KTO KARATAY
ÜNİVERSİTESİ

T.C.
KTO Karatay Üniversitesi
Fen Bilimleri Enstitüsü

ELEKTRİK VE BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
TEZLİ YÜKSEK LİSANS PROGRAMI

TOPLU TAŞIMA AĞLARINDA ZAMANA BAĞLI SORGU TABANLI BİR GÜZERGAH PLANLAMA TEKNİĞİ

Mustafa YILDIRIM

KONYA

Haziran 2019

TOPLU TAŞIMA AĞLARINDA ZAMANA BAĞLI SORGU TABANLI BİR GÜZERGAH PLANLAMA TEKNİĞİ

Mustafa YILDIRIM

KTO Karatay Üniversitesi Fen Bilimleri Enstitüsü

Elektrik ve Bilgisayar Mühendisliği Ana Bilim Dalı
Yüksek Lisans Programı

Yüksek Lisans Tezi

Haziran, 2019

Fen Bilimleri Enstitü Onayı

Fen Bilimleri Enstitüsü Müdür
Prof. Dr. Hüseyin Bekir Yıldız

Bu tezli yüksek lisans tezinin yapılması gereken bütün gerekliliklerinin yerine getirdiğini onaylıyorum.

Anabilim Dalı Başkanı
Dr. Öğr. Üyesi H. Oktay ALTUN

Mustafa YILDIRIM tarafından hazırlanan TOPLU TAŞIMA AĞLARINDA ZAMANA BAĞLI SORGU TABANLI BİR GÜZERGAH PLANLAMA TEKNİĞİ başlıklı bu çalışma 05.07.2019 tarihinde yapılan savunma sınavı sonucunda başarılı bulunarak jüri tarafından tezli yüksek lisans tezi olarak kabul edilmiştir.

Tez Danışmanı
Dr. Öğr. Üyesi H. Oktay ALTUN

Jüri Üyeleri

Başkan: Dr. Öğr. Üyesi Ahmet Reha BOTSALI

Üye: Dr. Öğr. Üyesi H. Oktay ALTUN

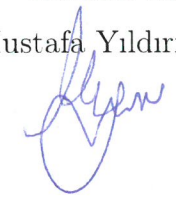
Üye: Dr. Öğr. Üyesi Amir YAVARİABDİ

Tez Bildirimi

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, ayrıca tez yazım kurallarına uygun olarak hazırlanan bu çalışmada orjinal olmayan her türlü kaynağa eksiksiz atıf yapıldığını, kullanılan verilerde herhangi bir değişiklik yapmadığımı, bu tezde sunduğum çalışmanın özgün olduğunu bildirir aksi bir durumda aleyhime doğabilecek tüm hak ve kayıplarını kabullendiğimi beyan ederim.

Haziran-2019

Mustafa Yıldırım



Özet

TOPLU TAŞIMA AĞLARINDA ZAMANA BAĞLI SORGU TABANLI BİR GÜZERGAH PLANLAMA TEKNİĞİ

Mustafa YILDIRIM

KTO Karatay Üniversitesi,
Fen Bilimleri Enstitüsü,
Elektrik ve Bilgisayar Mühendisliği Anabilim Dalı Yüksek Lisans Tezi

Tez Danışmanı: Dr. Öğr. Üyesi H. Oktay ALTUN

Haziran 2019

Bu çalışmada toplu taşıma ağlarında güzergah planlama problemini sorgu tabanlı bir algoritma ile çözdük. Bu yaklaşımda öncelikle toplu taşıma hatlarının güzergahlarında bulunan durakları, bu durakların sıralarını, ardışık duraklar arası geçme zamanlarını ve haftalık tarifelerini tablolaştırıyoruz. Akabinde; bu birincil tablolardan, hatları ikili ve daha çoklu kesiştirerek, kesişim duraklarının listelendiği yeni ikincil tablolar oluşturuyoruz. Tüm bu birincil ve ikincil tablolar üzerinde sorgular koşturarak alternatif tüm güzergahları oluşturuyoruz. Çıkan sonuçların en kısa yolculuk süresine göre sıralanması ile tüm çözümleri listeliyoruz. Geliştirdiğimiz algoritmayı öncelikle bir test senaryosu üzerinde uyguladık. Daha sonra Konya şehri için toplu taşıma güzergah uygulaması haline getirdik. Sorgu tabanlı uygulamamız tek veya daha çok hat kullanılan çözümlerde oldukça kısa sürelerde sorgulara cevap verebilmektedir.

Anahtar kelimeler: Toplu taşıma ağlarında güzergah planlama, sorgu tabanlı güzergah planlama algoritmaları

Abstract

A TIME-BASED ROUTE PLANNING TECHNIQUE IN MASS TRANSPORT NETWORKS

Mustafa YILDIRIM

KTO Karatay University,
The Graduate School of Natural and Applied Sciences,
Master of Science Thesis in Electrical and Computer Engineering

Advisor: Asst. Prof. H. Oktay ALTUN

June 2019

In this study, we solved the problem of route planning in public transport networks by a query based algorithm. In this approach, we first tabulate the stops located on the routes of public transport lines as well as the order of these stops, the passing times between consecutive stops and weekly tariffs. Subsequently; from these primary tables, we create new secondary tables listing intersection stops by intersecting the lines in binary and multiple-combinations. We are creating all alternative routes by running queries on all these primary and secondary tables. We list all the solutions by sorting the results according to the shortest journey time. We developed the algorithm firstly on a test scenario. Then we have programmed the public transport itinerary application for a city in Turkey, Konya. Our query-based application is able to respond, queries quite quickly in scenarios involving single or more lines.

Keywords: Route planning in public transport networks, query-based route planning algorithms

Teşekkür

Çalışmalarım boyunca bana her türlü fedakarlıklarını gösteren kıymetli eşim Fatma YILDIRIM'a, kıymetli anneme, destekleri ve tüm bilgi birikimiyle sürekli bana destek olan ve kodların yazılımında payı büyük olan ağabeyim Ali ÖZKAN'a, tezin yazılımında katkıları bulunan arkadaşım Hatice Tekiş'e ve her zaman desteğini arkamda hissettiğim Dr. Öğr. Üyesi H. Oktay ALTUN Hocam'a teşekkürü borç bilirim.

Bu tezi, tezimi hazırlarken kaybettiğim kıymetli babacığım Veli YILDIRIM'a ithaf ediyorum.

