BAŞKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

OTEL SEÇİMLİ GEZGİN SATICI PROBLEMİ İÇİN YENİ MATEMATİKSEL MODELLER

CEMAL AYKUT GENCEL

YÜKSEK LİSANS TEZİ

2019

OTEL SEÇİMLİ GEZGİN SATICI PROBLEMİ İÇİN YENİ MATEMATİKSEL MODELLER

NEW MATHEMATICAL MODELS FOR THE TRAVELING SALESMAN PROBLEM WITH HOTEL SELECTION

CEMAL AYKUT GENCEL

Başkent Üniversitesi
Lisansüstü Eğitim Öğretim ve Sınav Yönetmeliğinin
ENDÜSTRİ Mühendisliği Anabilim Dalı İçin Öngördüğü
YÜKSEK LİSANS TEZİ
olarak hazırlanmıştır.

2019

“Otel Seçimli Gezgin Satıcı Problemi İçin Yeni Matematiksel Modeller” başlıklı bu çalışma, jürimiz tarafından, 31/01/2019 tarihinde, **ENDÜSTRİ MÜHENDİSLİĞİ ANA BİLİM DALI’nda YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Başkan : Prof.Dr.Fulya ALTIPARMAK

Üye (Danışman) : Dr.Öğr.Üyesi Barış KEÇECİ

Üye : Dr.Öğr.Üyesi Tusan DERYA

ONAY

/ 02 /2019

Prof.Dr.Ömer Faruk ELALDI

Fen Bilimleri Enstitüsü Müdürü

BAŞKENT ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ
YÜKSEK LİSANS TEZ ÇALIŞMASI ORİJİNALLİK RAPORU

Tarih: 05 / 02 / 2019

Öğrencinin Adı, Soyadı : Cemal Aykut GENCEL

Öğrencinin Numarası : 21610318

Anabilim Dalı : Endüstri Mühendisliği Ana Bilim Dalı

Programı : Endüstri Mühendisliği Tezli Yüksek Lisans Programı

Danışmanın Unvanı/Adı, Soyadı : Dr. Öğr. Üyesi Barış KEÇECİ

Tez Başlığı : Otel Seçimli Gezgin Satıcı Problemi İçin Yeni Matematiksel Modeller

Yukarıda başlığı belirtilen Yüksek Lisans tez çalışmamın; Giriş, Ana Bölümler ve Sonuç Bölümünden oluşan, toplam 40 sayfalık kısmına ilişkin, 05 / 02 / 2019 tarihinde tez danışmanım tarafından Turnitin adlı intihal tespit programından aşağıda belirtilen filtrelemeler uygulanarak alınmış olan orijinallik raporuna göre, tezimin benzerlik oranı % 10'dur.

Uygulanan filtrelemeler:

1. Kaynakça hariç
2. Alıntılar hariç
3. Beş (5) kelimeden daha az örtüşme içeren metin kısımları hariç

"Başkent Üniversitesi Enstitüleri Tez Çalışması Orijinallik Raporu Alınması ve Kullanılması Usul ve Esaslarını" inceledim ve bu uygulama esaslarında belirtilen azami benzerlik oranlarına tez çalışmamın herhangi bir intihal içermediğini; aksinin tespit edileceği muhtemel durumda doğabilecek her türlü hukuki sorumluluğu kabul ettiğimi ve yukarıda vermiş olduğum bilgilerin doğru olduğunu beyan ederim.

Öğrenci İmzası:

Onay

05 / 02 / 2019

Dr. Öğr. Üyesi Barış KEÇECİ

TEŞEKKÜR

Sayın Hocam Dr.Öğr.Üyesi Barış KEÇECİ'ye tez süresince bana bilgi ve deneyimleri ile yol gösterdiği ve büyük destek olduğu için,

Sayın Hocam Dr.Öğr.Üyesi Tusan DERYA'ya tez süresinde bana sunduğu katkılar için, teşekkürlerimi sunarım.

ÖZ

OTEL SEÇİMLİ GEZGİN SATICI PROBLEMİ İÇİN YENİ MATEMATİKSEL MODELLER

Cemal Aykut GENCEL

Başkent Üniversitesi Fen Bilimleri Enstitüsü

Endüstri Mühendisliği Anabilim Dalı

Bu çalışma kapsamında Gezgin Satıcı Problemi (GSP)'nin bir çeşidi olan Otel Seçimli Gezgin Satıcı Problemi (OSGSP) ele alınmıştır. Klasik GSP başlangıç noktası ve bitiş noktası aynı olmak üzere tüm müşterilere en kısa yol veya zaman içinde uğramayı hedefler. Ele alınan problemin farkı ise müşterilerin gün içinde belirlenen bir zaman kısıtına göre ziyaret edilmesi ve her günün bir otelde sonlanmasıdır. Tüm müşterilere uğrama zorunluluğu bulunmaktadır. OSGSP'nin amacı toplam gün sayısını ve seyahat uzunluğunu enküçükleyen rotanın belirlenmesidir. Süre kısıtları dahilinde gün içerisinde yapılan seyahate gezi ve bu gezilerin birleşiminden oluşan çevrime ise tur adı verilmektedir. Her gezi kendisinden önceki gezinin tamamlandığı otelden başlar. Bu tez sonucunda literatürde yer alan matematiksel modeller dışında üç yeni model önerilmiştir. Önerilen yeni modeller ile literatürde yer alan mevcut iki modelin performans karşılaştırması yapılmıştır.

ANAHTAR SÖZCÜKLER: Gezgin Satıcı Problemi, Otel Seçimi, Matematiksel Modeller, Benchmark Problemleri, Optimizasyon.

Danışman: Dr.Öğr.Üyesi Barış KEÇECİ, Başkent Üniversitesi, Endüstri Mühendisliği Bölümü

ABSTRACT

NEW MATHEMATICAL MODELS FOR THE TRAVELING SALESMAN PROBLEM WITH HOTEL SELECTION

Cemal Aykut GENCEL
Başkent University, Instutute of Science and Engineering
Department of Industrial Engineering

In this study Traveling Salesman Problem with Hotel Selection (TSPHS) is discussed which is a variant of Traveling Salesman Problem (TSP). The TSP aims to visit all customers in the shortest way or time, with the same start point and end point. The difference of the TSPHS is that the customers are visited according to a time constraint determined during the day and each trip in a day ends in a hotel. All customers must be visited. The purpose of TSPHS is to determine the total number of days and the length of the trip that minimizes travel length. In terms of time constraints, the tour is a combination of daily trips. Each trip starts from the hotel where the previous trip was completed. As a result of this study, three new models were proposed. The performance of the existing two models in the literature was compared with the proposed new models.

KEYWORDS: Traveling Salesman Problem, Hotel Selection, Mathematical Models, Benchmark Problems, Optimization.

Instructor: Dr.Öğr.Üyesi Barış KEÇECİ, Başkent University, Industrial Engineering Department.

İÇİNDEKİLER LİSTESİ

Sayfa

TABLOLAR LİSTESİ

ŞEKİLLER LİSTESİ

KISALTMALAR

CSVG	Castro,Sörensen,Vansteenwegen,Goos
FGG	The Fox, Gavish and Graves
GG	Gavish and Graves
GSP	Gezgin Satıcı Problemi
OPL	Optimization Programming Language
OSGSP	Otel Seçimli Gezgin Satıcı Problemi
TSPHS	Travelling Salesman Problem with Hotes Selection
VSS	Vansteenwegen,Souffriau,Sörensen

1 GİRİŞ

Dünyada insanlara hizmet veren bir çok sektör bulunmaktadır. Günümüzde gıda, inşaat, otomotiv, sağlık başta olmak üzere bir çok sektör, üretim veya hizmet olmak üzere iki ana sektör başlığı altında gruplanabilmektedir. Gerek üretim gerekse hizmet sektöründe bulunan bir çok işletme, kurum veya kuruluş farklı nedenlerle sürekli iyileştirmeyi sağlamak için büyük yatırımlar yapmaktadır. Bu nedenlerin başında rekabet piyasasında bulunmak önemli bir rol aldığı gibi aynı zamanda çağın getirdiği yeniliklere ayak uydurmak ve insanlara daha iyi hizmet vererek refah seviyesini sürekli yükseltmeyi amaçlamak da yer almaktadır. Bu anlamda yapılan çalışmaların odak noktası, yıllar geçse de değişmeyen maliyet, zaman ve kalite olmuştur.

Araştırmacılar sektörün ihtiyacını görerek bir çok alanda çalışma yapmış ve yapmaya devam etmektedir. Bunlardan bir tanesi olan Gezgin Satıcı Problemi (GSP), çok sayıda noktaya en kısa sürede veya en kısa mesafede uğramayı amaçlayan bir problemdir.

GSP'nin özelleşmiş bir hali olan Otel Seçimli Gezgin Satıcı Problemi (OSGSP) bu çalışmaya konu olmuştur. OSGSP'nin ortaya çıkışında, bir satıcının günlük çalışma süresi içerisinde tüm müşterileri tek seferde ziyaret etmesinin mümkün olmayışı rol almıştır.

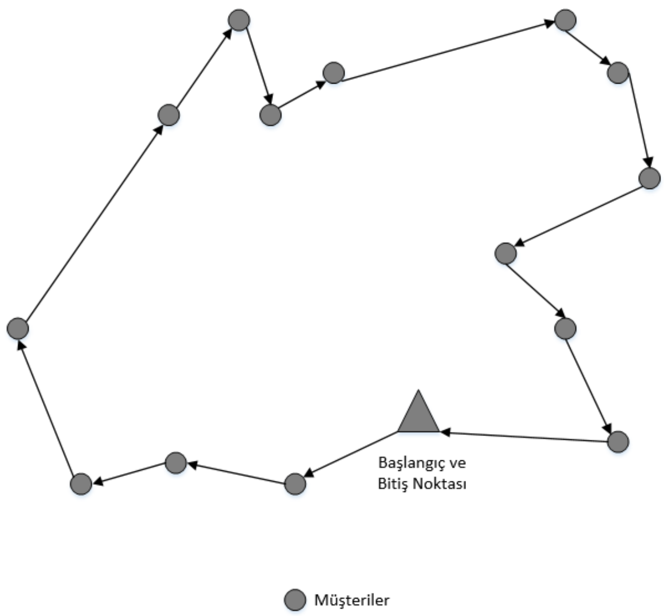
Bu çalışmada, OSGSP'nin çözümü için literatürden referans alınan matematiksel modellere alternatif modeller geliştirilmiş ve tüm modellerin aynı koşullar altında, aynı problem verileri üzerindeki performansları karşılaştırılmıştır. İlk bölümde problemin tanımı ve örnekler ile anlatımı, ikinci bölümde bu konu ile ilgili literatürdeki çalışmalar, takip eden bölümlerde ise modellerin tanıtılması, çözüm ve sonuçlar yer almaktadır.

2 GEZGİN SATICI PROBLEMİ ve LİTERATÜR ARAŞTIRMASI

2.1 Gezgin Satıcı Problemi

GSP'de bir başlangıç noktasından başlanarak hedeflenen noktalara uğranır ve tekrardan başlangıç noktasına ulaşıldığında rota tamamlanır, bu işlem "Tur" olarak adlandırılır. Tur içerisinde hedeflenen tüm noktalara uğrama zorunluluğu vardır ve bir noktaya yalnızca bir defa uğranabilir. Problemde noktalar arası mesafe veya zaman parametre olarak tanımlı olmalıdır.

