$\|C(L)\|$
1	95*	0.95*	36*	67,339*	82	0.82	30	3,779
2	0	0.00	0	208,794	47	0.47	17	4,730
3	90	0.90	31	116,999	95	0.95	32	3,977
4	34	0.34	13	65,495	83	0.83	29	3,328
5	0	0.00	0	245,484	83	0.83	27	4,549
6	0	0.00	0	227,590	59	0.59	18	5,235
7	0	0.00	0	394,904	158	0.79	60	12,071
8	0	0.00	0	346,714	151	0.76	58	11,786
9	0	0.00	0	613,232	143	0.72	53	13,613
10	0	0.00	0	2,800,185	132	0.66	45	14,532
11	-	-	-	-	45	0.23	16	20,519
12	0	0.00	0	2,664,581	132	0.66	47	16,388
13	26	0.26	9	143,088	73	0.73	24	4,122
14	35	0.35	13	71,951	72	0.72	24	4,214
15	34	0.34	13	57,927	63	0.63	23	4,114
16	88	0.88	31	147,233	81	0.81	27	4,376
17	81	0.81	33	52,899	69	0.69	27	3,383
18	0	0.00	0	376,471	68	0.68	24	4,748
19	0	0.00	0	531,313	161	0.81	59	12,073
20	0	0.00	0	5,433,414	127	0.64	42	17,658
21	0	0.00	0	921,212	138	0.69	48	12,970
22	0	0.00	0	565,408	133	0.67	52	10,736
23	0	0.00	0	587,276	146	0.73	53	11,294
24	0	0.00	0	907,855	120	0.60	41	12,752

*: *Optimal Çözüm*

7.3 *MD-SUKRK* Problemi için Elde Edilen Dolu Gitme Başına Düşen Maliyet ($R(l)$) ve Ortalama Yüzde Kazanç ($Sv(l)$) Miktarları

Rota ayrıtları için dolu gitme başına düşen maliyet ($R(l)$) ve rota ayrıtlarının çözümden elde ettikleri yüzde tasarruf miktarları ($Sv(l)$) önemli karşılaştırma kriterleridir. Bu kriterler kullanılarak, *DFK* 'den elde edilen maliyet dağıtımları analiz edilecektir. Dolu gitme başına düşen maliyet, rota ayrıtına atanan maliyetin; o rota ayrıtının dolu gitme maliyetine oranla ne kadar az veya fazla olduğunu göstermektedir. Bu oranın bire yakın veya eşit olması, rota ayrıtının kendi dolu gitme maliyetini ödediği; boş gitme maliyetine katlanmadığı anlamına gelmektedir. Yüzde tasarruf miktarı ise rota ayrıtına atanan maliyetin, kendi bireysel maliyetinden ne oranda az olduğunu göstermektedir. Yüzde tasarruf miktarlarının yüksek olması, *DFK* 'den elde edilen maliyetin dağıtımının ne kadar iyi olduğunu göstermektedir.

Tablo 7.5 'de *DFK* yönteminden kapsanan rota ayrıtları için elde edilen dolu gitme başına düşen maliyet ($R(l)$) ve ortalama yüzde kazanç ($Sv(l)$) miktarları gösterilmektedir. Dolu gitme başına düşen maliyetin bire yakın olması maliyet dağıtımı için daha iyidir. Çünkü böyle bir durumda rota ayrıtlarına, dolu gitme maliyetleri atanmıştır. Kapsanan bir rota ayrıtı için bu değerin alabileceği en yüksek değer 1.71 $((\eta_l = 1.00 + \rho_a = 0.80) \times ((1 - \theta_l = 0.05) = 0.95))$ 'tir. Kapsanmayan her bir rota ayrıtı içinse, 1.80 $(\eta_l = 1.00 + \rho_a = 0.80)$ 'tir. 1.71 değeri aynı zamanda bir rota ayrıtının elde edebileceği en düşük tasarruf miktarı olan %5 'i göstermektedir. Bu değerin alabileceği en düşük değer ise λ_l değerine karşılık gelen 0.2 'tir. Çünkü bu değer; kapsanan rota ayrıtlarına atanan maliyet için, dolu gitme maliyetleri oranında bir alt sınır tanımlar. Altı örnek için, $R(l)$ alt sınır olan 0.2 değerini almıştır. 21 örnek içinse, üst sınır olan 1.71 değerini almıştır. Ortalamalara bakıldığında ise, $R(l)$ değerlerinin ortalamasının 1.1 civarında olduğu görülmektedir. Diğer bir değişle, ortalamada kapsanan rota ayrıtlarına yaklaşık olarak kendi dolu

gitme maliyetleri atanmaktadır. Kapsanan rota ayrıtları için elde edilen $Sv(l)$ değerlerine bakıldığında, 14 örnek için minimum tasarruf miktarı olan %5 değerinin elde edildiği görülmektedir. Maksimum $Sv(l)$ değerine bakıldığında bütün örnekler için %75 'in üzerinde olduğu görülmektedir. Ortalamada ise, koalisyona dahil olan rota ayrıtları %40.1 oranında tasarruf sağlamaktadır.

Tablo 7.5: *MD-SUKRK* Probleminde *DFK* 'den Elde Edilen Rota Ayrıtları için Dolu Gitme Başına Düşen Maliyet ($R(l)$) ve Yüzde Kazanç ($Sv(l)$)

	$R(l)$			$Sv(l)$		
Örnek	Min	Ort	Mak	Min	Ort	Mak
1	0.2	1.1	1.6	8.8	39.2	88.4
2	0.3	1.1	1.7	5.0	41.6	86.1
3	0.3	1.2	1.7	5.0	35.3	84.4
4	0.3	1.1	1.7	5.0	41.4	82.5
5	0.3	1.1	1.7	5.0	39.3	83.5
6	0.4	1.0	1.7	6.3	43.9	78.8
7	0.2	1.1	1.7	5.0	36.5	87.3
8	0.2	1.2	1.7	5.0	36.4	86.9
9	0.5	1.1	1.7	5.0	39.7	71.6
10	0.2	1.0	1.7	5.0	44.4	86.5
11	0.3	1.1	1.7	5.0	41.8	83.1
12	0.3	1.1	1.7	6.6	41.8	86.2
13	0.4	1.1	1.7	5.0	40.1	78.2
14	0.3	1.1	1.7	8.4	37.5	83.0
15	0.4	1.1	1.7	8.3	40.1	77.8
16	0.2	1.1	1.7	5.0	40.8	88.1
17	0.2	1.1	1.7	5.0	39.3	88.5
18	0.3	1.0	1.7	6.5	42.2	81.9
19	0.4	1.1	1.6	8.9	37.7	77.2
20	0.3	1.1	1.7	5.0	38.1	80.9
21	0.4	1.1	1.6	9.4	36.6	75.6
22	0.3	1.0	1.7	5.7	43.5	80.9
23	0.3	1.0	1.7	5.0	42.8	84.9
24	0.3	1.0	1.7	7.3	42.0	84.4
Ortalama	0.3	1.1	1.7	6.1	40.1	82.8

7.4 *ÇKMD-SUKRK* Probleminin Optimal Olarak Çözülmesi

Tek koalisyonlu yapıda, sıkı kısıtlardan dolayı; birçok rota ayrıtı koalisyon dışında bırakılabilmektedir. Daha fazla sayıda rota ayrıtı kapsayabilmek için, çok koalisyonlu yapı önerilmiştir. Çok koalisyonlu yapının; çözüm süresine olan etkisini ve ne oranda rota ayrıtını kapsayabileceğini analiz etmek için, *ÇKMD-SUKRK* problemi *KK* yöntemiyle çözülmüştür.

Tablo 7.6 'de bu yöntemden elde edilen çözümler verilmektedir. Bu problem iki koalisyon üzerinden çözülmüştür. Problem rota ayrıtlarını iki koalisyona ayırarak, her bir koalisyon için *kararlı* bir maliyet dağıtımı bulmayı amaçlamaktadır. *MD-SUKRK* problemi için *KK*, 13 örnek için optimal çözümü bulabilmiştir. Bu problemin daha kolay çözülebilir olmasının sebebi, rota ayrıtlarını koalisyonlara ayırarak; *kararlı* bir maliyet dağıtımının bulunmasının daha kolay olmasıdır. Beklenildiği gibi, bu problemde daha fazla rota ayrıtı kapsanabilmiştir. Bu problemin ele alınmasındaki amaç, koalisyon sayısın artması durumunda daha rahat çözümlerin alınabileceğini ve daha fazla sayıda rota ayrıtı kapsanabileceğini göstermektir. 12 örnek için kapsanan rota ayrıtı sayısı %82 'nin üzerindedir. Koalisyon sayısı arttırılarak, kapsanan rota ayrıtı sayısı da arttırılabilir. Bu problem için, üretilen çevrim sayısında herhangi bir değişim söz konusu değildir. Ancak değişkenler ve kısıtlar koalisyonlar üzerinden tanımlandığı için, bu problemin içerdiği değişken ve kısıt sayısı yaklaşık olarak iki kat fazladır.

Tablo 7.6: *ÇKMD-SUKRK* Problemi için *KK* Yönteminden Elde Edilen Çözümler

Örnek	LCP Gap	Cplex Gap	Çöz. Süresi (sn)	$\lvert\hat{L}\rvert$	$\lvert\hat{L}\rvert/\lvert L\rvert$	$\lvert\hat{C}(L)\rvert$	$\lvert C(L)\rvert$
1*	1.0*	0.0*	80.9*	96*	0.96*	36*	67,339*
2*	0.4*	0.0*	522.0*	96*	0.96*	34*	208,794*
3*	2.4*	0.0*	1,105.1*	98*	0.98*	33*	116,999*
4*	0.3*	0.0*	536.1*	97*	0.97*	37*	65,495*
5*	0.1*	0.0*	4,110.2*	99*	0.99*	31*	245,484*
6*	0.1*	0.0*	2,532.7*	98*	0.98*	32*	227,590*
7	7.5	16.3	*ZL*	164	0.82	62	394,904
8	49.8	100.0	*ZL*	4	0.02	2	346,714
9	55.1	100.0	*ZL*	0	0.00	0	613,232
10	48.8	100.0	*ZL*	0	0.00	0	2,800,185
11	-	-	-	-	-	-	-
12	55.2	100.0	*ZL*	0	0.00	0	2,664,581
13*	0.8*	0.0*	1,609.8*	98*	0.98*	35*	143,088*
14*	0.2*	0.0*	93.8*	94*	0.94*	35*	71,951*
15*	1.1*	0.0*	113.8*	99*	0.99*	35*	57,927*
16*	0.1*	0.0*	1,604.9*	95*	0.95*	32*	147,233*
17*	0.3*	0.0*	54.0*	93*	0.93*	35*	52,899*
18*	0.1*	0.0*	1,868.4*	97*	0.97*	32*	376,471*
19*	1.5*	0.0*	5,406.3*	193*	0.97*	72*	531,313*
20	-	-	-	-	-	-	-
21	39.6	100.0	*ZL*	0	0.00	0	921,212
22	58.5	100.0	*ZL*	0	0.00	0	565,408
23	55.0	100.0	*ZL*	0	0.00	0	587,276
24	48.7	100.0	*ZL*	0	0.00	0	907,855

**: Optimal Çözüm*

7.5 Maliyet Dağıtım Yöntemlerinin Karşılaştırılması

Geliştirilen maliyet dağıtım yöntemlerini hem birbirleriyle hem de literatürde yer alan yöntemlerle karşılaştırmak için, çeşitli analizler yapılmıştır. Burada amaçlanan, hangi yönteminin hangi karşılaştırma kriterine göre daha iyi olduğunu tespit etmektir. Aynı zamanda, literatürde yer alan iki aşamalı yaklaşım ile önerilen yaklaşımın karşılaştırılması da amaçlanmaktır.