Mustafa YILDIRIM
Haziran-2019

İçindekiler

Şekil Listesi

Tablo Listesi

Kısaltmalar

Kısaltmalar	Açıklama
A*	A yıldız algoritması
ark.	Arkadaşları
MC	Mümkün Çözümler
NP Hard	Non-deterministic polynomial-time
QoS	Quality of Service (Servis Kalitesi)
SD	Seçilen Düğüm
RAPTOR	Round-Based Public Transit Optimized Router
T-share	Sadece taksi paylaşımı yapan bir ağ

Semboller

Simgeler	Açıklama
A	Birinci döngüdeki başlangıç hat adı
$A, .., Y$	Kurgusal harita gidiş durak etiketleri
$A', .., Y'$	Kurgusal harita dönüş durak etiketleri
B	Birinci döngüdeki varış hat adı
C	İkinci döngüdeki başlangıç hat adı
D	İkinci döngüdeki varış hat adı
M	Birinci döngüdeki hat adı
N	İkinci döngüdeki hat adı
n	Hat kesişim sayısı

1 Giriş

Toplu taşıma sistemlerinde yapılan güzergah planlama çalışmaları akıllı şehir sistemleri için çok önemlidir. Bu çalışmalar sayesinde en optimal güzergah hesaplanarak gittikçe artan trafik sıkışmasına bir çözüm sunulabilir. Gene bu çalışmalar, etkili bir trafik planlama stratejisi oluşturulmasının yanında, enerji tüketiminin, çevre kirliğinin ve toplam toplu ulaşım maliyetinin de azalmasına yardımcı olacaktır. Literatürde toplu taşıma ağları için 1) ön-hesaplama süresi, 2) sorgu süresi ve 3) hafıza gereksinimleri yönünden farklılık arzeden bir çok farklı algoritmalar geliştirilmiştir. Algoritmalardaki bu üç önemli özellik değiş-tokuş dengesine sahiptir. Mesela bazı algoritmalar herhangi bir ön hesaplama gerektirmemekle birlikte, önemli miktarda bir sorgu süresine sahiptirler. Bazıları ise ön hesaplama ve hafıza gereksinimleri açısından çok maliyetli olmakla birlikte, istenilen güzergah için hızlı güzergah sorgulama süresi çok kısadır [1].

Toplu taşıma için güzergah planlama problemleri, standart güzergah planlama problemlerine göre daha zor problemlerdir, çünkü zamana bağlıdırlar ve sorgularda çoklu kriterler istenebilir. Bu problemlerde zaman sınırlarının olduğu tek tip ulaşım vasıtaları varsa ona tek modlu, bu vasıtaların yanında bisiklet, binek araç veya yürüme gibi herhangi bir zaman sınırlamasının olmadığı vasıtalar da mevcutsa bunlara çoklu modlu problemler adı verilir [1]. Biz bu çalışmada tek modlu (sadece otobüs ve tramvay gibi tarifeli toplu ulaşım araçları için) yolcu güzergah problemi üzerinde bir çalışma gerçekleştirdik.

Güzergah planlama, en kısa yol bulma algoritmalarıyla veya zaman bağımlı sistemler için sorgu yaklaşımı ile yapılabilmektedir. En kısa yol bulma algoritmaları genellikle statik ağlarda kullanılmaktayken, tarife tabanlı sistemler de zaman bağımlı sistemler için kullanılmaktadır. Statik ağlarda en kısa yolu bulmak için genellikle graf (graph) teorisi yaklaşımları kullanılmaktadır. Literatürde yapılan çalışmalara baktığımızda statik ağlar için çeşitli sezgisel en kısa yol algoritmaları geliştirilmiş olsa da şu an bir en iyi çözüm hala sunulamamıştır. Bu problemler, gezgin satıcı problemi olarak geçmektedir ve çözülmesi zor olan (NP hard) problemlerdir. Yine de hesaplama karmaşıklığı (computational complexity) yönünden

kullanşlı çözümler önerilmiştir.

Naoum-Sawaya ve ark. evlerinden iş yerlerine giden çalışanlara araba tahsisi için olasılıksal (stochastic) karmaşık tam sayı programı kullanarak bir çalışma gerçekleştirmiştir [2]. Shang ve ark., birden fazla başlangıç noktasını ve limitli sayıda buluşma noktası olan bir varış noktasını birleştirmek için en düşük maliyetli yol planlama algoritması geliştirmişlerdir [3]. Algoritma toplu seyahat planlama sorgularına cevap vererek çalışmaktadır. Massobrio ve ark. tek kaynaklı ve çoklu varış senaryoları için yolcuların taksi paylaşımında genetik algoritmaları kullanarak bir çözüm önermiştir [4]. Zhu ve ark. QoS kısıtlamaları olan taşıtlar için seyahat mesafesini azaltmak amacıyla yeni bir algoritma geliştirmiştir [5]. T-share ve Via'dan farklı olarak araba, otobüs ve taksiler için yaygın kullanılan ulaşım ağları için yeni bir yaklaşım getirmiştir. Yine Zue ve ark., başka bir çalışmasında toplu taşıma araçları için çevrimiçi/dinamik bir güzergah planlamasına bir çözüm getirmiştir. Dolambaçlı yollar gibi QoS kısıtlamalarını ihlal eden talepleri filtrelemek için limitli arama alanında çalışan yeni bir algoritma geliştirmişlerdir. Bozkurt ve ark., seyahat süresi, mesafesi ve yakıt tüketimi gibi çoklu faktörlü yol planlaması için sezgisel yaklaşımlardan biri olan A* algoritması ile bellek kullanımında iyileştirmeler yapmıştır [6].

Zamana bağlı stokastik algoritmalar üzerinde de bir çok çalışma yapılmıştır. Gao ve ark., stokastik zaman bağımlı şebekelerde, optimal rotalama problemleri için teorik bir yapı kurmuşlardır. Optimal rotalama problemlerini türleri, stokastik bağımlılık ve gerçek zamanlı verinin var olmasına göre sınıflandırmışlardır. Problemin bazı değiskenleri üzerinde çalışmışlardır. Muhtemel en az zaman yolu bulmak için bir algoritma geliştirilmiştir. Bu algoritma muhtemel en az zaman yolunu ve seyahat süresinin gerçekleşme ihtimalini vermiştir [7]. Sürekli en kısa yola stokastik tutarlılık ve stokastik üstünlük, sürekli fonksiyonu alt ve üst sınırlar ile sıkıştırarak ulaşmaya çalışmıştır [8]. Öte yandan biz bu çalışmada toplu taşıma araçlarının zamanlarında bir stokastik bileşen olmadığını varsayıyoruz.

2 Arkaplan Bilgileri

Toplu taşıma ağında güzergah planlama algoritmalarının anlaşılabilmesi için bu bölümde bazı kavramsal altyapı bilgilerine kısaca değindik.