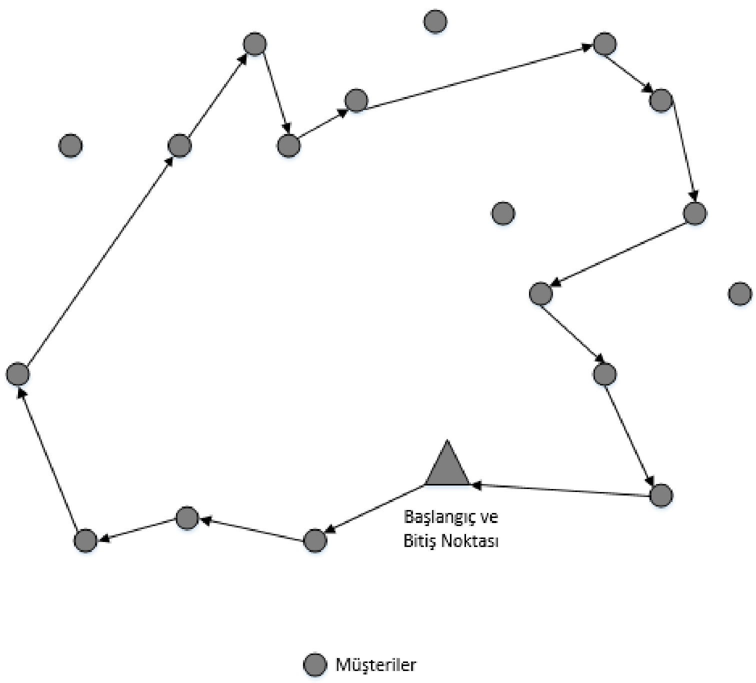
Şekil-1'de örnek bir GSP çözümü görsel olarak gösterilmiştir.



Şekil 1 Örnek Bir GSP Çözümü

Standart GSP mesafe veya zamanı enküçüklemeyi amaçlarken, farklı yaklaşımlarla bu problem çeşitlendirilmiştir. Örneğin GSP'deki uğrak noktaların getirileri de göz önünde bulundurulduğunda Seçici Gezgin Satıcı Problemi (SGSP) adı altında yeni bir problem çeşidi ortaya çıkmıştır. Bu problemin GSP'den en önemli farkı, belirlenen bir tur süresinin olması ve bu süre içerisinde başlangıç noktasına geri dönme zorunluluğu olmasıdır. Bu aşamada karar verici, noktalar arasında bir seçim yapmalı ve belirlenen zaman içerisinde en fazla noktaya nasıl uğramalıyım sorusuna cevap aramalıdır. Eğer noktaların farklı getirileri söz konusu ise bu sefer de belirlenen süre içerisinde, turu en fazla toplam getiri ile tamamlamak amaçlanmalıdır.

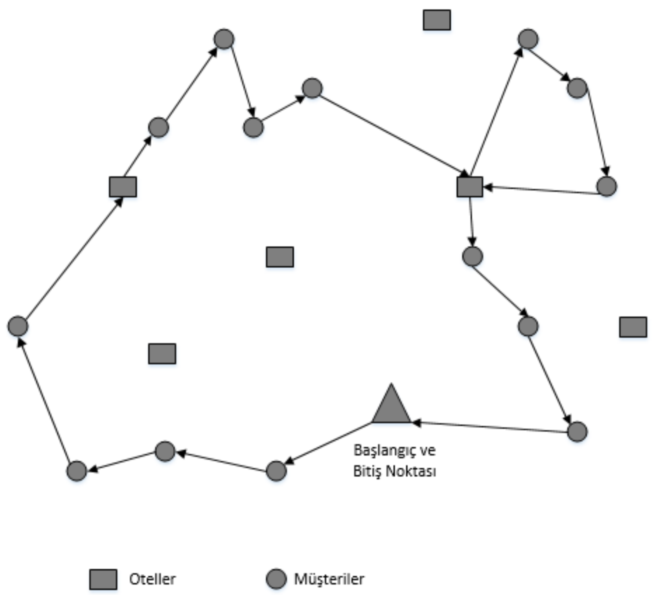
Şekil-2'de örnek bir SGSP çözümü görsel olarak gösterilmiştir.



Şekil 2 Örnek Bir SGSP Çözümü

Bu çalışmanın konusu olan OSGSP'de, klasik GSP'deki gibi tüm müşterilere uğrama zorunluluğu vardır. Dolayısıyla turu en az zaman veya maliyetle tamamlamayı amaçlar. Ancak tur süresince yapılan müşteri ziyaretleri, dinlenme zamanından dolayı kesikli şekilde yürütülmektedir. Bu çalışmada, iki dinlenme zamanı arasında müşteri ziyareti yapılarak aralıksız olarak geçen süre "Gezi" olarak adlandırılır. OSGSP'de gezi için süre kısıtı bulunmaktadır ve bu süre kısıtı içerisinde gezinin sonlanması gerekmektedir. Eğer bir gezi otelde sonlandıysa, bir sonraki gezi aynı otelden başlar. Gezilerin birleşiminden oluşan tur, GSP'de olduğu gibi başlangıç noktasında tamamlanır.

Şekil-3’te örnek bir OSGSP çözümü görsel olarak gösterilmiştir.



Şekil 3 Örnek Bir OSGSP Çözümü

OSGSP ile gerçek yaşam uygulamalarında birçok yerde karşılaşılmaktadır.

- Elektrikli araç kullanan bir satış personelinin araç menzil kapasitesi içerisinde gidebileceği müşteri sayısı sınırlıdır. Aracı şarj etmek için belirli istasyonlarda günlük gezilerin sonlandırılması gerekmektedir. Tüm satış noktaları tamamlandığında günlük gezilerin birleşiminden oluşan tur tamamlanacaktır [1].

- Uzun mesafeli taşıma yapan kamyon sürücüleri için günlük çalışma süreleri yasal olarak sınırlandırılmıştır. Bu süreler aşıldığında para cezaları uygulanabilmektedir. Bir kamyon şoförü uzun süreli bir tur yapmaktaysa sık sık park etme ve uygun bir park yerine ulaşma zorluğu ile karşılaşmaktadır. Gece uygun olmayan bir yerde konaklanması otopark para cezası riski oluşturmaktadır. Modern uygulamalarda coğrafi bilgi sistemleri ile hem konaklama hem de sevkiyat yapılacak noktaları içeren turları belirleyen yöntemler kullanılmaktadır [2].

- Sokaklarda yer alan trafik levhalarının coğrafi bir veritabanında tutulmasını sağlayan bir şirket düşünülürse her sokağın ziyaret edildiği bir tur planlaması yapılmalıdır. Verinin toplanması için orta ölçekli bir bölgede bile birkaç gün süren bir tur gerekmektedir. Bu durumda veri toplanan her gün sonunda bir konaklama noktası belirlenmelidir [3].

- Bir bölgeye yapılan uzun süreli turist gezilerinde her aktiviteye katılım sağlamak isteyen bir turist örneği de verilebilir. Her aktiviteye katılım olmadığı durumda Oryantiring Problemi olarak ele alınmaktadır [4].

- Yaya olarak taşıdığı belge miktarını azaltmak isteyen bir postacı, turunu alt dağıtım rotaları oluşturarak tamamlayabilir. Önceden araba veya bisiklet ile belli noktalara daha hafif ve bölünmüş çantalar yerleştirilirse postacının bu noktalara geliş rotası optimize edilebilir. Böylece her toplama ve dağıtım noktası arasındaki bölgelere gerekli iletim sağlanmış olacaktır [5].

2.2 Literatür Araştırması

GSP literatürde 1800'lü yıllardan bu yana çalışılmakta olan bir NP-zor (Non Polynomial-Hard) problem çeşididir. GSP, kapasite kısıtlamasının olmadığı tek araçlık klasik Hamilton turlu problem türüdür. Hamilton Turu, bir noktadan sadece bir kez geçen ve başlangıç noktası ile bitiş noktası aynı olan 19. yüzyılda yaşamış matematikçi William Hamilton'un adı ile anılan turdur. GSP'nin amacı en kısa yoldan turun tamamlanmasıdır. n adet noktadan oluşan bir problemde, ilk noktaya geldiğinde gezgin satıcının $n-1$ farklı yol seçeneği vardır. İkinci noktada $n-2$ farklı yol seçeneği vardır. Genel olarak $(n-1)!$ farklı durum sözkonusudur ve mümkün olan tur sayısı $(n-1)!/2$'dir. Bugüne kadar çözülebilen en büyük GSP İsveç'in tüm yerleşim alanlarını içeren 24.978 adet noktadan oluşan bir problemdir. 96 bilgisayarlı bir ağ üzerinde çözülmesi 3 yıl sürmüştür. 2001'de Almanya'nın 15.112 yerleşim noktasını ziyaret eden ve çözümü 22 yıl süren problemin sonucu 66.000 km olarak hesaplanmıştır. Şu anda çözülmeye çalışılan en büyük boyutlu GSP ise 1.904.711 şehirden oluşan dünya üzerinde kayıtlı tüm yerleşim yerini kapsayan en kısa yol problemidir. GSP'nin daha detaylı tarihçesi ile literatürde yer alan kaynaklar mevcuttu [6].

GSP çözümünde kesin çözüm (exact) yöntemleri, klasik sezgisel (heuristic) yöntemler ve metasezgisel (metaheuristic) yöntemler kullanılmaktadır. Kesin çözüm yöntemleri en iyi çözümü garanti eder ancak büyük boyutlu problemlerde sonuca ulaşmak çok fazla zaman alabilir ve programlama bilgisi gerektirir. Klasik sezgisel yöntemlerin uygulanması oldukça basit ve kısa olmasına rağmen en iyi çözümü garanti etmezler. Metasezgisel çözüm yöntemleri ile ise iyi bir programlama bilgisi ile en iyi çözüme çok yakın çözüm kısa süre içerisinde bulunabilir. Bu sebeble son yıllarda en fazla metasezgisel çözüm yöntemleri tercih edilmiştir. Kesin çözüm yöntemlerine örnek olarak “Branch & Bound” algoritması verilebilir. Diğer yöntemler “Lineer Programlama Yöntemleri” ve “Dinamik Programlama”dır. Sezgisel yöntemlere örnek olarak ise “Tur Oluşturan Sezgiseller”, “Turu Geliştiren Sezgiseller” ve “Melez Sezgiseller” verilebilir [7].

Şekil-4’te GSP için kullanılan çözüm yöntemleri gösterilmiştir.

Şekil 4 GSP Çözüm Yöntemleri

GSP için literatürdeki ilişkili problem türleri olarak aşağıdaki örnekler verilebilir.

- Çoklu Otel Seçimli Gezgin Satıcı Problemi (The Multiple Travelling Salesperson Problem With Hotel Selection (m-TSPHS)) OSGSP’nin bir genelleştirilmiş türüdür. Amaç, her müşterinin ziyaret edileceği, aynı depoda başlayıp biten ve her satıcı için bir tane olmak üzere toplam bir tur seti bulmaktır. Müşterilerin toplam ziyaret maliyeti enküçüklenmeye çalışılır [8].

- Otel Seçimli Oryantiring Problemi (Orienteering Problem with otel Selection) belirli puanları olan müşterilerin ziyaret edilmesini ve tüm müşterilere uğrama zorunluluğu olmaksızın toplanan skorun enbüyüklemesini amaçlar. Adını oryantiring sporundan alan bu problemde günlük süre kısıtları verildiği için geziler bir otelde sonlanır. Takip eden gezi yine aynı otelden başlar ve bu şekildeki gezilerin birleşiminden oluşan toplam tur tamamlanır. Bu konuda çalışılmış hem sezgisel hem de kesin sonuç veren yöntemler mevcuttur [9].

- Çoklu Gezgin Satıcı Problemi (The Multiple Traveling Salesperson Problem (MTSP)) birden çok satıcının tek bir depo noktasından yola çıkarak ara noktalara dağıtım yapması esnasındaki her satıcı için rotaların oluşturulması üzerinde kurgulanmıştır. Amaç, toplam mesafe ve süre gibi toplam maliyetlerin enküçüklenmesidir. Her dağıtım noktasına sadece bir kez uğranması esastır. Örnek olarak okul servislerinin planlanması verilebilir. Bu konuda çalışılmış hem sezgisel hem de kesin sonuç veren yöntemler mevcuttur [10].

