Journal of the Faculty of Engineering and Architecture of Gazi University 34:4 (2019) 2061-2078

Mühendislik Mimarlık Fakültesi Dergisi
Journal of The Faculty of Engineering and Architecture of Gazi University

Elektronik / Online ISSN : 1304 - 4915
Basılı / Printed ISSN : 1300 - 1884

Column generation based matheuristics for a vehicle routing problem with time windows and variable start time

Hande Küçükaydın
Department of Industrial Engineering, MEF University, İstanbul, 34396, Turkey

Highlights:

- A vehicle routing problem with time windows and variable service start time is considered.
- Two column generation based matheuristics, which successfully combine mathematical programming and metaheuristics, are developed.
- For each route obtained by matheuristics, the optimal start time from the depot is computed.

Keywords:

- Vehicle routing problem with time windows
- Route duration
- Column generation
- Matheuristics

Article Info:

Research Article
Received: 08.05.2018
Accepted: 14.05.2019

DOI:

10.17341/gazimmfd.421828

Acknowledgement:

This work was supported by grant no. 116C013 received from TÜBİTAK BİDEB.

Correspondence:

Author: Hande Küçükaydın
e-mail: hande.kucukaydin@mef.edu.tr
phone: +90 212 395 3631

Graphical/Tabular Abstract

The vehicle routing problem with time windows aims to determine the optimal set of routes for a fleet of vehicles serving customers with various demand, where each customer can be visited within certain time windows. In most of the studies, the objective is to minimize the total traveled distance and the departure time from the depot for each vehicle is known. In this paper, however, the vehicles can start serving customers at any desired time and the aim is to minimize the sum of traveling, serving, and waiting times of vehicles. In other words, the optimal departure time from the depot needs to be specified to minimize the route duration of vehicles. In order to solve the problem, two column generation based matheuristics are developed, where the first one utilizes iterated local search and the second one uses variable neighborhood search as metaheuristic. Both methods successfully identify the optimal departure time from the depot for each generated route.

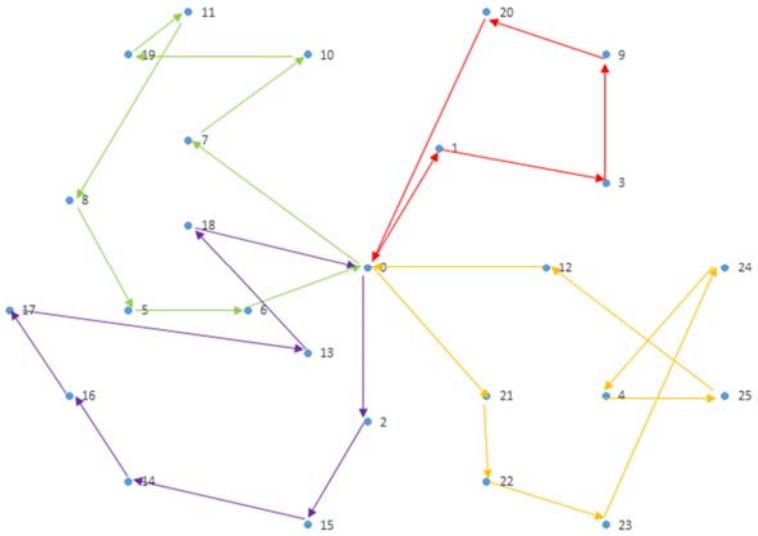

Figure A. Set of vehicle routes obtained on Solomon instance R104

Purpose: The aim of this study is to obtain efficient solution procedures for the vehicle routing problems for which minimization of route durations is of importance.

Theory and Methods: Two column generation based matheuristic algorithms are developed in order to find good feasible solutions for large instances, whereas a depth-first search algorithm is utilized as an exact method for solving small instances.

Results: Both the accuracy and the efficiency of the developed matheuristic algorithms are proven by the computational results.

Conclusion: In this paper, a vehicle routing problem with variable service start time is addressed. Two solution procedures which are able to find good feasible solutions within a small amount of time are obtained. These methods can be made use of when the problem is to be exactly solved by a branch-and-price approach.

Journal of the Faculty of Engineering and Architecture of Gazi University 34:4 (2019) 2061-2078

Mühendislik Mimarlık Fakültesi Dergisi
Journal of The Faculty of Engineering and Architecture of Gazi University
Elektronik / Online ISSN : 1304 - 4915
Basılı / Printed ISSN : 1300 - 1884

Zaman pencereli ve değişken başlama zamanlı bir araç rotalama problemi için sütun türetme temelli mat-sezgiseller

Hande Küçükaydın*
MEF Üniversitesi, Endüstri Mühendisliği Bölümü, İstanbul, 34396, Türkiye

ÖNEÇIKANLAR

- Zaman pencereli ve değişken başlama zamanlı bir araç rotalama problemi ele alındı
- Matematiksel programlama ile meta-sezgiselleri başarılı bir şekilde birleştiren sütun türetme temelli iki mat-sezgisel geliştirildi
- Mat-sezgisellerin elde ettiği her rota için eniyi depodan başlama zamanı hesaplandı

Makale Bilgileri
Araştırma Makalesi
Geliş: 08.05.2018
Kabul: 14.05.2019

DOI:
10.17341/gazimmfd.421828

Anahtar Kelimeler:
Zaman pencereli araç rotalama problemi, rota süresi, sütun türetme, mat-sezgiseller

ÖZET

Bu çalışmada, araçların kullanıldıkları süreye bağlı maliyetlerin oluştuğu ve araçların depodan başlama zamanının bir karar verici tarafından belirlendiği zaman pencereli bir araç rotalama problemi ele alınmaktadır. Problemi çözmek için biri yinelemeli yerel arama meta-sezgiselinden, diğeri değişken komşuluk arama meta-sezgiselinden yararlanan iki sütun türetme temelli mat-sezgisel geliştirilmiştir. Geliştirilen mat-sezgiseller ilk önce literatürden alınarak türetilen küçük bir veri kümesi üzerinde problemin eniyi sonucunu bulan kesin bir yöntem ile karşılaştırılarak kaliteli sonuçlar ürettiklerini kanıtlamışlardır. Yöntemlerin ürettikleri sonuçların doğruluk derecesinden emin olunduktan sonra, daha büyük 87 örnek üzerinde her mat-sezgisel her örnekte 3 kere çalıştırılarak test edilmiştir. Bilgisayımsal sonuçlar değişken komşuluk arama meta-sezgiseli kullanan mat-sezgiselin, daha kaliteli ve verimli sonuçlar vererek daha başarılı bir algoritma olduğunu göstermiştir. Bu sayede kesin bir yöntemle makul bir ana işlemci zamanında çözülemeyen büyük ölçülü problemler için çok kısa bir zaman içerisinde iyi bir olurlu çözüm elde etmek mümkün hale gelmiştir.

Column generation based matheuristics for a vehicle routing problem with time windows and variable start time

HIGHLIGHTS

- A vehicle routing problem with time windows and variable service start time is considered
- Two column generation based matheuristics, which successfully combine mathematical programming and metaheuristics, are developed
- For each route obtained by matheuristics the optimal start time from the depot is computed

Article Info
Research Article
Received: 08.05.2018
Accepted: 14.05.2018

DOI:
10.17341/gazimmfd.421828

Keywords:
Vehicle routing problem with time windows, route duration, column generation, matheuristics

ABSTRACT

In this study, a vehicle routing problem with time windows is investigated, where the costs depend on the total duration of vehicle routes and the starting time from the depot for each vehicle is determined by a decision maker. In order to solve the problem, two column generation based mat-heuristics are developed, where the first one makes use of the iterated local search and the second one uses the variable neighbourhood search. In order to assess the accuracy of the mat-heuristics, they are first compared with an exact algorithm on small instances taken from the literature. Since their performance are quite satisfactory, they are further tested on 87 large instances by running each algorithm 3 times for each instance. The computational results prove that the mat-heuristic using the variable neighbourhood search outperforms the other one. Hence, this enables to obtain a good feasible solution in a very short time when it is not possible to solve large instances with an exact solution method in a reasonable CPU time.

*Sorumlu Yazar/Corresponding Author: hande.kucukaydin@mef.edu.tr / Tel: +90 212 395 3631

1. GİRİŞ (INTRODUCTION)

Araç rotalama problemleri (ARP), belirli bir müşteri kümesine hizmet götürmek isteyen bir araç filosunun ulaşım maliyetlerini enküçükleyen rota kümesini oluşturmaya çalışır [1]. ARP, kolay tanımlanmasına ve modellenebilmesine rağmen kombinatoryal eniyileme alanında en çok tanınan ve çözümü en zorlu problemlerden birisidir. İlk olarak Dantzig ve Ramser [2] tarafından 1959 senesinde "kamyon sevkiyat problemi" olarak tanıtılmış ve o zamandan günümüze pratikte uygulanabilirliği ve öneminden dolayı sığalı ARP, zaman pencereli ARP, topla ve dağıt ARP gibi birçok türde ARP ele alınmıştır. Bu problemleri çözmek içinse birçok sezgisel ve kesin algoritmalar önerilmiştir.

ARP'nin gerçek hayat uygulamalarına atık toplama, kargo ve posta dağıtımı, okul servislerinin rotalarının düzenlenmesi, depozitolu ürünlerin geri toplanması, perakende ürün dağıtımı gibi birçok örnek verilebilir. De Bruecker vd. [3], Belçika'da cam atık toplayan bir şirketin atık toplama rotalarını ve en az maliyet getiren vardiya çizelgelerini belirleyerek atık toplama maliyetlerinden tasarruf sağlayabilmişlerdir. Kargo dağıtımının en güncel örneklerinden bir tanesi Männel ve Bortfeldt [4]'e aittir. Bu çalışmada yazarlar, tektürel bir araç filosunun rotalarını, toplam katedilecek mesafeyi enküçükleyecek şekilde belirlemişler ve kutuların yüklendikten sonra teslimat sırasında tekrar düzenlenmemesi için aynı boyuttaki kutuların aynı yerleştirme planına sahip olmasını sağlamışlardır. Brezilya'da kırsal alanda yaşayan öğrencilerin şehir merkezindeki okullara ulaşımının sağlanabilmesi için De Souza Lima vd. [5], çoktürel, sığalı ve karma yüklü bir araç filosuna sahip çok amaçlı bir okul servisi rotalama problemini dikkate almışlardır. Ele alınan bu problemin çözümüyle beraber yazarlar, sadece toplam rotalama maliyetlerini azaltmakla kalmamış, öğrencilerin toplam ağırlıklandırılmış seyahat sürelerini enküçüklemiş ve sürücülerin çalışma süreleri arasında denge kurmuşlardır. Hosseini vd. [6] ise teşvik usulüyle kullanılmış ürünleri geri toplayan, tektürel bir araç filosuna sahip bir şirket için eşzamanlı olarak toplama merkezlerinin lokasyonunu, araçların rotalarını, önerilecek teşvik miktarını ve talep noktalarından toplanacak kullanılmış ürün adedini bulmayı amaçlayan bir problemi ele almışlardır.

Geleneksel araç rotalama probleminin en klasik ve doğal uzantılarından biri, zaman pencereli deterministik ARP'dir [7, 8]. Zaman pencereli ARP'de, her müşteri kendi belirlediği bir zaman penceresi dâhilinde ziyaret edilebilir. Dolayısıyla bir araç, hizmet vereceği müşteriye zaman penceresinin başlangıcından önce vardığında başlangıca kadar beklemek zorundadır. Buna karşın zaman penceresinin bitişinden sonra müşteriye hizmet vermesi yasaktır. Zaman pencereli ARP'yi ele alan literatürdeki çoğu çalışma, müşterilerine bir araç filosuyla teslimat yapan şirketin amacının tüm araçların katettiği toplam mesafenin enküçüklenmesi olduğunu kabul eder. Bu tarz çalışmaların hepsinde her aracın depodan başlama zamanı kesin ve bellidir. Bu çalışmada ise diğer zaman pencereli ARP'den farklı olarak araçlar istenilen herhangi bir zamanda depodan ayrılarak kendilerine atanan rotada müşterilere hizmet vermeye başlayabilirler. Ele alınan deterministik problemin amacı, araçların seyahat, hizmet ve bekleme sürelerinin toplamını, yani toplam rota süresini enküçükleyen eniyi depodan hareket etme zamanını belirlemektir. Probleme zaman özelliğinin eklenmesi, ele alınan problemin gerçek hayatta karşılaşılan ARP'lere daha fazla yaklaşılmasını sağlar; çünkü örneğin dağıtım şirketleri kendi filoları yetersiz kaldığında müşterilere verilen hizmetin aksamaması için işi taşerona vermek durumunda kalırlar ve böyle durumlarda taşeron işletmelerin araçlarının kullanıldığı süreye bağlı bir masrafa maruz kalırlar. Bu sebepten dolayı da araçların depodan ne zaman yola çıkmaları gerektiğini belirtmek durumundadırlar. Taşeron firmalara ihtiyaç duyulan vakalar dışında da günlük hayatta, özellikle araçların bekleme sürelerinden kaynaklanan maliyetin azaltılmasının önem teşkil ettiği birçok duruma rastlanabilmektedir. Sözgelimi sürücülerin ve dağıtım, montaj gibi hizmetleri veren personelin saat başına aldıkları ücretler, toplam ulaşım maliyetinin kayda değer bölümünü oluşturduğu durumlarda, atıl zamanlarının enküçüklenmesi önem arz edecektir. Bu tarz örneklere montaj veya tamirat yapan teknisyenlerin ulaşıma dâhil olması dışında özellikle evde sağlık bakımı hizmetlerinde çalışan hemşirelerin tedavi bekleyen hastalara ulaştırılması problemlerinde sıkça rastlanmaktadır [9]. Bekleme süresinin dolayısıyla da rota süresinin enküçüklenmesinin önemli olduğu başka bir durum ise kolay bozulur ürün dağıtımında gözlemlenmektedir. Bu tarz dağıtımlarda kullanılan soğuk hava depolu araçların bekleme süreleri arttıkça enerji maliyetleri artmakta, dolayısıyla dağıtımı yapan firmaların işletim maliyetleri artmaktadır [10].