2.1 Graf Teorisi

Graf Teorisi, basitçe ifade edilecek olursa düğüm olarak adlandırılan noktalar ve bu noktaları birleştiren hatlardan oluşan ve geometrik ya da konumsal bir bilgi vermeyip, sadece düğümler arasındaki ilişkiyi gösteren çizgiler topluluğudur. Diğer bir ifadeyle graf gerçek hayatta karşılaşılan birçok problemi, mantıksal ilişki kurarak problemi göstermeye yarayan bir ağ yapısıdır. İlk olarak graf teorisinin doğuşu 1700'lü yıllardaki meşhur problem olan Königsberg'in 7 köprüsü problemi ile ortaya çıkmıştır. Königsberg şehri Pregel nehrinin kıyısına kurulmuş ve 7 ayrı köprü ile birbirine bağlanmış 4 farklı bölümden oluşmaktadır. Zamanla insanlar kendilerine, aynı köprüden bir kez daha geçmemek üzere tüm şehri dolaşmanın mümkün olup olmadığı sorusunu sormuşlardır. Ancak böyle bir rota çizilememiştir. Sadece Leonhard Euler bunun neden mümkün olamayacağını ispatlayabilmiştir.

Graf teorisi uygulamaları ile günlük yaşamda birçok problemin çözümü bulunabilmektedir. Fizik, kimya gibi temel bilim alanlarında, elektrik ve elektronik mühendisliğinde, ulaşımda, otoyolları ve havayollarında, bilgisayar mühendisliği ve bilgisayar bilimlerinde ve daha birçok alanda kullanılmaktadır. Grafta düğümlerin ve bu noktaları birleştiren kenarlar kümesinin tanımlanması gerekmektedir. Daha sonra da hangi kenarın hangi düğümleri bağladığının belirtilmesi gerekmektedir. Graf teorisi en kısa yol problemlerinde de sıkça kullanılan bir yaklaşım modelidir. Bir noktadan başka bir noktaya giderken en kısa yolun tercih edilmesini ve tüm maliyetin en aza indirilmesini amaçlar. Böyle bir problemde başlangıç ve varış noktaları için iki düğüm noktası ve düğümlerin birbirine bağlantısını gösteren kenar yolları ve ağırlıklar da kenarların uzunluklarını temsil etmektedir. Şehirler arası veya şehir içi iki nokta arası karayolu ile seyahat veya havayolu ile seyahat en kısa

yol problemlerinin başlıca örneklerindendir. Bu ve bunun gibi pek çok problem graflar ile ifade edilebilir [9].

2.2 En Kısa Yol Algoritmaları

En kısa yol problemleri, graf teorisinin en önemli uygulama alanlarından biridir. Problem, ağırlıklı bir graf üzerinde iki düğüm arasındaki toplam ağırlığı en düşük olan bağlantıları bulmayı amaçlayan bir optimizasyon problemidir. G adında bir grafımız olsun. $G = (V, A)$ şekilindeki gösterimde $V = V(G)$ düğümler kümesini ve $A = A(G)$ ise kenarlar kümesini temsil etmektedir. $A(u, v)$ yani u ve v düğümlerinin oluşturduğu kenarı temsil etmektedir. $l(u, v)$ ise bu kenarın uzunluğunu temsil etmektedir. Her kenar negatif olamayan uzunluğa sahiptir. s ve t de G grafının düğümlerindendir. Bir yolun uzunluğu, kenarların uzunluklarının toplamıdır. Problemin modellenmesi ile en kısa yol algoritmaları geliştirilmiştir.

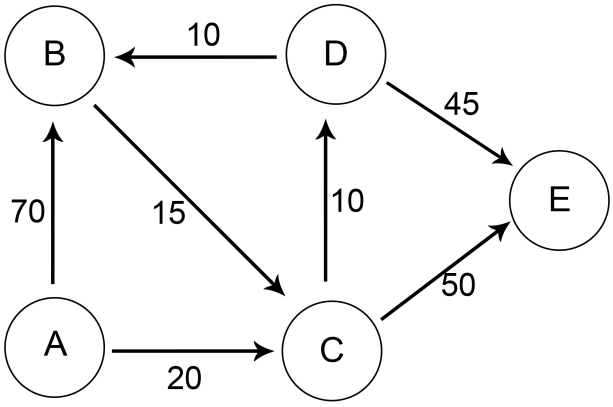
2.3 Statik Ulaşım Ağlarında En Kısa Yol Algoritmaları

2.3.1 Dijkstra algoritması

Dijkstra Algoritması yazılım dünyasının ötesinde matematik dünyasında ve konumuzla alakalı olan en kısa yol problemlerinde de sıkça kullanılan bir algoritmadır. Algoritmanın temel amacı Graf üzerindeki en kısa yolu bulmaktır. Algoritma 1956'da bilgisayar bilimcisi Edsger W. Dijkstra tarafından tasarlandı ve üç yıl sonra yayınlandı. Algoritmanın mantığı şu şekildedir. Bir tane başlangıç düğümü seçilir ve geri kalan tüm düğümlerle başlangıç düğümü arasındaki en kısa yolların bulunulmasını sağlayacak şekilde tekrarlanır [10].

Dijkstra algoritması, ağırlıklı ve yönlü graflar için geliştirilmiş olup en kısa yolu belirlerken Greedy yaklaşımını kullanmaktadır. Greedy yaklaşımı basitçe şöyledir. Bir problemin çözümünde ileride doğabilecek sonuçlar göz önüne alınmadan, mevcut şartlar altında en iyi olan seçimin yapılması gerektiğini savunan yaklaşımdır. Düğümleri dolaşırken başlangıç olarak seçtiğimiz düğümü eğer daha önce gitmiş isek o düğümü tekrar seçmiyoruz ve en az maliyetli olan başka bir düğümü seçiyoruz. Ayrıca Dijkstra algoritması negatif ağırlıkları desteklememektedir [11].