- Gezgin Alıcı Problemi (Traveling Purchaser Problem) m adet alışveriş yerinden n adet ürünü satın alacak bir alıcı (tüketici) için, alışveriş noktaları arasındaki ulaşım maliyeti ve her bir ürünün fiyatı bilinmek şartı ile, toplam ulaşım ve satın alma masraflarını enküçükleyen turun belirlenmesi problemidir. Literatürde kapasitelendirilmiş ve kapasitelendirilmemiş olarak iki çeşit Gezgin Alıcı Problemi yer almaktadır [11].

OSGSP ise ilk olarak 2012 yılında GSP'nin bir türü olarak çalışılmıştır [5]. Bu çalışmada iki indeksli bir formülasyon ile lokal arama algoritması kullanılmıştır. 2013 yılında daha iyi performans gösteren bir memetik algoritma üzerinde çalışılmıştır [2]. 2015 yılında şu ana kadar bilinen en iyi performansa sahip sezgisel yöntem üzerinde çalışılmıştır. Yerel arama algoritması ile önce düzenle sonra ayır (order-first split-second) metodunun kombinasyonu kullanılmıştır. 2015 yılında ise değişken komşu arama yöntemini kullanan OSGSP üzerinde çalışılmıştır [12].

3 OTEL SEÇİMLİ GEZGİN SATICI PROBLEMİ İÇİN MATEMATİKSEL MODELLER

OSGSP'de, s adet otelden oluşan otel kümesi ($i=1,\ldots,s$) ve n adet müşteriden oluşan müşteri kümesi ($i=s+1,\ldots,s+n$) bulunur. Her i müşterisi için servis veya ziyaret süresi olan T_i değeri bilinmektedir. Her i ve j kombinasyonu için i müşterisinden/otelinden j müşterisine/oteline gidiş süre değerleri $c_{i,j}$ verilir veya veriler X-Y koordinatlarına göre hesaplanır. Her tur için m adet geziden oluşan gezi kümesi ($d=1,\ldots,m$) bir süre sınırlaması olan C değeri ile sağlanmaktadır. İlk amaç toplam gezi sayısını ve sonrasında tur uzunluğunu enküçüklemektir. Turun başlangıç ve bitiş otel noktası olarak $i=1$ alınmıştır. Bir otel birden çok kez kullanılabildiği için OSGSP problemi her zaman başlangıç noktasından, başlangıç noktasına tek bir çevrim oluşturmayabilir [5].

Bu çalışmada, literatürde bulunan iki matematiksel model incelenmiş ve bu çalışmada üç yeni matematiksel model önerilmiştir.

İncelenen ilk matematiksel model [5] literatürde hali hazırda yer almakta olup bu çalışmada VSS olarak isimlendirilmiştir. Bu model karma tamsayılı doğrusal programlama modelleri arasında yer almaktadır.

Ele alınan ikinci matematiksel model, bu tez kapsamında yeni geliştirilmiştir. VSS modelindeki alt tur engelleme kısıtı hariç tüm kısıtlar aynen kullanımış, Gavish and Graves (GG) Formülasyonu [13] kullanılarak yeni alt tur engelleme kısıtı eklenmiştir. Bu çalışmada model VSSGG olarak isimlendirilmiştir.

Üçüncü model [2] literatürde hali hazırda yer almakta olup CSVG olarak isimlendirilmiştir. CSVG modelinde hem gezi sayısını hem de toplam mesafeyi en aza indiren ağırlıklı bir amaç fonksiyonu kullanılmıştır.

Ele alınan dördüncü matematiksel model, bu tez kapsamında yeni geliştirilmiştir. CSVG modelindeki alt tur engelleme kısıtı hariç tüm kısıtlar aynen kullanımış, Gavish and Graves (GG) Formülasyonu [13] kullanılarak yeni alt tur engelleme kısıtı eklenmiştir. Bu çalışmada model CSVGGG olarak isimlendirilmiştir.

Beşinci matematiksel model ise Fox, Gavish and Graves Formülasyonu [13] kullanılarak bu tez kapsamında yeni geliştirilmiş ve model FGG olarak isimlendirilmiştir.

Tüm modeller için ortak kullanılan simgeler, parametreler ve karar değişkenleri aşağıdaki gibi tanımlanmıştır. Ancak FGG için karar değişkeni dört indisli olup diğer modellerden farklıdır. FGG model açıklamasında ayrıca belirtilmiştir.

Simgeler

k ve *l*: İndisler

Parametreler

m: Gezi sayısı (modelde gezi sayısı sabit alınmaz ise alternatif formülasyon ile bu değer de enküçüklenebilir, örneğin CSVG veya CSVGGG modelleri)

s: Otel sayısı

n: Müşteri sayısı

c[*k,l*]: *k*. düğümden *l*. düğüme gitmenin süresi (mesafesi)

C: Her gezi için belirlenen süre kısıtı

T[*l*]: *l*. müşteride geçen zaman

Karar Değişkenleri

x[*k,l,d*]: *d*. gezide *k*. düğümden *l*. düğüme gidiliyorsa 1, diğer durumlarda 0.

3.1 VSS Modeli

Modelin amaç fonksiyonu ve kısıtları aşağıdaki gibidir;

Amaç Fonksiyonu;

$$\text{Enk} \sum_{d=1}^{\mathrm{m}} \sum_{k=1}^{\mathrm{s+n}} \sum_{l=1}^{s+n} x_{k,l,d} \left(c_{k,l} + T_l \right) \qquad (3.1)$$

Amaç fonksiyonunda (3.1) toplam seyahat süresi enküçüklenmektedir.

Kısıtlar;

$$\sum_{l=1}^{s+n} x_{1,l,1} = 1 \tag{3.2}$$

$$\sum_{k=1}^{s+n} x_{k,1,m} = 1 \tag{3.3}$$

(3.2) ve (3.3) numaralı kısıtlar turun başlangıç otelinde başlaması ve aynı otelde bitmesini garanti etmektedir.

$$\sum_{h=1}^{s} \sum_{k=1}^{s+n} x_{k,h,d} = 1 \qquad d = 1,..m \tag{3.4}$$

$$\sum_{h=1}^{s} \sum_{l=1}^{s+n} x_{h,l,d} = 1 \qquad d = 1,..m \tag{3.5}$$

(3.4) ve (3.5) numaralı kısıtlar her gezinin bir otelde başlaması ve bir otelde bitmesini garanti etmektedir.

$$\sum_{k=1}^{s+n} x_{k,h,d} - \sum_{l=1}^{s+n} x_{h,l,d+1} = 0 \qquad d = 1,..m-1;\ h = 1,..s \tag{3.6}$$

(3.6) numaralı kısıt bir gezi hangi otelde biterse takip eden gezinin aynı otelden başlamasını garanti eder.

$$\sum_{d=1}^{m} \sum_{k=1}^{s+n} x_{k,l,d} = 1 \qquad l = s+1,..s+n \tag{3.7}$$

(3.7) numaralı kısıt her müşterinin 1 kez ziyaret edilmesini sağlar.

$$\sum_{k=1}^{s+n} x_{k,h,d} - \sum_{l=1}^{s+n} x_{h,l,d} = 0 \qquad d = 1,..m;\ h = s+1,..s+n \tag{3.8}$$

(3.8) numaralı kısıt her gezi içerisinde müşteriler arası devamlılığı sağlar, hangi müşteriye gidiliyorsa o müşteriden çıkılır.

$$\sum_{k=1}^{s+n}\sum_{l=1}^{s+n} x_{k,l,d}\left(c_{k,l}+T_l\right) \leq C \qquad d=1,..m \tag{3.9}$$

(3.9) numaralı kısıt her gezi için süre kısıtlamasını sağlar. Bir gezi maksimum süreyi aşmadan tamamlanmalıdır.

Alt Tur Engelleme Kısıtı;

$$u_k - u_l + 1 \leq n(1-\sum_{d=1}^{m} x_{k,l,d}) \qquad k=s+1,..s+n;\ l=s+1,..s+n \tag{3.10}$$

(3.10) tur süresince alt turlar oluşmasını engeller.

$$x_{k,l,d} \in 0{,}1 \qquad d=1,..m; k=1,..s+n; l=1,..s+n \tag{3.11}$$

Kısıt (3.11) $x_{k,l,d}$ değerlerinin 1-0 değer almasını sağlar.

$$u_k \geq 0, u_l \geq 0 \qquad k=s+1,..s+n; l=s+1,..s+n \tag{3.12}$$

Kısıt (3.12) u_k veya u_l sıralama değerlerinin müşteri sayısı kadar değer almasını sağlar.

3.2 VSSGG modeli

Alt turları engellemek için kullanılan ilave yardımcı değişken $g_{k,l}$: *k.* düğümden *l.* düğüme hangi sırayla gidileceğini belirlemek için kullanılır.

Modelin amaç fonksiyonu ve kısıtları aşağıdaki gibidir;

Amaç Fonksiyonu;

Enk (3.1)

Kısıtlar;

(3.2) - (3.9) ve (3.11)

Alt Tur Engelleme Kısıtı;

$$\sum_{l=1}^{s+n} g_{l,k} - \sum_{l=1}^{s+n} g_{k,l} = 1 \qquad k = s+1,..s+n \qquad (3.13)$$

$$g_{k,l} \leq n \sum_{d=1}^{m} x_{k,l,d} \qquad k = 1,..s+n;\ l = 1,..s+n \qquad (3.14)$$

(3.13) ve (3.14) tur süresince alt turlar oluşmasını engeller.

$$g_{k,l} \leq 0 \qquad k = 1,..s+n;\ l = 1,..s+n \qquad (3.15)$$

Kısıt (3.15) $g_{k,l}$ değerlerinin müşteri sayısı kadar değer almasını sağlar.

3.3 CSVG Modeli

Modelin karar değişkenleri, amaç fonksiyonu ve kısıtları aşağıdaki gibidir;

Karar Değişkenleri;

x[*k,l,d*]: *d*. gezide *k*. düğümden *l*. düğüme gidiliyorsa 1, diğer durumlarda 0.

y[*d*]: *d*. gezi gerçekleştiyse 1, diğer durumda 0.

Amaç Fonksiyonu;

$$Enk \sum_{d=1}^{m} \sum_{k=1}^{s+n} \sum_{l=1}^{s+n} x_{k,l,d}(c_{k,l} + T_l) + M \sum_{d=1}^{m} y_d \qquad (3.16)$$

Amaç fonksiyonu (3.16) toplam seyahat süresini ve gezi sayısını enküçüklenmektedir.