Zamana bağlı maliyet fonksiyonlarını göz önüne alan veya araçların depodan hareket zamanını karar değişkeni olarak ele alan az sayıda çalışma bulunmaktadır. Bunlar arasından en göze çarpanlardan biri Liberatore vd. [11]'ne aittir. Bu çalışmada tektürel bir araç filosunun yumuşak zaman pencerelerine sahip müşterilere hizmet verdiği bir ARP ele alınmaktadır. Problemde müşterilerin zaman penceresi kısıtları, araçların doğrusal bir ceza ödemeleri koşuluyla, müşterileri zaman pencereleri dışında da ziyaret etmelerine izin verecek şekilde gevşetilmiştir. Problemin çözümü için yazarlar bir dal-fiyat algoritması önermektedir. Ancak Liberatore vd. [11]'nin amacı rota süresini ve bu yüzden de toplam bekleme süresini enküçüklemek değildir; müşterilerin zaman pencerelerinden sapmanın yol açtığı maliyeti enküçüklemeye çalışırlar. Bettinelli vd. [12], çoklu depoya sahip karma sığalı ve zaman pencereli bir ARP'de araçların depodan hareket zamanını karar değişkeni olarak almakta ve problemin çözümü için bir dal-kesme-fiyat algoritması önermektedirler. Buna karşın sadece sabit araç gideri ve uzaklığa bağlı rotalama giderlerini enküçüklemeye çalışırlar. Dabia vd. [13] ise araçların ayrıtlar üzerindeki yolculuk sürelerinin zamana bağlı olduğu bir problem için

depodan hareket zamanını karar değişkeni olarak tanımlamaktadır. Zaman pencereli ve zamana bağlı bir ARP önerirken bir dal-fiyat algoritması kullanırlar. Savelsbergh [14], zaman pencereli bir ARP'nin toplam rota süresini enküçüklemek amacıyla ayrıt değiş tokuş yöntemleri geliştirir. ARP'nin yanı sıra Ioachim vd. [15], zaman penceresi ve doğrusal düğüm bekleme maliyeti içeren bir en kısa yol problemini çözebilmek için bir dinamik programlama algoritması geliştirmişlerdir. Bu çalışma aslında Desaulniers ve Villeneuve [16]'ün çalıştıkları doğrusal bekleme maliyetli bir en kısa yol probleminin özel bir durumunu yansıtmaktadır.

Archetti ve Speranza [17], zaman pencereli ARP başta olmak üzere değişik türde birçok ARP'nin kesin çözümü için dal-fiyat (DF) algoritmasının, kendini kanıtlamış ve şu anda önde gelen kesin yöntembilim olduğunu vurgulamaktadır. DF algoritması, temelde bir dal-sınır ağacının içine dâhil edilmiş bir sütun türetme (ST) algoritmasından meydana gelir [18, 19]. ST algoritması her yinelemede özgün değişkenlerin sadece bir alt kümesini içeren kısıtlı bir ana problem ve bunun yanında bir fiyatlandırma alt problemi çözen, yinelenen bir algoritmadır [20]. ST'nin ARP'ye uygulanması durumunda fiyatlandırma alt problemi aslında bir *kaynak kısıtlı temel en kısa yol problemidir*; işlevi ise kısıtlı ana probleme dâhil edilmemiş sütunlar (yani değişkenler) arasında negatif indirgenmiş fiyatlı olanları belirlemektir. ST ve/veya DF yöntemlerini ARP için uygulayan literatürdeki makalelere Ceselli vd. [7], Gutiérrez-Jarpa vd. [21], Salani ve Vacca [22], Desaulniers vd. [23], Liberatore vd. [11], Bettinelli vd. [12] ve Dabia vd. [13]'nin çalışmaları örnek olarak verilebilir. Ceselli vd. [7], çoktürel bir araç filosunun değişik depolardan hareket ettiği, müşteri, depo, ürün ve araçlar arasında uyumsuzluğun olduğu ve bazı müşterilere hizmeti aksatmaya izin veren bir ARP'yi çözmek için bir ST algoritması geliştirirler. Gutiérrez-Jarpa vd. [21] ise zaman pencereli, seçici topla ve dağıt bir ARP için kesin DF algoritması önerirler. DF algoritmasının zaman pencereli, bölünmüş talebe sahip bir ARP'ye uygulandığı çalışma ise Salani ve Vacca [22]'ya aittir. Farklı olarak dal-sınır ağacının her düğümünde olurlu bir çözüm elde etmek için açgözlü bir sezgisel kullanırlar.

DF yöntemini uygulayabilmek için ARP'nin ikili tamsayılı gösteriminin aslında bir küme kapsama formülasyonu olarak verilmesi gerekir [24]. Ancak bu formülasyon üssel sayıda değişkene sahip olduğundan dolayı fiyatlandırma alt problemini kesin olarak çözmek, zaman açısından oldukça masraflıdır. Bu nedenle DF yöntemini uygulayan çalışmalar, genelde 100 müşteriden daha azını içeren örneklerde eniyi sonucu bulabilmektedirler veya ana işlemci zamanına bir üst sınır koymak zorunda kalmaktadırlar (sözgelimi [21] ve [25]). Bu yüzden bu çalışmada ele alınan ARP için çok daha verimli sonuçlar üretebilecek mat-sezgiseller ortaya koymak gerekmektedir. Mat-sezgiseller, meta-sezgisel ve matematiksel programlama tekniklerinin birlikte işlediği sezgisel algoritmalardır [26]. Diğer bir deyişle sezgisel bir çerçevede matematiksel programlama modellerinden faydalanırlar; bunu yaparken matematiksel programlama modellerinin çözüldüğü evreleri sezgisellerin içine dâhil ederler. Bunu gerçekleştirmedeki hedef, meta-sezgisellerle karşılaştırıldığında eniyi çözüme olabildiğince yakın sonuçlar, yani kaliteli sonuçlar elde etmek ve kesin yöntemler/matematiksel programlama teknikleriyle karşılaştırıldığında ise daha makul bir ana işlemci zamanı içerisinde sonuçlar, yani verimli sonuçlar elde etmektir. Rotalama problemlerini çözen mat-sezgiseller üç sınıfa ayrılmaktadır [17]: ayrıştırma yaklaşımları, iyileştirme sezgiselleri ve dal-fiyat/sütun türetme (DF/ST) temelli yaklaşımlar. Üçüncü sınıfta bulunan yaklaşımlar genelde ST algoritmasının zamanından önce durdurulmasını, böylelikle kesin yöntemin yakınsaklığının hızlandırılarak eniyiliğin kaybedildiği algoritmaları içerir.

DF/ST temelli yaklaşımlarda en çok kısıtlı bir ana sezgiselin kullanıldığı algoritmalara rastlanmaktadır. Bu kısıtlı ana sezgisel, bir küme bölüntüleme formülasyonun fiyatlandırma algoritmasının türettiği bir sütun alt kümesi üzerinde çözüldüğü bir DF yaklaşımının içine gömülür. Böylelikle hızlı bir şekilde olurlu bir çözüm elde edilir. Kısıtlı ana sezgisel sayesinde sınırlarda hızlı bir iyileştirme elde edilir ve kesin yöntem böylelikle hızlandırılmış olur.

Kısıtlı ana sezgiselin kullanıldığı ilk çalışmalardan birisi Danna ve Pape [27]'ye aittir. Bu çalışmada yazarlar DF algoritmasının erkenden iyi sonuç elde edebilmesi için dal-fiyat algoritmasının içine bir kısıtlı ana sezgisel yerleştirip ana problemi, o ana kadar elde edilmiş en iyi tam sayılı çözümü başlangıç çözüm olarak kabul eden bir yerel arama algoritması ile çözmüşlerdir. Geliştirdikleri bu hibrit algoritmayı hem geleneksel DF algoritması hem de bir meta-sezgisel olan geniş komşuluk arama yöntemiyle zaman pencereli ARP üzerinde karşılaştırmışlardır. Bu karşılaştırmalar göstermektedir ki kesin bir yöntem olan DF algoritmasına göre daha hızlı bir şekilde eniyi çözüme ulaşmışlardır ve bunun yanında bir meta-sezgisele göre kısıtlı bir zaman içerisinde daha kaliteli sonuçlar elde etmişlerdir. Archetti vd. [28] kısıtlı ana sezgisel içeren DF algoritmasından, ayrık teslimatlı bir ARP'nin eniyi çözümünü bulmak için yararlanmışlardır. Makul bir ana işlemci zamanı içerisinde sonuç elde edebilmek için kısıtlı ana sezgiseli, türetilen sütunların bir alt kümesinde çalıştırmışlardır. Alt problemde sütunlar her rotadaki müşterilerin ziyaret sırasını verirken, ana problem teslimat miktarlarını belirlemiştir. Yaptıkları bilgisayımsal deneyler sonucunda literatürdeki benchmark test örneklerinin bilinen eniyilik boşluklarını iyileştirmişler ve birçok örnek için yeni en iyi sonuçlar elde ederek literatürdeki mevcut yöntemlere göre daha kaliteli sonuçlara ulaşmışlardır. İçine kısıtlı bir ana sezgisel yerleştirilen DF algoritmaları yukarıda verilen çalışmaların haricinde özellikle çoklu oyunculu oryantiring problemlerini çözmek için kullanılmışlardır. Belirlenen bir seyahat süresi kısıtı altında, tüm düğümlere uğrama kısıtının gevşetildiği ve her uğranan düğümden bir kâr elde edilerek, toplanan kârın enbüyüklenmeye çalışıldığı ARP türü literatürde çoklu oyunculu oryantiring problemi (ÇOOP) olarak bilinmektedir [29]. Archetti vd. [30] her müşterinin bir talebi olduğu sığalı ÇOOP'ni ele almışlardır.

Tamamlanmamış hizmete izin veren sığalı ÇOOP ise Archetti vd. [30] tarafından kısıtlı ana sezgisel içeren bir DF algoritması ile çözülmüştür. Archetti vd. [32] ve Archetti vd. [33] ise sırasıyla yine sığalı ÇOOP'ni ve bölünmüş talebe sahip ÇOOP'ni bir DF yöntemi yardımıyla ele almışlardır. Bu çalışmaların hepsinin ortak özelliği, fiyatlandırma alt problemini eniyiyi bulacak şekilde çözmüş ve sütunların sadece bir alt kümesinin kısıtlı ana sezgiselde küme bölüntüleme formülasyonunu çözmek için kullanılmış olmasıdır. Bu şekilde bir mat-sezgisel geliştiren tüm bu makalelerdeki çözüm yöntemi aslında problemin yaklaşık çözümünü elde etmek için değil, problemi kesin olarak çözmek için kullanılmıştır.

Buna karşılık bir de sütunları sezgisel türeten yöntemler bulunmaktadır. Sütun türetme iki farklı yolla gerçekleştirilir: (Eş. 1) kısıtlı ana sezgiselin verdiği dual bilgisini kullanmayan bir sezgisel, (Eş. 2) sadece kısıtlı sayıda sütunun türetildiği dual bilgisini kullanan bir ST algoritması. Birinci yol sütunları türetmek için sadece bir sezgisele ve bir küme bölüntüleme modeline ihtiyaç duymaktadır ve bu sebepten uygulanması ikinci yoldan daha kolaydır. Bunun doğal sonucu olarak literatürdeki birçok yöntem birinci yolu tercih etmektedir. Bu tarz çalışmalara [34-39] örnek olarak verilebilir. Aghezzaf vd. [34] her rotanın bir tedarikçide başlayıp bittiği ve araçlara çoklu rotaların atandığı bir periyodik envanter rotalama problemi ele almışlardır. Yazarların problemi çözmek için geliştirdikleri formülasyonda, her araç rotası ikili bir değişkenle temsil edilmekte ve sütunların Clarke-and-Wright sezgiseliyle türetildiği bir ST algoritması kullanılmaktadır. Geliştirdikleri yöntemin performansını analiz etmek için çoklu rota içeren kendi modellerini, tekli rota içeren klasik modelle karşılaştırmışlar ve kendi modellerinin tasarruf açısından çok daha kaliteli sonuçlar elde ettiğini gözlemlemişlerdir. Daha sonra bu problemin, her biri değişik çevrim süresine sahip olabilen, ama aynı araca atanan bir rota koleksiyonu içeren uzantısı Raa ve Aghezzaf [35] tarafından incelenmiştir. Ancak bu ve bir önceki çalışmada rota çevrim süreleri gün bazında tam sayılı değer almamaktadır. Rota çevrim sürelerinin tam sayılı değer alabildiği çalışma ise Raa ve Aghezzaf [36] tarafından gerçekleştirilmiştir. Bu son iki çalışmada da benzer şekilde sütunların sezgisel türetildiği bir ST algoritması kullanılmıştır.

Parragh vd. [37] bir sürücü kısıtlı, heterojen, merkezi olmayan istasyonlar arası araç rotalama problemini çözmek için bir değişken komşuluk arama yöntemini ST algoritmasının içine entegre etmişlerdir. İlk önce sentetik veri üzerinde yöntemin kısa süre içerisinde kaliteli sonuçlar elde ettiğini göstermişler, daha sonrasında ise klasik ST algoritmasıyla süre açısından bir karşılaştırma yapıp yöntemlerinin verimliliğini doğrulamışlardır. Parragh ve Schmid [38] ise sütunların yine bir değişken komşuluk arama algoritmasıyla türetilip küme kapsama formülasyonuna iletildiği mat-sezgiseli, merkezi olmayan istasyonlar arası araç rotalama problemini çözmek amacıyla kullanmışlardır. Belirli bir sayıda yineleme gerçekleştikten sonra, o ana kadar elde edilmiş en iyi sonucu iyileştirmek için bir geniş komşuluk arama algoritmasından faydalanarak verimli ve kaliteli sonuçlar üretebilen bir mat-sezgisel elde etmişlerdir. Son olarak, bu makaleler arasında en dikkat çekici olanı Cacchiani vd. [39]'nin periyodik ARP üzerine olan çalışmasıdır. Yazarlar bir küme kapsama formülasyonu önermekte ve sütunları bir yinelemeli yerel arama (YYA) algoritması ile türetmektedir.

Bu çalışmanın amacı, bir zaman pencereli ARP için toplam rota süresini enküçükleyen depodan hareket zamanını belirlemektir. Problemi çözmek için iki farklı sütun türetme temelli mat-sezgisel önerilmektedir. Mat-sezgisellerden yararlanılmasındaki amaç yukarıda belirtildiği gibi, kesin yöntemlerin vereceği sonuçlara meta-sezgisellerden daha fazla yaklaşabilen, doğruluk derecesi yüksek kaliteli sonuçlar elde edebilmek ve kesin yöntemlerden ise çok daha hızlı bir şekilde sonuçlara ulaşabilmek, yani verimli sonuçlar elde edebilmektir.

Çalışmanın geri kalanı şu biçimde planlanmıştır: 2. bölümde problemin matematiksel modeli ve özellikleri tanıtılmaktadır. 3. bölümde ise geliştirilen çözüm yöntemleri anlatılmaktadır. Bilgisayımsal deney sonuçları 4. bölümde verilmektedir. Son olarak, 5. bölümde sonuçlardan söz edilmektedir.