Şekil 2.1 örneği üzerinden Dijkstra Algoritması'nı anlatalım. İlk adım, A Düğümü'nü başlangıç düğümü olarak kabul edelim. A Düğümü'nden B ve C



Şekil 2.1: Dijkstra Algoritması örneği

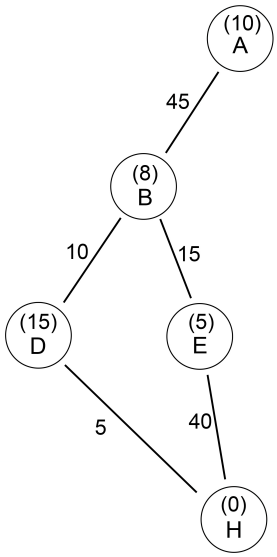
Düğüm	**SD**	**A**	**B**	**C**	**D**	**E**
A	A	—	70, A	20, A	sonsuz	sonsuz
A,C	C	X	70, A	—	30, C	70, C
A, C, D	D	X	70, A	X	—	75, C
A, C, D, B	B	X	—	X	X	75, C
A, C, D, B, E	E	X	X	X	X	—

Tablo 2.1: Dijkstra Algoritması örneği sunuçları

Düğüm'lerine gidilebilmektedir. A Düğümü'nden B Düğümü'ne gitmenin maliyeti 70 ve A'nın kendisi kadar, C Düğümü'ne gitmenin maliyeti ise 20 ve yine A'nın kendisi kadardır. İkinci adımda, gidilmesi en az maliyetli olan C Düğümü seçilir. C Düğümü'nden D ve E Düğüm'lerine gidilebilmektedir. C Düğümü'nden D Düğümü'ne gitmenin maliyeti $20+10 = 30$ ve C'nin kendisi kadar, ve E Düğümü'ne gitmenin maliyeti ise $20 + 50 = 70$ ve C'nin kendisi kadardır. Üçüncü adımda, D Düğümü seçilir. D Düğümü'nden sadece E Düğümü'ne gidilebilmektedir. D Düğümü'nden E Düğümü'ne gitmenin maliyeti $30 + 45 = 75$ ve D Düğümü'nün kendisi kadardır. Dördüncü adımda, B Düğümü seçilir. B Düğümü'nden sadece C Düğümü'ne gidilebilmektedir. B Düğümü'nden C Düğümü'ne gitmenin maliyeti $70 + 30 = 100$ ve B Düğümü'nün kendisi kadardır. Beşinci adım, seçilecek Düğüm olarak sadece E Düğümü kaldığı için onu E Düğümü'nden hiçbir düğüme gidilemediği için algoritma burada sonlanır. Tüm işlem Tablo 2.1'de verilmiştir.

2.3.2 A* Algoritması

Bilgisayar bilimlerinde A star (A*), genellikle yol bulma ve grafik geçişinde kullanılan bir algoritmadır. A* algoritması, derinlik öncelikli bir arama yöntemidir. Noktalar ve aranan düğümler arasında yer belirleme süreci verimlidir. A* algoritması aynı zamanda gerçek maliyet öncelikli arama yöntemi ile sezgisel maliyet öncelikli arama (önce-en-iyi arama) yönteminin hibritleşmesi ile oluşmuş maliyet öncelikli bir arama yöntemidir.



Şekil 2.2: A star algoritması örneği

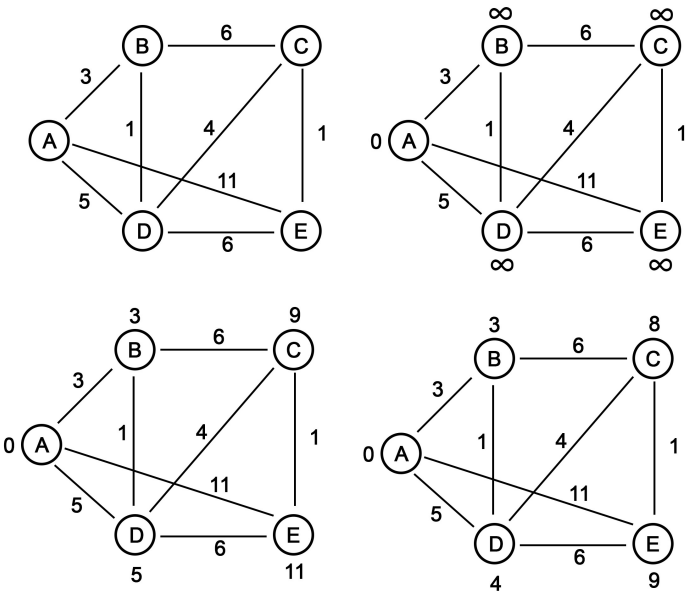
A* algoritması ağaçtaki ziyaret edilen düğümlerin sıraya konmasını belirlemek için bilgi ve sezgisel düşünmeden yola çıkarak x Düğümü'nün maliyet fonksiyonunu $f(x)$ kullanır. Maliyet fonksiyonu, kökten mevcut duruma gelişin maliyeti $g(x)$ ve mevcut durumdan hedefe gidişin maliyeti $h(x)$ iki fonksiyonun toplamından oluşur. Bu seçeneklerden en düşük maliyetli olandan ilerlenir ve diğer seçeneklerin maliyetleri de hafızada tutulur. Bu işlem hedefi buluncaya kadar devam eder. Fakat henüz daha hedefe ulaşmadan önce herhangi bir adımdaki $f(x)$ önceki adımlardaki $f(x)$'lerden daha yüksek çıkarsa geri adım atılır ve düşük maliyetli yoldan devam edilir [12].

Şekil 2.2 örneği üzerinden A* algoritmasını anlatalım. Örnek düğümler, düğümler arası mesafeler ve düğümlerin ağırlıkları verimiştir. Bu verileri ışığında A Düğümü'nden H Düğümü'ne gitmek için A* algoritmasını koşturalım. İlk adım olarak, A Düğümü'nden B Düğümü'ne gidilir. A Düğümü'nden B Düğümü'ne gitmenin maliyeti $45 + 8 = 53$, $f(B) = 53$'dür. İkinci adım, B Düğümü'nden D ve E

düğümlerine gidilebilmektedir. B Düğümü'nden D Düğümü'ne gitmenin maliyeti $45+10+15=70$, $f(D)=70$ ve B'den E'ye gitmenin maliyeti ise $45+15+5=65$, $f(E)=65$'dir. Üçüncü adım, daha düşük maliyetli olan E Düğümü'nü seçeriz. E Düğümü'nden H Düğümü'ne gitme maliyeti $45+15+40=100$, $f(H)=100$ dür. $f(H)$ değeri $f(E)$ büyük olduğu için D Düğümü'ne tekrar döneriz. D Düğümü'nden H Düğümü'ne varma maliyeti $45+10+5=60$, $f(H)=60$ dır. Buna A'da H Düğümü'ne A-B-D-H yolunu izleyerek varılabilir.

2.3.3 Bellman - Ford algoritması

Bellman-Ford algoritması, tek kaynaklı en kısa yol problemleri için etkili bir yöntemdir. Bellman-Ford algoritması, Dijkstra algoritmasında olduğu gibi bir başlangıç düğümünden diğer tüm düğümlere olan en kısa yolu bulmaktadır [13]. Bazı durumlarda grafın eksi maliyetli değerler alması söz konusu olabilmektedir [14]. Dijkstra algoritması kenar maliyetleri sıfır veya artı olan graflar için doğru sonuç verirken Dijkstra'dan farklı olarak Bellman-Ford algoritmasının kenar maliyetleri eksi olan graflar için de uygulanabilmektedir.