Kısıtlar;

$$\sum_{k=1}^{s}\sum_{l=1}^{s} x_{k,l,d} = 0 \qquad d = 1,..m \tag{3.17}$$

Kısıt (3.17) ele alınan referans yayında bulunmamaktadır. Bu kısıt, herhangi bir tur içindeki oteller arasında geçişi önlemek için eklenmiştir.

$$\sum_{d=1}^{m}\sum_{k=1}^{s+n} x_{k,l,d} = 1 \qquad l = s+1,..s+n \tag{3.18}$$

Kısıt (3.18) her müşterinin 1 kez ziyaret edilmesini sağlar.

$$\sum_{k=1}^{s+n} x_{k,h,d} - \sum_{l=1}^{s+n} x_{h,l,d} = 0 \qquad h = s+1,..s+n\,; d = 1,..m \tag{3.19}$$

Kısıt (3.19) her gezi içerisinde devamlılığı sağlar, hangi müşteriye gidiliyorsa o müşteriden geziye devam edilir.

$$\sum_{k=1}^{s}\sum_{l=1}^{s+n} x_{k,l,d} - y_d = 0 \qquad d = 1,..m \tag{3.20}$$

$$\sum_{l=1}^{s}\sum_{k=1}^{s+n} x_{k,l,d} - y_d = 0 \qquad d = 1,..m \tag{3.21}$$

Kısıt (3.20) ve (3.21) eğer gerçeklenen geziler olacaksa bu gezilerin bir otelde başlaması ve bir otelde bitmesini garanti etmektedir.

$$\sum_{k=1}^{s+n}\sum_{l=1}^{s+n} x_{k,l,d}(c_{k,l} + T_l) \leq C \qquad d = 1,..m \tag{3.22}$$

Kısıt (3.22) her gezi için süre kısıtlamasını sağlar.

$$\sum_{l=2}^{s+n} x_{1,l,1} = 1 \tag{3.23}$$

$$\sum_{k=2}^{s+n} x_{k,1,d} \geq y_d - y_{d+1} \qquad d = 1,..m-1 \qquad (3.24)$$

Kısıt (3.23) ve (3.24) turun başlangıç otelinde başlaması ve aynı otelde bitmesini garanti etmektedir.

$$\sum_{k=1}^{s+n} x_{k,h,d} + y_d \geq \sum_{l=1}^{s+n} x_{h,l,d+1} + y_{d+1} \qquad h = 1,..s\,; d = 1,..m-1 \qquad (3.25)$$

$$\sum_{k=1}^{s+n} x_{k,h,d} - \sum_{l=1}^{s+n} x_{h,l,d+1} \leq 1 - y_{d+1} \qquad h = 1,..s\,; d = 1,..m-1 \qquad (3.26)$$

Kısıt (3.25) ve (3.26) bir gezi hangi otelde biterse takip eden gezinin aynı otelden başlamasını garanti eder.

$$x_{k,l,d} \leq y_d \qquad d = 1,..m; k = 1,..s+n; l = 1,..s+n \qquad (3.27)$$

Kısıt (3.27) bir gezide en az bir müşteri/otel ziyaret edildiğinde, o gezi için karşılık gelen değişkenin 1 değerini aldığı, yani kullanıldığı anlamına gelir.

$$y_d \geq y_{d+1} \qquad d = 1,..m-1 \qquad (3.28)$$

Kısıt (3.28) gezilerin 1. günden başlayarak ardışık günlerde yapılmasını sağlar.

$$y_d \in 0{,}1 \qquad d = 1,..m \qquad (3.29)$$

Kısıt (3.29) y_d değerlerinin 1-0 değer almasını sağlar.

Alt Tur Engelleme Kısıtı;

(3.10) - (3.12)

3.4 CSVGGG

Alt turları engellemek için kullanılan ilave yardımcı değişken $g_{k,l}$: *k.* düğümden *l.* düğüme hangi sırayla gidileceğini belirlemek için kullanılır.

Modelin amaç fonksiyonu ve kısıtları aşağıdaki gibidir;

Amaç Fonksiyonu;

Enk (3.16)

Kısıtlar;

(3.11) ve (3.17) - (3.29)

Alt Tur Engelleme Kısıtı;

(3.13) - (3.15)

3.5 FGG Modeli

FGG modelinin diğer modellerden farkı, alt tur engelleme kısıtı olmaksızın karar değişkeni içerisine sıralama indisi eklenerek geliştirilmiş bir model olmasıdır. Bu modelin geliştirilmesinin amacı alt tur engelleme kısıtlı modellere göre performansının nasıl olduğunun ölçülmesidir.

Modelin karar değişkenleri, amaç fonksiyonu ve kısıtları aşağıdaki gibidir;

Karar değişkenleri;

x[*k,l,d,u*]: *d*. gezide *k*. düğümden *l*. düğümüne *u.* sırada gidiliyorsa 1, diğer durumlarda 0.

Amaç Fonksiyonu;

$$\text{Enk} \sum_{\mathrm{d}=1}^{\mathrm{m}} \sum_{\mathrm{k}=1}^{\mathrm{s+n}} \sum_{\mathrm{l}=1}^{s+n} \sum_{u=1}^{s+n} x_{k,l,d,u} \left(c_{k,l} + T_l\right) \tag{3.30}$$

Amaç fonksiyonunda (3.30) toplam seyahat süresi enküçüklenmektedir.

$$\sum_{l=2}^{s+n} x_{1,l,1,1} = 1 \tag{3.31}$$

$$\sum_{k=2}^{s+n} \sum_{u=1}^{s+n} x_{k,1,m,u} = 1 \tag{3.32}$$

Kısıt (3.31) ve (3.32) turun 1.sırada başlangıç otelinde başlaması ve son sırada aynı otelde bitmesini garanti etmektedir.

$$\sum_{l=1}^{s}\sum_{k=1}^{s+n}\sum_{u=1}^{s+n} x_{k,l,d,u} = 1 \qquad d = 1,..m \tag{3.33}$$

$$\sum_{k=1}^{s}\sum_{l=1}^{s+n}\sum_{u=1}^{s+n} x_{k,l,d,u} = 1 \qquad d = 1,..m \tag{3.34}$$

Kısıt (3.33) ve (3.34) her gezinin bir otelde başlaması ve bitmesini garanti etmektedir.

$$\sum_{k=1}^{s+n}\sum_{u=1}^{s+n} x_{k,h,d,u} - \sum_{l=1}^{s+n} x_{h,l,d+1,1} = 0 \qquad d = 1,..m-1;\ h = 1,..s \tag{3.35}$$

Kısıt (3.35) bir gezi hangi otelde biterse takip eden gezinin aynı otelden başlamasını garanti eder.

$$\sum_{d=1}^{m}\sum_{k=1}^{s+n}\sum_{u=1}^{s+n} x_{k,l,d,u} = 1 \qquad l = s+1,..s+n \tag{3.36}$$

Kısıt (3.36) her müşterinin 1 kez ziyaret edilmesini sağlar.

$$\sum_{k=1}^{s+n}\sum_{u=1}^{s+n} x_{k,h,d,u} - \sum_{l=1}^{s+n}\sum_{u=1}^{s+n} x_{h,l,d,u} = 0 \qquad d = 1,..m; h = s+1,..s+n \tag{3.37}$$

Kısıt (3.37) her gezi içerisinde devamlılığı sağlar, hangi müşteriye gidiliyorsa o müşteriden geziye devam edilir.

$$\sum_{k=1}^{s+n}\sum_{l=1}^{s+n}\sum_{u=1}^{s+n} x_{k,l,d,u}(c_{k,l} + T_l) \leq C \qquad d = 1,..m \tag{3.38}$$

Kısıt (3.38) her gezi için süre kısıtlamasını sağlar.

$$\sum_{d=1}^{m}\sum_{l=1}^{s+n}\sum_{u=1}^{s+n} x_{h,l,d,u}u - \sum_{d=1}^{m}\sum_{k=1}^{s+n}\sum_{u=1}^{s+n} x_{k,h,d,u}u = 1 \qquad h = s+1,..s+n \qquad (3.39)$$

Kısıt (3.39) her müşteri çifti için ziyaret sıralamasını belirler.

$$x_{k,l,d,u} \in 0,1 \qquad d = 1,..m;\ k = 1,..s+n;\ l = 1,..s+n;\ u = 1,..s+n \qquad (3.40)$$

Kısıt (3.40) $x_{k,l,d,u}$ değerlerinin 1-0 değer almasını sağlar.

VSS, VSSGG, CSVG ve CSVGGG modellerinde $O(n^3)$ sayıda, FGG modelinde ise $O(n^4)$ sayıda karar değişkeni bulunmaktadır; yanı sıra VSS, VSSGG, CSVGGG ve FGG modellerinde $O(n^2)$ sayıda, CSVG modelinde ise $O(n^3)$ sayıda kısıt bulunmaktadır.

4 SAYISAL ANALİZLER

4.1 Test Problemleri

Bu tezde daha önce OSGSP çalışmalarında kullanılan test verileri [14], [15], [16] kullanılmıştır. Büyük ölçekli probemler SET-1, küçük ölçekli problemler SET-2 olmak üzere bu çalışmada 2 farklı veri seti üzerinde modeller test edilmiştir.

SET-1'de 16 problem bulunmaktadır, bu veri seti içerisindeki r101 ve r201 olarak isimlendirilen problemler rastgele coğrafi koordinat verilerinden oluşmaktadır. Kümelendirilmiş verilerden oluşan problemler c101 ve c201 olarak isimlendirilmiştir. Yarı kümelenmiş bir problem, rastgele oluşturulmuş veri ile kümelendirilmiş verilerin bir karışımını içermektedir. Yarı kümelendirilmiş problemler rc101 ve rc201 olarak isimlendirilmiştir. r101, c101 ve rc101 verileri kısa dönem planlamada kullanılırken r201, c201 ve rc201 verileri uzun dönemli planlama için kullanılmaktadır. Bu problemlerin tümü 100 müşteri noktası içermektedir, pr1-pr10 test verileri ise 48 ile 288 tane arasında değişken müşteri noktası içeren problem setlerinden oluşmaktadır.

SET-2'de 52 farklı problem bulunmaktadır, bu problemler SET-1'deki problemlerin içerisindeki müşteri noktalarından küçük bir küme seçilerek oluşturulmuştur [5]. SET-2 problemleri, referans alınan problemin yanına seçilen müşteri sayısı eklenerek isimlendirilmiştir. Her problem için 10, 15, 30 ve 40 müşteriden oluşacak şekilde problemler türetilmiştir. SET-2 problemleri içerisinde uzun dönem planlama için kullanılan r201, c201 ve rc201 problemleri yer almamıştır.

Veri setlerindeki problemlere göre müşteri sayısı (n), gezi süresi (C), otel sayısı (s) ve gezi sayısı (m) Tablo-1 ve Tablo-2'de gösterilmektedir. CSVG ve CSVGGG modelleri, gezi sayısını kendi hesapladığı için bu modellerde "m" bir parametre olarak kullanılmamıştır. Ayrıca data setlerinde yer alan müşteri ve otellerin X-Y koordinat bilgileri model içerisinde noktalar arası mesafe hesabında kullanılmıştır.

Veri setlerindeki en büyük ölçekli problem müşteri sayısına göre 288 müşteri, otel sayısına göre başlangıç oteli hariç 5 ekstra otel içermektedir. En küçük problem ise müşteri sayısına göre 10 müşteri, otel sayısına göre başlangıç oteli hariç 1 ekstra otel içermektedir.

Tablo-1'de yer alan SET-1 verilerinde müşteri sayısı 48 ile 288 arasında değişmektedir. Otel sayısı tüm problemlerde başlangıç oteli hariç 5 olarak verilmiştir. Gezi sayısı 2 ile 9 arasında değişkenlik göstermektedir. Gezi süreleri ise en az 230 en fazla 3390 olarak verilmiştir.

Tablo 1 SET-1 Test Verileri

SET1-Problem	n	C	s	m
c101	100	1236	6	9
c201	100	3390	6	3
pr01	48	1000	6	2
pr02	96	1000	6	3
pr03	144	1000	6	4
pr04	192	1000	6	5
pr05	240	1000	6	6
pr06	288	1000	6	7
pr07	72	1000	6	3
pr08	144	1000	6	4
pr09	216	1000	6	5
pr10	288	1000	6	7
r101	100	230	6	9
r201	100	1000	6	2
rc101	100	240	6	8
rc201	100	960	6	2

Tablo-2'de yer alan SET-2 verilerinde müşteri sayısı 10 ile 40 arasında değişmektedir. Otel sayısı tüm problemlerde başlangıç oteli hariç 1 olarak verilmiştir. Gezi sayısı 1 ile 4 arasında değişkenlik göstermektedir. Gezi süreleri ise en az 230 en fazla 1236 olarak verilmiştir.