2. PROBLEMİN TANIMI VE MATEMATİKSEL MODELİ
(PROBLEM DESCRIPTION AND ITS MATHEMATICAL MODEL)

Bu çalışmada, tek depoya ve tektürel, sığalı araç filosuna sahip bir dağıtım şirketi tarafından belirli bir müşteri kümesine hizmet verilen bir zaman pencereli ARP ele alınmaktadır. Dağıtım şirketinin herhangi bir aracı, bir müşteriyi ziyaret ettiğinde müşterinin belirtmiş olduğu talep miktarı kadar teslimat yapar. Dağıtım şirketi teslimat yaparken her müşterinin kendi belirlediği zaman penceresi kısıtına uymak durumundadır. Bu durumda herhangi bir araç, bir müşterinin zaman penceresinin başlangıcından önce müşterinin bulunduğu noktaya vardığı takdirde zaman penceresinin başlangıcını beklemek zorundadır. Buna karşılık araç, müşterinin zaman penceresinin bitişinden sonra müşteriye dağıtım hizmetini verememektedir. Bu çalışmada diğer araç rotalama problemlerinden farklı olarak araçlar istenilen herhangi bir zamanda depodan ayrılarak kendilerine atanan rotada müşterilere hizmet vermeye başlayabilmektedirler. Ancak her araç, sürücülerin vardiya sürelerine göre belirlenmiş azami bir süre için kullanılabilmektedir ve her aracın atandığı rota depoda başlamakta, depoda bitmektedir. Herhangi bir aracın ulaşım maliyeti klasik ARP'lerden farklı olarak kendisine atanan rotanın süresine bağlı bir fonksiyondur. Dağıtım şirketinin amacı, toplam ulaşım giderini, yani toplam rota sürelerini enküçükleyen araç rotalarını ve bir rotaya atanan her aracın depodan çıktığı zamanı belirlemektir. Bunu yaparken dağıtım şirketi şu kısıtlara uymak durumundadır: araç sığa kısıtı, araçların azami süre kısıtı, müşterilerin zaman penceresi kısıtları, her müşterinin talebinin karşılanması kısıtı, her müşterinin sadece bir araç rotasında yer alması

kısıtı ve her rotanın depoda başlayıp depoda sona ermesi kısıtı. Bu problemin formülasyonu için V düğüm kümesi ve A ayrıt kümesine sahip $G = (V, A)$ ile gösterilen yönlü bir çizge ele alınır. V düğüm kümesi, $V = N \cup \{0, n+1\}$ şeklinde tanımlanır. Burada N, hizmet götürülmesi gereken n tane müşterinin bulunduğu düğüm kümesini, 0 ve $n+1$ düğümleri ise sırasıyla deponun başlangıç ve bitiş kopyalarını gösterir. Her $i \in N$ müşterisinin $d_i \geq 0$ ile ifade edilen bir talep miktarı, $s_i \geq 0$ ile ifade edilen hizmet süresi ve $[a_i, b_i]$ kapalı aralığı ile gösterilen bir zaman penceresi bulunur. a_i zaman penceresinin başlangıcını belirtirken, b_i zaman penceresinin bitişini gösterir. Deponun iki kopyası için ilgili değerler ise $d_0 = d_{n+1} = s_{n+1} = 0$ şeklindedir. Kısacası deponun talep miktarı ve araçlar depoya döndüğünde aracı boşaltmak için gereken süre sıfırdır. Buna karşın, aracı depoda yüklemek gerektiğinden $s_0 > 0$ olarak kabul edilir. Herhangi bir aracın depodan hareket zamanı bir karar değişkeni olduğundan, deponun başlangıç ve bitiş kopyalarının zaman pencerelerinin belirgin sınırları yoktur. Bu sebepten $[a_0, b_0] = [a_{n+1}, b_{n+1}] = (-\infty, +\infty)$ kabul edilir. Bunların yanı sıra depoda bekleyen her aracın sığası Q ile belirtilir. Her araç depodan hareket edip, kendisine atanan rotadaki müşterilere istedikleri talebi teslim edip, tekrar depoya döneceğinden dolayı, bu müşterilerin toplam talep miktarının Q'yu geçmemesi gerekir. Aynı zamanda her aracı kullanan sürücünün S ile gösterilen bir vardiya süresi vardır. Bu sebeple her aracın kullanıldığı sürenin, yani kendisine atanan rotanın toplam süresinin S'yi geçmemesi gerekir. Son olarak, bir $i \in V$ düğümünden $j \in V$ düğümüne seyahat etmek için gereken süreyi t_{ij} olarak tanımladıktan sonra, ayrıt kümesi olan A matematiksel olarak şu şekilde ifade edilebilir: $A = \{(i,j): i,j \in V, i \neq j, a_i + s_i + t_{ij} \leq b_j\}$.

Bu problemi çözmek için yukarıda verilen yönlü çizge üzerinde çalışan bir küme kapsama formülasyonu oluşturulmuştur. Böylelikle problemi ST temelli matsezgiseller ile çözmek mümkün hale gelmiştir. Bu şekildeki bir formülasyonda her sütun, yani her değişken, olurlu bir araç rotasına karşılık gelmektedir. Olurlu bir araç rotası ise yukarıda belirtilen tüm kısıtları sağlayan ve herhangi bir müşteriyi en fazla bir kere ziyaret eden bir rotadır. Bu olurlu tüm rotaları içeren küme ise Ω ile temsil edilir. Ω kümesindeki her r rotasının maliyetine, yani süresine, c_r, $i \in N$ müşterisinin $r \in \Omega$ rotasında ziyaret edilip edilmediğini belirten gösterge parametreye g_{ir} ve de $r \in \Omega$ rotasının eniyi çözümde yer alıp almadığını gösteren ikili karar değişkenine x_r denirse, problemi çözen küme kapsama formülasyonu (KKF) Eş. 1, Eş. 2, Eş. 3 gibi verilir:

KKF: $Enk \sum_{r\in\Omega} c_r x_r$ (1)

öyle ki $\sum_{r\in\Omega} g_{ir} x_r \geq 1, \forall i \in N$ (2)

$x_r \in \{0,1\}, \forall r \in \Omega$ (3)

(Eş. 1) ile gösterilen amaç fonksiyonu, toplam maliyeti, diğer bir deyişle toplam rota sürelerini enküçükler. (Eş. 2) numaralı kısıt kümesi, her müşterinin en az bir kere ziyaret edilerek hizmet almasını sağlar. Son olarak (Eş. 3) numaralı kısıt kümesi, karar değişkenlerinin ikili tamsayı değerler almasını garanti eder.

3. SÜTUN TÜRETME TEMELLİ MAT-SEZGİSELLER
(COLUMN GENERATION BASED MATHEURISTICS)

(Eş. 1, Eş. 2, Eş. 3) ile gösterilen **KKF** formülasyonunda kullanılan, deponun başlangıç ve bitiş kopyaları olan 0 ve $n+1$ düğümleri arasında yer alan tüm olurlu rotaları içeren Ω kümesinin kardinalitesi n ile gösterilen müşteri sayısının artması ile üssel olarak artmaktadır [24]. Bu çalışmada ele alınan probleme ait G çizgesi aslında tam bir çizgedir. Bu çizgeden sadece zaman penceresi olurluğunu bozan ayrıtlar, algoritmalar çalıştırılmadan önce bir ön işleme ile çıkartılmaktadır. Bu sebepten dolayı 0 düğümünden başlayıp $n+1$ düğümünde biten toplam olası rota sayısı, $P(n,k)$'nın n müşteri sayısının k'lı permütasyonunu simgelediği kabul edilirse $\sum_{k=1}^{n} P(n,k) = \sum_{k=1}^{n} \frac{n!}{(n-k)!}$ formülüyle bulunur [40]. n müşteri sayısının 9 olduğu durumda formül kullanılarak toplam rota sayısının 986.409 olduğu kolaylıkla görülebilir. Müşteri sayısının 1 artması dâhi toplam rota sayısını çok büyütmektedir. Ancak bu rotaların bir bölümü olurlu değildir, yani Ω kümesine ait değildir. Eğer KKF formülasyonu, tüm rotalar türetilerek çözülmek istenirse deponun başlangıç ve bitiş kopyaları olan 0 ve $n+1$ düğümlerinin arasındaki tüm yolların tespit edilmesi gerekmektedir. Bu yolların tespiti ise bir derinliğine arama algoritmasının yardımıyla gerçekleştirilebilir [41, 42]. Derinliğine arama algoritması (DAA), özyinelemeli bir algoritmadır. Başlangıçta tüm düğümleri ziyaret edilmemiş kabul eder ve herhangi bir düğümü seçerek arama yapmaya başlar. Bir düğümü ziyaret ettiğinde o düğümü ziyaret edilmiş olarak işaretler ve o düğümün komşusu olan ziyaret edilmemiş tüm düğümleri yinelemeli olarak ziyaret eder. Problemde ihtiyaç duyulan sadece olurlu rotalar olduğu için bir her düğüme sıra geldiğinde o düğümün eklendiği rotayı olursuz hale getirip getirmediği kontrol edilir ve eğer olursuz hale getiriyorsa o düğümle devam edilmemesi için budanır [41, 42]. Böylelikle elde edilen toplam rota sayısı $\sum_{k=1}^{n} \frac{n!}{(n-k)!}$ sayısına ulaşmaz ve arama süresi kısalır.

DAA algoritması olabilecek en verimli şekilde kodlandığında bile, rota sayısının, müşteri sayısının artması ile üssel olarak büyümesinden dolayı, olurlu tüm rotaları polinom zamanda türetmek mümkün değildir [19, 24]. Bu sebeplerden ötürü, KKF problemini makul bir ana işlemci zaman dâhilinde çözebilmek için formülasyonun doğrusal gevşetmesinden faydalanılacaktır. Feillet [24], bu doğrusal gevşetme ST yöntemiyle çözüldüğünde, amaç fonksiyonunun eniyi değeri için çok sıkı alt sınırlar elde edildiğini vurgular. ST ise her yinelemede özgün değişkenlerin sadece bir alt kümesini içeren kısıtlı bir ana problem ve bunun yanında bir fiyatlandırma alt problemi çözen yinelenen bir algoritmadır. Fiyatlandırma alt problemi bir kaynak kısıtlı temel en kısa yol problemidir (KKTEKYP)

[20]. Özgün değişkenlerin sadece bir alt kümesinin kullanılmasının sebebi, Ω kümesinin kardinalitesinin n ile üssel olarak artmasıdır. k. yinelemede kullanılan alt küme $\Omega_k \subseteq \Omega$ olarak ifade edilirse, bu yinelemede çözülmesi gereken ana problem şu şekilde formüle edilir:

$$\text{KKF}_k\text{: } Enk \sum_{r\in\Omega_k} c_r x_r \quad (4)$$

$$\text{öyle ki } \sum_{r\in\Omega_k} g_{ir} x_r \geq 1, \forall i \in N \quad (5)$$

$$0 \leq x_r \leq 1, \forall r \in \Omega_k \quad (6)$$

Cplex gibi bir doğrusal programlama çözücüsünden faydalanılarak (Eş. 4, Eş. 5, Eş. 6) ile verilen formülasyonun eniyi primal ve dual çözümleri elde edilir. Fiyatlandırma alt problemi ise yukarıda değinildiği gibi bir KKTEKYP'dir. Ancak KKTEKYP'nin ayrıt maliyetleri k. kısıtlı ana problemin, yani (Eş. 4, Eş. 5, Eş. 6) ile verilen KKF_k probleminin dual çözümüne bağlıdır. KKTEKYP'nin amacı ise bu dual çözüme ilişkin, $\Omega \setminus \Omega_k$ kümesinde bulunan negatif indirgenmiş maliyetli değişkenleri (sütunları) tespit etmektir. Eğer böyle bir değişken yoksa k. kısıtlı ana problemin eniyi çözümü, ana problem için, yani (Eş. 1, Eş. 2, Eş. 3) ile verilen KKF formülasyonun doğrusal gevşetmesi için de eniyi çözümdür. Tek yapılması gereken $r \in \Omega \setminus \Omega_k$ olan tüm x_r değişkenlerinin değerine 0 vermektir. Aksi halde tespit edilen tüm negatif indirgenmiş maliyetli değişkenler (k+1). yinelemeye başlamadan Ω_k kümesine eklenerek Ω_{k+1} kümesi elde edilir.

KKTEKYP'nde elde edilen en düşük maliyetli yani en kısa yol temel bir yol olmak, yani hiçbir düğümün birden fazla ziyaret edilmediği bir yol olmak durumundadır. Üzerinde çalışılan çizge, çevrimsiz bir çizge ise, elde edilen yolları temel olmak için zorlamaya gerek yoktur. Ancak ayrıtların üzerinde negatif maliyetler tanımlı olduğunda çizge çevrimsel ise temel kısıtların varlığı kaçınılmazdır. Dror [43], KKTEKYP'nin çevrimsel çizgeler için çözümünün kuvvetle NP-zor olduğunu vurgular. Özetle ST algoritmasının başarısı, KKTEKYP'nin çözümünün başarısına sıkı bir şekilde bağlıdır, çünkü KKTEKYP'nin çözüm süresi ST algoritmasının çözüm süresinde belirleyici etmendir.

Ele alınan bu problemin eniyi çözümü ancak bir DF algoritması ile elde edilebilir. Bunun için de yukarıda bahsedilen ST algoritması, bir dal-sınır ağacının içine dâhil edilir. Ancak problemde araçlar herhangi bir zamanda depodan hareket edebildiğinden ve toplam yolculuk, hizmet ve bekleme sürelerine bağlı maliyet oluşturduğundan, KKTEKYP'nin sonsuz sayıda Pareto-eniyi durumu veya başka deyişle sonsuz sayıda etiketi dikkate alması gerekir. Fiyatlandırma alt probleminin bu özelliği, problemin çözümünü çok zor hale getiren en kritik sorunlardan birisidir. Sonsuz sayıda etiket sorununun üstesinden gelebilmek için eniyi çözüme götürmeyecek olurlu araç rotalarını eleyen üstünlük kurallarından yararlanan dinamik programlama algoritmaları geliştirmek gerekir. Literatürdeki mevcut üstünlük kuralları burada ele alınan ARP için uygun olmadığından probleme uygun yeni üstünlük kuralları keşfetmek kaçınılmazdır. Ayrıca literatürdeki çalışmalar, daha kolay problemler için geliştirilen DF algoritmalarının, 25 müşteri içeren küçük örneklerde bile bazen makul bir sürede hiç sonuç elde edemediklerini göstermektedir. Diğer yandan Raidl ve Puchinger [44], literatürde ST ve/veya DF yöntemlerini tümleşik bir şekilde bir meta-sezgiselle birleştiren algoritmalarda fiyatlandırma alt problemlerini kesin çözmek yerine karmaşık bir fiyatlandırma meta-sezgiselinin geliştirilmesinin eksikliğinden bahsederler. Bu sebeplerden dolayı bu çalışmanın amacı ST algoritmasının sütunları rafine bir meta-sezgiselden aldığı mat-sezgiseller geliştirmektir. Böylelikle, ele alınan bu zor problem için bilindiği kadarıyla ilk defa verimli çözüm yöntemleri geliştirilmiş olacaktır.