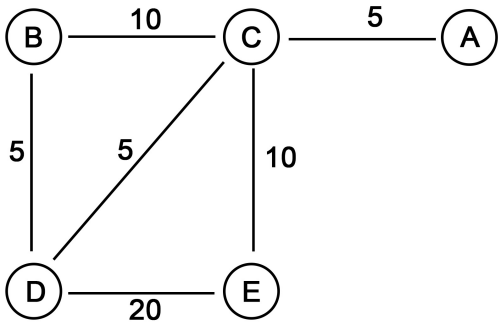
Şekil 2.3: Bellman-Ford algoritması çözüm adımları

Bellman-Ford algoritmasının adımları şunlardır: Her düğümün diğer düğümlere

olan uzaklıkları hesaplanarak bir tabloda tutulmaktadır. Her düğümün tablosu, komşu düğümlere gönderilmektir. Her düğüm, komşusundan uzaklıklar tablosunu aldığında, diğer düğümlere olan en kısa yollar hesaplanır ve tablo değerleri değişiklikler ile güncellenmektir.

2.3.4 Floyd algoritması

Floyd algoritması Dijkstra algoritmasının daha genel halidir [15]. Çünkü Floyd algoritması, şebekedeki herhangi iki düğüm arasındaki en kısa yolu belirlemektedir. Floyd algoritması graf üzerinde her bir düğüm için diğer düğümlere olan en kısa yolları bulan en genel algoritmadır. Floyd algoritmasının verdiği sonuç, Dijkstra algoritmasının graftaki düğüm sayısı kadar çalıştırılmasıyla da elde edilebilmektedir. Floyd algoritmasının yoğun graflarda daha iyi sonuçlar vermektedir. Bunun sebebi, çok seyrek graflarda Dijkstra Algoritması'nın düğüm sayısının katı kadar çalıştırılmasıyla iyi sonuçlar alınabilmektedir. Floyd Algoritması, grafın komşuluk matrisi, sonucun tutulacağı uzaklık matrisi ve en kısa yol bilgilerinin tutulacağı rota matrisi bilgilerini kullanmaktadır. Başlangıç düğümünün komşuluk matrisindeki maliyeti, başlangıç maliyeti olarak kabul edilmektedir. Floyd algoritması, n düğümlü şebekeyi n satırlı ve n sütunlu kare matris olarak göstermektedir. Matrisin (i, j) elemanı, i. düğümden j. düğüme olan uzaklığı vermektedir. Doğrudan i, j'ye bağlıysa düğümler bağlantı değeri almaktadır. Eğer düğümler arasında doğrudan bağlantı yok ise bu düğümler arasındaki maliyetler yapılmaktadır. Uzaklık matrisinde ise, başlangıçta hiçbir bilgi olmadığını göstermek amacıyla tüm bölmeler -1 ile doldurulmaktadır. Şekil 2.4'de Floyd Algoritması ile çözüm adımları verilmistir.



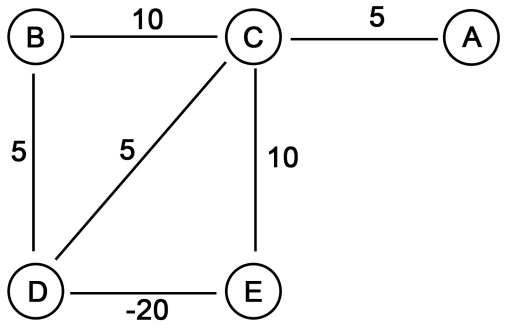
Şekil 2.4: Floyd Algoritması graf örneği

Şekil 2.4'de verilen graf için A Düğümü'nden E Düğümü'ne olan yol bilgileri

Güzergah	**Maliyet**
Yol 1 : A →B → E	20
Yol 2 : A → D → E	25
Yol 3 : A → B → D → E	35
Yol 4 : A → D → B → E	20

Tablo 2.2: Floyd algoritması yol bilgileri

Çizelge Tablo 2.2'de verilmistir. Burada dikkat edilecek noktalar şunlardır: İki düğüm arasında birden fazla düğüm bulunabilmektedir. Rotada bulunan düğüm sayısı önemli değildir (yol 4, 4 düğüm içerirken; yol 2, 3 düğüm içermektedir). En kısa uzunluğu veren birden fazla yol olabilmektedir. Graftan diğer bilgiler de gözlemlenebilmektedir. Her yol göz önünde bulundurularak, her düğümden en az bir kere geçilmelidir. Negatif ağırlıklara izin verildiğinde problem farklı olarak değerlendirilmektedir. Negatif ağrılıklar döngüde gözüktüğü zaman en kısa yol bulunamamaktadır. Şekil 2.5'de negatif ağırlıklı Floyd algoritması grafı görülmektedir.



Şekil 2.5: Negatif ağırlıklı Floyd algoritması graf örneği

Şekil 2.5'deki grafda C, E, D ve C düğümleri arasında bir döngü bulunmaktadır. B Başlangıç Düğümü aynı zamanda son düğümdür. Bu döngünün maliyeti 10–20 + 5 = 5 değerini almaktadır. Eğer bu döngü herhangi bir yola eklenirse, yolun değerini 5 azaltacaktır. İki defa eklenirse, yolun değerini $2 \times 5 = 10$ azaltacaktır. Floyd algoritması yolu bulmaz, sadece uzunlukları bulmaktadır. Negatif değerlerin döngüye girmediği varsayılmaktadır. En kısa uzaklıkları bulmak için izlenecek adımlar listelenmektedir.

	A	**B**	**C**	**D**	**E**
A	0	10	0	5	0
B	10	0	5	5	10
C	0	5	0	0	0
D	5	5	0	0	20
E	0	10	0	20	0

Tablo 2.3: Floyd algoritması başlangıç matrisi

2.3.5 ALT Algoritması

A* Algoritması, referans noktaları ve üçgen eşitsizliği temeline dayanan bu yöntem, Goldberg ve Harrelson tarafından geliştirilmiştir [16]. Yöntem ağ üzerinde önceden belirlenen referans noktaları üzerinden, A* Algoritması'ndaki sezgisel fonksiyonun hesaplanmasını esas alır. Çalışmada referans noktalarının en basit haliyle rastlantısal olarak seçilebileceği belirtilmiş ve buna ek olarak farklı referans noktası seçim önerileri geliştirilmiştir. Algoritma, tüm düğümlerin referans noktalarının tümüne önceden hesaplanmış uzaklıkları kullanılarak çalıştırılır. Üçgen eşitsizliğinden yararlanan algoritma, iki düğüm arasındaki mesafenin düğümlerin referans noktalarına ayrı ayrı mesafelerinin farkından büyük olacağı gerçeğinden yararlanır. Bu gerçekten hareketle düğümler arası mesafe için bir alt sınır belirlenmiş olur. Algoritma etiketlemeye alt sınırı en küçük olan düğümden başlayarak en kısa yola hızlı bir şekilde ulaşmayı hedefler [17,18]. Zhao ve ark. zamana bağımlı ağlarda ALT algoritmasını ön işlemlerle hızlandırmışlardır [19]. Nannicini ve ark. ise algoritmayı zamana bağımlı ağlara çift yönlü olarak uyarlamıştır [20].