Yöntemler ilk olarak; *ihlal miktarları (İM), ortalama ihlal oranları (OİO)* ve *maksimum ihlal oranları (MİO)* üzerinden karşılaştırılacaktır. *İM*, yöntemden elde edilen maliyet dağıtımının *kararlılık* kısıtını kaç çevrim için ihlal ettiğini göstermektedir. İhlal oranları ise, bu ihlalin ne kadar olduğunu oransal olarak göstermektedir. Bu ihlallerin sayısal ve oransal olarak az olması, yöntemin iyi olduğunu göstermektir. Eğer *İM* değeri sıfır ise, bu durumda o yöntemden elde edilen maliyet dağıtımının, *kararlı* bir maliyet dağıtımı olduğu söylenebilir. Aynı zamanda, ilgili örneğin *çekirdeği* boş küme değildir. *İM* değeri sıfırdan farklı olması durumda; ihlal oranları karşılaştırma açısından önemlidir. Eğer ihlal oranları yüksekse; ilgili yöntemin kötü sonuçlar verdiği söylenebilir.

Bu kriterlere ek olarak, yöntemler; birim dolu gitme başına düşen maliyet ($R(l)$), ortalama yüzde kazanç ($Sv(l)$) ve çözüm süreleri üzerinden de karşılaştırılacaktır. Bu kriterleri kullanmaktaki amaç, yöntemlerin benzer çözümler vermesi durumunda; hangi yöntemin daha tercih edilebilir olduğuna karar vermektir.

Tablo 7.7 ve 7.8 'de maliyet dağıtım yöntemlerinden elde edilen *ihlal miktarları (İM), ortalama ihlal oranları (OİO)* ve *maksimum ihlal oranları (MİO)* verilmektedir. Bu tablolarda gösterilen çözümler, *SUKRK* probleminin optimal çözümü kullanılarak elde edilmiştir. İlk olarak *SUKRK* problemi *KK* yöntemi kullanılarak optimal çözülmüştür. Bu çözümden elde edilen çevrimler ve toplam minimum

maliyet kullanılarak, bütün rota ayrıtlarına düşen maliyet hesaplanmıştır. Bu hesaplamalar, literatürde kullanılan ve önerilen yöntemler kullanılarak yapılmıştır. *İM* göz önüne alındığında, en düşük değerler sırasıyla *s-Nükleolus* ve *q-Nükleolus* 'dan elde edilmiştir. En yüksek değerler ise sırasıyla *UOMD* ve *SD* 'den elde edilmiştir. *OİO* dikkate alındığında; *MTİO*, *s-Nükleolus*, *%s-Nükleolus*, *q-Nükleolus* ve *%q-Nükleolus* %0.1 'in altında çözümler vermiştir. Bu yöntemlerden elde edilen maliyet dağıtımları, ihlallere sebep olsa da ihlal oranları ortalamaya bakıldığında oldukça düşüktür. *MİO* için en yüksek değerler; *UOMD* için %51.0, *SD* için %211.8, *MTİO* için %1.2, *MMİO* için %15.0 'tır. Diğer yöntemler için *MİO* değerleri %0.1 'in altındadır. *MTİO* ve *MMİO* yöntemleri kıyaslandığında beklenildiği gibi; *MMİO* daha yüksek sayıda ihlale sebep olurken, *MMİO* daha yüksek oranda ihlale sebep olmuştur. Bu sonuç dikkate alındığında iki yöntem arasında bir ödünleşimin olduğu söylenebilir. Eğer amaç ihlal sayısını düşük tutmak ise *MTİO* yöntemi, eğer amaç ihlal oranlarını düşük tutmak ise *MMİO* yöntemi seçilmelidir. *Nükleolus* temelli yöntemler dikkate alındığında ise, oldukça iyi sonuçların alındığı söylenebilir. Bu yöntemler kullanılarak, dokuz örnek için *kararlı* bir maliyet dağıtımı elde edilebilmiştir. Başka bir değişle, bu örneklerin *çekirdekleri* boş küme değildir. Önermelerde ispat edildiği gibi; bu örneklerin doğrusal gevşetmeleri, tam sayılı çözümler vermektedir.

Tablo 7.9 ve 7.10 'de maliyet dağıtım yöntemlerinden elde edilen dolu gitme başına düşen maliyet ($R(l)$) verilmektedir. Maksimum $R(l)$ değerinin 1.8 olması, en az bir rota ayrıtına kendi bireysel maliyetinin (g_l) atandığını gösterir. Böyle bir durumda, bu rota ayrıtının yüzde kazancı sıfırdır. Maksimum $R(l)$ değerinin 1.8 'den büyük olması, bu rota ayrıtlarına kendi bireysel maliyetlerinden fazla maliyet dağıtıldığını gösterir. Maksimum $R(l)$ değerinin 1.7 olması, ise rota ayrıtının %5 oranında tasarruf sağladığını gösterir. Bu kısıtı yalnızca *MTİO* ve *MMİO* yöntemleri dikkate aldığı için, en iyi maksimum $R(l)$ değerleri bu yöntem için elde

edilmiştir. En kötü değer ise *SD* için elde edilmiştir. Bu yöntemde en az bir rota ayrıtına atanan maliyet, kendi bireysel maliyetinden yüksektir. *Nükleolus* bazlı yöntemler değerlendirildiğinde benzer çözümlerin alındığı söylenebilir. Bu yöntemler, koalisyona katılan rota ayrıtları için herhangi bir tasarruf garanti etmez. Bu nedenle, bu yöntemler için en az bir rota ayrıtının elde ettiği kazanım oranı sıfırdır.

Tablo 7.11 'de rota ayrıtları için elde edilen ortalama yüzde kazanç ($Sv(l)$) miktarları gösterilmektedir. En düşük $Sv(l)$ değeri sırasıyla, *SD* ve *UOMD* 'den elde edilmiştir. En yüksek değer ise *MTİO* yönteminden elde edilmiştir. *Nükleolus* temelli maliyet dağıtım yöntemleri ise benzer çözümler vermiştir. Bütün metotlar için ortalama kazanç miktarı %30 'un üzerindedir.

Tablo 7.12 'de maliyet dağıtım yöntemleri için harcanan süreler verilmektedir. Maliyet dağıtım yöntemlerini karşılaştırırken kullanılan en önemli ölçütlerden bir tanesi de, yöntemler için harcanan sürelerdir. Tabloda verilen süreler, yalnızca yöntemler için harcanan süreleri göstermektedir. *SUKRK* problemini çözmek için harcanan süre, bu sürelere dahil edilmemiştir. Beklenildiği gibi *UOMD* ve *SD*, birçok örnek için bir saniyenin altında sürelerde çözüm vermişlerdir. Bu yöntemler için harcanan süre sırasıyla maksimum 2.5 ve 4.2 saniyedir. *MMİO* bir örnek hariç tüm örnekler için 232 saniyenin altında sonuç vermiştir. En yüksek çözüm süresi ise 991.3 saniye ile 20.örneğe aittir. *MTİO* yönteminden ise iki örnek (10. ve 12.örnekler) için, iki saatlik zaman limiti altında olurlu bir çözüm alınamamıştır. Bir örnek içinse zaman limitine takılarak çözüm vermiştir. *Nükleolus* bazlı çözüm yöntemlerinden ise çözüm süresi, diğer yöntemlere kıyasla daha uzundur. *s-Nükleolus* yöntemi, dört örnek için zaman limitine takılmıştır. *q-Nükleolus* yöntemi ise, sekiz örnek için zaman limitine takılmıştır. Bunun sebebi, modelde yer alan kısıtların problemi çözmeyi zorlaştırması olabilir. Bu yöntemlerin amaç fonk-

siyonları miktar olarak değil de, yüzde olarak hesaplandığında ise; çözüm sürelerinin kısaldığı görülmektedir. *%s-Nükleolus* yöntemi, yalnızca iki örnek için zaman limitine takılmıştır. *%q-Nükleolus* yöntemi ise, yalnızca bir örnek için zaman limitine takılmıştır.

Maliyet dağıtım yöntemlerinden elde edilen çözümleri gösteren tablolar birlikte değerlendirildiğinde, *%s-Nükleolus* ve *%q-Nükleolus* yönteminin iyi sonuçlar verdiği söylenebilir. Bu yöntemler için hem ihlal miktarı oldukça düşüktür hem de çözüm, zaman limiti içerisinde alınabilmektedir. *s-Nükleolus* ve *q-Nükleolus* yöntemi düşük ihlale sahip çözümler verse de, bazı örnekler için zaman limiti içerisinde çözüm almak mümkün olmamıştır. Doğrusal gevşetmesi tam sayılı çözüm veren bütün örnekler için *nükleolus* temelli yöntemler *kararlı* maliyet dağıtımı vermiştir. Bu da yapılan önermeleri doğrulamaktadır. *MMİO* yöntemi kısa sürelerde düşük ihlal oranı sahip çözümler verdiği söylenebilir. Ancak ihlal sayısı yüksektir. *UOMD* ve *SD* 'den ise çok hızlı çözümler elde edilmesine rağmen, ihlal miktarları oldukça yüksektir. *SD* için en az bir rota ayrıtına kendi bireysel maliyetinden fazla maliyet atanmıştır. *MTİO* ise ihlal sayısı olarak iyi çözümler vermesine rağmen çözüm süresi oldukça uzundur.

Tablo 7.7: Maliyet Dağıtım Yöntemleri için İhlal Miktarları ve İhlal Oranları-1

	UOMD			SD			MMİO			MTİO		
Örnek	İM	OİO	MİO	İM	OİO	MİO	İM	OİO	MİO	İM	OİO	MİO
1	4,326	4.5	36.8	3,503	4.1	137.6	152	0.0	0.2	9	0.4	2.9
2	24,159	5.4	37.6	13,129	4.6	38.9	534	0.0	0.5	37	0.5	3.2
3	3,474	3.6	30.5	3,052	3.6	25.1	207	0.0	0.2	11	0.6	6.1
4	8,928	5.9	36.4	8,603	6.3	59.3	415	0.0	0.4	56	0.1	1.5
5	6,878	3.9	25.0	5,114	3.1	28.2	249	0.0	0.2	29	0.0	1.1
6	12,166	4.8	34.5	10,295	4.5	36.0	614	0.0	0.3	82	0.1	1.5
7	35,645	4.9	43.1	44,406	5.3	114.0	946	0.0	0.4	40	0.5	5.6
8	41,427	5.1	35.3	38,709	4.4	29.3	974	0.0	0.3	3	5.7	15.0
9	38,048	4.6	30.7	39,769	4.1	211.8	749	0.0	0.2	4	1.5	4.6
10	218,113	5.0	35.1	192,339	4.8	65.2	21,133	0.0	0.7	-	-	-
11	-	-	-	-	-	-	-	-	-	-	-	-
12	168,708	5.1	39.2	203,832	5.3	129.8	4,906	0.0	0.3	-	-	-
13	12,471	6.0	44.8	10,288	6.1	46.8	563	0.2	1.0	61	0.6	5.2
14	20,872	8.4	46.6	14,067	6.7	38.7	653	0.2	1.2	162	0.4	5.0
15	2,407	3.3	18.9	2,117	3.5	26.0	253	0.0	0.2	22	0.1	1.3
16	10,162	4.1	27.8	10,541	4.6	40.6	684	0.0	0.3	156	0.0	1.1
17	10,017	7.4	40.9	11,925	7.0	78.4	1,217	0.1	1.1	194	0.5	8.4
18	34,750	6.6	40.7	33,120	6.8	45.7	1,143	0.0	0.5	177	0.0	2.0
19	59,599	5.0	34.2	47,352	4.6	50.3	1,374	0.0	0.4	80	0.4	9.8
20	392,653	4.8	38.7	240,033	3.5	60.2	7,055	0.0	0.4	666	0.1	2.1
21	198,421	6.2	42.4	156,812	5.1	51.8	9,954	0.1	1.0	164	0.7	12.8
22	69,346	5.6	48.5	76,795	5.5	75.7	3,692	0.0	0.5	334	0.2	5.6
23	56,963	5.4	46.4	58,348	5.5	90.8	2,314	0.0	0.5	504	0.1	2.2
24	169,661	6.9	51.0	136,279	6.0	153.9	7,212	0.0	0.8	739	0.2	8.6