Bu çalışma esnasında ele alınan problemi çözmek için iki sütun türetme temelli mat-sezgisel geliştirilmiştir. İki mat-sezgiselin birbirinden farkı fiyatlandırma alt problemini farklı meta-sezgisellerle çözmesi, dolayısıyla kısıtlı ana problemin ihtiyaç duyduğu sütunları farklı şekilde elde etmesidir. İlk mat-sezgisel, meta-sezgisel olarak yinelemeli yerel arama algoritması kullanırken, ikinci mat-sezgisel değişken komşuluk arama algoritmasından meta-sezgisel olarak faydalanmaktadır. İki mat-sezgisel de ST algoritmasını her bir rotanın tek bir müşteriyi ziyaret ettiği müşteri sayısı kadar sütun ile başlatır ve ST algoritması negatif indirgenmiş maliyetli sütun türetmeye devam ettiği sürece çalıştırılır. Ancak sütun havuzunda biriktirdikleri sütun sayısı 3000'i aştığında son 500 yineleme boyunca bazda hiç yer almayan değişkenler yani sütunlar havuzdan çıkartılır. ST algoritması işini tamamladığındaysa sütun havuzunda saklanan tüm sütunlar Ω kümesine eklenerek (Eş. 1, Eş. 2, Eş. 3) ile verilen ikili tamsayılı programlama KKF formülasyonu, yani küme kapsama formülasyonu, Cplex 12.7 çözücüsü ile çözülerek ele alınan ARP için olurlu bir çözüm, dolayısıyla ARP'nin eniyi amaç fonksiyon değeri için bir üst sınır elde edilir. Bu şekilde oluşturulan iki yöntem sütun türetme temelli mat-sezgisellerdir, çünkü mat-sezgiseller, meta-sezgisel ve matematiksel programlama tekniklerinin birlikte işlediği sezgisel algoritmalardır.

3.1. Yinelemeli Yerel Arama Kullanan Sütun Türetme Temelli Mat-Sezgisel
(Column Generation Based Matheuristic Using Iterated Local Search)

Yinelemeli yerel arama (YYA) algoritması tek çözüm temelli bir yaklaşımdır; bir yerel arama sezgiselinin yinelenen uygulamasını ve farklılaştırma için bir pertürbasyon mekanizmasını içerir [45, 46]. Birçok sezgiselde kullanılan basit bir prensibe dayanmaktadır. Bir başlangıç çözümle başlar ve bu başlangıç çözüme bir yerel arama uygular. Daha sonra her yinelemede yerel arama ile elde edilen yerel çözüme pertürbasyon uygular. Pertürbasyon operatörü, eldeki geçerli çözüme uygulanan rassal büyük bir hamledir. Buradaki amaç eldeki geçerli çözümün bir parçasını tutarken başka bir çekim havzasına ulaşmak için çözümün diğer parçasını güçlü bir şekilde bozmaktır. Her yinelemede geçerli çözüme pertürbasyon uygulandıktan sonra elde edilen yeni çözüme (bu çözüm

çoğunlukla olurlu bir çözüm değildir) bir yerel arama uygulanarak, yeni bir yerel olurlu çözüm elde edilir. Elde edilen yeni yerel çözüm belirli koşulları sağlıyorsa yeni geçerli çözüm olarak kabul edilir. Bu süreç baştan belirlenen durma koşulları sağlanana kadar devam eder [45, 46].

YYA algoritmasının fiyatlandırma alt problemini çözmeye başlaması için bir başlangıç çözüme ihtiyacı vardır. Bu çalışmada başlangıç çözümü oluşturmak için araçların azami süre kısıtını gevşeten bir en ucuz ekleme sezgiseli oluşturulmuştur. Sezgisel, boş müşteri kümesiyle başlayıp her yinelemede en az maliyeti veren tek bir müşteri ekleyerek, $T_0 = 0$ zamanında depoda başlayıp bir müşteri alt kümesini ziyaret edip tekrar depoda biten bir rota oluşturur. Müşteriler rotaya eklenirken aracın sığa kısıtı ve müşterilerin zaman pencereleri dikkate alınır. Bu durumda elde edilen başlangıç çözüm azami süre kısıtını ihlal eden bir çözüm olabilmektedir. Olurlu çözüm bulma sorumluluğu ise yerel arama algoritmasına yüklenmiştir. Fiyatlandırma algoritmasıyla çözülmek istenilen problem bir KKTEKYP'dir; yani deponun başlangıç ve bitiş kopyaları arasında negatif maliyetli ayrıtlar üzerinden geçen bir kısa yoldur. Kısıtlı ana problemden alınan dual fiyatların her biri bir müşteriye yani bir düğüme ait olduğundan dolayı yüksek bir dual fiyata sahip bir düğüme ulaşan tüm ayrıtların negatif maliyete sahip olma olasılığı yüksektir. Bu nedenle YYA algoritmasındaki komşuluklar, düğüm hamleleri temel alınarak oluşturulmuştur. Algoritma boyunca kullanılan olası komşuluklar şu şekildedir:

- Ekleme: Rotaya önceden ziyaret etmediği yeni bir düğüm ekleme.
- Çıkarma: Rotada ziyaret edilen bir düğümü çıkarma.
- Taşıma: Rotada ziyaret edilen bir düğümü bulunduğu pozisyondan alıp başka bir pozisyona taşıma.
- Yer değiştirme: Rotada ziyaret edilen bir düğümü çıkarıp yerine aynı pozisyona ziyaret edilmemiş bir düğüm ekleme.
- Değiş tokuş: Farklı sırada ziyaret edilen iki düğümün karşılıklı ziyaret sıralarını değiştirme.

Bu hamleler, aracın sığa kısıtı ve müşterilerin zaman penceresi kısıtları dikkate alınarak gerçekleştirilirken, araçların azami süre kısıtları ise yukarıda belirtildiği gibi pertürbasyon sırasında ihlal edilebilmiştir. Bu beş komşuluğun hepsi pertürbasyon esnasında kullanılırken yerel arama esnasında sadece ilk iki komşuluk kullanılmıştır. Pertürbasyon evresinde yerel aramadan gelen çözüme peş peşe 5 rassal hamle uygulanarak bir çözüm bulunur. Daha sonra pertürbasyondan gelen bu çözüme tekrar yerel arama uygulanarak, yerel olurlu tek bir çözüm (rota/sütun) elde edilir. Eğer bu son elde edilen rotanın maliyeti negatif ise YYA algoritmasının başında boş küme olarak başlatılan negatif indirgenmiş maliyetli sütunlar kümesine eklenir. YYA algoritması sona erdiğinde ise bu kümede bulunan sütunlar ST algoritmasının sütun havuzuna aktarılır.

YYA algoritmasının durdurulması için uyarlamalı bir mekanizma kullanılmıştır. Eğer yapılan yineleme sayısı fazla ise, sezgisel iyi kalitede çözümler üretir, ancak bu uzun zaman alır. ST algoritmasının başında genellikle çok fazla yineleme yapmaya ihtiyaç yoktur, çünkü KKTEKYP'nin çözümleri hızlı bir şekilde elde edilmektedir. İlerleyen yinelemelerde ise ana problemin çözümü zorlaşır ve YYA'nın daha fazla yineleme gerçekleştirmesi gerekmektedir [39]. Bu sebeple bu çalışmada YYA'nın durdurulması için gereken yineleme sayısı uyarlamalı bir şekilde belirlenmiştir. *yinelemeMax* kadar bir yineleme sayısı gerçekleştikten sonra eğer YYA negatif indirgenmiş maliyetli yollar üretebildiyse durdurulmuştur. Üretemediyse yineleme sayısı arttırılarak, yani *yinelemeMax* iki katına çıkartılarak YYA'ya devam edilmiştir. Ayrıca yapılan toplam yineleme sayısına bakılıp eğer bu sayı *MaxyinelemeMax* ile belirtilen bir üst sınırı geçtiyse YYA yine durdurulmuştur. Bu çalışmada, hem literatürdeki örnekler incelenerek hem de yapılan deneme sonuçlarına bakılarak *yinelemeMax* parametresi 50 ve *MaxyinelemeMax* parametresi 500 olarak belirlenmiştir.

YYA algoritmasının sözde kodu Şekil 1'de gösterilmektedir. Sözde kodda verilen *neg_indirg_maliyet_sütun*, negatif indirgenmiş maliyetli sütunlar kümesini, SYA ise yerel arama evresinde kullanılan salınımlı yerel arama algoritmasını ifade etmektedir.

Girdi: *yinelemeMax*, *MaxyinelemeMax*, $\boldsymbol{\pi}$ dual değişken vektörü
1. $x_0 \leftarrow En\_Ucuz\_Ekleme\_Sezgiseli(\boldsymbol{\pi})$
2. $yineleme \leftarrow 1$
3. $neg\_indirg\_maliyet\_sütun \leftarrow \emptyset$
4. $x_{en_iyi} \leftarrow$SYA$(x_0, \boldsymbol{\pi})$
5. $maliyet_{en\_iyi} \leftarrow maliyet(x_{en\,iyi}, \boldsymbol{\pi})$
6. **While** $yineleme < yinelemeMax$ **do**
7. $\quad x_{pert} \leftarrow$ PERTURB(x_{en_iyi})
8. $\quad x_{pert} \leftarrow$ SYA$(x_{pert}, \boldsymbol{\pi})$
9. $\quad$ **If** $maliyet(x_{pert}, \boldsymbol{\pi}) < maliyet_{en\_iyi}$
10. $\quad\quad x_{en\_iyi} \leftarrow x_{pert}$
11. $\quad\quad maliyet_{en\_iyi} \leftarrow maliyet(x_{pert}, \boldsymbol{\pi})$
12. $\quad$ **End**
13. $\quad$ **If** $maliyet(x_{pert}) < 0$
14. $\quad\quad neg\_indirg\_maliyet\_sütun \leftarrow neg\_indirg\_maliyet\_sütun \cup \{x_{pert}\}$
15. $\quad$ **End**
16. $\quad$ **If** $maliyet_{en\_iyi} \geq 0 \wedge yinelemeMax < MaxyinelemeMax$
17. $\quad\quad yinelemeMax \leftarrow 2 \times yinelemeMax$
18. $\quad$ **End**
19. $\quad yineleme \leftarrow yineleme + 1$
20. **End**
21. **Return** $neg\_indirg\_maliyet\_sütun$

Şekil 1. Yinelemeli Yerel Arama (YYA) algoritması
(Iterated Local Search Algorithm)

YYA algoritmasının yerel arama evresinde yukarıda belirtildiği üzere salınımlı yerel arama algoritmasından (SYA) yararlanılmıştır. Bu algoritmanın Cuervo vd. [47] tarafından, zaman penceresiz bir topla ve dağıt ARP için geliştirilmiş YYA algoritmasında başarılı bir şekilde uygulandığı gözlemlenmiştir. Bu çalışmadaki problemde SYA esnasında da azami süre kısıtının gevşetilmesine izin verilmiştir. Ancak bu kısıta uymayan çözümler cezalandırılmıştır. SYA algoritması, her rotanın maliyetini hesaplarken aşılan süre miktarını γ ceza parametresiyle çarpmakta ve rotanın maliyetine eklemektedir. γ ceza parametresi ilk yinelemede γ_0 değeriyle başlatılmakta, olurlu

bir çözüm bulunamadığında $\delta > 1$ diye bir katsayı ile çarpılarak arttırılmaktadır. SYA her yinelemede eldeki geçerli çözümün tüm komşularının amaç fonksiyon değerlerini γ ceza parametresini göz önünde bulundurarak hesaplamakta ve aralarından en düşük maliyetli komşuyu seçmektedir. Eğer seçilen komşu olurlu bir çözümse, geçerli olurlu çözümden daha iyi olup olmadığı kontrol edilmekte ve daha iyiyse bu komşu çözüm yeni geçerli olurlu çözüm olmakta ve γ ceza parametresi tekrar γ_0 değerini almaktadır. Aksi takdirde SYA algoritması sona erdirilmektedir.

Böylelikle SYA algoritması pertürbasyon evresinden gelen çözümden bir yerel minimum elde etmektedir. Şekil 2, SYA algoritmasının sözde kodunu göstermektedir. Bu algoritmada yapılan denemeler sonucunda γ_0 parametresinin değeri sıfır, δ katsayısının değeri ise iki olarak belirlenmiştir. γ parametresinin değerinin arttırılması gerektiği durumlarda eğer değeri γ_0'a eşitse, yeni değeri δ katsayısının değerine eşitlenmiştir.

Girdi: x_0 başlangıç çözümü, γ_0 ceza parametresi başlangıç değeri, δ ceza katsayısı, $\boldsymbol{\pi}$ dual değişken vektörü
1. **If** x_0 olurlu bir çözüm ise
2. $\quad x_{en\_iyi\_olurlu} \leftarrow x_0$
3. $\quad maliyet_{en\_iyi\_olurlu} \leftarrow maliyet(x_0, \boldsymbol{\pi})$
4. **End**
5. **Else**
6. $\quad maliyet_{en\_iyi\_olurlu} \leftarrow +\infty$
7. **End**
8. $\gamma \leftarrow \gamma_0$
9. $x_{geçerli} \leftarrow x_0$
10. $durma\ koşulu \leftarrow false$
11. **While** $\neg durma\_koşulu$ **do**
12. $\quad x_{geçerli} \leftarrow en\_düşük\_maliyetli\_komşu(x_{geçerli}, \gamma, \boldsymbol{\pi})$
13. $\quad$ **If** $x_{geçerli}$ olurlu bir çözüm ise
14. $\quad$ **If** $maliyet(x_{geçerli}, \boldsymbol{\pi}) < maliyet_{en\_iyi\_olurlu}$
15. $\quad\quad x_{en\_iyi\_olurlu} \leftarrow x_{geçerli}$
16. $\quad\quad maliyet_{en\_iyi\_olurlu} \leftarrow maliyet(x_{geçerli}, \boldsymbol{\pi})$
17. $\quad\quad \gamma \leftarrow \gamma_0$
18. $\quad$ **End**
19. $\quad$ **Else**
20. $\quad\quad durma\_koşulu \leftarrow true$
21. $\quad$ **End**
22. **End**
23. **Else**
24. $\gamma \leftarrow \delta \times \gamma$
25. **End**
26. **Return** $x_{en\ iyi\ olurlu}$

Şekil 2. Salınımlı Yerel Arama (SYA) Algoritması
(Oscillating Local Search Algorithm)