Tablo 2 SET-2 Test Verileri

SET2-Problem	N	C	s	m	SET2-Problem	N	C	s	m	SET2-Problem	N	C	s	m
c101.k10	10	1236	2	1	pr04.k30	30	1000	2	2	pr09.k10	10	1000	2	1
c101.k15	15	1236	2	2	pr04.k40	40	1000	2	2	pr09.k15	15	1000	2	1
c101.k30	30	1236	2	3	pr05.k10	10	1000	2	1	pr09.k30	30	1000	2	2
c101.k40	40	1236	2	4	pr05.k15	15	1000	2	1	pr09.k40	40	1000	2	2
pr01.k10	10	1000	2	1	pr05.k30	30	1000	2	1	pr10.k10	10	1000	2	1
pr01.k15	15	1000	2	1	pr05.k40	40	1000	2	2	pr10.k15	15	1000	2	1
pr01.k30	30	1000	2	1	pr06.k10	10	1000	2	1	pr10.k30	30	1000	2	1
pr01.k40	40	1000	2	2	pr06.k15	15	1000	2	1	pr10.k40	40	1000	2	2
pr02.k10	10	1000	2	1	pr06.k30	30	1000	2	2	r101.k10	10	230	2	2

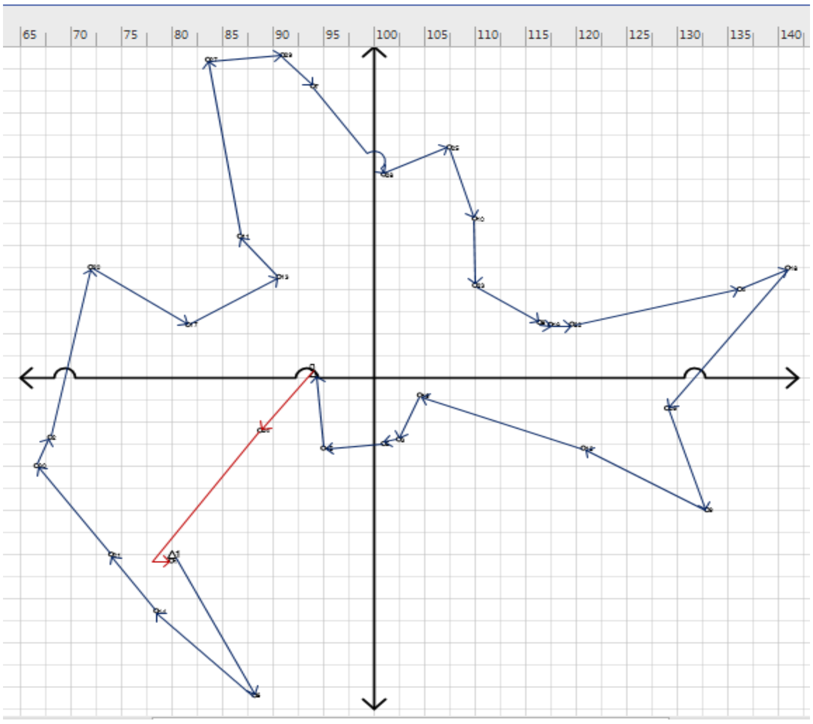
Tablo-2 devam ediyor

pr02.k15	15	1000	2	1	pr06.k40	40	1000	2	2	r101.k15	15	230	2	2
pr02.k30	30	1000	2	2	pr07.k10	10	1000	2	1	r101.k30	30	230	2	3
pr02.k40	40	1000	2	2	pr07.k15	15	1000	2	1	r101.k40	40	230	2	4
pr03.k10	10	1000	2	1	pr07.k30	30	1000	2	2	rc101.k10	10	240	2	1
pr03.k15	15	1000	2	1	pr07.k40	40	1000	2	2	rc101.k15	15	240	2	2
pr03.k30	30	1000	2	1	pr08.k10	10	1000	2	1	rc101.k30	30	240	2	4
pr03.k40	40	1000	2	2	pr08.k15	15	1000	2	1	rc101.k40	40	240	2	4
pr04.k10	10	1000	2	1	pr08.k30	30	1000	2	2					
pr04.k15	15	1000	2	1	pr08.k40	40	1000	2	2					

4.2 Sayısal Sonuçlar

Literatürden referans alınan ve yeni geliştirilen metematiksel modeller IBM ILOG CPLEX Optimization Studio 12.6.1.0 programı kullanılarak OPL ile kodlanmıştır. VSS, CSVG, VSSGG, CSVGGG, FGG modellerinin OPL kodları sırasıyla Ek-1, Ek-2, Ek-3, Ek-4, Ek-5'te yer almaktadır. Optimal sonuçları bilinen veri setleri Intel Core i7-4470 CPU 3.40 GHz ve 8 GB RAM özelliklerindeki bilgisayar ile 10800 saniye zaman sınırı, 1024 MB kullanılabilir bellek sınırı ve düğüm dosyasının hard disk üzerine sıkıştırılmış biçimde yazılması için ilgili parametreler değiştirilerek CPLEX yardımıyla çözdürülmüştür. Bu parametreler dışında CPLEX'in mevcut parametrelerinde bir değişiklik yapılmamıştır. Modellerin geçerliliği optimal değerler ile karşılaştırılarak doğrulanmıştır. Sonrasında performans artışı için geliştirilen alternatif modellerle de aynı koşullarda aynı veri setleri çözdürülerek modeller karşılaştırılmıştır.

Şekil-5'te test verilerinden örnek bir problemin kodlanan modeller kullanılarak alınan çözümü yer almaktadır. Çizimde otel ve müşteri koordinatları gerçeğe uygun olarak x-y ekseni üzerine yerleştirilmiştir.



Şekil 5 pr8-30 Test Verisine Ait Çözümün Görsel Gösterimi

SET-1 ve SET-2 problemlerinin literatürde bilinen optimal sonuçları, literatürdeki modeller ve geliştirilen alternatif modellerin çözümleri ile birlikte Tablo-3 ve Tablo-4'te gösterilmiştir. Belirlenen süre kısıtı dahilinde olası bir çözüme ulaşılamadığı durumlar “-” ile ifade edilmiştir. Ayrıca tabloda koyu renk olarak belirtilen değerler çözümün optimal sonuca ulaştığını ifade etmektedir. CSVG ve CSVGGG modellerinin çözüm değerleri içerisinde gezi sayısı kadar *M* değeri içermektedir. SET-1 için *M*=10000 iken, SET-2 için *M*=5000 olarak belirlenmiştir.

Tablo-3'e göre SET-1 için VSS ve FGG modelleri hiçbir problemde optimal çözüme ulaşamamıştır. Literatürde yer alan optimal değerler 1412.2 ile 9559.9 arasında değer almaktadır. CSVG ve CSVGGG modelleri çözülürken büyük *M* katsayısı olarak 10000 değeri alındığı için hesaplanan 21412.2 değeri, gezi sayısı kadar *M* değeri çıkarılarak optimal olarak işaretlenmiştir. VSS modelinin bulduğu 1412.2 değeri optimal değere ulaşmış olsa dahi model verilen çözüm süresi limitlerinde henüz sonuçlanmadığı için optimal olarak değerlendirilmemiştir. CSVGGG modelinin bulduğu 21413.4 değeri optimal değere CPLEX'te belirlenen hassasiyet oranında yaklaştığı için model sonlanmış ve optimal olarak işaretlenmiştir. FGG modeli hiçbir problemde optimale ulaşamamıştır.

Tablo 3 SET-1 Optimal Değer Karşılaştırma Tablosu

SET1-Problem	Opt.	VSS	CSVG	VSSGG (Yeni)	CSVGGG (Yeni)	FGG (Yeni)
c101	9591.1	-	-	-	-	-
c201	9559.9	9821.8	-	-	-	-
pr01	1412.2	1412.2	21412.2	1412.2	21413.4	-
pr02	2548.8	-	102809	-	-	-
pr03	3404.2	-	-	-	-	-
pr04	4215.3	-	-	-	-	-
pr05	4948.9	-	-	-	-	-
pr06	5960.9	-	-	-	-	-
pr07	2070.3	-	32078.2	-	-	-
pr08	3367.7	-	-	-	-	-
pr09	4414.9	-	-	-	-	-
pr10	5932.0	-	-	-	-	-
r101	1695.5	-	-	-	-	-
r201	1642.8	1643.7	21660.8	-	21758.8	-
rc101	1673.4	-	-	-	-	-
rc201	1642.7	-	-	-	-	-

Tablo-4'e göre SET-2 için CSVG, VSSGG ve CSVGGG modelleri 46 adet olmak üzere eşit sayıda ve en fazla optimale ulaşan veri sayısına sahiptir. Optimale ulaşma açısından en düşük performanslı model 30 adet ile FGG olarak görülmüştür. Literatürde yer alan optimal değerler 237.5 ile 3866.1 arasında değişmektedir.

Tablo 4 SET-2 Optimal Değer Karşılaştırma Tablosu

SET2-Problem	Opt.	VSS	CSVG	VSSGG (Yeni)	CSVGGG (Yeni)	FGG (Yeni)
c101.k10	955.1	**955.1**	**5955.1**	**955.1**	**5955.1**	**955.1**
c101.k15	1452.2	1452.2	**11452.2**	**1452.2**	**11452.2**	**1452.3**
c101.k30	2863.2	2864.9	17863.6	2868.8	17863.3	-
c101.k40	3866.1	3882	23876.6	3875.8	23938.2	-
pr01.k10	426.6	**426.6**	**5426.8**	**426.6**	**5426.6**	**426.6**
pr01.k15	590.4	**590.4**	**5590.4**	**590.4**	**5590.4**	**590.4**
pr01.k30	964.8	**964.8**	**5964.8**	**964.8**	**5964.8**	**964.8**
pr01.k40	1160.5	**1160.5**	**11160.5**	**1160.5**	**11160.5**	-
pr02.k10	661.9	**661.9**	**5661.9**	**661.9**	**5661.9**	**661.9**
pr02.k15	745.6	**745.6**	**5745.6**	**745.6**	**5745.6**	**745.6**
pr02.k30	1078.3	**1078.3**	**11078.3**	**1078.3**	**11078.3**	-
pr02.k40	1336.9	**1336.9**	**11336.9**	**1336.9**	**11336.9**	-
pr03.k10	553.3	**553.3**	**5553.3**	**553.3**	**5553.3**	**553.3**
pr03.k15	632.9	**632.9**	**5632.9**	**632.9**	**5632.9**	**632.9**
pr03.k30	952.5	**952.5**	**5952.5**	**952.5**	**5952.5**	-
pr03.k40	1303.4	**1303.4**	**11303.4**	**1303.4**	**11303.4**	-
pr04.k10	476.4	**476.4**	**5476.4**	**476.4**	**5476.4**	**476.4**
pr04.k15	683.4	**683.4**	**5683.4**	**683.4**	**5683.4**	**683.4**
pr04.k30	1091.6	**1091.6**	**11091.6**	**1091.6**	**11091.6**	**-**

Tablo-4 devam ediyor.

pr04.k40	1259.5	**1259.5**	**11259.5**	**1259.5**	**11259.5**	**-**
pr05.k10	528.9	**528.9**	**5528.9**	**528.9**	**5528.9**	**528.9**
pr05.k15	621.2	**621.2**	**5621.2**	**621.2**	**5621.2**	**621.2**
pr05.k30	924.7	**924.7**	**5924.7**	**924.7**	**5924.7**	**924.7**
pr05.k40	1200.7	**1200.7**	**11200.7**	**1200.7**	**11200.7**	-
pr06.k10	597.4	**597.4**	**5597.4**	**597.4**	**5597.4**	**597.4**
pr06.k15	685.2	**685.2**	**5685.2**	**685.2**	**5685.2**	**685.2**
pr06.k30	1063.2	**1063.2**	**11063.2**	**1063.2**	**11063.2**	-
pr06.k40	1242.9	1250.2	**11242.9**	**1242.9**	**11242.9**	-
pr07.k10	670.2	**670.2**	**5670.2**	**670.2**	**5670.2**	**670.2**
pr07.k15	795.3	**795.3**	**5795.3**	**795.3**	**5795.3**	**795.3**
pr07.k30	1130.4	**1130.4**	**11130.4**	**1130.4**	**11130.4**	-
pr07.k40	1407.0	**1407**	**11407**	**1407**	**11407**	-
pr08.k10	573.4	**573.4**	**5573.4**	**573.4**	**5573.4**	**573.4**
pr08.k15	707.2	**707.2**	**5707.2**	**707.2**	**5707.2**	**707.2**
pr08.k30	1006.2	**1006.2**	**11006.2**	**1006.2**	**11006.2**	-
pr08.k40	1222.2	**1222.2**	**11222.2**	**1222.2**	**11222.2**	-
pr09.k10	645.5	**645.5**	**5645.5**	**645.5**	**5645.5**	**645.5**
pr09.k15	771.7	**771.7**	**5771.7**	**771.7**	**5771.7**	**771.7**
pr09.k30	1091.4	**1091.4**	**11091.4**	**1091.4**	**11091.4**	**1091.4**
pr09.k40	1284.4	**1284.4**	**11284.4**	**1284.4**	**11284.4**	-
pr10.k10	461.5	**461.5**	**5461.5**	**461.5**	**5461.5**	**461.5**
pr10.k15	611.9	**611.9**	**5611.9**	**611.9**	**5611.9**	**611.9**
pr10.k30	918.9	**918.9**	**5918.9**	**918.9**	**5918.9**	**918.9**
pr10.k40	1200.4	**1200.4**	**11200.4**	**1200.4**	**11200.4**	-
r101.k10	272.8	**272.8**	**10272.8**	**272.8**	**10272.8**	**272.8**
r101.k15	379.8	**379.8**	**10379.8**	**379.8**	**10379.8**	**379.8**
r101.k30	655.2	-	15655.2	-	15655.2	-
r101.k40	862.8	-	25908.9	-	-	-
rc101.k10	237.5	**237.5**	**5237.5**	**237.5**	**5237.5**	**237.5**
rc101.k15	303.2	303.2	**10303.2**	**303.2**	**10303.2**	**303.2**
rc101.k30	705.5	683.8	20683.8	707.5	20683.8	-
rc101.k40	850.3	-	-	-	-	-