Tablo 7.8: Maliyet Dağıtım Yöntemleri için İhlal Miktarları ve İhlal Oranları-2

	s-Nükleolus			%s-Nükleolus			q-Nükleolus			%q-Nükleolus		
Örnek	İM	OİO	MİO	İM	OİO	MİO	İM	OİO	MİO	İM	OİO	MİO
1	65	0.0	0.0	65	0.0	0.0	61	0.0	0.0	61	0.0	0.0
2	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0
3	57	0.0	0.0	57	0.0	0.0	54	0.0	0.0	54	0.0	0.0
4	52	0.0	0.0	52	0.0	0.0	49	0.0	0.0	49	0.0	0.0
5	83	0.0	0.0	77	0.0	0.0	82	0.0	0.0	76	0.0	0.0
6	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0
7	201	0.0	0.1	261	0.0	0.0	255	0.0	0.0	208	0.0	0.0
8	214	0.0	0.0	218	0.0	0.0	280	0.0	0.0	211	0.0	0.0
9	210	0.0	0.0	205	0.0	0.0	272	0.0	0.0	253	0.0	0.0
10	375	0.0	0.0	496	0.0	0.0	364	0.0	0.0	498	0.0	0.0
11	-	-	-	-	-	-	-	-	-	-	-	-
12	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0
13	64	0.0	0.0	102	0.0	0.0	63	0.0	0.0	62	0.0	0.0
14	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0
15	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0
16	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0
17	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0
18	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0
19	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0	0	0.0	0.0
20	554	0.0	0.0	790	0.0	0.0	428	0.0	0.0	722	0.0	0.0
21	278	0.0	0.0	265	0.0	0.0	254	0.0	0.0	240	0.0	0.0
22	197	0.0	0.0	193	0.0	0.0	276	0.0	0.0	189	0.0	0.0
23	295	0.0	0.0	296	0.0	0.0	302	0.0	0.0	283	0.0	0.0
24	277	0.0	0.0	262	0.0	0.0	251	0.0	0.0	246	0.0	0.0

Tablo 7.9: Maliyet Dağıtım Yöntemlerinden Rota Ayrıtları için Dolu Gitme Başına Düşen Maliyet ($R(l)$) - 1

	UOMD			SD			MMİO			MTİO		
Örnek	Min	Ort	Mak	Min	Ort	Mak	Min	Ort	Mak	Min	Ort	Mak
1	1.0	1.2	1.8	0.9	1.3	4.3	0.3	1.2	1.7	0.2	1.2	1.7
2	1.0	1.3	1.8	0.9	1.3	2.2	0.3	1.3	1.7	0.3	1.3	1.7
3	1.0	1.2	1.8	0.9	1.2	2.1	0.3	1.2	1.7	0.3	1.2	1.7
4	1.0	1.2	1.8	0.9	1.2	2.9	0.3	1.2	1.7	0.3	1.2	1.7
5	1.0	1.2	1.8	0.9	1.2	2.0	0.3	1.2	1.7	0.3	1.2	1.7
6	1.0	1.2	1.8	0.9	1.2	1.8	0.4	1.1	1.7	0.4	1.1	1.7
7	1.0	1.3	1.8	0.9	1.3	3.9	0.2	1.2	1.7	0.2	1.2	1.7
8	1.0	1.2	1.8	1.0	1.3	2.3	0.2	1.3	1.7	0.2	1.3	1.7
9	1.0	1.2	1.8	0.9	1.3	5.6	0.4	1.2	1.7	0.5	1.2	1.7
10	1.0	1.2	1.8	0.9	1.3	3.0	0.3	1.2	1.7	-	-	-
11	-	-	-	-	-	-	-	-	-	-	-	-
12	1.0	1.2	1.8	0.8	1.2	2.7	0.2	1.2	1.7	-	-	-
13	1.0	1.3	1.8	0.9	1.3	2.1	0.3	1.2	1.7	0.3	1.2	1.7
14	1.0	1.3	1.8	1.0	1.3	2.4	0.3	1.3	1.7	0.3	1.3	1.7
15	1.0	1.2	1.8	1.0	1.2	2.1	0.5	1.2	1.7	0.4	1.2	1.7
16	1.0	1.1	1.8	0.9	1.2	1.9	0.3	1.1	1.7	0.3	1.1	1.7
17	1.0	1.3	1.8	1.0	1.3	3.2	0.2	1.3	1.7	0.2	1.3	1.7
18	1.0	1.2	1.8	0.9	1.2	2.4	0.3	1.2	1.7	0.3	1.2	1.7
19	1.0	1.2	1.8	0.9	1.3	2.3	0.4	1.2	1.7	0.4	1.2	1.7
20	1.0	1.3	1.8	0.9	1.3	2.9	0.4	1.3	1.7	0.3	1.3	1.7
21	1.0	1.3	1.8	0.9	1.3	2.0	0.4	1.3	1.7	0.3	1.3	1.7
22	1.0	1.2	1.8	0.9	1.2	2.5	0.3	1.1	1.7	0.3	1.1	1.7
23	1.0	1.2	1.8	0.9	1.2	2.4	0.2	1.2	1.7	0.2	1.2	1.7
24	1.0	1.2	1.8	0.7	1.3	4.6	0.3	1.2	1.7	0.2	1.2	1.7

Tablo 7.10: Maliyet Dağıtım Yöntemlerinden Rota Ayrıtları için Dolu Gitme Başına Düşen Maliyet ($R(l)$) - 2

	s-Nükleolus			%s-Nükleolus			q-Nükleolus			%q-Nükleolus		
Örnek	Min	Ort	Mak	Min	Ort	Mak	Min	Ort	Mak	Min	Ort	Mak
1	0.2	1.2	1.8	0.2	1.2	1.8	0.3	1.2	1.8	0.2	1.2	1.8
2	0.2	1.3	1.8	0.2	1.3	1.8	0.2	1.3	1.8	0.2	1.3	1.8
3	0.2	1.2	1.8	0.2	1.2	1.8	0.2	1.2	1.8	0.2	1.2	1.8
4	0.3	1.1	1.8	0.3	1.1	1.8	0.3	1.1	1.8	0.3	1.1	1.8
5	0.2	1.2	1.8	0.2	1.2	1.8	0.2	1.2	1.8	0.2	1.2	1.8
6	0.4	1.1	1.8	0.4	1.1	1.8	0.4	1.1	1.8	0.4	1.1	1.8
7	0.2	1.3	1.8	0.2	1.3	1.8	0.2	1.3	1.8	0.2	1.3	1.8
8	0.2	1.3	1.8	0.2	1.3	1.8	0.2	1.3	1.8	0.2	1.3	1.8
9	0.4	1.2	1.8	0.4	1.2	1.8	0.5	1.2	1.8	0.5	1.2	1.8
10	0.2	1.2	1.8	0.2	1.2	1.8	0.3	1.2	1.8	0.3	1.2	1.8
11	-	-	-	-	-	-	-	-	-	-	-	-
12	0.2	1.2	1.8	0.2	1.2	1.8	0.2	1.2	1.8	0.2	1.2	1.8
13	0.2	1.2	1.8	0.2	1.2	1.8	0.2	1.2	1.8	0.2	1.2	1.8
14	0.2	1.3	1.8	0.2	1.3	1.8	0.2	1.3	1.8	0.2	1.3	1.8
15	0.6	1.2	1.8	0.6	1.2	1.8	0.5	1.2	1.8	0.5	1.2	1.8
16	0.2	1.1	1.8	0.2	1.1	1.8	0.2	1.1	1.8	0.2	1.1	1.8
17	0.2	1.3	1.8	0.2	1.3	1.8	0.2	1.3	1.8	0.2	1.3	1.8
18	0.2	1.2	1.8	0.2	1.2	1.8	0.2	1.2	1.8	0.2	1.2	1.8
19	0.3	1.2	1.8	0.3	1.2	1.8	0.3	1.2	1.8	0.3	1.2	1.8
20	0.3	1.3	1.8	0.3	1.3	1.8	0.3	1.3	1.8	0.3	1.3	1.8
21	0.3	1.3	1.8	0.3	1.3	1.8	0.3	1.3	1.8	0.3	1.3	1.8
22	0.3	1.1	1.8	0.3	1.1	1.8	0.3	1.1	1.8	0.3	1.1	1.8
23	0.2	1.2	1.8	0.2	1.2	1.8	0.2	1.2	1.8	0.2	1.2	1.8
24	0.2	1.2	1.8	0.2	1.2	1.8	0.2	1.2	1.8	0.2	1.2	1.8

Tablo 7.11: Maliyet Dağıtım Yöntemlerinden Rota Ayrıtları için Ortalama Yüzde Kazançlar ($Sv(l)$)

Örnek	UOMD	SD	MMİO	MTİO	s-Nükleolus	%s-Nükleolus	q-Nükleolus	%q-Nükleolus
1	31.0	29.6	33.1	33.4	33.3	33.5	33.2	33.4
2	27.0	27.1	28.5	28.9	28.5	28.6	28.5	28.6
3	33.8	33.0	33.5	33.5	33.5	33.6	33.7	33.5
4	34.0	33.4	36.1	36.2	36.4	36.5	36.4	36.5
5	36.1	35.1	35.0	35.1	34.8	34.8	34.8	34.8
6	35.0	34.8	36.6	36.7	36.5	36.5	36.6	36.6
7	30.5	28.5	31.0	30.9	30.6	30.6	30.6	30.6
8	31.5	29.7	29.8	29.9	29.9	29.6	29.7	29.5
9	33.8	30.6	32.6	32.6	32.3	32.4	32.4	32.3
10	32.2	30.8	32.3	-	32.3	32.3	32.3	32.3
11	-	-	-	-	-	-	-	-
12	34.7	32.3	34.2	-	34.1	34.1	34.1	34.1
13	28.7	28.8	32.8	33.2	33.4	33.4	33.4	33.4
14	26.7	27.1	29.1	29.5	29.3	29.3	29.4	29.4
15	34.6	34.1	34.0	34.3	33.5	33.5	33.8	33.8
16	36.7	35.8	37.0	37.1	36.5	36.7	36.9	36.8
17	29.2	27.5	28.6	29.6	28.6	28.6	28.6	28.6
18	34.5	32.6	34.6	34.9	35.2	34.6	35.2	35.2
19	30.9	30.7	31.7	31.8	31.6	31.7	31.6	31.7
20	29.0	27.6	29.2	29.5	29.1	29.1	29.1	29.1
21	26.4	26.5	27.6	27.9	28.3	28.4	28.3	28.3
22	35.4	34.2	36.9	37.1	37.1	37.3	37.1	37.1
23	34.0	32.9	34.8	35.1	35.4	35.3	35.3	35.3
24	31.7	30.0	31.6	31.9	31.8	31.9	31.5	31.4

Tablo 7.12: Maliyet Dağıtım Yöntemleri için Harcanan Süre (sn)

Örnek	UOMD	SD	MMİO	MTİO	s-Nükleolus	%s-Nükleolus	q-Nükleolus	%q-Nükleolus
1	0.1	0.1	3.3	4.2	6.5	11.9	22.4	11.8
2	0.2	0.2	14.6	163.6	9.8	20.0	10.3	14.4
3	0.1	0.1	5.5	13.0	47.2	362.3	438.1	241.7
4	0.1	0.1	3.5	5.1	3.9	21.0	6.3	5.5
5	0.2	0.2	12.0	152.9	70.1	898.5	1,052.0	994.2
6	0.3	0.2	12.7	161.7	11.8	24.5	12.4	21.1
7	0.3	0.2	34.1	350.1	1,365.0	3,130.4	*ZL*	1,446.2
8	0.3	0.3	89.3	299.7	4,585.4	1,858.4	*ZL*	1,445.3
9	0.5	0.3	138.1	853.6	1,375.1	5,117.9	*ZL*	2,502.0
10	1.1	1.1	266.6	-	*ZL*	*ZL*	*ZL*	*ZL*
11	-	-	-	-	-	-	-	-
12	1.4	1.4	231.1	-	220.5	240.1	238.3	422.9
13	0.1	0.1	6.1	98.5	169.2	121.4	531.9	131.4
14	0.2	0.1	5.2	12.4	3.2	5.5	3.2	3.7
15	0.1	0.1	2.8	4.7	2.4	2.5	2.0	2.6
16	0.1	0.1	9.1	21.5	5.5	7.2	5.6	7.0
17	0.1	0.1	2.6	6.5	2.5	3.4	2.1	3.1
18	0.3	0.3	21.6	384.0	20.0	28.8	22.5	24.2
19	0.3	0.3	45.9	654.4	30.8	63.1	30.8	48.4
20	2.5	4.2	991.3	*ZL*	6,723.9	1,038.1	*ZL*	1,009.3
21	0.5	0.6	78.5	2,716.2	*ZL*	5,824.9	*ZL*	3,236.1
22	0.4	0.3	41.3	1,613.4	5,321.4	3,092.1	*ZL*	1,911.9
23	0.4	0.4	75.5	3,050.4	*ZL*	*ZL*	*ZL*	1,778.9
24	0.5	0.4	70.5	4,482.7	*ZL*	4,227.8	*ZL*	4,340.7