Burada bahsedilmesi gereken başka önemli bir nokta ise yerel arama evrelerinde uygulanan düğüm ekleme ve çıkarma hamlelerinden sonra oluşan yeni rotaların depodan eniyi başlama zamanının bulunmasının gerekliliğidir. Aksi takdirde kısıtlı ana probleme eklenecek yeni keşfedilmiş rotaların c_r ile gösterilen maliyet parametreleri hatalı olacaktır. Her hamle uygulanırken aslında anlık kısmi iki yol oluşmakta ve sonra bu iki kısmi yolun tek bir rota verecek şekilde birleştirilmesi gerekmektedir. Klasik ARP'de bunu yapmak kolaydır, çünkü rotanın maliyeti toplam katedilen yol olduğundan iki kısmi yolun uzunluklarının toplamı yeni rotanın maliyetine eşittir. Ancak bu çalışmada ele alınan ARP'de rotaların maliyetleri sürelerine bağlıdır ve birleştirilen iki kısmi yoldan elde edilen yeni rotadaki toplam bekleme süresi önceden hesaplanamaz. Bu zorluğun üstesinden gelmek için Savelsbergh [14]'in birleştirme teoremi kullanılmıştır. Savelsbergh [14]'in bu teoremi, rotayı müşterilerin zaman pencereleri açısından olursuz hale getirmeden ziyaret edilen bir düğümün hizmet başlangıç zamanının ne kadar ötelenebileceğini belirten *ileri zaman payı* kavramını kullanmaktadır. Teorem, F_1 ve F_2 ileri zaman payına sahip (V_0, …, V_i) ve (V_{i+1}, …, V_0) şeklindeki iki olurlu kısmi yol birleştirildiğinde, T_0 ile belirtilen, birinci kısmi yolun depodan başlama zamanının $F = min\{F_1, F_2 + w_1 + T_{i+1} - (T_i + \bar{t}_{i,i+1})\}$ kadar ötelenebildiğini veya öne çekilebildiğini hesaplamaktadır. Burada w_1, birinci kısmi yoldaki toplam bekleme süresini, T_i, i düğümünde hizmete başlama zamanını, $\bar{t}_{i,i+1}$ ise i düğümünden i+1 düğümüne gitmek için gereken sürenin ve i düğümündeki hizmet süresinin toplamını göstermektedir. Böylelikle iki kısmi yol birleştirilerek elde edilen (V_0, …, V_i, V_{i+1}, …, V_0) yolunun eniyi depodan başlama zamanı $T_0 + F$ olarak hesaplanabilmektedir. Yeni elde edilen yoldaki toplam bekleme süresini hesaplamak içinse ilk önce, birleşmeden sonra depodan başlama zamanı T_0 olarak bırakıldığında, V_{i+1} düğümündeki hizmete başlama zamanı olan T_{i+1}'deki artışı ya da azalmayı gösteren λ, $T_i + \bar{t}_{i,i+1} - T_{i+1}$ olarak tanımlanır. Kısacası λ, birinci kısmi yolun başlama zamanı T_0 olarak kaldığında, V_{i+1} düğümüne ulaşma zamanı ile T_{i+1} arasındaki farkı temsil eder. Bu durumda λ, hem pozitif hem negatif değer alabilmektedir. λ'nın haricinde bir de *geri zaman payı* kavramı tanıtılmaktadır. Geri zaman payı, ziyaret edilen bir düğümün hizmete başlama zamanının yolda ek bekleme süresi yaratmadan ne kadar geri çekilebileceğini hesaplamaktadır. (V_{i+1}, …, V_0) kısmi yolunun geri zaman payına B_2 ve bekleme yaptığı süreye w_2 denirse birleştirilen yoldaki toplam bekleme süresi olan w, Tablo 1'de verilen şekilde hesaplanabilmektedir [14]:

Tablo 1. İki kısmi yolun birleştirilmesiyle elde edilen yeni yoldaki toplam bekleme süresi
(Total waiting time on the new path obtained by concatenating two partial paths)

	$\lambda \geq 0$	$\lambda < 0$
$w_2 = 0$	w_1	$w_1 + \max\{0, -\lambda - B_2\}$
$w_2 > 0$	$w_1 + \max\{0, w_2 - \lambda\}$	$w_1 + w_2 - \lambda$

Böylelikle ileri zaman payı olan F, sıfırdan büyük veya sıfıra eşitse w'dan büyük olup olmadığı kontrol edilmektedir. Eğer $F \geq w$ ise $T_0 + F$ zamanında başlayan yeni rotadaki en az bekleme süresi sıfırdır, aksi takdirde $w - F$ kadardır. $F \geq 0$ olması durumu, T_0 ile belirtilen birinci kısmi yolun depodan başlama zamanının, yeni elde edilen (V_0, …, V_i, V_{i+1}, …, V_0) yolu için ötelenmesi gerektiğini gösterir. Eğer tam tersine $F < 0$ ise birinci kısmi yolun depodan başlama zamanının öne çekilmesi gerekir. Her iki durumda da yeni elde edilen rotanın depodan başlama zamanı $T_0 + F$ olarak hesaplanır. $F < 0$ durumunda yeni rotadaki toplam bekleme süresini hesaplamak için ikinci kısmi yolunun bekleme süresi olan w_2 değerine bakılmaktadır. Bu değer sıfırdan büyükse $T_0 + F$ zamanında başlayan yeni rotadaki en az bekleme süresi $w + |F|$ olarak hesap edilmekte, aksi takdirde sıfır olmaktadır. Bu sayede YYA algoritmasının yerel arama evresinde elde edilen her olurlu rotanın depodan eniyi başlama zamanı ve toplam süresi hesaplanmış olmaktadır. Daha önce belirtildiği üzere YYA algoritması aslında ST algoritmasının her yinelemede çözdüğü KKF_k probleminin kullanacağı Ω_k rota

alt kümesine dâhil edilmek üzere, negatif indirgenmiş maliyetli sütunları türetmektedir. ST algoritması ise YYA algoritmasını, negatif indirgenmiş maliyetli sütun bulamayana kadar her yinelemede çağırır. Bu şekilde sonuncu yineleme gerçekleştirildikten sonra, o ana kadar tespit edilmiş tüm negatif indirgenmiş maliyetli sütunlar, boş olan Ω kümesine eklenerek KKF problemi ikili tamsayılı programlama formülasyonu olarak çözülür. Böylelikle özgün problem için olurlu bir çözüm elde edilir.

3.2. Değişken Komşuluk Arama Kullanan Sütun Türetme Temelli Mat-Sezgisel
(Column Generation Based Matheuristic Using Variable Neighborhood Search)

Değişken komşuluk arama algoritması (DKA), hem yerel minimumun iniş yönünde hem de bunları içeren vadilerden kaçma yönünde sistematik olarak komşuluk değiştirme fikrini kullanan bir algoritmadır [48]. Hansen vd. [48]'nin vurguladığı üzere, bu fikir aşağıda belirtilen üç olguya dayanmaktadır: i) Bir komşuluk yapısı için yerel minimum olan bir çözüm başka bir komşuluk yapısı için yerel minimum olmayabilir. ii) Bir global minimum, tüm komşuluk yapılarında yerel minimumdur. iii) Birçok problem için, bir veya birçok komşuluk yapısında yerel minimum olan çözümler göreceli olarak birbirine yakındır. Üçüncü olgu, deneysel bir olgudur ve yerel bir minimumun, global minimum hakkında bize bilgi verdiğini gösterir. Bu yüzden de daha iyisi bulunana kadar yerel minimumun komşulukları sırayla organize bir şekilde incelenmelidir.

Bu sebeplerden dolayı DKA, ele alınan problemin *X* çözüm uzayındaki bir *x* başlangıç noktası etrafında ilk önce bir komşuluk yapısı serisi tanımlar. Bu seride *lmax* tane iç içe geçmiş yapı bulunmaktadır. Bir başka ifadeyle $N_i(x)$, bir *x* çözümünün *i*. komşuluğu olsun. Bu durumda komşuluk yapıları, $N_1(x) \subseteq N_2(x) \subseteq \cdots \subseteq N_{lmax}(x)$ ve $X \subseteq N_1(x) \cup N_2(x) \cup \ldots \cup N_{lmax}(x)$ sağlar. Algoritma $l = 1$ ile başlar. İlk olarak seçilen *x* başlangıç noktası silkelenir. Başka bir deyişle $N_l(x)$ komşuluğundan rasgele bir x' noktası seçilir. Seçilen bu x' noktası başlangıç noktası olacak şekilde bir yerel arama algoritması uygulanarak yerel bir x'' eniyisi elde edilir. Eğer bu x'' denilen yerel eniyi çözüm, *x* noktasından daha iyiyse, yeni *x* noktası x'' olur ve *k* değeri tekrar 1 değerini alarak ilk komşuluk olan $N_1(x)$ içerisinde tekrar silkeleme yapılır. Ancak x'' denilen yerel eniyi, *x* noktasından daha iyi bir nokta değilse aynı *x* noktası ile *l*'nin değeri 1 arttırılarak $N_{l+1}(x)$ içinde silkelenir. Bu işleme *l*, *lmax* değerini alana kadar devam edilir. Algoritma ise en başta belirlenen durma koşulları gerçekleşene kadar devam eder [48]. İkinci mat-sezgiselde kullanılan DKA'nın silkeleme fazı için 9 komşuluk yapısı belirlenmiştir (N_l, l = 1,…,9) ve komşuluklar verilen bir rotayı üç farklı şekilde değiştirebilmektedir. Her yinelemede bir rotaya ya düğüm eklenmekte, ya rotadan düğüm çıkartılmakta ya da rotada düğüm değiş tokuşu yapılmaktadır. Her komşulukta hangi düğümün seçileceği, (Eş. 4, Eş. 5, Eş. 6) ile gösterilen kısıtlı ana problemin verdiği dual bilgisiyle belirlenmektedir. Böylelikle her *i* düğümünün seçilme olasılığı, dual değişkeni olan π_i ile doğru orantılıdır. Herhangi bir komşuluk bir rotayı, $\left\lceil \frac{l}{3} \right\rceil$ düğüm kadar etkilemektedir. N_1, N_4 ve N_7 komşulukları, silkelenen rotada bulunmayan sırasıyla 1, 2 ve 3 tane düğüm ekler. N_2, N_5 ve N_8 komşulukları silkelenen rotada ziyaret edilen 1, 2 ve 3 tane düğümü rotadan çıkartır. Son olarak N_3, N_6 ve N_9 komşulukları ise silkelenen rotada ziyaret edilen 1, 2 ve 3 tane düğümü ziyaret edilmeyen 1, 2 ve 3 tane düğümle değiş tokuş eder. Silkelemede amaç, yeni negatif indirgenmiş maliyetli sütunlar elde etmek olduğundan yeni eklenecek düğümlerin seçilme olasılığı π_i değerleriyle doğru, çıkartılacak düğümlerin seçilme olasılığı ise π_i değerleriyle ters orantılıdır. Ekleme yapılırken her düğüm, zaman penceresi olurluğunu bozmayan ilk olası pozisyona eklenmiştir. Silkeleme esnasında ilk mat-sezgiselde olduğu gibi aracın sığa kısıtı ve müşterilerin zaman pencereleri dikkate alınmış, araçların azami süre kısıtları gevşetilmiştir. Silkeleme evresinin hemen arkasından yerel arama evresi geldiğinden olurlu çözüm bulma sorumluluğu yerel arama algoritmasına yüklenmiştir. Yerel arama evresinde ise 3.1 bölümünde ayrıntıları açıklanan ve sözde kodu verilen SYA algoritması kullanılmıştır.

SYA algoritmasının amacı silkeleme evresinden gelen rasgele çözümden bir yerel minimum elde etmek olduğundan dolayı bu çözümü geliştirmek için bir yerel arama algoritması kullanılması gerekir. Silkeleme evresinde rotalar arası komşuluk operatörleri kullanıldığı için bu evrede rota içi ilk gelişim tipi yerel arama algoritması kullanılmıştır. Bu durumda ilk önce ilk sırada ziyaret edilen depo haricindeki ilk düğüm mevcut pozisyonundan kaldırılmış ve zaman penceresi olurluğunu bozmayan ilk olası pozisyona eklenmiştir. Eğer bu yeni pozisyon rotanın maliyetini azaltmışsa, yerinden oynatılan düğüm yeni pozisyonunda sabitlenmiş ve aramaya rotada bir sonraki pozisyonda ziyaret edilen düğümle devam edilmiştir. Aksi takdirde ise ilk düğüm bir sonraki olası pozisyona kaydırılmıştır. Bu durum, düğüm için uygun bir pozisyon bulunamayana kadar devam ettirilmiş ve sırayla bütün ziyaret edilen düğümler için denenmiştir. Kısacası yerel arama algoritması ya rota maliyetini azaltan bir pozisyon bulunana kadar ya da rotadaki sonuncu düğüm sonuncu olası pozisyona kaydırılana kadar devam ettirilmiştir.

DKA algoritmasının hem silkeleme hem de yerel arama evrelerinde uygulanan operatörlerden sonra oluşan yeni rotaların eniyi depodan başlama zamanının bulunması gereklidir. Bunun içinse 3.1 bölümünde uygulaması açıklanan Savelsbergh [14]'in birleştirme teoreminden aynı şekilde yararlanılmıştır.

Bu mat-sezgiselin YYA algoritmasından faydalanan mat-sezgiselden en büyük farkı, seçilen meta-sezgiseli yani DKA'yı sütun türetme algoritmasının her yinelemesinde çözdüğü kısıtlı ana problemin temel olurlu çözümünde yer alan sütunlara tek tek uygulamasıdır. Bunun sebebi ise Parragh ve Schmid [49] tarafından bu uygulama ile bir zaman pencereli, merkezi olmayan istasyonlar arası rota planlama problemi için başarılı bir melez ST algoritması

geliştirilmiş olmasıdır. DKA esnasında negatif indirgenmiş maliyet açısından bilinen en iyi rotayı geliştirildiğinde tekrar N_1 komşuluğu ile başlanmış, yoksa bir sonraki komşuluk olan N_{l+1} ile devam edilmiştir. DKA ise ya N^{yeni} tane yeni negatif indirgenmiş maliyetli sütun bulunduğunda ya da sonuncu yani 9. komşuluk negatif indirgenmiş maliyetli sütun türetemediğinde sonlandırılmıştır. Bu çalışmada yapılan denemeler sonucunda N^{yeni} parametresi 2 olarak belirlenmiştir. DKA algoritmasının çalışma prensibi Şekil 3'te verilen sözde kod ile özetlenmiştir.