SET-1 ve SET-2'deki her problem için VSS, CSVG, VSSGG, CSVGG, FGG modellerinin belirlenen süre kısıtı dahilindeki çözüm süreleri Tablo-5 ve Tablo-6'da gösterilmiştir. Belirlenen süre kısıtı dahilinde olası bir çözüme ulaşılamadığı durumlar "-" ile ifade edilmiştir.

Tablo-5'e göre SET-1 için tüm modellerin çözüm süreleri 418.08 ile CPLEX'te belirlenen üst limit olan 10800 arasında değişkenlik göstermektedir. CSVGGG modeli 1 problemde diğer modellere göre daha kısa sürede optimal sonuca ulaşmıştır. FGG modeli hiçbir uygun çözüm sağlayamamıştır.

Tablo 5 SET-1 Çözüm Süresi Karşılaştırma Tablosu (sn)

SET1-Problem	VSS	CSVG	VSSGG (Yeni)	CSVGGG (Yeni)	FGG (Yeni)
c101	-	-	-	-	-
c201	10800	-	-	-	-
pr01	10800	1496.46	582.71	418.08	-
pr02	-	10800	-	-	-
pr03	-	-	-	-	-
pr04	-	-	-	-	-
pr05	-	-	-	-	-
pr06	-	-	-	-	-
pr07	-	10800	-	-	-
pr08	-	-	-	-	-
pr09	-	-	-	-	-
pr10	-	-	-	-	-
r101	-	-	-	-	-
r201	10800	10800	-	10800	-
rc101	-	-	-	-	-
rc201	-	-	-	-	-

Tablo-6'ya göre SET-2 için tüm modellerin çözüm süreleri 0.09 ile CPLEX'te belirlenen üst limit olan 10800 arasında değişkenlik göstermektedir. VSSGG modeli 41 problem ile daha kısa sürede optimal değere ulaşan model olarak en yüksek performansı göstermektedir.

Tablo 6 SET-2 Çözüm Süresi Karşılaştırma Tablosu

SET2-Problem	VSS	CSVG	VSSGG (Yeni)	CSVGGG (Yeni)	FGG (Yeni)
c101.k10	62.73	0.53	0.20	0.44	0.25
c101.k15	10800	4.85	0.62	16.01	27.39
c101.k30	10800	10800	10800	10800	-
c101.k40	10800	10800	10800	10800	-
pr01.k10	0.24	0.39	0.14	0.30	0.90
pr01.k15	1.33	1.19	0.31	1.40	2.86
pr01.k30	36.66	30.97	1.39	44.40	240.07
pr01.k40	27.55	74.97	34.37	100.17	-
pr02.k10	0.17	0.40	0.14	0.30	0.23
pr02.k15	0.27	0.84	0.30	1.12	2.26
pr02.k30	1044.13	161.29	71.65	124.19	-
pr02.k40	206.11	126.80	59.67	216.33	-
pr03.k10	0.18	0.81	0.09	0.30	0.36
pr03.k15	0.27	1.22	0.14	1.19	1.58
pr03.k30	2.73	9.36	0.86	7.19	-
pr03.k40	338.32	485.26	85.27	78.06	-
pr04.k10	0.25	0.50	0.11	0.28	0.37
pr04.k15	0.33	0.83	0.22	0.94	1.75
pr04.k30	25.63	34.66	9.14	84.51	-
pr04.k40	28.24	36.38	20.14	184.94	-
pr05.k10	0.22	0.23	0.13	0.22	0.22

Tablo-6 devam ediyor.

pr05.k15	0.5	0.80	0.20	1.17	2.29
pr05.k30	2.79	7.91	0.80	12.39	47.14
pr05.k40	46.89	385.71	45.91	442.62	-
pr06.k10	0.20	0.31	0.11	0.42	0.36
pr06.k15	0.44	0.75	0.22	1.94	1.72
pr06.k30	1593.72	19.03	9.33	45.08	-
pr06.k40	10800	610.59	100.68	97.88	-
pr07.k10	0.17	0.23	0.09	0.58	0.19
pr07.k15	0.52	1.00	0.28	2.15	4.74
pr07.k30	33.70	35.97	8.13	170.56	-
pr07.k40	2564.45	617.92	210.20	223.91	-
pr08.k10	0.27	0.45	0.13	0.38	0.50
pr08.k15	0.51	0.94	0.14	1.47	3.53
pr08.k30	170.31	68.83	13.29	165.91	-
pr08.k40	284.03	43.99	267.42	177.62	-
pr09.k10	0.34	0.44	0.22	0.40	0.64
pr09.k15	0.41	1.00	0.28	0.89	1.47
pr09.k30	19.92	22.21	5.01	132.60	787.04
pr09.k40	37.86	153.69	28.53	365.60	-
pr10.k10	0.22	0.55	0.11	0.37	0.14
pr10.k15	0.62	1.22	0.20	1.48	1.58
pr10.k30	222.68	8.86	1.31	72.32	958.52
pr10.k40	4280.42	292.53	40.90	252.83	-
r101.k10	0.33	0.58	0.14	1.09	0.66
r101.k15	0.95	2.96	0.51	4.51	59.16
r101.k30	-	10800	-	10800	-
r101.k40	-	10800	-	-	-
rc101.k10	18.99	2.82	0.19	0.78	0.34
rc101.k15	10800	3.96	0.55	4.12	4.85
rc101.k30	10800	10800	10800	10800	-
rc101.k40	-	-	-	-	-

SET-1 ve SET-2’deki her problem için belirlenen süre kısıtı dahilinde modellerin olası bir çözüm bulduğu problemler için Gevşetilmiş Çözüm Değer Aralığı değeri hesaplanmış ve Tablo-7 ve Tablo-8’de gösterilmiştir. Belirlenen süre kısıtı dahilinde olası bir çözüme ulaşılamadığı durumlar “-” ile ifade edilmiştir. CSVG modellerinde tur sayısı *M* ile hesaplandığı ve bu değer amaç fonksiyonunda yer aldığı için; modelin henüz tur sayısını belirleyemediği durumlarda *M**tur sayısı farkı kadar Gevşetilmiş Çözüm Değer Aralığı değeri etkilenmektedir.

Tablo-7’de SET-1 için GAP değerleri hesaplanmış olup 0.01% ile 68.38% arasında değişkenlik göstermektedir. FGG modeli optimal çözüme ulaşamadığı için GAP değeri hesaplanamamıştır.

Tablo 7 SET-1 Gevşetilmiş Çözüm Değer Aralığı Yüzdesi (GAP) Tablosu

SET1-Problem	VSS	CSVG	VSSGG (Yeni)	CSVGGG (Yeni)	FGG (Yeni)
c101	-	-	-	-	-
c201	3.06%	-	-	-	-
pr01	1.10%	0.01%	0.01%	0.01%	-
pr02	-	68.38%	-	-	-
pr03	-	-	-	-	-
pr04	-	-	-	-	-
pr05	-	-	-	-	-
pr06	-	-	-	-	-
pr07	-	0.12%	-	-	-
pr08	-	-	-	-	-
pr09	-	-	-	-	-
pr10	-	-	-	-	-
r101	-	-	-	-	-
r201	0.70%	0.14%	-	0.58%	-
rc101	-	-	-	-	-
rc201	-	-	-	-	-

Tablo-8'de SET-2 için GAP değerleri hesaplanmış olup 0.00% ile 36.92% arasında değişkenlik göstermektedir. Veri setleri arasında SET-2 daha küçük boyutlu olduğu için GAP değerleri çoğunlukla hesaplanabilmekte ve modeller yüksek performanslı olarak çalışabilmektedir.

Tablo 8 SET-2 Gevşetilmiş Çözüm Değer Aralığı Karşılaştırma Tablosu

SET2-Problem	VSS	CSVG	VSSGG (Yeni)	CSVGGG (Yeni)	FGG (Yeni)
c101.k10	0.00%	0.01%	0.00%	0.00%	0.00%
c101.k15	1.08%	0.01%	0.00%	0.00%	0.01%
c101.k30	2.23%	0.18%	0.61%	0.03%	-
c101.k40	2.72%	0.33%	1.71%	0.45%	-
pr01.k10	0.00%	0.01%	0.00%	0.00%	0.00%
pr01.k15	0.00%	0.01%	0.00%	0.00%	0.00%
pr01.k30	0.00%	0.01%	0.00%	0.00%	0.00%
pr01.k40	0.01%	0.01%	0.00%	0.00%	-
pr02.k10	0.00%	0.00%	0.00%	0.00%	0.00%
pr02.k15	0.00%	0.01%	0.00%	0.00%	0.00%
pr02.k30	0.01%	0.01%	0.01%	0.01%	-
pr02.k40	0.01%	0.01%	0.01%	0.01%	-
pr03.k10	0.00%	0.00%	0.00%	0.00%	0.00%
pr03.k15	0.00%	0.01%	0.00%	0.00%	0.00%
pr03.k30	0.00%	0.01%	0.00%	0.01%	-
pr03.k40	0.01%	0.01%	0.01%	0.01%	-
pr04.k10	0.00%	0.00%	0.00%	0.00%	0.00%
pr04.k15	0.00%	0.00%	0.00%	0.00%	0.00%
pr04.k30	0.01%	0.01%	0.00%	0.01%	-
pr04.k40	0.01%	0.00%	0.00%	0.00%	-
pr05.k10	0.00%	0.00%	0.00%	0.00%	0.00%

Tablo-8 devam ediyor

pr05.k15	0.00%	0.01%	0.00%	0.01%	0.00%
pr05.k30	0.00%	0.01%	0.00%	0.01%	0.00%
pr05.k40	0.01%	0.01%	0.00%	0.01%	-
pr06.k10	0.00%	0.00%	0.00%	0.00%	0.00%
pr06.k15	0.00%	0.01%	0.00%	0.00%	0.00%
pr06.k30	0.01%	0.01%	0.01%	0.01%	-
pr06.k40	1.93%	0.01%	0.01%	0.01%	-
pr07.k10	0.00%	0.00%	0.00%	0.00%	0.00%
pr07.k15	0.00%	0.01%	0.00%	0.00%	0.00%
pr07.k30	0.01%	0.01%	0.00%	0.01%	-
pr07.k40	0.01%	0.01%	0.00%	0.01%	-
pr08.k10	0.00%	0.00%	0.00%	0.01%	0.00%
pr08.k15	0.00%	0.00%	0.00%	0.00%	0.00%
pr08.k30	0.01%	0.01%	0.00%	0.00%	-
pr08.k40	0.01%	0.01%	0.01%	0.00%	-
pr09.k10	0.00%	0.00%	0.00%	0.00%	0.00%
pr09.k15	0.00%	0.00%	0.00%	0.01%	0.00%
pr09.k30	0.00%	0.01%	0.00%	0.01%	0.00%
pr09.k40	0.01%	0.01%	0.00%	0.01%	-
pr10.k10	0.00%	0.00%	0.00%	0.00%	0.00%
pr10.k15	0.00%	0.00%	0.00%	0.00%	0.00%
pr10.k30	0.00%	0.01%	0.00%	0.00%	0.00%
pr10.k40	0.01%	0.01%	0.00%	0.01%	-
r101.k10	0.00%	0.00%	0.00%	0.00%	0.00%
r101.k15	0.00%	0.01%	0.00%	0.01%	0.00%
r101.k30	-	0.10%	-	0.13%	-
r101.k40	-	19.62%	-	-	-
rc101.k10	0.00%	0.01%	0.00%	0.00%	0.00%
rc101.k15	2.41%	0.01%	0.00%	0.00%	0.00%
rc101.k30	36.92%	25.07%	14.69%	24.90%	-
rc101.k40	-	-	-	-	-

Literatürden referans seçilen modeller ile geliştirilen alternatif modellerin Optimale Ulaşılan Veri Sayısı, Daha Kısa Sürede Optimale Ulaşılan Veri Sayısı, Optimale Ulaşamayıp Optimale Daha Yakın Sonuca Ulaşılan Veri Sayısı ve Ortalama Çözüm Süresi (sn) kriterleri altında performans karşılaştırması yapılmış olup Tablo-9, Tablo 10'da özet olarak gösterilmiştir.