7.6 Önerilen Yaklaşım ile Klasik Yaklaşımın Karşılaştırılması

Bu tablolardan çıkarılacak en önemli sonuç, önerilen yaklaşım ile *kararlı* bir maliyet dağıtımının daha kolay elde edilebileceğidir. Tablolardan da görüldüğü gibi, klasik yaklaşım (iki aşamalı) ile *kararlı* bir maliyet dağıtımını elde etmek birçok örnek için mümkün değildir. *Nükleolus* bazlı yöntemlerle yalnızca dokuz örnek için *kararlı* bir maliyet dağıtımı elde edilebilmiştir. Diğer örnekler için ihlal sayıları ve oranları düşük olmasına rağmen, *kararlılık* kısıtı tam anlamıyla sağlanamamıştır. Ancak tez kapsamında önerilen yaklaşım ile böyle bir şey söz konusu değildir. Önerilen yaklaşım ile *kararlılık* kısıtı hiçbir çevrim için ihlal edilmemiştir. Ayrıca *nükleolus* bazlı yöntemler için, en az bir rota ayrıtına kendi bireysel maliyeti atanmıştır. Böyle bir durumda, rota ayrıtı koalisyona dahil olmasına rağmen herhangi bir tasarruf elde edememiştir. Başka bir ifadeyle, bu rota ayrıtının koalisyona dahil olması ile dahil olmaması durumunda herhangi bir fark yoktur. Bu tarz bir durum, koalisyonun devamlılığını tehdit edecektir. Diğer yandan, önerilen yaklaşım ile koalisyona katılan rota ayrıtlarının belirli bir oranda tasarruf elde edeceği garanti edilmektedir. Bu da kurulan koalisyonun sürdürülebilir olmasına katkıda bulunmaktadır. Tablolarda raporlanmasa da, bu kısıtı *q-Nükleolus* ve *%q-Nükleolus* yöntemlerine eklediğimizde; birçok örnek için zaman limiti altında olurlu çözümler alınamamıştır. Çözüm alınan örnekler için ise, ihlal sayıları ve oranları oldukça fazladır.

Klasik yaklaşımın diğer bir dezavantajı ise problemin iki aşamalı olarak çözülmesi gerekmesidir. *SUKRK* problemini çözdükten sonra, *kararlı* bir maliyet dağıtımı bulmak için başka bir matematiksel modelin çözülmesi gerekmektedir. Bu da probleminin çözüm süresi için fazladan zaman harcanacağı anlamına gelmektedir. Unutulmaması gereken diğer bir nokta ise *SUKRK* probleminin çözümünün, maliyet dağıtım yöntemleri için bağlayıcı olmasıdır. Bu problem için çözüm alına-

maması durumunda; herhangi bir maliyet dağıtımı elde edilmeyecektir. Böyle bir durumda ise, *SUKRK* problemini çözmek için bir çözüm yöntemine ihtiyaç vardır. Ancak bu durumda da çözüm yönteminin optimal çözüm verdiğinden emin olunması gerekir. Aksi durumda, çözdürülen örneğin *çekirdeği* boş küme olmasa bile maliyet dağıtımı *kararlı* bir çözüm vermeyecektir. Dolayısıyla *SUKRK* probleminin optimal olarak çözdürülmesi son derece önemlidir. *MD-SUKRK* problemi için geliştirilen *DFK* yöntemi bu probleme uyarlanarak; *SUKRK* problemi optimal olarak çözdürülebilir. Ancak böyle bir durumda, elde edilen çözüm optimal olsa bile kısıtlı sayıda çevrim üzerinden elde edilmiş olacaktır. Dolayısıyla *nükleolus* bazlı yöntemler, bu kısıtlı sayıda çevrim ile başlayacaktır. Bu durumda üretilmeyen çevrimlere karşılık gelen *kararlılık* kısıtlarının yeniden tanımlanması gerekebilir. Bu nedenle, *SUKRK* için tek başına çözüm yöntemi geliştirmek yeterli değildir. Aynı zamanda, *nükleolus* bazlı yöntemler için de satır türetme bazlı bir yöntemin geliştirilmesi gerekmektedir. Tüm bunlar göz önüne alındığında, önerilen yaklaşımın problemin çözümü için daha akılcı ve faydalı olduğu söylenebilir.

Yukarıda tartışılan sonuçları daha detaylı analiz etmek için çözdürülen örnek sayısı 30 'a çıkartılmıştır. Tablo 7.13 'de 30 örnek için detaylı bilgiler verilmektedir. Bu örnekler bir öncekilere kıyasla daha büyük boyutludur. 150 düğümlük ve 200 rota ayrıtlık örnekler analizlere dahil edilmiştir.

Tablo 7.13: Çözdürülen Örnekler Hakkında Genel Bilgiler-2

Örnek	$\|N\|$	$\|L\|$	CPR
1-3	100	100	0.5
4-6	100	100	0.8
7-9	100	150	0.5
10-12	100	150	0.8
13-15	100	200	0.5
16-18	100	200	0.8
19-21	150	150	0.5
22-24	150	150	0.8
25-27	150	200	0.5
28-30	150	200	0.8

7.7 *Önerme 3* 'ün Sayısal Olarak İspatlanması

Bölüm 3.3 'de verilen *Önerme 3* matematiksel olarak ispat edilmiştir. *Önerme 3* 'e göre, *SUKRK* probleminin *çekirdeğinin* boş küme olmaması için; bu problemin doğrusal gevşetmesinin tam sayılı çözüm vermesi gerekmektedir. Bu önermenin doğruluğunu sayısal olarak da ispat etmek için; 30 örnek kullanılarak *SUKRK* problemi ve doğrusal gevşetmesi (*DG-SUKRK*), *KK* ile optimal olarak çözdürülmüştür. *SUKRK* probleminin optimal çözümünden elde edilen minimum maliyet, *Öncül-Nükleolus* kullanılarak dağıtılmaya çalışılacaktır. Eğer *Öncül-Nükleolus* 'un amaç fonksiyonu değeri sıfır ise, çözdürülen örnek için *çekirdekte* yer alan bir maliyet dağıtımının bulunduğu anlamına gelir.

Tablo 7.14 'de, *SUKRK* ve *DG-SUKRK* problemi için, *KK* ile elde edilen çözümler verilmektedir. Sırasıyla ikinci, üçüncü ve dördüncü sütunda, bu problemlerden elde edilen amaç fonksiyonu değerleri ve arasındaki fark verilmektedir. Çözdürü-

len 30 örneğin 8 tanesinde *DG-SUKRK* tam sayılı çözüm vermiştir. *DG-SUKRK* probleminin tam sayılı çözüm verdiği sekiz örnek için, *Öncül-Nükleolus* 'un amaç fonksiyonu sıfırdır. Dolayısıyla bu örnekler için *çekirdek* boş küme değildir. Diğer örnekler içinse, *çekirdekte* yer alan bir maliyet dağıtımı bulmak mümkün değildir. Tabloda yer alan son sütunda, maliyet dağıtımında kullanılan tasarruf kısıtı ve alt sınır kullanılması durumunda ortaya çıkan ihlal miktarı analiz edilmiştir. Bu ihlal miktarları *MMİO* yönteminden elde edilen maliyet dağıtımı kullanılarak hesaplanmıştır. Maliyet dağıtımları için alt sınır ve yüzde tasarruf garantisi tanımlandığında, *kararlı* bir maliyet dağıtımı elde etmek mümkün değildir. İhlal miktarları düşük olmasına rağmen, bu kısıtları sağlayacak şekilde *kararlı* bir maliyet dağıtımı bulmak mümkün değildir. Başka bir ifadeyle önerilen kısıtlarla iki aşamalı yaklaşımın kullanılması durumunda; *kararlı* bir maliyet dağıtımı bulmak mümkün olmayacaktır.

Tablo 7.14: *SUKRK* Problemi için Kararlı Maliyet Dağıtımı Bulunması

Örnek	SUKRK	DG-SUKRK	Fark	Öncül-Nükleolus	MMİO
1	113,216.6	113,206.4	10.22	0.26	6.15
2*	106,937.8	106,937.8	0.00*	0.00*	14.48
3	107,163.5	107,160.0	3.43	0.10	7.36
4	117,249.9	117,248.7	1.27	0.03	13.25
5	98,631.3	98,626.6	4.67	0.15	5.15
6*	97,201.1	97,201.1	0.00*	0.00*	8.89
7*	177,618.7	177,618.7	0.00*	0.00*	47.39
8	180,574.1	180,562.3	11.85	0.19	54.59
9*	179,099.7	179,099.7	0.00*	0.00*	11.91
10*	184,304.5	184,304.5	0.00*	0.00*	23.95
11	168,826.9	168,820.3	6.61	0.12	35.02
12	166,998.6	166,996.2	2.45	0.05	6.59
13	228,890.3	228,874.7	15.53	0.20	13.56
14	232,172.5	232,165.4	7.16	0.09	10.04
15	212,532.3	212,528.7	3.58	0.05	4.71
16	219,269.2	219,268.8	0.42	0.01	22.80
17	-	-	-	-	-
18*	199,002.3	199,002.3	0.00*	0.00*	10.81
19	172,937.8	172,897.4	40.34	0.71	6.32
20*	162,226.2	162,226.2	0.00*	0.00*	20.40
21	167,708.1	167,698.5	9.57	0.17	8.74
22	169,634.0	169,621.7	12.37	0.22	26.70
23	113,639.7	113,637.6	2.12	0.05	20.99
24	145,207.5	145,193.6	13.91	0.30	8.98
25	207,039.7	207,026.1	13.62	0.21	4.21
26	186,005.2	185,994.9	10.30	0.16	10.82
27	200,760.1	200,737.4	22.70	0.33	9.67
28	245,212.6	245,208.9	3.73	0.05	11.74
29*	215,196.4	215,196.4	0.00*	0.00*	4.58
30	208,262.3	208,258.1	4.13	0.06	7.13

**: Çekirdek Boş Küme Değil*

DG-SUKRK: SUKRK Probleminin Doğrusal Gevşetmesi

7.8 Satır ve Sütun Türetmek için Geliştirilen Sezgisellerin Performansı

Satır ve sütun türetmek için yalnızca tam sayılı modeller kullanmak harcanan zaman açısından etkin olmayabilir. Bu nedenle, çözüm yönteminin etkinliğini arttırmak için dört farklı sezgisel yöntem önerilmiştir. Bu sezgisellerin geliştirilmesindeki amaç, tam sayılı modellere duyulan ihtiyacı azaltmaktır. Bu sezgisellerin performansını analiz etmek için, *MD-SUKRK* problemi hem sezgisellerle hem de sezgiseller olmadan çözdürülmüştür. Böylelikle, sezgisellerin satır ve sütun türetmek için harcanan zamanı; ne kadar kısalttıkları analiz edilecektir.