Girdi: π dual değişken vektörü, Ω_k^{baz}, l_{max}, N^{yeni}
1. **For** $x_r \in \Omega_k^{baz}$ **do**
2. $l \leftarrow 1$
3. $x_{en\_iyi} \leftarrow x_r$
4. $maliyet_{en\_iyi} \leftarrow maliyet(x_{en\_iyi}, \pi)$
5. $durma\ koşulu \leftarrow false$
6. **While** $\neg durma\_koşulu$ **do**
7. $x' \leftarrow$ SİLKELE(x_{en_iyi}, π)
8. $x'' \leftarrow$ SYA(x', π)
9. **If** $maliyet(x'', \pi) < maliyet_{en\_iyi}$
10. $x_{en\_iyi} \leftarrow x''$
11. $maliyet_{en\_iyi} \leftarrow maliyet(x'', \pi)$
12. $l \leftarrow 1$
13. **End**
14. **Else**
15. $l \leftarrow l + 1$
16. **End**
17. **If** $maliyet(x'', \pi) < 0$
18. $neg\_indirg\_maliyet\_sütun \leftarrow neg\_indirg\_maliyet\_sütun \cup \{x''\}$
19. **End**
20. **If** $l \geq l_{max} \vee |neg\_indirg\_maliyet\_sütun| \geq N^{yeni}$
21. $durma\ koşulu \leftarrow true$
22. **End**
23. **End**
24. **End**
25. **Return** $neg\_indirg\_maliyet\_sütun$

Şekil 3. Değişken Komşuluk Arama (DKA) Algoritması
(Variable Neighborhood Search Algorithm)

DKA kullanan ST temelli mat-sezgisel, YYA kullanan mat-sezgiselden farklı olarak KKF$_k$ problemini her çözdüğünde DKA algoritmasına, düğümlerin dual değerlerinin yanı sıra bazda yer alan sütunları aktarmaktadır. DKA algoritması bazda yer alan her sütunu başlangıç çözüm olarak kullanarak, her baz sütun için N^{yeni} tane yeni negatif indirgenmiş maliyetli sütun türetmeye çalışır. Ancak bu durum, DKA kullanan ST temelli mat-sezgiselin çok yineleme yapmasına sebep olabilmesinden dolayı ST algoritmasının durma koşulu olarak, DKA'nın negatif indirgenmiş maliyetli sütun türetememesinin yanı sıra azami bir yineleme sayısı da benimsenmiştir. Bu azami sayının değeri 1000 olarak alınmıştır.

4. BİLGİSAYIMSAL SONUÇLAR
(COMPUTATIONAL RESULTS)

Bu çalışmada ele alınan ARP'ni çözmek için geliştirilen iki sütun türetme temelli mat-sezgiselin nasıl kalitede sonuçlar verdiğini görmek ve birbirleriyle karşılaştırmak için Solomon [50]'un iyi bilinen benchmark test örnekleri kullanıldı. Bu test örneklerinin her biri 100 müşteri içermekte ve kümeli, rasgele kümeli ve rasgele olmak üzere üç sınıfa ayrılmaktadır. Kümeli test örnekleri sınıfında C101, C102, C103, C104, C105, C106, C107, C108, C109, rasgele kümeli test örnekleri sınıfında RC101, RC102, RC103, RC104, RC105, RC106, RC107, RC108 ve rasgele test örnekleri sınıfında R101, R102, R103, R104, R105, R106, R107, R108, R109, R110, R111, R112 adlı örnekler yer almaktadır. Her örnek için araç sığası, müşteri talep miktarları, müşteri ve depo koordinatları, azami süre ve de zaman pencereleri özgün örneklerde verildiği şekilde belirlenmiştir.

İlk önce geliştirilen iki yöntemin hangi kalitede sonuçlar elde ettiğini anlayabilmek için tüm örneklerin sadece ilk 10 müşterisi alınarak ayrı bir veri kümesi elde edilmiştir. Amaç, bu küçük veri kümesi üzerinde Bölüm 3'te açıklanan DAA yardımıyla KKF problemini her örnek için kesin olarak çözmek ve iki mat-sezgiselin elde ettikleri en iyi olurlu çözümün, yani en iyi üst sınırın, eniyi amaç fonksiyon değerinden ne kadar uzak olduğunu tespit etmektir. Ancak hem kesin yöntemin bu veri kümesinde makul bir zamanda sonuç verebilmesi hem de müşterisi sayısının az olmasından dolayı eniyi çözümde tek bir aracın izleyeceği tek bir rota ile tüm müşterilerin ziyaret edilmesine engel olmak için, araç sığaları ve azami kullanım süreleri özgün değerlerinin yarısına eşitlenmiştir. Böylelikle elde edilen eniyi çözümlerde birden fazla araç rotası yer almasının yanı sıra, DAA algoritması birçok düğümü olursuz hale geldiğinden dolayı budayarak hızlı bir şekilde bir sonuca ulaşabilmiştir. Mat-sezgisellerin ve kesin yöntemin çalışması esnasında çözülmesi gereken doğrusal programlama problemleri ve ikili tamsayılı doğrusal problemler için Cplex 12.7 çözücüsünden faydalanılmıştır. Algoritmalar C# dilinde kodlanmış ve tüm deneyler 64 GB belleğe sahip, on çekirdekli Intel Xeon 2.20 GHz işlemcili, Windows 10 işletim sistemiyle çalışan bir iş istasyonunda gerçekleştirilmiştir.

Tablo 2, 10 müşterili örneklerin sonuçlarını sunmaktadır. Her mat-sezgisel her bir örnek için 3 kere çalıştırılmıştır. YYAMS yinelemeli yerel arama kullanan mat-sezgiseli, DKAMS değişken komşuluk arama kullanan mat-sezgiseli temsil etmektedir. Tablodaki ilk AİZ sütunu kesin yöntemin her test örneği için harcadığı ana işlemci zamanını, diğer iki AİZ sütunu ise her mat-sezgiselin her test örneği için 3 kere çalıştırılması sonucu elde edilen ortalama ana işlemci zamanını saniye cinsinden göstermektedir. z^* sütunu ise DAA kullanan kesin yöntemin her örnekte elde ettiği eniyi amaç fonksiyon değerini vermektedir. İki mat-sezgiselin üç kere çalıştırılması sonucu elde ettikleri en iyi üst sınırın, eniyi amaç fonksiyon değerinden yüzde sapması EEİS ile, üç çalıştırmanın yüzde sapmalarının ortalaması ise OEİS ile verilmiştir. Mat-sezgisellerin bir kere çalıştırılması sonucu elde edilen üst sınır $\boldsymbol{z}$ olarak adlandırılırsa, o çalıştırılma için eniyi amaç fonksiyonu değerinden yüzde sapma, $100 \times (\boldsymbol{z} - \boldsymbol{z}^*)/\boldsymbol{z}^*$ olarak hesap edilir. Son olarak OSS, üç çalıştırmanın türettiği ortalama sütun sayısını sunmaktadır.

Tablo 2'de üç test örneği sınıfı için ayrı ayrı her yöntemin elde ettiği ortalama AİZ, mat-sezgisellerin elde ettikleri ortalama EEİS, OOİS ve OSS değerleri verilmiştir. Ayrıca bu ortalama değerler, üç yöntem için tüm örnekler dikkate alınarak genel ortalama başlığı altında tablonun son satırında gösterilmektedir. Genel ortalama değerlerine bakıldığında en başarılı yöntemin DKAMS olduğu açıkça görülmektedir. DKAMS'nin üç kere çalıştırılması sonucu elde ettiği en iyi üst sınır, kesin yöntemin elde ettiği eniyi amaç fonksiyonu

Tablo 2. 10 müşteri için kesin yöntemin ve mat-sezgisel algoritmaların elde ettiği bilgisayımsal sonuçlar (Computational results obtained by the exact method and mat-heuristic algorithms for 10 customers)

	Kesin Yöntem		YYAMS				DKAMS			
	AİZ (sn.)	*z**	AİZ (sn.)	EEİS (%)	OEİS (%)	OSS	AİZ (sn.)	EEİS (%)	OEİS (%)	OSS
C101	0,15	990,70	1,57	0,00	0,00	60,33	0,50	0,00	1,02	472,33
C102	2,88	990,70	1,31	3,60	5,49	176,67	0,63	0,11	0,24	799,00
C103	2,88	990,70	1,18	0,28	2,65	193,33	0,71	0,00	0,16	826,33
C104	33,90	990,10	0,86	3,38	5,70	220,33	0,63	0,07	0,13	854,67
C105	0,26	990,10	1,90	0,00	0,00	63,67	0,51	0,00	0,00	642,00
C106	0,19	990,70	4,41	0,00	1,12	75,00	0,50	0,00	0,00	704,00
C107	0,19	990,10	3,54	0,00	0,00	118,00	0,57	0,00	0,00	534,33
C108	0,39	989,20	1,72	0,00	0,23	174,67	0,51	0,00	0,00	556,00
C109	0,53	989,20	1,37	0,00	5,48	236,67	0,85	0,00	0,14	801,67
Ortalama	4,60		1,99	0,81	2,30	146,52	0,60	0,02	0,19	687,81
RC101	0,13	286,40	2,39	0,00	0,00	123,33	0,50	0,00	0,00	469,33
RC102	4,79	270,10	2,84	0,26	1,05	233,67	0,42	0,26	0,68	538,00
RC103	4,69	270,10	2,68	0,26	1,05	286,00	0,58	0,00	0,15	879,00
RC104	71,70	267,20	3,73	1,12	4,88	354,00	0,51	0,97	1,35	830,67
RC105	0,53	279,70	4,47	0,00	0,43	199,00	0,51	0,00	0,21	561,00
RC106	0,19	278,40	2,70	0,00	0,00	214,67	0,52	0,00	0,95	702,00
RC107	1,20	268,50	1,63	0,00	1,13	165,00	0,50	0,00	0,41	487,00
RC108	11,62	266,30	1,70	2,25	2,74	277,00	0,50	0,00	0,55	441,33
Ortalama	11,86		2,77	0,49	1,41	231,58	0,51	0,15	0,54	613,54
R101	0,12	399,40	11,38	0,00	0,00	73,00	0,45	0,00	0,15	850,67
R102	0,16	330,40	2,81	0,00	0,80	132,00	0,44	2,39	2,49	525,67
R103	0,39	330,40	3,83	0,00	0,00	151,33	0,56	0,00	1,43	1429,33
R104	2,11	298,70	2,37	0,00	0,48	201,33	0,50	0,00	0,00	449,67
R105	0,19	353,70	4,05	0,00	0,95	114,33	0,38	0,00	0,00	583,33
R106	0,41	320,60	2,81	0,00	0,45	190,33	0,51	0,00	0,45	474,67
R107	0,39	320,60	2,91	0,00	1,27	161,67	0,50	0,00	0,45	470,33
R108	3,00	298,70	2,46	0,00	0,96	200,00	0,49	0,00	0,00	459,67
R109	0,17	332,80	4,92	0,00	0,74	190,33	0,50	0,00	0,00	493,67
R110	0,14	314,40	3,94	0,00	1,11	205,00	0,44	0,00	0,00	578,33
R111	0,36	320,60	2,90	0,00	0,45	272,33	0,51	0,00	0,34	508,33
R112	0,39	298,70	4,00	1,44	1,56	229,33	0,50	0,00	0,00	615,33
Ortalama	0,65		4,03	0,12	0,73	176,75	0,48	0,20	0,44	619,92
Genel Ortalama	5,18		3,01	0,43	1,40	182,92	0,53	0,13	0,39	639,97

değerinden ortalamada %0,13 saparken, YYAMS yöntemi için bu ortalama yüzde sapma %0,43'e çıkmaktadır. Her iki yöntem için elde ettikleri üç yüzde sapma değerinin ortalaması kıyaslanırsa, YYAMS'nin ortalama %1,40, DKAMS'nin ortalama %0,39 eniyiden saptığı gözlemlenmektedir. Bu da DKAMS yönteminin daha kaliteli sonuçlar verdiğini açıkça göstermektedir. Aslında bu elde edilen ortalama yüzde sapma değerler her ne kadar DKAMS'nin daha kaliteli sonuçlar elde ettiğini gösterse de, genel olarak iki yöntem de iyi kalitede sonuçlar üretebilmektedir. YYAMS 29 test örneğinin 21 tanesinde, DKAMS ise 25 tanesinde eniyi sonucu bulmuştur. Bu durum Tablo 2'de %0,00 sapma değerleri ile gözlemlenmektedir. Ancak kıyaslamaya ana işlemci zamanı ile devam edildiğinde, iki mat-sezgiselin sırasıyla ortalamada 3,01 ve 0,53 saniye ile ortalamada 5,18 saniye ile sonuç elde edebilen kesin yöntemden daha hızlı oldukları açıktır. Ancak, DKAMS bu kadar küçük örneklerde dahi YYAMS'den ortalamada 5,71 kat daha hızlıdır. Test örnekleri sınıfları ayrı ayrı ele alındığında DKAMS kümeli ve rasgele kümeli örneklerde daha kaliteli ve verimli sonuçlar elde etmektedir. YYAMS ise sadece rasgele örneklerde bir tek üç çalıştırma sonucu elde edilen en iyi sonucun eniyi amaç fonksiyon değerinden yüzde sapması konusunda DKAMS'den daha başarılıdır. DKAMS'nin genel anlamdaki başarısını, bu kadar küçük örneklerde dâhi türetebildiği sütun sayısı ile bağdaştırmak mümkündür. YYAMS ortalamada 182,92 sütun türetebilirken, DKAMS ortalamada 639,97 sütun türetmeyi başarmıştır.