2.3.6 Geometrik Kalıplar Algoritması

Geometrik Kalıplar Algoritması, Almanya ulusal demiryolu ağı için zaman parametreli olarak Schulz ve ark. tarafından geliştirilmiştir [21]. Wagner ve ark ise algoritmayı tüm ağırlıklı ağlar için genelleştirmişlerdir [22]. Geometrik Kalıplar Algoritması, grafın her kenarı için, en kısa yol çözümünün o kenardan başladığı düğümleri içeren bir geometrik şekil oluşturarak önceden hesaplanmış bu veri yapısı üzerinden Dijkstra Algoritması ile çözüm üretir. Böylece algoritmanın çözüme ulaşamayacak düğümlere sapması engellenerek çözümün hesaplanma süresi iyileştirilmiş olur. Algoritma daha sonra gerçek zamanlı trafik durumuna göre dinamik olarak çalışacak şekilde düzenlenmiş ve ön işlem adımının hızlandırılması sağlanmıştır [18, 23].

2.3.7 Kenar Bayrakları Algoritması

Kenar Bayrakları Algoritması, önceden hesaplanmış kenar bayraklarına dayanan bir algoritmadır. Alt graflara bölünmüş grafın her alt grafındaki bağlantılar diğer alt graflara en kısa yol üzerinde bulunup bulunmadıklarına göre işaretlenir. Böylece arama algoritması varış Düğümü'nün bulunduğu alt grafa en kısa yol üzerinde bulunan bağlantılar üzerinden kolayca çalıştırılabilir. Köhler ve ark., kenar bayralları algoritmasını hızlandırmak için farklı önişlem adımı içeren birkaç iyileştirme önermişlerdir [24]. Önerilen ön işlem adımlarının klasik yaklaşıma göre daha az zaman aldığı ve sorgu süresini de önemli ölçüde azalttığı kaydedilmiştir [18].

2.3.8 Reach Algoritması

Algoritma Gutman tarafından geliştirilmiştir. Yöntem grafın her düğümü için önceden hesaplanan reach ölçüsüne göre çalışır [24]. Grafın bir düğümü (v) için reach ölçüsü şöyle hesaplanır: Herhangi iki düğüm arasında v Düğümü'nden geçen tüm yollar için başlangıç Düğümü'nden v Düğümü'ne ve v Düğümü'nden varış Düğümü'ne toplam maliyeti küçük olan değerlerin en büyüğü v Düğümü'nün reach ölçüsü olarak belirlenir. v Düğümü'nün u-başlangıç ve u-varış düğümleri için reach ölçüsü 18'dir. Derinlik öncelikli arama temelinde çalışan algoritma her adımda, toplam maliyeti ilgili düğümün reach ölçüsünden küçük olan veya hedef düğüme öklit uzaklığı reach ölçüsünden küçük olan düğüm ile devam ederek en kısa yola ulaşmaya çalışır. Reach Algoritması, A* Algoritması ile birleştirilerek en kısa yol çözümüne daha hızlı ulaşabilmektedir [18].

2.3.9 Geçiş Düğümleri Algoritması

Anayol hiyerarşileri algoritması temeline dayanan Geçiş Düğümleri Algoritması'nda graf üzerinde önceden hesaplama ile elde edilmiş veriler ile arama algoritmasının hızlandırılması esas alınmaktadır [25]. Geçiş Düğümleri Algoritmasın'da graf, birbirlerine belli bir uzak olmayan öklid mesafesinde (D) olan düğümlerden oluşan alt ağlara bölünür. Bir alt ağın her düğümünden diğer tüm düğümlere en kısa yollar hesaplanarak, en kısa yollardan en az biri üzerinde bulunan düğümler belirlenir. Bu düğümlerden D mesafesi içinde kalan son düğümler geçiş düğümü olarak belirlenir. Geçiş düğümleri hesaplandıktan sonra tüm düğümlerden geçiş düğümlerine, geçiş düğümleri arası ve tüm geçiş düğümlerinden tüm düğümlere en kısa yol mesafeleri hesaplanarak bir tabloya kaydedilir. Arama işlemi bu tablo üzerinden gerçekleştirilir [18].

2.3.10 Toplu Ulaşım Ağlarında En Kısa Yol Problemi

$G = (V, E)$ şeklinde gösterilen bir toplu ulaşım ağında V, ağda bulunan durakların kümesi olarak tanımlanır. Bazı tanımlamalarda önemli noktalar (POI-Point-of-Interest) da V kümesine dâhil edilebilmektedir. En kısa yol probleminin temel çözüm yaklaşımları paralel bağlantı içermeyen ağlar için geliştirildiklerinden, bu algoritmalar toplu ulaşım ağına uyarlanırken oluşturulacak grafların bu varsayıma uygun olarak tasarlanmaları gerekmektedir [26]. Çünkü toplu ulaşım ağında iki durak arasında birden fazla hatla seyahat etmek mümkün olduğundan ilkel haliyle graf paralel bağlantılar içerecektir. Ayrıca toplu ulaşım ağında seyahatler zamana bağlı olarak planlandığı için problem geleneksel en kısa yol probleminden faklı olarak bir zaman parametresi içermektedir.