Tablo-9'a göre SET-1 için FGG ve VSS modelleri optimal değerlere ulaşamamıştır. SET-1 içerisinden yalnızca 1 problem CSVG, CSVGGG ve VSSGG modelleri tarafından çözülmüştür. Bu problem için daha kısa sürede optimal sonuç veren model CSVGGG'dir. CPLEX programında üst limit olarak verilen 3 saatlik süre kısıtında optimal sonuca 2 problem ile en çok VSS ve CSVG modelleri yaklaşmıştır. FGG ve VSS modelleri optimal sonuca ulaşamadıkları için ortalama çözüm süresi hesaplanamamıştır. Optimale ulaşan çözümler için hesaplanan ortalama çözüm süreleri karşılaştırıldığında modeller arasında CSVGGG'nin en yüksek performansa sahip olduğu görülmektedir.

Tablo 9 SET-1 Sonuç Değerlendirme Tablosu

	Optimale Ulaşılan Veri Sayısı				
SET1	**VSS**	**CSVG**	**VSSGG (Yeni)**	**CSVGGG (Yeni)**	**FGG (Yeni)**
	0	1	1	1	0
	Daha Kısa Sürede Optimale Ulaşılan Veri Sayısı				
	VSS	**CSVG**	**VSSGG (Yeni)**	**CSVGGG (Yeni)**	**FGG (Yeni)**
	0	0	0	1	0
	Optimale Daha Yakın Sonuca Ulaşan Veri Sayısı				
	VSS	**CSVG**	**VSSGG (Yeni)**	**CSVGGG (Yeni)**	**FGG (Yeni)**
	2	2	0	0	0
	Ortalama Çözüm Süresi (sn)				
	VSS	**CSVG**	**VSSGG (Yeni)**	**CSVGGG (Yeni)**	**FGG (Yeni)**
	-	1496	582	418	-

Tablo-10'a göre SET-2 için FGG modeli optimale ulaşma açısından en düşük performansa sahiptir. Daha kısa sürede optimal sonuç veren modeller incelendiğinde VSSGG modeli 41 adet problemde yüksek performans göstermiştir. CPLEX programında üst limit olarak verilen 3 saatlik süre kısıtında optimal sonuca 2 veri ile CSVG, VSSGG ve CSVGGG modelleri yaklaşmıştır. Ortalama çözüm süreleri karşılaştırıldığında modeller arasında CSVGGG'nin en yüksek performansa sahip olduğu görülmektedir. En düşük performans 257 saniye ile VSS modelinindir.

Tablo 10 SET-2 Sonuç Değerlendirme Tablosu

SET2	**Optimale Ulaşılan Veri Sayısı**				
	VSS	**CSVG**	**VSSGG (Yeni)**	**CSVGGG (Yeni)**	**FGG (Yeni)**
	43	46	46	46	30
	Daha Kısa Sürede Optimale Ulaşılan Veri Sayısı				
	VSS	**CSVG**	**VSSGG (Yeni)**	**CSVGGG (Yeni)**	**FGG (Yeni)**
	2	1	**41**	2	0
	Optimale Daha Yakın Sonuca Ulaşan Veri Sayısı				
	VSS	**CSVG**	**VSSGG (Yeni)**	**CSVGGG (Yeni)**	**FGG (Yeni)**
	0	2	2	2	0
	Ortalama Çözüm Süresi (sn)				
	VSS	**CSVG**	**VSSGG (Yeni)**	**CSVGGG (Yeni)**	**FGG (Yeni)**
	257	71	22	66	72

Tüm modeller değerlendirildiğinde, ayrıca bir alt tur engelleme kısıtı olmadan model içerisinde alt turlar engellenmek istendiğinde performansta düşüş yaşanmaktadır. Diğer modellerde 3 indis ile birlikte alt tur engelleme kısıtı kullanılırken, FGG modelinde 4 indis kullanılarak alt tur engelleme kısıtı kaldırılmış, buna karşılık modelin çözülmesi zorlaşmıştır.

5 SONUÇ VE ÖNERİLER

Bu çalışma ile literatürde bulunan OSGSP matematiksel modellerine göre performans anlamında daha iyi yeni alternatif modeller oluşturulmaya çalışılmıştır.

SET-1 problemleri büyük ölçekli, SET-2 problemleri küçük ölçekli problemler olarak adlandırılacak olursa, bu tablolara göre;

Optimale ulaşılan veri sayısı kriter olarak alındığında, büyük ve küçük ölçekli problemlerde CSVG, VSSGG ve CSVGGG modellerinin diğerlerine göre daha etkin olduğu görülmektedir. Daha kısa sürede optimal değere ulaşma sayısı kriter olarak alındığında; büyük ölçekli problemlerde CSVGGG, küçük ölçekli problemlerde ise VSSGG modelinin diğer modellere göre daha etkin olduğu görülmektedir. Ortalama çözüm süresi kriteri altında ise büyük ölçekli problemlerde CSVGGG, küçük ölçekli problemlerde VSSGG modelinin diğer modellerden daha etkin olduğu görülmektedir.

Bu sonuçlardan çıkarımla gerçek hayatta küçük ölçekli problemlerde hızlı sonuç alınmak istendiği durumda VSSGG modelinin kullanılması önerilmektedir.

VSS modeli otel, müşteri, gezi süresi, gezi sayısı, noktalar arası geçen zaman, noktalarda geçen zaman parametrelerini kullanarak çözüm sağlarken, CSVG modeli bu parametrelerden gezi sayısı bilgisini kullanmadan kendisi hesaplamaktadır, bu nedenle gezi sayısı bilinmediği durumda CSVG modellerinin kullanılması uygun olacaktır.

Gelecek çalışmalarda sezgisel veya metasezgisel yöntemler ile daha büyük boyutlu prolemlerin çözümü üzerinde çalışmalar yürütülebilir. Modellere araç kapasitesi, otellerde geçen süre, öncelikli müşteriler gibi parametreler eklenerek çalışmanın yelpazesi genişletilebileceği gibi sıralı veya paralel çoklu tur planlama üzerine çalışmalar da yapılabilir. Ayrıca OSGSP çok amaçlı olarak ele alınarak yeni modeller geliştirilebilir.

KAYNAKLAR LİSTESİ

[1] M. Castro, K. Sörensen, P. Vansteenwegen ve P. Goos, *A Fast Metaheuristic For The Travelling Salesperson Problem With Hotel Selection,* 2015.

[2] M. Castro, K. Sörensen, P. Vansteenwegen ve P. Goos, «A Memetic Algorithm For The Travelling Salesperson Problem With Hotel Selection,» *Computers & Operations Research,* 2013.

[3] P. Vansteenwegen, W. Souffriau ve K. Sörensen, «Solving the mobile mapping van problem : A hybrid metaheuristic for capacitated arc routing with soft time windows,» *Computers &OperationsResearch,* 2010.

[4] W. Souffriau, P. Vansteenwegen, J. Vertommen, G. V. Berghe ve D. V. Oudheusden, «A Personalised Tourist Trip Design Algorithm for Mobile Tourist Guides,» *Applied Artificial Intelligence,* no. 22, p. 964–985, 2008.

[5] P. Vansteenwegen, W. Souffriau ve K. Sörensen, «The Travelling Salesperson Problem With Hotel Selection,» *Journal of the Operational Research Society,* 2012.

[6] G. Laporte, *A Short History of the Traveling Salesman Problem,* Montr´eal, Canada, 2006.

[7] S. ÖZKAN, *GEZGİN SATICI PROBLEMİNİN ÇÖZÜMÜNE YÖNELİK ALGORİTMİK YAKLAŞIMLAR,* Ankara, 2010.

[8] M. Castro, K. Sörensen, P. Goos ve P. Vansteenwegen, *The multiple travelling salesperson problem with hotel selection,* Belgium, 2014.

[9] A. Divsalar, P. Vansteenwegen, K. Sörensen ve D. Cattrysse, «A memetic algorithm for the orienteering problem with hotel selection,» *European Journal of Operational Research,* 2014.

[10] T. BEKTAŞ, «The Multiple Traveling Salesman Problem: An Overview of Formulations And Solution Procedures,» *OMEGA,* 2006.

[11] D. M. F. DEMİRAL ve A. K. DEMİRAL, %1 içinde *GEZGİN ALICI PROBLEMİNE YÖNELİK BİR MODEL ÖNERİSİ*, 2014.

[12] M. M. Sousa, I. M. Coelho ve L. B. Gonçalves, «Latin American Computing Conference (CLEI),» %1 içinde *A Variable Neighborhood Search Heuristic for the Traveling Salesman Problem with Hotel Selection*, 2015.

[13] T. Öncan, K. Altınel ve G. Laporte, «A Comparative Analysis of Several Asymmetric Traveling Salesman Problem Formulations,» *Computers & Operations Research,* 2009.

[14] «ANT/OR,» [Çevrimiçi]. Available: http://antor.uantwerpen.be/instances-in-the-paper-a-memetic-algorithm-for-the-travelling-salesperson-problem-with-hotel-selection/. [Erişildi: 16 Aralık 2018].

[15] M. M. Solomon, «Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints,» JSTOR, 1987, p. 254–265.

[16] J. Cordeau, G. Laporte ve A. Mercier, «A unified tabu search heuristic for vehicle routing problems with time windows,» J Opl Res Soc, 2001, p. 928–936.