Geliştirilen iki mat-sezgiselin kesin yöntem karşısında performanslarından emin olunduktan sonra, performanslarını birbiriyle daha iyi karşılaştırabilmek için daha büyük üç veri kümesi oluşturuldu. Bunun için ilk önce özgün örneklerde bulunan ilk 25 müşteri, sonra ilk 50 müşteri ve en son olarak tüm

100 müşteriyi kullanılarak her örnek için 3 farklı veri kümesi elde edilmiştir. 25 müşterili örneklerden elde edilen sonuçlar Tablo 3'te, 50 müşterili örneklerden elde edilen sonuçlar Tablo 4'te, 100 müşterili örneklerden elde edilen sonuçlar ise Tablo 5'te sunulmuştur. Her mat-sezgisel her bir örnek için yine 3 kere çalıştırılmıştır. Tüm tablolarda $z_{\text{en iyi}}$, her iki mat-sezgisel yöntemin 3 çalıştırılması sonucu elde edilen en iyi üst sınır değerini, Yöntem bu en iyi üst sınırın hangi yöntem tarafından bulunduğunu ve OEİS, yöntemlerin her 3

Tablo 3. 25 müşteri için mat-sezgisel algoritmaların elde ettiği bilgisayımsal sonuçlar
(Computational results obtained by mat-heuristic algorithms for 25 customers)

			YYAMS			DKAMS		
	$z_{en\ iyi}$	Yöntem	AİZ (sn.)	OEİS (%)	OSS	AİZ (sn.)	OEİS (%)	OSS
C101	2466,40	YYAMS	9,31	0,58	786,00	0,70	3,81	1249,00
C102	2503,90	DKAMS	19,71	1,41	1040,33	0,69	1,56	1200,33
C103	2525,10	DKAMS	5,48	6,18	708,33	1,27	1,11	1775,33
C104	2505,50	DKAMS	8,04	1,52	1762,33	1,13	0,76	1873,00
C105	2483,00	DKAMS	22,95	1,76	795,67	0,78	2,44	1201,67
C106	2497,70	DKAMS	28,62	3,19	527,67	0,63	2,23	868,67
C107	2503,50	YYAMS	11,73	1,65	761,00	1,05	1,22	1559,67
C108	2490,80	DKAMS	11,11	2,94	813,33	0,77	1,25	1040,00
C109	2501,20	DKAMS	6,00	4,86	988,00	1,09	0,91	1089,00
Ortalama			13,66	2,68	909,19	0,90	1,70	1317,41
RC101	725,90	DKAMS	35,42	3,92	712,67	0,78	1,54	1110,67
RC102	603,60	YYAMS	44,07	4,24	1809,33	0,85	5,30	1294,00
RC103	593,10	YYAMS	35,40	0,19	2237,67	1,76	1,66	1906,33
RC104	583,90	DKAMS	19,62	13,39	1604,00	1,52	1,10	1498,00
RC105	687,50	DKAMS	20,99	4,36	855,67	0,69	3,65	1175,67
RC106	600,30	YYAMS	51,11	7,41	1888,00	0,71	0,58	915,00
RC107	550,70	YYAMS	51,54	4,45	1916,67	0,88	3,98	1147,33
RC108	550,00	YYAMS	55,79	3,62	1821,67	0,97	13,92	1411,00
Ortalama			39,24	5,20	1605,71	1,02	3,97	1307,25
R101	998,30	DKAMS	28,40	3,90	229,00	0,48	1,24	789,00
R102	904,20	YYAMS	13,35	5,01	387,00	0,51	2,46	652,33
R103	767,00	DKAMS	10,77	4,12	719,00	0,91	0,98	1658,33
R104	700,40	DKAMS	19,27	7,67	1181,33	0,97	3,83	1561,67
R105	807,60	DKAMS	34,10	5,42	468,67	0,63	2,48	1813,67
R106	752,90	DKAMS	19,66	4,74	600,33	0,90	1,93	2101,33
R107	726,70	DKAMS	9,13	12,61	550,00	0,78	0,60	1326,33
R108	683,90	DKAMS	14,54	8,32	688,33	0,95	2,15	1248,67
R109	698,70	DKAMS	29,73	1,69	1002,33	0,79	1,76	1387,00
R110	718,00	DKAMS	14,28	13,62	534,00	0,97	0,84	2022,67
R111	687,80	YYAMS	19,44	11,77	1038,00	0,76	6,53	1278,67
R112	687,70	DKAMS	18,56	6,36	788,00	0,85	2,83	1433,67
Ortalama			19,27	7,10	682,17	0,79	2,30	1439,44
Genel Ortalama			23,04	5,20	1007,39	0,89	2,57	1365,10

çalıştırma sonucunun bulunmuş en iyi üst sınırdan yüzde sapmasının ortalamasını göstermektedir. Tablo 3'te görüldüğü üzere, 25 müşterili örneklerde YYAMS 29 test örneğinin 9 tanesinde en iyi üst sınırı elde edebilirken, DKAMS ise geri kalan 20 örnekte en iyi üst sınırı elde ederek kaliteli sonuç elde etme başarısını kanıtlamıştır. Bunun haricinde YYAMS'nin 25 müşterili örneklerde 3 kere çalıştırılma sonucu elde ettiği OEİS ortalamada %5,20 iken DKAMS ortalamada %2,57 ile daha iyi bir sonuç yakalamıştır.

Ayrıca DKAMS ortalama 1365,10 sütun türetirken, YYAMS ortalamada ancak 1007,39 sütun türetebilmiştir. Böylelikle DKAMS, 25 müşterili örneklerde YYAMS'den daha kaliteli sonuç elde etme başarısını gösterebilmiştir. Son olarak YYAMS ortalamada 23,04 saniyede sonuca ulaşırken, DKAMS ortalamada 0,89 saniye ile çok daha hızlı bir algoritma olduğunu göstermektedir.

50 müşterili örneklere bakıldığında YYAMS'nin 4 örnekte, DKAMS'nin ise 25 örnekte en iyi üst sınırı elde ederek daha

Tablo 4. 50 müşteri için mat-sezgisel algoritmaların elde ettiği bilgisayımsal sonuçlar
(Computational results obtained by mat-heuristic algorithms for 50 customers)

			YYAMS			DKAMS		
	$z_{en\ iyi}$	Yöntem	AİZ (sn.)	OEİS (%)	OSS	AİZ (sn.)	OEİS (%)	OSS
C101	5169,30	DKAMS	32,27	2,24	861,33	1,17	0,90	2378,00
C102	5070,90	DKAMS	27,94	1,79	2772,33	1,59	0,97	645,33
C103	5021,50	DKAMS	19,42	5,25	1344,67	1,73	1,53	2500,67
C104	5002,30	YYAMS	21,64	3,83	1389,33	4,50	3,32	1362,00
C105	5027,90	DKAMS	36,92	2,93	1848,33	1,38	1,09	2734,00
C106	4933,00	YYAMS	40,95	3,48	1813,00	1,47	4,68	2134,67
C107	5038,60	DKAMS	32,38	3,50	2056,00	1,43	0,42	2561,00
C108	4996,10	DKAMS	29,26	4,80	1661,00	1,81	1,22	964,00
C109	5011,00	DKAMS	29,65	3,60	1878,67	2,42	0,99	1186,33
Ortalama			30,05	3,49	1736,07	1,94	1,68	1829,56
RC101	1534,90	DKAMS	39,90	21,89	1531,67	1,48	2,56	2063,00
RC102	1395,80	DKAMS	52,82	19,94	1560,67	1,71	3,36	591,67
RC103	1275,20	DKAMS	56,91	8,55	2293,00	2,67	3,17	1090,67
RC104	1078,20	YYAMS	714,84	7,69	1913,33	2,81	12,26	591,33
RC105	1400,00	DKAMS	30,29	20,54	2352,00	1,25	2,21	2548,00
RC106	1238,70	DKAMS	26,31	34,87	1419,67	1,45	6,46	1377,00
RC107	1191,70	DKAMS	42,49	12,83	1503,00	1,47	7,79	636,33
RC108	1163,90	DKAMS	38,91	8,71	1840,67	1,43	2,86	1810,67
Ortalama			125,31	16,88	1801,75	1,78	5,08	1338,58
R101	1764,10	YYAMS	59,70	1,83	672,33	1,17	1,79	2256,67
R102	1558,30	DKAMS	23,40	8,70	987,33	1,05	1,91	2661,33
R103	1373,90	DKAMS	29,89	11,49	1192,00	1,52	2,49	1294,67
R104	1177,80	DKAMS	29,78	17,89	1479,00	2,46	3,33	1043,00
R105	1494,70	DKAMS	63,64	7,29	1091,67	1,53	1,16	1763,67
R106	1345,60	DKAMS	28,41	8,93	1279,00	1,44	2,18	590,00
R107	1282,10	DKAMS	33,30	14,93	1229,67	1,74	0,61	2127,67
R108	1260,10	DKAMS	35,74	10,63	1315,00	2,50	2,47	623,00
R109	1313,00	DKAMS	30,54	13,25	1368,00	1,28	2,21	1267,67
R110	1278,20	DKAMS	29,49	8,93	1567,33	1,44	0,09	1255,67
R111	1271,60	DKAMS	40,98	9,48	1984,67	1,40	1,23	2632,67
R112	1215,50	DKAMS	28,14	13,55	1604,33	1,53	1,22	1123,67
Ortalama			36,09	10,57	1314,19	1,59	1,72	1553,31
Genel Ortalama			58,82	10,12	1579,62	1,75	2,64	1579,80

başarılı olduğu Tablo 4 yardımıyla görülmektedir. Bu örneklerde DKAMS, 25 müşterili örneklere göre daha da başarılı olmuştur. Aradaki başarı farkı OEİS değerleriyle daha belirgin ortaya çıkmaktadır. YYAMS'nin elde ettiği ortalama OEİS değeri %10,12 iken DKAMS ortalamada %2,64 ile daha başarılı olmuştur. Ancak YYAMS ve DKAMS bu örneklerde birbirine çok yakın sayıda sütun türetmiştir. YYAMS ortalamada 1579,62, DKAMS ortalamada 1579,80 sütunu, sütun havuzuna eklemiştir. İki algoritma verimlilik açısından karşılaştırıldığında ise YYAMS ortalamada 58,82 saniye zaman harcarken, DKAMS'nin aynı örneklerde ortalamada 1,75 saniye zaman harcayarak daha verimli olduğu görülmektedir.

Tablo 5'te sergilenen 100 müşterili örnek test sonuçlarında durum değişmemektedir. YYAMS sadece 3 örnekte en iyi üst sınırı veren yöntem olurken, 50 müşterili örneklere göre başarısını artıran DKAMS geri kalan 26 örnekte en iyi üst sınırı vermiştir. Bunun haricinde YYAMS ortalamada %13,91 OEİS değeri elde ederken, DKAMS için bu değer %4,95 olmaktadır. Bu iki tür karşılaştırma DKAMS yönteminin YYAMS yönteminden eniyi çözüme daha yakın, yani çok daha kaliteli çözümler elde ettiğini göstermektedir.

Özellikle örneklerin boyutu büyüdükçe DKAMS çok daha fazla sayıda en iyi çözümü elde etmekte ve ortalamada en iyi üst sınırdan daha az sapmaktadır. Bu durum test örnekleri sınıf bazında incelendiğinde de değişmektedir. DKAMS, Tablo 3, Tablo 4 ve Tablo 5 ile verilen bütün veri kümelerindeki kümeli, rasgele kümeli ve rasgele sınıfların her birinde daha kaliteli sonuç elde etme başarısını yinelemektedir. Ancak 100 müşterili örneklerde türetilen sütun sayısı göz önünde bulundurulursa YYAMS'nin ortalamada 1813,57, DKAMS'nin ortalamada 1813,76 sütun türettiği gözlemlenir. Yani iki mat-sezgiselin türettikleri sütun sayısı neredeyse birebir aynı olmasına rağmen

Tablo 5. 100 müşteri için mat-sezgisel algoritmaların elde ettiği bilgisayımsal sonuçlar
(Computational results obtained by mat-heuristic algorithms for 100 customers)

			YYAMS			DKAMS		
	$z_{\text{en iyi}}$	Yöntem	AİZ (sn.)	OEİS (%)	OSS	AİZ (sn.)	OEİS (%)	OSS
C101	10373,20	DKAMS	50,65	5,66	1218,67	3,40	2,39	879,67
C102	10426,20	YYAMS	90,62	0,61	2169,00	3,00	2,87	1232,67
C103	10386,90	DKAMS	53,54	1,37	2006,33	5,39	1,57	1393,33
C104	10214,90	YYAMS	62,39	3,75	2133,00	28,74	3,38	1756,00
C105	10369,60	DKAMS	62,38	5,06	1830,33	4,16	0,19	2464,00
C106	10376,50	DKAMS	60,42	4,14	1724,33	2,33	3,24	1153,33
C107	10270,60	DKAMS	69,19	4,01	2039,67	3,50	2,19	1927,33
C108	10343,70	DKAMS	69,99	2,25	1675,67	3,00	1,67	1576,00
C109	10330,20	DKAMS	49,94	5,75	1982,00	4,84	0,56	2764,33
Ortalama			63,24	3,62	1864,33	6,49	2,01	1682,96
RC101	2866,30	DKAMS	53,80	23,78	1179,00	3,61	1,06	1559,33
RC102	2697,50	DKAMS	53,37	19,36	1401,33	7,35	1,66	1936,33
RC103	2628,70	DKAMS	64,33	14,22	2150,00	10,34	1,11	1563,00
RC104	2748,40	DKAMS	68,08	4,36	1765,00	12,17	2,19	836,67
RC105	2751,30	DKAMS	49,30	18,39	1399,00	4,44	3,42	2180,00
RC106	1765,00	YYAMS	54,26	55,45	1729,33	5,57	56,21	2179,67
RC107	2379,50	DKAMS	57,72	31,47	2269,67	5,38	7,46	2606,67
RC108	2461,10	DKAMS	58,63	28,43	2040,00	6,18	5,35	2138,00
Ortalama			57,44	24,43	1741,67	6,88	9,81	1874,96
R101	3063,40	DKAMS	101,77	2,86	1430,00	2,31	2,14	2606,33
R102	2756,10	DKAMS	63,44	12,23	1220,67	2,94	2,30	1532,00
R103	2423,90	DKAMS	75,06	17,40	1863,67	6,52	1,26	2276,00
R104	2334,00	DKAMS	55,28	16,45	1955,00	13,17	7,97	2260,33
R105	2550,70	DKAMS	116,24	15,13	1677,00	2,67	6,22	756,67
R106	2415,30	DKAMS	74,32	17,31	1874,67	6,18	1,57	1687,67
R107	2370,10	DKAMS	72,21	19,16	1670,67	11,84	6,46	2186,33
R108	2480,30	DKAMS	79,48	4,79	2483,00	14,08	2,08	1631,33
R109	2398,90	DKAMS	83,38	15,30	2272,33	4,75	3,98	2005,00
R110	2354,60	DKAMS	79,29	13,99	2235,00	5,38	3,99	1422,67
R111	2319,60	DKAMS	92,73	18,24	2126,33	6,32	3,64	2685,00
R112	2331,10	DKAMS	59,56	22,58	1073,00	6,42	5,26	1403,33
Ortalama			67,81	13,92	1812,89	6,75	5,01	1811,51
Genel Ortalama			68,32	13,91	1813,57	6,76	4,95	1813,76

DKAMS büyük örneklerde YYAMS'den çok daha kaliteli sonuçlar elde edebilmiştir. Bu da aslında DKAMS'nin türettiği sütunların YYAMS'nin türettiği sütunlardan çok daha kaliteli olduğuna işarettir. Son olarak tüm örneklerde DKAMS'nin en hızlı yani en verimli algoritma olduğu anlaşılmaktadır. 100 müşterili örneklerde, 25 ve 50 müşterili örneklere göre sonuç değişmemiştir. DKAMS bu en büyük örneklerde ortalamada 6,76 saniyede çözüme ulaşmayı başarırken, YYAMS ortalamada 68,32 saniye ile DKAMS'nin çok gerisinde kalmıştır. Dolayısıyla tüm büyüklükteki örnekler incelendiğinde DKAMS, YYAMS'nin yaklaşık 16 katı kadar hızlı sonuca ulaşmaktadır. Sonuç olarak iki mat-sezgisel ana işlemci zamanları, elde ettikleri üst sınırların kalitesi ve türettikleri sütun sayıları açısından kıyaslandığında DKAMS, çok daha kaliteli ve verimli sonuçlar elde ederek yüksek başarım göstermiştir.