Toplu ulaşım ağında hatlar arası yürüme bağlantıları da önemli bir yere sahiptir. Özellikle farklı ulaşım seçenekleri (metro ve otobüs vb.) arasındaki aktarımlarda duraklar farklı konumlarda olacağından duraklar arası yürüme bağlantıları problemin en kısa yol çözümünü etkileyecektir. Yürüme bağlantılarının yanında toplu ulaşım ağında bir yolculuk çözümünün aktarım sayısı da çözümün kalitesini önemli derecede etkileyen bir faktördür. Bu sebeple çoğu durumda en düşük maliyetli çözümün aynı zamanda en az aktarım gerektiren çözüm olması gerekecek ve böylece problem çok amaçlı hale gelecektir. Aktarım sayısı ve toplam yürüme mesafeleriyle birlikte çözüm kalitesini etkileyen diğer bir önemli faktör de yolculuğun bir sonraki adımındaki hat için durakta bekleme süreleridir. Örneğin aynı yolculuk maliyetine sahip iki çözümden daha az bekleme süresi içeren çözümün kalitesinin çoğu durumda daha yüksek olacağı açıktır. Zaman parametresi ve paralel bağlantılar göz önüne alınarak literatürde toplu ulaşım ağı için iki farklı yaklaşım geliştirilmiş ve ağ için oluşturulacak graf bu yöntemlere göre yeniden tasarlanmıştır [18].

2.3.11 Zaman Genişletme Yaklaşımı

Zaman genişletme yaklaşımında düğüm kavramı, bir durakta belli bir zamandaki (kalkış veya varış) olayına karşılık gelir. Böylece toplu ulaşım ağındaki tüm hatlar için hattın bir duraktan kalkış ve durağa varış olayı ağda bir düğüm olarak gösterilir [27]. Aktarım durağı olarak belirlenen bir durakta, bir önceki hat varış düğümü ile bir sonraki hat kalkış düğümü arasında oluşturulacak bağlantılarda ağırlık genellikle zamanlara göre oluşturulur. Bu tür bağlantılar oluşturulurken hareket düğümünün zaman parametresinin varış düğümü zaman parametresinden büyük olması gerekir [18, 28]. Zaman genişletme yaklaşımında düğüm sayısının artışına bağlı olarak ağ, önemli ölçüde büyümektedir.

2.3.12 Zamana Bağımlı Yaklaşım

Bu yaklaşımda graf geleneksel yaklaşımda olduğu gibi duraklar üzerinden tanımlanır. Ancak paralel bağlantıları engellemek amacıyla bir duraktan geçen her hat için ayrı bir düğüm tanımlanır. Zaman bağımlı yaklaşımda bağlantı ağırlıkları ise algoritmanın çalışması sırasında çözüme giren düğüm dikkate alınarak dinamik olarak belirlenir [27]. Zaman bağımlı yaklaşımın çoğu çalışmada zamanı genişletme yaklaşımına göre daha yüksek performans sergilediği görülmüştür [27–29]. Ayrıca arama algoritmasını hızlandıracak ön işlem adımlarının oluşturulduğu birçok hızlandırma tekniğinde zaman bağımlı yaklaşımın uygulanma kolaylığına dikkat çekilmiştir [18].

2.4 Dinamik Ulaşım Ağlarında En Kısa Yol Algoritmaları

Bu bölüm (zamanlamaya dayalı) toplu taşıma ağlarında yolculuk planlamasını ele almaktadır. Bu senaryoda, bir zaman çizelgesi verilmektedir. Genel olarak, bir zaman çizelgesi bir dizi duraktan (otobüs durakları veya tren platformları gibi), bir rota setinden (otobüs veya tren hatları gibi) ve bir dizi yolculuktan oluşmaktadır. Geziler, günün belirli bir saatinde belirli bir rota boyunca durakları ziyaret eden bireysel araçlara karşılık gelir. Yolculuklar, ayrıca her biri bir çift (başlangıç/varış yeri) durdurma ve aracın durmadan seyahat ettiği (kalkış/varış) süreleri olarak verilen temel bağlantı dizilerine ayrılabilir. Ek olarak, yürüyüş yolları yakındaki duraklar arasındaki yürüyüş bağlantılarını modellemektedir. Karayolu ağları için önemli bir fark, toplu taşıma ağlarının doğal olarak zamana bağlı olmalarıdır, çünkü ağın belirli bölümleri yalnızca zaman içinde belirli, ayrı noktalarda geçirilebilir. Bu nedenle, ilk zorluk zaman çizelgesinin uygun şekilde modellenmesi ile ilgilidir. Yol ağlarındayken, en kısa yol hesaplarında (genellikle en kısa zamada yolculuk) çoğu zaman yeterli olur. Ama çoğu zaman çeşitli optimizasyon kriterlerinin de dikkate alınması gerekebilir. İyi bir yolculuk planlaması için sorguların hızlandırılması uzun zamandır çözülmeyen sorunlardandır. Yalnızca temel sorguları hızlı bir şekilde cevaplamak için değil, aynı zamanda gecikmeleri içeren, parasal maliyet gibi ek kriterleri optimize etmek için çok sayıda algoritma geliştirilmiştir [1].

2.4.1 Modelleme

Optimal seyahatleri hesaplayan algoritmaların çalışabilmesi için ilk olarak gereken zaman çizelgesini modellemektir. Bu modeleme için en iyi bilinen $G = (V, A)$ notasyonu kullanılır. G de en uygun seyahatlere karşılık gelir. Bunu yapmak için iki ana yaklaşımı gözden geçirmektedir (zaman zenişletme ve zamana bağlı) Müller-Hannemann ve ark. [1, 27].

2.4.2 Toplu Ulaşım Ağları En Kısa Yol Çalışmaları

En kısa yol problemi, literatürde yaklaşık 50 yıl gibi bir geçmişe sahip olmasına rağmen, probleme özellikle Dijkstra Algoritması'nı temel alan birçok çözüm yaklaşımı geliştirilmiştir. Bilgisayar teknolojisinin ve buna bağlı olarak bilgisayarların hesaplama kapasite ve performanslarının da zaman içindeki hızlı gelişimi ile en kısa yol problemi için çözüm yöntemleri, çok büyük graflar için bile optimum çözümleri mikro saniyeler ile ölçülebilen sürelerde hesaplayabilmektedir [1].

Başlangıçta çoğu araştırmacı ve akademisyen, toplu ulaşım ağında da en kısa yol problemini çözerken, klasik yöntemleri kullanmışlardır. Bu anlamda ilk çalışmalar Dijkstra Algoritması'nın zaman parametreli olarak çözüm üretmesini hedefleyen sürümlerinden oluşmaktadır. Kısaca, Tarifeli (scheduled) Ağ olarak adlandırılabilecek bir toplu ulaşım ağının zaman genişletmeli veya zaman bağımlı olarak ele alınması durumunda, ağın katlanarak büyümesi sebebiyle klasik yaklaşımların optimum çözümleri yeterince etkin bir performansta üretemediği görülmüştür. Bu sebeple son birkaç yılda araştırmacılar, problemi graf yapısından kurtarıp daha basit veri yapıları ile ele almıştır [30,31]. Böylece yeni algoritmaların çözüm hesaplama performanslarının artırılması hedeflenmiştir. Bu yeni bakış açısı problemin trafik, hava koşulları gibi olasılıklı (probabilistic) parametrelere bağlı değişim gösteren faktörlerden etkilenmesi sebebiyle önişlem adımı içeren yöntemlerin gerçek uygulamalarda kullanılabilir çözümler üretemeyeceği görüşüne dayanır. Bu çerçevede geliştirilen yöntemler, ağı basit veri yapılarıyla ele alan ve dinamik programlama temelinde dayanan yöntemlerdir.