Tablo 7.15 'de satır ve sütun türetmede kullanılan sezgisellerin ve tam sayılı modellerin harcadıkları zamanlar verilmektedir. Burada, harcanan zamandan kast edilen; problemi *DFK* ile çözene kadar geçen süredir. Bu analizi yapmak için, ilk olarak bu sezgiseller çıkarılarak ve yalnızca tam sayılı modeller (*fiyatlandırma problemi* ve *satır türetme alt problemi*) kullanılarak satır ve sütun türetme için harcanan zaman hesaplanmıştır. Tabloda verilen süreler saniye cinsindendir. Tam sayılı problemler kullanılarak harcanan zaman tablonun ikinci sütununda raporlanmaktadır. Üçüncü sütunda ise sezgiseller kullanılarak harcanan zamanlar verilmektedir. Son sütunda ise bu sezgisellerin satır ve sütun türetmek için harcanan zamanları yüzde olarak ne kadar iyileştirdiği gösterilmektedir. 30 örneğin tamamı için sezgiseller harcanan zamanı azaltmıştır. En düşük iyileşme oranı 14.örnekte sağlanmıştır. Bu örnek için sezgiseller %4.2 oranında bir iyileşme göstermiştir. Ortalama olarak bakıldığında, %20.6 olarak bir iyileşme sağlanmıştır. Bu tabloya göre, satır ve sütun türetmek için geliştirilen sezgisellerin iyi çözümler verdiği ve tam sayılı modellere duyulan ihtiyacı azalttıkları söylenebilir.

Tablo 7.15: Satır ve Sütun Türetme Sezgisellerinin Performansı

Örnek	STA+FP	SSTS+STA+FP	%İyileşme
1	162.7	106.1	-34.8
2	182.3	156.8	-14.0
3	172.0	141.6	-17.7
4	184.0	167.4	-9.0
5	245.2	186.4	-24.0
6	297.9	242.4	-18.6
7	588.5	562.4	-4.4
8	522.2	419.9	-19.6
9	325.5	274.3	-15.7
10	375.1	291.7	-22.2
11	504.0	425.6	-15.6
12	959.5	386.3	-59.7
13	780.3	688.6	-11.8
14	593.3	568.2	-4.2
15	868.0	679.4	-21.7
16	882.8	802.7	-9.1
17	1,169.2	1,018.8	-12.9
18	1,081.5	1,016.8	-6.0
19	809.5	551.2	-31.9
20	1,916.3	1,297.4	-32.3
21	1,189.7	895.9	-24.7
22	1,249.9	833.1	-33.3
23	1,255.4	1,117.7	-11.0
24	1,228.1	1,064.3	-13.3
25	1,626.7	1,335.7	-17.9
26	2,141.7	1,634.3	-23.7
27	2,726.5	1,810.5	-33.6
28	1,915.7	1,465.6	-23.5
29	1,877.8	1,386.0	-26.2
30	2,057.5	1,534.5	-25.4
Ortalama	996.3	768.7	-20.6

FP : Fiyatlandırma Problemi

STA : Satır Türetme Alt Problemi

SSTS : Satır ve Sütun Türetme Sezgiselleri

7.9 Maliyet Dağıtımında Kullanılan Parametrelerin Çözüme Olan Etkisi

MD-SUKRK probleminde, maliyet dağıtımında kullanılan alt sınırın (λ_l) ve minimum yüzde tasarruf garantisinin (θ_l) çözüme olan etkisi *DFK* ile test edilmiştir. Burada amaç, bu parametrelerin çözüme olan etkisini analiz etmektir. Bu parametreler, modeli kısıtladıkları için amaç fonksiyonu değerini ve kapsanan rota ayrıtı sayısını etkileyebilmektedir. Bu parametreler arttırıldığında, ilgili koşulları sağlayabilmek için; daha fazla sayıda rota ayrıtı koalisyon dışında bırakılacaktır. Bu nedenle, amaç fonksiyonu değerinin artması beklenmektedir.

Tablo 7.16 'da maliyet dağıtımı için kullanılan farklı alt sınırlar (λ_l) için, *DFK* ile elde edilen çözümler verilmektedir. Bu çözümler elde edilirken garanti edilen yüzde tasarruf miktarı %5 olarak alınmıştır. Dört farklı alt sınır değeri için elde edilen *LCP Gap* ve kapsanan rota ayrıtı oranı ($|\hat{L}|/|L|$) verilmiştir. Beklenildiği gibi alt sınır değeri arttıkça *LCP Gap* değeri artmakta; kapsanan rota ayrıtı sayısı ise düşmektedir. Bunun temel sebebi, bu değer arttığında ilgili kısıtın daha sıkı olmasıdır.

Kısıtın sıkılaşmasından dolayı, bu kısıtı sağlayan çözümleri bulmak güçleşmektedir. Çözüm yöntemi bu kısıtı sağlayabilmek için daha fazla sayıda rota ayrıtını koalisyon dışında bırakmaktadır. Bu nedenle *LCP Gap* artmış, kapsanan rota ayrıtı sayısı da düşmüştür.

Alt sınırı kaldırdığımızda yani, ilgili değeri sıfıra eşitlediğimizde; beklenenin aksine *LCP Gap* artmış, kapsanan rota ayrıtı oranı ise düşmüştür. Bu durumun tam aksinin beklenmesine rağmen yaşanması; alternatif çözümlerin varlığından kaynaklanmış olabilir. Alternatif çözümlerden dolayı satır ve sütun türetme aşamasında optimal çözüme yakınsama hızı düşmüş olabilir. Bu nedenle zaman limitinin de etkisiyle, beklenenin tam aksi durum yaşanmış olabilir. Ancak zaman limiti

kaldırıldığında, en kötü durumda; 0.2 ile elde edilen çözümlerin alınması beklenir.

Tablo 7.17 'de maliyet dağıtımı için kullanılan farklı minimum yüzde tasarruf garantileri (θ_l) için, *DFK* ile elde edilen çözümler verilmektedir. Bu çözümler elde edilirken alt sınır değeri 0.2 olarak alınmıştır. Dört farklı tasarruf garantisi değeri için elde edilen *LCP Gap* ve kapsanan rota ayrıtı oranı ($|\hat{L}|/|L|$) tabloda verilmiştir. Beklenildiği gibi garanti verilen minimum yüzde tasarruf değeri arttığında, *LCP Gap* artmış; kapsanan rota ayrıtı sayısı ise azalmıştır. Bunun temel sebebi, yüzde tasarruf garantisi arttıkça ilgili kısıtı sağlayan çözümleri bulabilmek için çözüm yöntemi daha fazla sayıda rota ayrıtını koalisyon dışında bırakmasıdır. Minimum yüzde tasarruf garantisini kaldırıldığında yani ilgili parametre sıfıra eşitlendiğinde, 14 örnek için *DFK* yöntemi optimal çözüm vermiştir. Geri kalan örnekler ise, *DFK* zaman limitine takılarak çözüm vermiştir. Ayrıca ortalamada rota ayrıtlarının %83 kapsanabilmiştir.

Tablo 7.16: Maliyet Dağıtımı için Kullanılan Alt Sınırın (λ_l) Çözüme Etkisi

	$\lambda_l = 0.0$		$\lambda_l = 0.2$		$\lambda_l = 0.5$		$\lambda_l = 1.0$	
Örnek	LCP Gap	$\|\hat{L}\|/\|L\|$	LCP Gap	$\|\hat{L}\|/\|L\|$	LCP Gap	$\|\hat{L}\|/\|L\|$	LCP Gap	$\|\hat{L}\|/\|L\|$
1	3.0	0.83	2.8	0.84	3.9	0.78	16.2	0.56
2	1.3	0.83	1.1	0.84	6.6	0.58	12.8	0.46
3	2.5	0.95	2.5	0.95	7.8	0.74	16.4	0.63
4	1.4	0.83	1.4	0.83	1.1	0.84	12.2	0.57
5	3.0	0.80	2.8	0.79	3.0	0.80	8.6	0.73
6	9.7	0.69	9.7	0.69	1.8	0.85	17.7	0.55
7	2.3	0.68	3.7	0.61	2.5	0.67	11.3	0.46
8	2.8	0.69	2.8	0.69	2.6	0.71	9.3	0.52
9	14.7	0.51	6.5	0.63	5.7	0.65	12.6	0.58
10	4.8	0.69	7.5	0.62	11.4	0.55	13.4	0.56
11	5.8	0.69	5.1	0.63	4.3	0.55	11.0	0.51
12	1.2	0.87	1.7	0.85	3.2	0.79	14.1	0.63
13	1.8	0.79	1.8	0.79	19.1	0.40	18.4	0.47
14	2.2	0.81	2.3	0.80	4.9	0.70	15.6	0.48
15	3.0	0.77	3.0	0.77	3.7	0.74	17.5	0.48
16	5.6	0.68	18.4	0.43	7.8	0.64	18.8	0.45
17	14.3	0.46	11.0	0.52	12.6	0.48	17.8	0.40
18	7.7	0.65	8.2	0.64	27.9	0.36	18.6	0.51
19	31.6	0.34	29.9	0.33	25.1	0.44	17.2	0.59
20	4.0	0.75	4.0	0.75	5.4	0.73	16.8	0.51
21	6.8	0.69	1.5	0.91	2.8	0.81	13.9	0.61
22	1.0	0.81	5.1	0.69	10.6	0.57	11.3	0.55
23	5.9	0.65	5.9	0.65	5.7	0.66	16.3	0.51
24	1.4	0.84	1.4	0.84	7.3	0.68	16.0	0.57
25	1.9	0.89	1.1	0.92	39.5	0.24	30.4	0.39
26	37.7	0.20	34.9	0.23	32.0	0.27	24.5	0.40
27	37.6	0.21	38.7	0.19	37.1	0.21	26.2	0.39
28	6.3	0.75	3.6	0.83	40.7	0.26	23.0	0.50
29	42.8	0.22	8.1	0.69	43.8	0.22	30.8	0.39
30	41.1	0.23	39.6	0.26	41.0	0.23	33.4	0.35
Ortalama	10.2	0.66	8.9	0.67	14.0	0.57	17.4	0.51

$\theta_l = \%5$

Tablo 7.17: Maliyet Dağıtımı için Kullanılan Tasarruf Garantisinin (θ_l) Çözüme Etkisi

	$\theta_l = \%0$		$\theta_l = \%5$		$\theta_l = \%10$		$\theta_l = \%20$	
Örnek	LCP Gap	$\lvert\hat{L}\rvert/\lvert L\rvert$	LCP Gap	$\lvert\hat{L}\rvert/\lvert L\rvert$	LCP Gap	$\lvert\hat{L}\rvert/\lvert L\rvert$	LCP Gap	$\lvert\hat{L}\rvert/\lvert L\rvert$
1	1.0*	0.95*	2.8	0.84	2.4	0.84	13.0	0.60
2	0.4*	0.96*	1.1	0.84	6.8	0.57	10.4	0.51
3	2.4*	0.96*	2.5	0.95	4.2	0.84	9.4	0.74
4	0.4*	0.96*	1.4	0.83	2.7	0.76	8.4	0.63
5	0.1*	0.97*	2.8	0.79	1.6	0.84	4.7	0.74
6	0.1*	0.98*	9.7	0.69	1.0	0.87	6.9	0.74
7	0.6*	0.96*	3.7	0.61	8.1	0.47	9.6	0.50
8	1.1*	0.95*	2.8	0.69	5.5	0.59	7.5	0.54
9	0.3*	0.98*	6.5	0.63	5.7	0.65	5.1	0.67
10	0.7*	0.97*	7.5	0.62	11.8	0.55	16.6	0.46
11	0.3	0.89	5.1	0.63	2.9	0.69	8.7	0.56
12	0.3	0.99	1.7	0.85	3.2	0.79	8.3	0.69
13	0.8	0.94	1.8	0.79	10.5	0.54	16.8	0.43
14	2.9	0.78	2.3	0.80	6.6	0.65	9.2	0.62
15	0.4	0.95	3.0	0.77	9.9	0.61	18.3	0.47
16	0.6	0.95	18.4	0.43	9.4	0.60	13.8	0.50
17	0.0*	0.98*	11.0	0.52	22.1	0.33	24.1	0.30
18	0.0*	0.99*	8.2	0.64	30.0	0.34	14.0	0.52
19	0.7	0.93	29.9	0.33	3.7	0.79	28.9	0.34
20	1.7	0.97	4.0	0.75	4.0	0.76	6.6	0.67
21	2.3	0.83	1.5	0.91	2.1	0.86	7.4	0.69
22	18.9	0.46	5.1	0.69	15.2	0.51	6.0	0.67
23	0.0*	0.99*	5.9	0.65	4.9	0.66	24.0	0.34
24	38.2	0.24	1.4	0.84	2.2	0.81	9.9	0.64
25	40.7	0.22	1.1	0.92	1.1	0.92	41.5	0.21
26	37.3	0.19	34.9	0.23	34.9	0.23	35.5	0.22
27	38.6	0.18	38.7	0.19	37.2	0.21	36.6	0.21
28	1.7	0.97	3.6	0.83	7.0	0.73	38.7	0.30
29	0.2*	0.99*	8.1	0.69	5.8	0.73	8.1	0.70
30	1.4	0.86	39.6	0.26	38.5	0.27	7.1	0.73
Ortalama	6.5	0.83	8.9	0.67	10.0	0.63	15.2	0.53

$\lambda_l = 0.2$

*: *Optimal Çözüm*

7.10 Koalisyon Sayılarına Göre *MD-SUKRK* Probleminden Elde Edilen Çözümler

Yukarıdaki sonuçlardan da görüldüğü gibi, bazı örnekler için kapsanan rota ayrıtı sayısı oldukça düşüktür. Maliyet dağıtımları üzerine, alt sınır ve yüzde tasarruf garantisi kısıtları konulduğunda; bütün rota ayrıtlarını kapsayan *kararlı* bir koalisyon oluşturmak mümkün değildir. Bu nedenle, koalisyon oluşturulduktan sonra; koalisyon dışında kalan rota ayrıtları için *DFK* yöntemiyle yeni bir koalisyon daha oluşturulmuştur. Bu şekilde toplam 3 koalisyon oluşturularak analizler yapılmıştır.