5. SİMGELER (SYMBOLS)

a_i : i müşterisinin zaman penceresinin başlangıcı
A : ayrıt kümesi
b_i : i müşterisinin zaman penceresinin sonu
B_i : i. kısmi yolun geri zaman payı, i = 1, 2
c_r, : r rotasının süresi
d_i : i müşterisinin talep miktarı
F : iki kısmi yolun birleştirilmesiyle elde edilen yolun ileri zaman payı
F_i :i. kısmi yolun yolun ileri zaman payı, i = 1, 2
g_{ir}, :i müşterisinin r rotasında ziyaret edilip edilmediğini belirten gösterge parametre
$G(V, A)$ ayrıt kümesi A, düğüm kümesi olan V çizge
KKF : küme kapsama formülasyonu
KKF_k :sütun türetme algoritmasinin k. yinelemesinde çözülen küme kapsama formülasyonu

$kmax$: değişken komşuluk arama algoritmasında kullanılan komşuluk yapısı sayısı
N : müşteri kümesi
$N_i(x)$: bir x çözümünün i. Komşuluğu
N^{yeni} : değişken komşuluk arama algoritmasının silkeleme evresinden türetilen azami negatif indirgenmiş maliyetli sütun sayısı
S : araçların sığası
s_i : i müşterisine hizmet vermek için harcanan süre
S : araçların azami süre miktarları
t_{ij} : i düğümünden j düğümüne seyahat etmek için gereken süre
$\bar{t}_{ij}$: i düğümünden j düğümüne seyahat etmek için gereken süre ile i düğümündeki hizmet süresinin toplamı
T_i : i düğümünü ziyaret eden bir aracın hizmete başladığı zaman
V : düğüm kümesi
w_i : i. kısmi yoldaki bekleme süresi, i = 1, 2
W : iki kısmi yolun birleştirilmesiyle elde edilen yoldaki bekleme süresi
x_r : r rotasının eniyi çözümde yer alıp almadığını gösteren ikili karar değişkeni
γ : salınımlı yerel arama algoritmasında olursuz çözümleri cezalandırmak için kullanılan ceza parametresi
λ : iki kısmi yol birleştikten sonra V_{i+1} düğümündeki yeni hizmete başlama zamanı olan
δ salınımlı yerel arama algoritmasında olurlu çözüm bulamadığında γ ceza parametresinin çarpıldığı katsayı
π_i : i. düğümün dual değişkeni
Ω : küme kapsama formülasyonunda olurlu tüm rotaları içeren küme
Ω_k : sütun türetme algoritmasının k. yinelemesinde Ω'nin alt kümesi

6. SONUÇLAR (CONCLUSIONS)

Bu çalışmada, tek depoya ve tektürel, sığalı araç filosuna sahip bir dağıtım şirketi tarafından belirli bir müşteri kümesine hizmet verilen bir zaman pencereli araç rotalama problemi ele alınmıştır. Dağıtım şirketinin herhangi bir aracı, bir müşteriyi ziyaret ettiğinde müşterinin belirtmiş olduğu talep miktarı kadar teslimat yapar. Dağıtım şirketi teslimat yaparken her müşterinin kendi belirlediği zaman penceresi kısıtına uymak durumdadır. Diğer araç rotalama problemlerinden farklı olarak araçlar istenilen herhangi bir zamanda depodan ayrılarak kendilerine atanan rotada müşterilere hizmet vermeye başlayabilirler. Ancak her araç, sürücülerin vardiya sürelerine göre belirlenmiş azami bir süre için kullanılabilmektedir ve her aracın atandığı rota depoda başlar, depoda biter. Herhangi bir aracın ulaşım gideri klasik araç rotalama problemlerinden farklı olarak kendisine atanan rotanın süresine bağlı bir fonksiyondur. Dağıtım şirketinin amacı, toplam ulaşım giderini enküçükleyen araç rotalarını ve bir rotaya atanan her aracın depodan çıktığı zamanı belirlemektir.

Özetlenen bu araç rotalama problemini çözebilmek için iki başarılı sütun türetme temelli mat-sezgisel geliştirilmiştir. Buradaki amaç meta-sezgisellerle karşılaştırıldığında doğruluk derecesi açısından daha iyi sonuçlar elde etmek ve kesin yöntemler/matematiksel programlama teknikleriyle karşılaştırıldığında ise verimlilik açısından daha iyi sonuçlar elde etmektir. İki mat-sezgisel sütunları türetirken yararlandıkları meta-sezgiseller açısından birbirinden ayrılmaktadır. Birinci mat-sezgisel, meta-sezgisel olarak yinelemeli yerel arama kullanırken ikinci mat-sezgisel değişken komşuluk arama kullanmaktadır. İlk önce bu iki mat-sezgiselin hangi kalitede çözümler elde ettiğini anlamak için derinliğine arama algoritması yardımıyla küme kapsama formülasyonun ihtiyaç duyduğu tüm olurlu rotaları türetebilen kesin bir yöntem geliştirilmiştir. Daha sonra iki mat-sezgiselin performansları, küçük veri kümeleri üzerinde kesin yöntemle karşılaştırılmış ve iki mat-sezgiselin kaliteli ve verimli yöntemler olduğu gözlemlenmiştir. Bu mat-sezgiseller arasından hem verimlilik hem doğruluk derecesi açısından en başarılı algoritma, değişken komşuluk arama algoritma meta-sezgiselinden faydalanan sütun türetme temelli mat-sezgisel olmuştur. Özellikle toplam 87 büyük örnek üzerinde yapılan bilgisayımsal deneylerle ortalamada 3,13 saniyede çalışarak çok hızlı bir algoritma olduğunu kanıtlamıştır. Böylelikle ele alınan bu problem için kesin bir yöntem geliştirilebildiği zaman, kesin yöntemin makul bir sürede bir çözüm sunamadığı durumlar için hızlı çözüm elde etmek mümkün olmuştur.

TEŞEKKÜR (ACKNOWLEDGEMENT)

Bu çalışma TÜBİTAK BİDEB 2232 Yurda Dönüş Araştırma Burs Programı kapsamında 116C013 numaralı proje ile desteklenmiştir.

KAYNAKLAR (REFERENCES)

1. Toth P. ve Vigo D., Vehicle Routing: Problems, Methods, and Applications, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 2014.
2. Dantzig, G.B. ve Ramser, J.H., The truck dispatching problem, Management Science, 6 (1), 80-91, 1959.
3. De Bruecker P., Beliën J., De Boeck L., De Jaeger S., Demeulemeester E., A model enhancement approach for optimizing the integrated shift scheduling and vehicle routing problem in waste collection, European Journal of Operational Research, 266 (1), 278-290, 2018.
4. Männel D. ve Bortfeldt A., Solving the pickup and delivery problem with three-dimensional loading constraints and reloading ban, European Journal of Operational Research, 264 (1), 2018.
5. de Souza Lima F.M., Pereira D.S.D., da Conceição S.V., de Camargo R.S., A multi-objective capacitated rural school bus routing problem with heterogeneous fleet and mixed loads, 4OR A Quarterly Journal of Operations Research, 15 (4), 359-386, 2017.
6. Hosseini M.B., Dehghanian F., Salari M., Selected capacitated location-routing problem with incentive-dependent returns in designing used products collection

network, European Journal of Operational Research, 272 (2), 2019.
7. Ceselli A., Righini G., Salani M., A column generation algorithm for a rich vehicle routing problem, Transportation Science, 43 (1), 56-69, 2009.
8. Desaulniers G., Madsen O.B., Ropke S., The Vehicle Routing Problem With Time Windows, Vehicle Routing: Problems, Methods, and Applications, Editörler: Toth P. ve Vigo D., Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 119-159, 2014.
9. François V., Neighborhood Search Algorithms for Multi-Trip Vehicle Routing, Doktora Tezi, Liège Üniversitesi, HEC Liège, Belçika, 2018.
10. Hsu C.-I, Hung S.-F., Li H.-C., Vehicle routing problem with time-windows for perishable food delivery, Journal of Food Engineering, 80 (2), 465-475, 2007.
11. Liberatore F., Righini G., Salani M.A., A column generation algorithm for the vehicle routing problem with soft time windows, 4OR A Quarterly Journal of Operations Research, 9 (1), 49-82, 2011.
12. Bettinelli A., Ceselli A., Righini G., A branch-and-cut-and-price algorithm for the multi-depot heterogenous vehicle routing problem with time Windows, Transportation Research Part C, 19 (5), 723-740, 2011.
13. Dabia S., Ropke S., van Woensel T., De Kok T., Branch and price for the time-dependent vehicle routing problem with time windows, Transportation Science, 47 (3), 380-396, 2013.
14. Savelsbergh M.W.P., The vehicle routing problem with time windows: Minimizing route duration, ORSA Journal On Computing, 4 (2), 146-154, 1992.
15. Ioachim I, Gélinas S., Soumis F., Desrosiers J., A dynamic programming algorithm for the shortest path problem with time windows and linear node costs, Networks, 31 (3), 193-204, 1998.
16. Desaulniers G., Villeneuve D., The shortest path problem with time windows and linear waiting costs, Transportation Science, 34 (3), 312-319, 2000.
17. Archetti C., Speranza M.G., A survey on matheuristics for routing problems, EURO Journal on Computational Optimization, 2 (4), 223-246, 2014.
18. Desaulniers G., Desrosiers J., Ioachim I., Solomon M. M., Soumis F., Villeneuve D., A unified framework for deterministic time constrained vehicle routing and crew scheduling problems, Fleet Management and Logistics, Editörler: Crainic T.G. ve Laporte G., Kluwer, Boston, 57-93, 1998.
19. Desrosiers J. ve Lübbecke M.E., A primer in column generation, Column Generation, Editörler: Desaulniers G., Desrosiers J., Solomon M.M., Springer, New York, 1-32, 2005.
20. Lübbecke M.E., Desrosiers J., Selected topics in column generation, Operations Research, 53 (6), 1007-1023, 2005.
21. Gutiérrez-Jarpa G., Desaulniers G., Laporte G., Marianov V., A branch-and-price algorithm for vehicle routing problem with deliveries, selective pickups and time windows, European Journal of Operational Research, 206 (2), 341-349, 2010.
22. Salani M. ve Vacca I., Branch and price for the vehicle routing problem with discrete split deliveries and time windows, European Journal of Operational Research, 213 (3), 470-477, 2011.
23. Desaulniers G., Lessard F., Hadjar A., Tabu search, partial elementarity, and generalized *k*-path inequalities for the vehicle routing problem with time windows, 42 (3), 387-404, 2008.
24. Feillet D., A tutorial on column generation and branch-and-price for vehicle routing problems, 4OR A Quarterly Journal of Operations Research, 8 (4), 407-424, 2010.
25. Azi N., Gendreau M., Potvin J.-Y., An exact algorithm for a vehicle routing problem with time windows and multiple use of vehicles, European Journal of Operational Research,, 202 (3), 756-763, 2010.
26. Boschetti M., Maniezzo V., Roffilli M., Röhler A.B., Matheuristics: Optimization, simulation and control, Hybrid Metaheuristics, Lecture Notes in Computer Science 6373, Editörler: Blesa M., Blum C., Raidl G., Roli A., Sampels M., Springer Berlin Heidelberg, 171-177, 2010.
27. Danna E. ve Pape C., Branch-and-price heuristics: a case study on the vehicle routing problem with time windows, Column Generation, Editörler: Desaulniers G., Desrosiers J., Solomon M.M., Springer, ABD, 99-129, 2005.
28. Archetti C., Bianchessi N., Speranza M.G., A column generation approach for the split delivery vehicle routing problem, Networks, 58 (4), 241-254, 2011.
29. Archetti C., Speranza M.G., Vigo D., Vehicle Routing Problems with Profits, Vehicle Routing: Problems, Methods, and Applications, Editör: Toth P. ve Vigo D., Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 273-298, 2014.
30. Archetti C., Bianchessi N., Speranza M.G., Optimal solutions for routing problems with profits, Discrete Applied Mathematics, 161 (4-5), 547-557, 2013.
31. Archetti C., Bianchessi N., Speranza M.G., The capacitated team orienteering problem with incomplete service, Optimization Letters, 7 (7), 1405-1417, 2013.
32. Archetti C., Bianchessi N., Hertz A., Speranza M.G., Incomplete service and split deliveries in a routing problem with profits, Networks, 63 (2), 135-145, 2014.
33. Archetti C., Bianchessi N., Hertz A., Speranza M.G., The split delivery capacitated team orienteering problem, Networks, 63 (1), 16-33, 2014.
34. Aghezzaf E-H, Raa B., Landeghem H.V., Modeling inventory routing problems in supply chains of high consumption products, European Journal of Operational Research, 169 (3), 1048-1063, 2006.
35. Raa B., Aghezzaf E.-H., Designing distribution patterns for long-term inventory routing with constant demand rates, International Journal of Production Economics, 112 (1), 255-263, 2008.
36. Raa B., Aghezzaf E.-H., A practical solution approach for the cyclic inventory routing problem, European

Journal of Operational Research, 192 (2), 429-441, 2009.

37. Parragh S. N., Cordeau J.F., Doerner K. F., Hartl R. F., Models and algorithms for the heterogeneous dial-a-ride problem with driver-related constraints, OR Spectrum, 34 (3), 593-633, 2012.

38. Parragh S. N. ve Schmid V., Hybrid column generation and large neighborhood search for the dial-a-ride problem, Computers and Operations Research, 40 (1), 490-497, 2013.

39. Cacchiani V., Hemmelmayr V. C., Tricoire F., A set covering based heuristic algorithm for the periodic vehicle routing problem, Discrete Applied Mathematics, 163 (1), 53-64, 2014.

40. Dasgupta S., Papadimitriou C., Vazirani U., Algorithms, McGraw-Hill Education, New York, 2006.

41. Russell S. ve Norvig P., Artificial Intelligence: A Modern Approach, Prentice Hall, Upper Saddle River, New Jersey, 2010.

42. Cormen T.H., Leiserson C.E., Rivest R.L., Stein C., Introduction to Algorithms, The MIT Press, Cambridge, Massahusetts, London, England, 2009.

43. Dror M., Note on the complexity of the shortest path models for column generation in VRPTW, Operations Research, 42 (5), 977-978, 1994.

44. Raidl G. R., Puchinger J., Combining (Integer) Linear Programming Techniques and Metaheuristics for Combinatorial Optimization, Hybrid Metaheuristics: An Emerging Approach to Optimization, Editörler: Blum C., Blesa M., Roli A., Sampels M., Springer Berlin Heidelberg, 31-62, 2008.

45. Talbi E.-G., Metaheuristics: From Design to Implementation, New Jersey, Wiley, 2009.

46. Glover F., Kochenberger G. A., Handbook of Metaheuristics, Dordrecht, Kluwer Academic Publishers, 2003.

47. Cuervo D.P., Goos P., Sörensen K., Arráiz E., An iterated local search algorithm fort he vehicle routing problem with backhauls, 237 (2), 454-464, 2014.

48. Hansen P., Mladenovic N., Perez J.A.M., Variable neighborhood search: methods and applications, Annals of Operations Research, 175 (1), 367-407, 2010.

49. Parragh S.N., Schmid V., Hybrid column generation and large neighborhood search for the dial-a-ride problem, Computers and Operations Research, 40 (1), 490-497, 2013.

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