EKLER LİSTESİ

EK-1 VSS Modeli OPL KODU

```
using CPLEX;
int s = ...;
int n = ...;
int m = ...;
int C = ...;
float T[1..s+n] = ...;
float XPos[1..s+n] = ...;
float YPos[1..s+n] = ...;
float c[1..(s+n)][1..(s+n)];
dvar boolean x[1..(s+n)][1..(s+n)][1..m];
dvar float+ u[(s+1)..(s+n)];
execute Compute_c
{
	for(var k=1; k<=s+n; k++)
		for(var l=1; l<=s+n; l++)
		{
			if(k!=l)
				c[k][l]   =   Opl.trunc(Opl.sqrt(   Opl.pow(XPos[k]-
XPos[l],2) + Opl.pow(YPos[k]-YPos[l],2) ) * 10)/10;
		}
	}

minimize
	sum(d in 1..m, k in 1..s+n, l in 1..s+n: k!=l)
	  (x[k,l,d]*(c[k,l]+T[l]));

subject to
{
	sum(l in 1..s+n) x[1,l,1] == 1;

	sum(k in 1..s+n) x[k,1,m] == 1;

forall (d in 1..m)
  sum(h in 1..s, k in 1..s+n)x[k,h,d]==1;

forall (d in 1..m)
  sum(h in 1..s, l in 1..s+n)x[h,l,d]==1;

if(m>=2)
{
	forall (d in 1..(m-1), h in 1..s)
		sum(k in 1..s+n)x[k,h,d]-sum(l in 1..s+n)x[h,l,(d+1)]==0;
}

forall (i in s+1..s+n)
  sum(d in 1..m, k in 1..s+n)x[k,i,d]==1;

forall (d in 1..m, i in s+1..s+n)
  sum(k in 1..s+n)x[k,i,d]-sum(l in 1..s+n)x[i,l,d]==0;

forall (d in 1..m)
  sum(k in 1..s+n, l in 1..s+n)(x[k,l,d]*(c[k,l]+T[k]))<=C;

forall (i in s+1..s+n, j in s+1..s+n)
  u[i]-u[j]+1<=n*(1-(sum(d in 1..m)x[i,j,d])); }
```

EK-2 VSS-Gravish and Graves OPL KODU

```
using CPLEX;
int s = ...;
int n = ...;
int m = ...;
int C = ...;
float T[1..s+n] = ...;
float XPos[1..s+n] = ...;
float YPos[1..s+n] = ...;
float c[1..(s+n)][1..(s+n)];
dvar boolean x[1..(s+n)][1..(s+n)][1..m];
dvar float+ g[1..(s+n)][1..(s+n)];
execute Compute_c
{for(var k=1; k<=s+n; k++)
             for(var l=1; l<=s+n; l++)
             {if(k!=l)
                            c[k][l]   =   Opl.trunc(Opl.sqrt(   Opl.pow(XPos[k]-
XPos[l],2) + Opl.pow(YPos[k]-YPos[l],2) ) * 10)/10; }
}
 minimize
  sum(d in 1..m, k in 1..s+n, l in 1..s+n: k!=l)
    (x[k,l,d]*(c[k,l]+T[l]));
subject to
{
  sum(l in 1..s+n) x[1,l,1] == 1;

  sum(k in 1..s+n) x[k,1,m] == 1;

  forall (d in 1..m)
    sum(h in 1..s, k in 1..s+n) x[k,h,d] == 1;

  forall (d in 1..m)
    sum(h in 1..s, l in 1..s+n) x[h,l,d] == 1;

  if(m>=2)
  {
    forall (d in 1..(m-1), h in 1..s)
      sum(k in 1..s+n) x[k,h,d] == sum(l in 1..s+n) x[h,l,(d+1)];
  }

  forall (i in s+1..s+n)
    sum(d in 1..m, k in 1..s+n) x[k,i,d] == 1;

  forall (d in 1..m, i in s+1..s+n)
    sum(k in 1..s+n) x[k,i,d] == sum(l in 1..s+n) x[i,l,d];

  forall (d in 1..m)
    sum(k in 1..s+n, l in 1..s+n) x[k,l,d]*(c[k,l]+T[k]) <= C;

  forall (i in s+1..s+n)
    sum(j in 1..s+n: i!=j) g[j,i] - sum(j in 1..s+n: i!=j) g[i,j] == 1;

  forall ( i in 1..s+n, j in 1..s+n: i!=j)
    g[i,j] <= (n)*sum(d in 1..m) x[i,j,d];
}
```

EK-3 CSVG Modeli OPL KODU

```
using CPLEX;
int s = ...;
int n = ...;
int m = 15;//set1 ve set2 de max trip sayısı 9 dur, bundan büyük bir değer
alınmıştır
int C = ...;
int M = 10000;//set1 çalıştırılacaksa 10000, set2 çalıştırılacaksa 5000
alınmalı
float T[1..s+n] = ...;
float XPos[1..s+n] = ...;
float YPos[1..s+n] = ...;
float c[1..(s+n)][1..(s+n)];
dvar boolean x[1..(s+n)][1..(s+n)][1..m];
dvar boolean y[1..m];
dvar float+ u[(s+1)..(s+n)];
execute Compute_c
{
	for(var k=1; k<=s+n; k++)
		for(var l=1; l<=s+n; l++)
		{
			if(k!=l)
				c[k][l] = Opl.trunc(Opl.sqrt( Opl.pow(XPos[k]-
XPos[l],2) + Opl.pow(YPos[k]-YPos[l],2) ) * 10)/10;
		}
}
minimize
M*sum(d in 1..m)y[d] + sum(d in 1..m, k in 1..s+n, l in 1..s+n: k!=l)
(x[k,l,d]*(c[k,l]+T[l]));
subject to
{
forall(d in 1..m)
  sum(h in 1..s, g in 1..s) x[h,g,d]==0;

forall (l in s+1..s+n)
  sum(d in 1..m, k in 1..s+n)x[k,l,d]==1;

forall(l in s+1..s+n, d in 1..m)
  sum(k in 1..s+n)x[k,l,d]==sum(k in 1..s+n)x[l,k,d];

forall(d in 1..m)
  sum(h in 1..s, l in 1..s+n: l!=h)x[h,l,d]==y[d];

forall(d in 1..m)
  sum(h in 1..s, k in 1..s+n: k!=h)x[k,h,d]==y[d];

forall(d in 1..m)
  sum(k in 1..s+n, l in 1..s+n:k!=l) (c[k,l]+T[l])*x[k,l,d]<=C;

sum(l in 1..s+n:l!=1) x[1,l,1]==1;

forall (d in 1..m-1)
  sum(k in 1..s+n:k!=1) x[k,1,d]>=y[d]-y[d+1];

forall(h in 1..s, d in 1..m-1)
  sum(k in 1..s+n)x[k,h,d]+y[d]>=sum(k in 1..s+n)x[h,k,d+1]+y[d+1];
```

```
forall(h in 1..s, d in 1..m-1)
  sum(k in 1..s+n)x[k,h,d]-sum(k in 1..s+n)x[h,k,d+1]<=1-y[d+1];

forall(d in 1..m, k in 1..s+n, l in 1..s+n: k!=l)
  x[k,l,d]<=y[d];

forall (d in 1..m-1)
  y[d]>=y[d+1];

forall (k in s+1..s+n, l in s+1..s+n)
  u[k]-u[l]+1<=(n)*(1-(sum(d in 1..m)x[k,l,d]));

}
```

EK-4 CSVG-Gravish and Graves OPL KODU

```
using CPLEX;
int s = ...;
int n = ...;
int m = 15;//set1 ve set2 de max trip sayısı 9 dur, bundan büyük bir değer alınmıştır
int C = ...;
int M = 10000;//set1 çalıştırılacaksa 10000, set2 çalıştırılacaksa 5000 alınmalı
float T[1..s+n] = ...;
float XPos[1..s+n] = ...;
float YPos[1..s+n] = ...;
float c[1..(s+n)][1..(s+n)];
dvar boolean x[1..(s+n)][1..(s+n)][1..m];
dvar boolean y[1..m];
dvar float+ g[1..(s+n)][1..(s+n)];
execute Compute_c
{
	for(var k=1; k<=s+n; k++)
		for(var l=1; l<=s+n; l++)
		{
			if(k!=l)
				c[k][l] = Opl.trunc(Opl.sqrt( Opl.pow(XPos[k]-XPos[l],2) + Opl.pow(YPos[k]-YPos[l],2) ) * 10)/10;
		}
}
minimize
M*sum(d in 1..m)y[d] + sum(d in 1..m, k in 1..s+n, l in 1..s+n: k!=l)(x[k,l,d]*(c[k,l]+T[l]));

subject to
{
forall(d in 1..m)
  sum(h in 1..s, g in 1..s) x[h,g,d]==0;

forall (l in s+1..s+n)
  sum(d in 1..m, k in 1..s+n)x[k,l,d]==1;

forall(l in s+1..s+n, d in 1..m)
  sum(k in 1..s+n)x[k,l,d]==sum(k in 1..s+n)x[l,k,d];

forall(d in 1..m)
  sum(h in 1..s, l in 1..s+n: l!=h)x[h,l,d]==y[d];

forall(d in 1..m)
  sum(h in 1..s, k in 1..s+n: k!=h)x[k,h,d]==y[d];

forall(d in 1..m)
  sum(k in 1..s+n, l in 1..s+n:k!=l) (c[k,l]+T[l])*x[k,l,d]<=C;

sum(l in 1..s+n:l!=1) x[1,l,1]==1;

forall (d in 1..m-1)
  sum(k in 1..s+n:k!=1) x[k,1,d]>=y[d]-y[d+1];

forall(h in 1..s, d in 1..m-1)
```

```
  sum(k in 1..s+n)x[k,h,d]+y[d]>=sum(k in 1..s+n)x[h,k,d+1]+y[d+1];

forall(h in 1..s, d in 1..m-1)
  sum(k in 1..s+n)x[k,h,d]-sum(k in 1..s+n)x[h,k,d+1]<=1-y[d+1];

forall(d in 1..m, k in 1..s+n, l in 1..s+n: k!=l)
  x[k,l,d]<=y[d];

forall (d in 1..m-1)
  y[d]>=y[d+1];

forall (k in s+1..s+n)
    sum(l in 1..s+n: k!=l) g[l,k] - sum(l in 1..s+n: k!=l) g[k,l] == 1;

forall ( k in 1..s+n, l in 1..s+n: k!=l)
    g[k,l] <= (n)*sum(d in 1..m) x[k,l,d];
}
```

EK-5 FGG OPL KODU

```
using CPLEX;
int s = ...;
int n = ...;
int m = ...;
int C = ...;
float T[1..s+n] = ...;
float XPos[1..s+n] = ...;
float YPos[1..s+n] = ...;
float c[1..(s+n)][1..(s+n)];
dvar boolean x[1..(s+n)][1..(s+n)][1..m][1..(s+n)];
execute Compute_c
{
	for(var k=1; k<=s+n; k++)
		for(var l=1; l<=s+n; l++)
		{
			if(k!=l)
				c[k][l]   =   Opl.trunc(Opl.sqrt(   Opl.pow(XPos[k]-
XPos[l],2) + Opl.pow(YPos[k]-YPos[l],2) ) * 10)/10;
		}
	}
 minimize
	sum(d in 1..m, k in 1..s+n, l in 1..s+n, u in 1..s+n: k!=l)
	  (x[k,l,d,u]*(c[k,l]+T[l]));

subject to
{
	sum(l in 2..s+n) x[1,l,1,1] == 1;

	sum(k in 2..s+n, u in 1..s+n) x[k,1,m,u] == 1;

forall (d in 1..m)
  sum(h in 1..s, k in 1..s+n, u in 1..s+n)x[k,h,d,u]==1;

forall (d in 1..m)
  sum(h in 1..s, l in 1..s+n, u in 1..s+n)x[h,l,d,u]==1;

if(m>=2)
{
	forall (d in 1..(m-1), h in 1..s)
		sum(k in 1..s+n,u in 1..s+n)x[k,h,d,u]-sum(l in 1..s+n)
x[h,l,(d+1),1]==0;
}

forall (i in s+1..s+n)
  sum(d in 1..m, k in 1..s+n, u in 1..s+n)x[k,i,d,u]==1;

forall (d in 1..m, i in s+1..s+n)
  sum(k in 1..s+n, u in 1..s+n)x[k,i,d,u]-sum(l in 1..s+n, u in 1..s+n)
x[i,l,d,u]==0;

forall (d in 1..m)
  sum(k in 1..s+n, l in 1..s+n, u in 1..s+n)(x[k,l,d,u]*(c[k,l]+T[k]))<=C;

forall (i in s+1..s+n)
```

```
  sum(d in 1..m, k in 1..s+n : i!=k, u in 1..s+n) u*x[i,k,d,u]-sum(d in 1..m, k
in 1..s+n: i!=k, u in 1..s+n)u*x[k,i,d,u]==1;
}
```