İlk olarak bütün rota ayrıtları üzerinden *DFK* yöntemi çözdürülerek ilk koalisyon kurulmuştur. Dışarıda kalanlar için tekrar *DFK* yöntemi kullanılarak ikinci bir koalisyon oluşturulmuştur. Son olarak ilk iki koalisyon içerisinde yer almayan rota ayrıtları üzerinden *DFK* yöntemi kullanılmıştır. Böylelikle toplam üç farklı koalisyon oluşturulmuştur. Burada amaç, koalisyon sayısı artması durumunda; çözüm süresinin ve kapsanan rota ayrıtı oranının nasıl değişeceğini gözlemlemektir. Ayrıca, çözdürülen 30 örnekli setin boyutu; 24 'lü sete kıyasla daha büyük olduğu için; *DFK* yönteminin performansı da analiz edilecektir.

Tablo 7.18 'de *MD-SUKRK* problemi için oluşturulan koalisyonlardan elde edilen *LCP Gap* ve çözüm süreleri verilmektedir. Üç koalisyon için elde edilen çözümler *DFK* yönteminden elde edilmiştir. Bütün rota ayrıtları üzerinden çözülen *1.Koalisyonda* hiçbir örnek için optimal çözüm elde edilememiştir. *DFK* yöntemi, tıpkı 24 örnekte olduğu gibi bu örnek seti için de *KK* 'ya kıyasla daha iyi çözümler vermiştir. *1.Koalisyon* çözüldükten sonra; koalisyon dışında kalan rota ayrıtları üzerinden *2.Koalisyon* oluşturulmuştur. Bu koalisyonda 10 örnek için optimal çözüm elde edilebilmiştir. Beklenildiği gibi *LCP Gap* değerleri *1.Koalisyona* kıyasla oldukça düşüktür. Son olarak, *1. ve 2.Koalisyon* dışında kalan rota ayrıtları üzerinden *3.Koalisyon* çözülmüştür. Bu koalisyonda 21 örnek için optimal çözüm elde

edilebilmiştir.

Tablo 7.19 'de *MD-SUKRK* problemi için oluşturulan koalisyonlar için kapsanan rota ayrıtı sayısı ve oranı, seçilen çevrim sayısı ve üretilen çevrim sayıları verilmektedir. Beklenildiği gibi koalisyon sayısı artıkça, toplamda kapsanan rota ayrıtı sayısı artmaktadır. Çözdürülen 3 örnek (3, 4 ve 21) için tek bir koalisyon oluşturulması yeterlidir. Çözdürülen 12 örnek için ise sadece iki koalisyon oluşturmak yeterlidir. Bunun temel sebebi eldeki rota ayrıtlarıyla *kararlı* bir maliyet dağıtımının oluşturulmasının mümkün olmamasıdır. Her bir koalisyonda verilen rota ayrıtı kapsanma oranı ($|\hat{L}|/|L|$), toplam oranı göstermektedir. Tek bir koalisyonla minimum %34 rota ayrıtı kapsayabiliyorken, bu oran iki koalisyon için %66, üç koalisyon için %76 'tır. *1.Koalisyon* için *DFK* 'den elde edilen çözümler oldukça iyidir. Az sayıda çevrim üreterek, çok sayıda rota ayrıtını kapsayabilen çevrimler bulunabilmiştir. Diğer koalisyon yapıları için ise, problemin boyutu küçüldüğü için optimal çözümü kısa sürede ve az sayıda çevrim üreterek bulabilmiştir.

Tablo 7.18: *MD-SUKRK* Problemi için Oluşturulan Koalisyonlardan Elde Edilen Çözümler-1

	KK		1.Koalisyon		2.Koalisyon		3.Koalisyon	
Örnek	LCP Gap	ÇS	LCP Gap	ÇS	LCP Gap	ÇS	LCP Gap	ÇS
1	1.0*	89.6*	3.1	*ZL*	2.3*	169.9*	3.5*	1.2*
2	40.0	*ZL*	11.8	*ZL*	3.9	*ZL*	2.7*	3,253.7*
3	4.3	*ZL*	2.5	*ZL*	0.4*	1.2*	0.4*	1.2*
4	33.4	*ZL*	1.4	*ZL*	2.6*	3.2*	2.6*	3.2*
5	60.2	*ZL*	1.6	*ZL*	0.4*	731.5*	0.5*	1.2*
6	55.8	*ZL*	15.2	*ZL*	0.2*	4,090.9*	0.0*	1.2*
7	36.4	*ZL*	2.3	*ZL*	1.4	*ZL*	1.3*	147.3*
8	38.6	*ZL*	2.8	*ZL*	2.3	*ZL*	2.7*	2.2*
9	45.8	*ZL*	6.6	*ZL*	3.4	*ZL*	1.3*	6,000.6*
10	49.1	*ZL*	8.1	*ZL*	1.8	*ZL*	2.2*	1.3*
11	48.5	*ZL*	10.1	*ZL*	4.0	*ZL*	1.9	*ZL*
12	59.8	*ZL*	1.2	*ZL*	1.9*	607.3*	2.2*	2.2*
13	48.9	*ZL*	1.8	*ZL*	1.1	*ZL*	1.4*	3.8*
14	50.6	*ZL*	3.2	*ZL*	5.9	*ZL*	9.0*	2.7*
15	55.1	*ZL*	4.3	*ZL*	7.2	*ZL*	3.1	*ZL*
16	48.8	*ZL*	5.9	*ZL*	6.9	*ZL*	5.9	*ZL*
17	-	-	28.2	*ZL*	6.0	*ZL*	2.5	*ZL*
18	55.2	*ZL*	7.5	*ZL*	2.6	*ZL*	1.5*	6,384.4*
19	50.5	*ZL*	31.6	*ZL*	2.6	*ZL*	3.0*	146.6*
20	48.2	*ZL*	4.0	*ZL*	2.0	*ZL*	1.6*	202.1*
21	54.8	*ZL*	1.2	*ZL*	0.4*	2.9*	0.4*	3.1*
22	50.7	*ZL*	1.0	*ZL*	0.7*	2,554.1*	0.6*	5.9*
23	-	-	6.1	*ZL*	3.4	*ZL*	2.1	*ZL*
24	57.3	*ZL*	1.4	*ZL*	0.9*	1,068.9*	0.8*	12.9*
25	55.0	*ZL*	1.1	*ZL*	1.7*	432.2*	0.8*	3.1*
26	48.9	*ZL*	37.6	*ZL*	7.3	*ZL*	1.3	*ZL*
27	51.8	*ZL*	37.9	*ZL*	11.2	*ZL*	9.2	*ZL*
28	62.2	*ZL*	4.1	*ZL*	10.6	*ZL*	8.1*	4.3*
29	60.8	*ZL*	44.8	*ZL*	18.4	*ZL*	5.4	*ZL*
30	58.5	*ZL*	38.5	*ZL*	14.8	*ZL*	5.8	*ZL*

**: Optimal Çözüm*

ÇS: Çözüm Süresi (saniye)

Tablo 7.19: *MD-SUKRK* Problemi için Oluşturulan Koalisyonlardan Elde Edilen Çözümler-2

	1.Koalisyon				2.Koalisyon				3.Koalisyon			
Örnek	$\|\hat{L}\|$	$\|\hat{L}\|/\|L\|$	$\|\hat{C}(L)\|$	$\|C(L)\|$	$\|\hat{L}\|$	$\|\hat{L}\|/\|L\|$	$\|\hat{C}(L)\|$	$\|C(L)\|$	$\|\hat{L}\|$	$\|\hat{L}\|/\|L\|$	$\|\hat{C}(L)\|$	$\|C(L)\|$
1	82	0.82	30	3,779	7*	0.89*	3*	124*	0*	0.89*	0*	44*
2	47	0.47	17	4,730	31	0.78	13	1,261	4*	0.82*	2*	193*
3	95	0.95	32	3,977	0*	0.95*	0*	10*	0*	0.95*	0*	10*
4	83	0.83	29	3,328	0*	0.83*	0*	113*	0*	0.83*	0*	113*
5	83	0.83	27	4,549	9*	0.92*	3*	230*	0*	0.92*	0*	34*
6	59	0.59	18	5,235	35*	0.94*	14*	988*	0*	0.94*	0*	20*
7	102	0.68	34	8,312	10	0.75	4	1,064	2*	0.76*	1*	484*
8	103	0.69	36	8,012	12	0.77	5	740	0*	0.77*	0*	272*
9	94	0.63	37	6,471	25	0.79	10	1,399	5*	0.83*	2*	617*
10	90	0.60	34	6,917	45	0.90	18	1,289	0*	0.90*	0*	49*
11	80	0.53	29	8,429	26	0.71	10	2,126	13	0.79	5	740
12	130	0.87	48	6,550	2*	0.88*	1*	162*	0*	0.88*	0*	129*
13	158	0.79	60	12,071	10	0.84	4	931	0*	0.84*	0*	454*
14	151	0.76	58	11,786	23	0.87	9	1,198	0*	0.87*	0*	297*
15	143	0.72	53	13,613	17	0.80	5	1,736	12	0.86	5	800
16	132	0.66	45	14,532	18	0.75	7	1,931	14	0.82	6	1,043
17	45	0.23	16	20,519	86	0.66	26	12,738	23	0.77	8	2,738
18	132	0.66	47	16,388	32	0.82	12	2,575	6*	0.85*	2*	699*
19	51	0.34	19	6,658	82	0.89	30	3,082	4*	0.91*	2*	99*
20	113	0.75	37	7,857	10	0.82	4	536	4*	0.85*	2*	238*
21	139	0.93	48	6,636	0*	0.93*	0*	47*	0*	0.93*	0*	47*
22	121	0.81	43	7,207	3*	0.83*	1*	405*	0*	0.83*	0*	214*
23	94	0.63	28	12,026	19	0.75	7	2,071	7	0.80	3	1,040
24	126	0.84	41	9,735	4*	0.87*	1*	348*	0*	0.87*	0*	202*
25	183	0.92	61	15,242	4*	0.94*	2*	139*	0*	0.94*	0*	77*
26	38	0.19	15	17,269	98	0.68	32	11,375	36	0.86	13	1,982
27	39	0.20	14	16,190	102	0.71	32	10,944	29	0.85	11	1,617
28	160	0.80	64	7,337	16	0.88	7	565	0*	0.88*	0*	169*
29	43	0.22	14	12,556	77	0.60	28	8,303	47	0.84	18	2,758
30	53	0.27	18	13,909	83	0.68	28	8,188	32	0.84	14	1,702

**: Optimal Çözüm*

8. SONUÇ ve ÖNERİLER

Bu tez kapsamında, tam kamyon yükü gönderici iş birliğinde *kararlı* koalisyonun bulunması için *kararlı* bir maliyet dağıtımının elde edilmesi amaçlanmıştır. İşbirlikçilerin seçimi, iş birliğinin en düşük toplam maliyetinin belirlenmesi ve iş birliğinden elde edilen getirinin iş birlikçiler arasında paylaştırılması birlikte değerlendirilerek *Maliyet Dağıtımlı Sayı ve Uzunluk Kısıtlı Rota Kapsama (MD-SUKRK)* problemi tanımlanmıştır. Bu aşamaların ayrı değerlendirilmesi durumunda *çekirdekte* yer alan bir maliyet dağıtımının bulunması için gerek ve yeter koşullar tanımlanmıştır. Bu gerek ve yeter koşullar kullanılarak *kararlı* bir maliyet dağıtımının bulunmasının güç olduğu gösterilmiştir.

MD-SUKRK problemi için geliştirilen bütün matematiksel modeller ve çözüm yöntemleri ve *SUKRK* problemi için geliştirilen maliyet dağıtım yöntemleri *Java* programlama diliyle kodlanarak; *Cplex OPL 12.9* çözdürücüsüyle çözdürülmüştür. Geliştirilen modeller, çözüm yöntemleri ve maliyet dağıtım yöntemleri; belirli kurallara göre rastgele üretilen örnekler kullanılarak test edilmiştir.

MD-SUKRK problemi ilk olarak optimal olarak çözdürülmeye çalışılmıştır. Bunun için bütün olurlu çevrimler zaman kısıtı olmadan üretilmiştir. Üretilen çevrimler üzerinden iki saatlik zaman limiti altında modelin optimal çözümünün (kesin çözümünün) bulunması amaçlanmıştır. Ancak iki saatlik zaman limiti altında bazı örnekler için olurlu çözümler elde edilememiştir. Bu nedenle, dal-fiyat ve kesi (*DFK*) yöntemi geliştirilmiştir. *DFK*, dal-sınır ve satır ve sütun türetme yöntemlerini bir araya getirmektedir. Ayrıca, *DFK* 'nin performansını arttırmak için; olurlu çözümler veren bir sezgisel yöntem de geliştirilmiştir.

Satır ve sütun türetme yönteminde amaçlanan, bütün olurlu çevrimleri üretmek için harcanan zamanı azaltmak ve bu çevrimlerin sayısını azaltmaktır. Çünkü bu

problemde her çevrime karşılık gelen bir değişken, bir de kısıt bulunmaktadır. Bunun için problemin doğrusal gevşetilmesi kullanılarak dual değişkenler tanımlanmıştır. Bu dual değişkenler üzerinden de fiyatlandırma problemi tanımlanmıştır. Fiyatlandırma probleminin amacı minimum indirgenmiş maliyetli çevrimi bulmaktır. Ayrıca satır ve sütun türetmek için dört farklı sezgisel yöntem geliştirilmiştir. Bu sezgisellerle amaçlanan, fiyatlandırma problemine duyulan ihtiyacı azaltmaktır. Sezgisel yöntemler, sırayla çalıştırılarak farklı arama uzaylarında çözüm aranmıştır. Fiyatlandırma problemi ve sezgisel yöntemler kullanılarak üretilen çevrim sayısı ve üretmek için harcanan zaman ciddi oranda azalmıştır.

Bu yöntemler dışında, *SUKRK* probleminden elde edilen toplam maliyeti dağıtmak için altı farklı maliyet dağıtım yöntemi de geliştirilmiştir. Geliştirilen bütün yöntemler, birbirleriyle ve literatürde yer alan iki yöntemle karşılaştırılmıştır. Maliyet dağıtım yöntemleri; *kararlılık* kısıtı ihlal sayısı, ortalama ve en yüksek oranda ihlal miktarı, rota ayrıtlarının yüzde kazanımları ve dolu gitme başına düşen maliyet kullanılarak karşılaştırılmıştır. Yapılan testler sonucunda, geliştirilen *nükleolus* temelli yöntemlerin; diğer yöntemlere kıyasla daha iyi çözümler verdiği gözlemlenmiştir.

Yapılan analizler ve testler sonucunda, literatürde kullanılan klasik yaklaşımla (iki aşamalı) *kararlı* bir koalisyonun bulunmasının güç olduğu gösterilmiştir. Ayrıca önerilen yaklaşımın kullanılması durumunda *kararlı* bir maliyet dağıtımının bulunabileceği ve bunun sonucunda *kararlı* bir koalisyonun kurulabileceği gösterilmiştir.

Bu çalışma kapsamında çok koalisyonlu yapı için matematiksel model geliştirilse de, tek koalisyonlu yapı dikkate alınmıştır. Ancak yapılan analizler sonucunda, geliştirilen matematiksel modelin; çok koalisyonlu yapı için iyi çözümler vermediği görülmüştür. Bu nedenle, bu tarz problem için de çeşitli çözüm yöntemleri geliş-

tirilebilir. Bunun için akla ilk gelen sezgisel yöntem, ilk olarak tek koalisyonlu olarak problemin çözülmesidir. Bu problem çözüldükten sonra, koalisyon dışında kalan oyuncular üzerinden aynı problem tekrar tekrar çözülebilir. Ancak bu yöntem olurlu bir çözüm verse de optimalden uzak olabilir. Bu nedenle, oyuncuları koalisyonlara bölerek *kararlı* bir yapı oluşturan sezgisel yöntemler denenebilir.

Satır ve sütun türetme yöntemleri için geliştirilen sezgisellerin optimallik garantisi yoktur. Bu nedenle, hem satır hem de sütun türetmek için tam sayılı programlama modeline (*fiyatlandırma problemi* ve *satır türetme alt problem*) ihtiyaç vardır. Bu modelin ihtiyaç duyulması bile, çözüm yöntemi için harcanan zamanı arttıracaktır. Bu nedenle, bu problem *Kaynak Kısıtlı En Kısa Yol (Shortest Path Problem with Resource Constraints)* problemine dönüştürülerek daha kısa zamanda çözülebilir. Bunun için *baskınlık kurallarının (dominance rules)* tanımlanması gerekir. Bu tarz bir yaklaşımın geliştirilmesi durumunda tam sayılı modelin çözülmesine gerek yoktur. Bu nedenle, satır ve sütun türetmek için harcanan zaman ciddi oranda azaltılabilir. Burada dikkat edilmesi gereken nokta, bu yaklaşımının performansının tanımlanacak *baskınlık kurallarına* bağlı olmasıdır. Bu nedenle, bu yaklaşım her zaman fiyatlandırma problemini; tam sayılı programlama modeline kıyasla, daha kısa sürede çözülebileceğinin garantisini vermez.

Geliştirilen *DFK* yöntemi yeterli süre verildiğinde optimal çözüm vermektedir. Ancak çözülen örneklerden de görüldüğü gibi, zaman limiti altında optimal çözüm elde etmek mümkün değildir. Bu nedenle, *DFK* yöntemine alternatif olarak; meta-sezgisel yöntemler (*tabu arama, benzetimli tavlama, genetik algoritma* gibi) de geliştirilebilir. Ancak böyle bir durumda, başlangıçta olurlu çözüm veren bir algoritmaya ve komşuluk tanımlarına ihtiyaç vardır. Bu tarz yöntemlerin en büyük dezavantajı, *kararlı* bir maliyet dağıtımının bulunmasıdır. Bunun için de çeşitli yöntemlerin geliştirilmesi gerekebilir.

MD-SUKRK problemi için geliştirilen yöntemlerden elde edilen maliyet dağıtımı birçok olurlu çözümden yalnızca bir tanesidir. Modelde yer alan kısıtları sağlayan birden fazla maliyet dağıtımı olabilir. Bu durumda hangi maliyet dağıtımının daha iyi olduğunun kararının verilmesi gerekebilir. Bu nedenle, bu maliyet dağıtımları arasından seçim yapan bir matematiksel model geliştirilebilir. Örneğin, modelde yer alan kısıtları (*bireysel rasyonellik, bütçe dengesi* ve *kararlılık*) sağlayacak şekilde oyunculara (rota ayrıtlarına) atan maliyetler arasındaki farkı veya toplam farkı minimize eden bir model geliştirilebilir. Bu tarz modeller doğrusal programlama modeli olacağı için çözüm elde etmek zor olamayacaktır.

SUKRK problemi için geliştirilen maliyet dağıtım yöntemleri, farklı problemler için de test edilebilir. Bunun için, ilgili problemin işbirlikçi oyun olarak tanımlanması gerekir. Daha sonra, problemden elde edilen fayda veya maliyet; geliştirilen maliyet dağıtım yöntemleri kullanılarak oyunculara dağıtılabilir. Böylelikle, geliştirilen yöntemlerin diğer problemler için de kullanılabileceği gösterilebilir.

Geliştirilen *nükleolus* temelli maliyet dağıtım yöntemlerini kullanarak çözüm almak büyük boyutlu örnekler için mümkün olmayabilir. Bu yöntemlerde, her bir olurlu çevrime karşılık bir *kararlılık* kısıtı vardır. Büyük boyutlu bazı örnekler için makul sürelerde bu tarz çevrimleri üretmek mümkün olmayabilir. Bu nedenle, bu durumda *nükleolus* temelli maliyet dağıtım yöntemleri tanımlanamaz. Tıpkı *MD-SUKRK* probleminde olduğu gibi, bu tarz bütün olurlu çevrimlerin üretilemediği örnekler için satır türetme yaklaşımları geliştirilerek *nükleolus* temelli maliyet dağıtım yöntemleri kullanılabilir. Bunun için, bu tez kapsamında tanımlanan sezgisel yöntemler ve matematiksel model kullanılabilir.

KAYNAKLAR

Akyol, D. E. ve De Koster, R. B. (2018). Determining time windows in urban freight transport: A city cooperative approach. *Transportation Research Part E: Logistics and Transportation Review*, 118:34–50.

Anshelevich, E., Dasgupta, A., Kleinberg, J., Tardos, E., Wexler, T., ve Roughgarden, T. (2008). The price of stability for network design with fair cost allocation. *SIAM Journal on Computing*, 38(4):1602–1623.

Archetti, C., Corberan, A., Plana, I., Sanchis, J. M., ve Speranza, M. G. (2016). A branch-and-cut algorithm for the Orienteering Arc Routing Problem. *Computers & Operations Research*, 66:95–104.

Audy, J.-F., D'Amours, S., Lehoux, N., ve Ronnqvist, M. (2010). Coordination in collaborative logistics. In *International workshop on supply chain models for shared resource management. Brussels*, pages 21–22.

Bachrach, Y., Elkind, E., Meir, R., Pasechnik, D., Zuckerman, M., Rothe, J., ve Rosenschein, J. S. (2009). The cost of stability in coalitional games. In *Algorithmic Game Theory*, pages 122–134. Springer. 00066.

Bartolini, E., Cordeau, J.-F., ve Laporte, G. (2013). An Exact Algorithm for the Capacitated Arc Routing Problem with Deadheading Demand. *Operations Research*, 61(2):315–327. 00007.

Berger, S. ve Bierwirth, C. (2010). Solutions to the request reassignment problem in collaborative carrier networks. *Transportation Research Part E: Logistics and Transportation Review*, 46(5):627–638. 00056.

Bertsimas, D., Farias, V. F., ve Trichakis, N. (2011). The price of fairness. *Operations research*, 59(1):17–31.

Bode, C. ve Irnich, S. (2012). Cut-First Branch-and-Price-Second for the Capacitated Arc-Routing Problem. *Operations Research*, 60(5):1167–1182. 00031.

Chabot, T., Bouchard, F., Legault-Michaud, A., Renaud, J., ve Coelho, L. C. (2018). Service level, cost and environmental optimization of collaborative transportation. *Transportation Research Part E: Logistics and Transportation Review*, 110:1–14.

Chabrier, A. (2006). Vehicle Routing Problem with elementary shortest path based column generation. *Computers & Operations Research*, 33(10):2972–2990. 00223.

Chardaire, P. (2001). The core and nucleolus of games: A note on a paper by Gothe-Lundgren et al. *Mathematical programming*, 90(1):147–151.

Chen, B. ve Gurel, S. (2012). Efficiency analysis of load balancing games with and without activation costs. *Journal of Scheduling*, 15(2):157–164.

Chen, Y. J. ve Zhang, J. (2012). Design of price mechanisms for network resource allocation via price of anarchy. *Mathematical programming*, 131(1):333–364.

Christodoulou, G., Chung, C., Ligett, K., Pyrga, E., ve van Stee, R. (2010). On the price of stability for undirected network design. *Approximation and Online Algorithms*, pages 86–97.

Correa, J. R., Schulz, A. S., ve Stier-Moses, N. E. (2007). Fast, fair, and efficient flows in networks. *Operations Research*, 55(2):215–225.

Cruijssen, F., Dullaert, W., ve Fleuren, H. (2007). Horizontal Cooperation in Transport and Logistics: A Literature Review. *Transportation Journal (American Society of Transportation & Logistics Inc)*, 46(3):22–39. 00115.

Cruijssen, F., Dullaert, W., ve Joro, T. (2010). Freight transportation efficiency through horizontal cooperation in Flanders. *International Journal of Logistics Research and Applications*, 13(3):161–178.

Dai, B. ve Chen, H. (2012). Profit allocation mechanisms for carrier collaboration in pickup and delivery service. *Computers & Industrial Engineering*, 62(2):633–643. 00043.

De Vos, B. ve Raa, B. (2018). Stability analysis of cost allocation methods for inventory routing. *IFAC-PapersOnLine*, 51(11):1682–1688.

Defryn, C., Sorensen, K., ve Cornelissens, T. (2016). The selective vehicle routing problem in a collaborative environment. *European Journal of Operational Research*, 250(2):400–411.

Dell'Amico, M., Righini, G., ve Salani, M. (2006). A branch-and-price approach to the vehicle routing problem with simultaneous distribution and collection. *Transportation Science*, 40(2):235–247. 00162.

Desrosiers, J., Gauthier, J. B., ve Lubbecke, M. E. (2013). Row-reduced column generation for degenerate master problems. *European Journal of Operational Research*.

du Merle, O., Villeneuve, D., Desrosiers, J., ve Hansen, P. (1999). Stabilized column generation. *Discrete Mathematics*, 194(3):229–237. 00321.

Ergun, O., Kuyzu, G., ve Savelsbergh, M. (2007a). Reducing truckload transportation costs through collaboration. *Transportation Science*, 41(2):206–221. 00086.

Ergun, O., Kuyzu, G., ve Savelsbergh, M. (2007b). Shipper collaboration. *Computers & Operations Research*, 34(6):1551–1560.

Estevez-Fernandez, A., Borm, P., Meertens, M., ve Reijnierse, H. (2009). On the core of routing games with revenues. *International Journal of Game Theory*, 38(2):291–304. 00007.

Feillet, D., Dejax, P., ve Gendreau, M. (2005). The Profitable Arc Tour Problem: Solution with a Branch-and-Price Algorithm. *Transportation Science*, 39(4):539–552. 00055.

Feldman, M. ve Tamir, T. (2012). Conflicting Congestion Effects in Resource Allocation Games. *Operations Research*, 60(3):529–540.

Frisk, M., Gothe-Lundgren, M., Jornsten, K., ve Ronnqvist, M. (2010). Cost allocation in collaborative forest transportation. *European Journal of Operational Research*, 205(2):448–458.

Gale, D. ve Shapley, L. S. (1962). College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15.

Gothe-Lundgren, M., Jornsten, K., ve Varbrand, P. (1996). On the nucleolus of the basic vehicle routing game. *Mathematical programming*, 72(1):83–100.

Guajardo, M. ve Jornsten, K. (2015). Common mistakes in computing the nucleolus. *European Journal of Operational Research*, 241(3):931–935. 00015.

Guajardo, M. ve Ronnqvist, M. (2015). Operations research models for coalition structure in collaborative logistics. *European Journal of Operational Research*, 240(1):147–159. 00014.

Guajardo, M., Ronnqvist, M., Flisberg, P., ve Frisk, M. (2018). Collaborative transportation with overlapping coalitions. *European Journal of Operational Research*, 271(1):238–249.

Hernandez, F., Feillet, D., Giroudeau, R., ve Naud, O. (2016). Branch-and-price algorithms for the solution of the multi-trip vehicle routing problem with time windows. *European Journal of Operational Research*, 249(2):551–559. 00009.

Ito, T., Kakimura, N., Kamiyama, N., Kobayashi, Y., ve Okamoto, Y. (2017). Efficient stabilization of cooperative matching games. *Theoretical Computer Science*, 677:69–82.

Kimms, A. ve Cetiner, D. (2012). Approximate nucleolus-based revenue sharing in airline alliances. *European Journal of Operational Research*, 220(2):510–521.

Kimms, A. ve Kozeletskyi, I. (2016). Core-based cost allocation in the cooperative traveling salesman problem. *European Journal of Operational Research*, 248(3):910–916. 00002.

Konemann, J., Pashkovich, K., ve Toth, J. (2020). Computing the nucleolus of weighted cooperative matching games in polynomial time. *Mathematical Programming*, pages 1–27.

le Blanc, H., Cruijssen, F., Fleuren, H., ve de Koster, M. (2006). Factory gate pricing: An analysis of the Dutch retail distribution. *European Journal of Operational Research*, 174(3):1950–1967.

Liu, L., Qi, X., ve Xu, Z. (2018). Simultaneous penalization and subsidization for stabilizing grand cooperation. *Operations Research*, 66(5):1362–1375.

Lozano, L., Duque, D., ve Medaglia, A. L. (2016). An Exact Algorithm for the Elementary Shortest Path Problem with Resource Constraints. *Transportation Science*, 50(1):348–357. 00008.

Lu, W. ve Quadrifoglio, L. (2019). Fair cost allocation for ridesharing services modeling mathematical programming and an algorithm to find the nucleolus. *Transportation Research Part B: Methodological*, 121:41–55.

Lubbecke, M. E. ve Desrosiers, J. (2005). Selected topics in column generation. *Operations Research*, 53(6):1007–1023.

Lysgaard, J. ve Wohlk, S. (2014). A branch-and-cut-and-price algorithm for the cumulative capacitated vehicle routing problem. *European Journal of Operational Research*, 236(3):800–810. 00025.

Muter, b., Birbil, . l., ve Bulbul, K. (2013). Simultaneous column-and-row generation for large-scale linear programs with column-dependent-rows. *Mathematical Programming*, 142(1-2):47–82. 00019.

Muter, I., Cordeau, J.-F., ve Laporte, G. (2014). A Branch-and-Price Algorithm for the Multidepot Vehicle Routing Problem with Interdepot Routes. *Transportation Science*, 48(3):425–441. 00023.

Oukil, A., Amor, H. B., Desrosiers, J., ve El Gueddari, H. (2007). Stabilized column generation for highly degenerate multiple-depot vehicle scheduling problems. *Computers & Operations Research*, 34(3):817–834.

Ozbaygin, G., Ekin Karasan, O., Savelsbergh, M., ve Yaman, H. (2017). A branch-and-price algorithm for the vehicle routing problem with roaming delivery locations. *Transportation Research Part B: Methodological*, 100:115–137.

Ozener, O. ve Ergun, O. (2008). Allocating Costs in a Collaborative Transportation Procurement Network. *Transportation Science*, 42(2):146–165.

Ozener, O. O., Ergun, O., ve Savelsbergh, M. (2013). Allocating cost of service to customers in inventory routing. *Operations Research*, 61(1):112–125.

Peng, Z., Shan, W., Jia, P., Yu, B., Jiang, Y., ve Yao, B. (2020). Stable ridesharing matching for the commuters with payment design. *Transportation*, 47(1):1–21.

Rasulkhani, S. ve Chow, J. Y. (2019). Route-cost-assignment with joint user and operator behavior as a many-to-one stable matching assignment game. *Transportation Research Part B: Methodological*, 124:60–81.

Resnick, E., Bachrach, Y., Meir, R., ve Rosenschein, J. S. (2009). The cost of stability in network flow games. In *Mathematical Foundations of Computer Science 2009*, pages 636–650. Springer.

Rousseau, L.-M., Gendreau, M., ve Feillet, D. (2007). Interior point stabilization for column generation. *Operations Research Letters*, 35(5):660–668. 00087.

Schmeidler, D. (1969). The nucleolus of a characteristic function game. *SIAM Journal on applied mathematics*, 17(6):1163–1170.

Shapley, L. S. (1971). Cores of convex games. *International journal of game theory*, 1(1):11–26.

Sun, L., Rangarajan, A., Karwan, M. H., ve Pinto, J. M. (2015). Transportation cost allocation on a fixed route. *Computers & Industrial Engineering*, 83:61–73. 00002.

Vacca, I., Salani, M., ve Bierlaire, M. (2013). An Exact Algorithm for the Integrated Planning of Berth Allocation and Quay Crane Assignment. *Transportation Science*, 47(2):148–161. 00038.

Vanovermeire, C. ve Sorensen, K. (2014). Integration of the cost allocation in the optimization of collaborative bundling. *Transportation Research Part E: Logistics and Transportation Review*, 72:125–143. 00003.

Wang, X., Agatz, N., ve Erera, A. (2018). Stable matching for dynamic ride-sharing systems. *Transportation Science*, 52(4):850–867.

Xu, H., Chen, Z.-L., Rajagopal, S., ve Arunapuram, S. (2003). Solving a practical pickup and delivery problem. *Transportation science*, 37(3):347–364.

Yang, F., Dai, Y., ve Ma, Z.-J. (2020). A cooperative rich vehicle routing problem in the last-mile logistics industry in rural areas. *Transportation Research Part E: Logistics and Transportation Review*, 141:102024.

Zhong, L. ve Bai, Y. (2019). Three-sided stable matching problem with two of them as cooperative partners. *Journal of Combinatorial Optimization*, 37(1):286–292.

TEZDEN TÜRETİLEN YAYINLAR, SUNUMLAR VE PATENTLER:

- Oner, N., Kuyzu, G., 2022. Nucleolus based Cost Allocation Methods for a Class of Constrained Lane Covering Games, *Computers & Industrial Engineering*.
- Oner, N., Kuyzu, G., 2021. Core Stable Coalition Selection in Collaborative Truckload Transportation Procurement, *Transportation Research Part E: Logistics and Transportation Review*, 154:102447.
- Oner, N., Kuyzu, G., 2019. Nucleolus Based Cost Allocation Methods for Collaborative Truckload Transportation Procurement, *INFORMS Annual Meeting*, October 2019, Seattle, Washington, United States.
- Öner, N., Kuyzu, G., 2018. Tam Kamyon Yükü Gönderici İş Birliğinde Kısmi Kararlı Maliyet Dağıtımın Bulunması, *38.Yöneylem Araştırması Endüstri Mühendisliği Ulusal Kongresi (YAEM 2018)*, Haziran 2018, Anadolu Üniversitesi, Eskişehir, Türkiye.
- Oner, N., Kuyzu, G., 2017. Coalition Selection in Collaborative Truckload Transportation Procurement, Computational, *Science and Engineering Student Conference (CSESC),* April 2017, Purdue University, West Lafayette, Indiana, USA.
- Oner, N., Kuyzu, G., 2016. Coalition Selection in Collaborative Truckload Transportation Procurement, *The 29th Conference of the European Chapter on Combinatorial Optimization (ECCO 2016*), May 2016, Eötvös University, Budapest, Hungary.
- Öner, N., Kuyzu, G., 2015. Tam Kamyon Yükü Gönderici İş Birliğinde Güzergah Planlama ve Maliyet Dağıtımının En İyilenmesi, *35.Yöneylem Araştırması Endüstri Mühendisliği Ulusal Kongresi (YAEM 2015)*, Eylül 2015, Orta Doğu Teknik Üniversitesi (ODTÜ), Ankara, Türkiye.