Aşağıda toplu ulaşım ağında yolculuk planlama problemini ele alan bazı temel çalışmalar verilmiştir [18]. Meng ve ark. [26], Singapur toplu ulaşım ağı üzerinde uyguladıkları yöntemle problemi çoklu modlu (metro ve otobüs vb.) ve çoklu kriterli (en az seyahat mesafesi, en düşük ücret ve en erken varış zamanı gibi) ele alarak bir çözüm yaklaşımı geliştirmişlerdir [18].

Schulz ve ark. [21] çalışmalarında, optimizasyon kriteri olarak sadece seyahat süresini ele almış ve Dijkstra Algoritması'nı farklı hızlandırma teknikleriyle

birleştirerek ortalama sorgu süresinin azaltılmasını amaçlamışlardır. Çalışmada varış zamanının önceden bilinememesine bağlı olarak zaman parametreli problemde çift yönlü arama yaklaşımının kullanılamayacağı ancak varış zamanı için belirlenebilecek bir üst sınır değeri ile çift yönlü arama yapmanın mümkün olacağı belirtilmiştir. Graf yapısının zaman genişletme yaklaşımına göre oluşturulduğu çalışmada grafın daraltılması ve buna bağlı olarak sorgu süresinin azaltılması için çeşitli yaklaşımlar önerilmiştir [18].

Brodal ve Jacob [32] yaptıkları çalışmada, toplu ulaşım ağında başlangıç durağından itibaren her aktarımda erişilebilen duraklara göre katmanlara ayırmışlardır. Daha sonra bu katmanları aktarım yapılabilen duraklar arasında bağlantılar oluşturarak birbirine bağlamış ve her bir katman üzerinde Dijkstra Algoritması'nı kullanarak en erken varış zamanı kriterine göre yolculuk çözümleri üretmişlerdir. Literatürde Katmanlı Dijkstra (Layered Dijkstra - LD) olarak adlandırılan algoritma, aktarım sayısı ve varış zamanı kriterlerine göre optimum çözümler üreterek problemi çok kriterli olarak ele alan etkin bir yöntemdi [18].

Barrett ve ark. [33], toplu ulaşım ağının bağlantılarını düğümler arası mümkün olan ulaşım seçeneklerine göre etiketleyerek bir çözüm yaklaşımı geliştirmişler ve bu etiketleme işlemi için farklı yöntemler önermişlerdir. Daha sonra önişlem adımında etiketlere göre uyarladıkları Dijkstra Algoritması'nın hızlandırma teknikleri ile geliştirilmiş versiyonlarıyla en kısa yol çözümlerini aramışlar ve kullandıkları yöntemlerin sonuçlarını karşılaştırmışlardır [18].

Bielli ve ark. [34], toplu ulaşım ağını nesne yönelimli (object-oriented) olarak modelleyerek hiyerarşik yaklaşımla ağ üzerindeki kaynak-hedef düğüm çiftleri için en kısa yolları hesaplamışlardır. Kullanıcının belirleyeceği aktarım sayısına göre çalışan bir uygulama ile yöntemi farklı boyuttaki ağlar üzerinde test ederek sonuçları karşılaştırmışlardır. Ayrıca yöntemin gerçek zamanlı trafik durumuna uyarlanabileceğini belirtmişlerdir [18]. Müller-Hannemann ve ark. [27], toplu ulaşım ağını zaman genişletme yaklaşımı ve zamana bağlı yaklaşım modeline göre modelleyerek tek kriterli ve çok kriterli problemler için en kısa yol çözümleri hesaplamışlardır. Zaman genişletme yaklaşımı ile oluşturdukları yöntemin problemi en az aktarım kriterine göre de çözebileceğini belirmişlerdir. Yöntemlerin testinde zamana bağlı yaklaşım yönteminin açık bir şekilde daha yüksek performansla çözüm ürettiği görülmektedir. Ayrıca elde ettikleri test sonuçlarına göre çok kriterli problemin sorgu süresinin 10 kat daha yüksek olduğu görülmüştür [18].

Sanders ve Schultes [25], son yıllarda geliştirilen hızlandırma teknikleri ile sorgu süresi milyon kat azalan Dijkstra Algoritması'nın toplu ulaşım ağına uyarlanması üzerinde yaptıkları çalışmada yöntemlerin özellikle ağırlıkların dinamik olarak değiştiği ağlarda nasıl kullanılacağı üzerinde durmuşlardır [18]. Wagner ve Will-

halm [23], son birkaç yılda önişlemlerle hızlandırılan Dijkstra Algoritması'nı toplu ulaşım ağına uyarlamış ve hızlandırma algoritmalarının birleşimlerinden oluşan yeni yaklaşımların sonuçlarını test etmişlerdir. Birçok hızlandırma tekniği amaç yönelimli (goal-directed) ve çift yönlü arama teknikleri ile birleştirilerek, sorgu işlemlerinin dikkate değer bir şekilde hızlandırılabileceğini söylemişlerdir. Ancak hızlandırma tekniği seçiminin yüksek oranda problemin (kabul edilebilir önişlem süresi, kullanılabilir bellek alanı vb. gibi) kendi kısıtlarına bağlı olacağını belirtmişlerdir [18].

Batz ve ark. [35], daralma hiyerarşileri algoritmasını zamana bağlı yaklaşıma göre değiştirip algoritmayı bağlantıların zamana bağlı olarak ağırlıklandırıldığı ağlar için genelleştirmişlerdir. Yeni yaklaşımın çift yönlü arama yönteminin kullanılabileceği ilk zaman parametreli hiyerarşik hızlandırma tekniği olduğunu söylemişlerdir [18]. Koszelew [36], toplu ulaşım ağını en kısa seyahat süresi kriterine göre çözüm üreten iki yeni yaklaşım geliştirmiş ve bu çözüm yaklaşımlarını sorgu süresi ve çözüm kalitesi açılarından karşılaştırmıştır. Çalışmada standart genişlik öncelikli arama algoritması ile oluşturulmuş iki boyutlu aktarım matrisleri ve iki durak arasındaki en kısa mesafeyi gösteren uzaklık matrisi kullanılmıştır. Aktarım matrisleri bir önişlem adımı ile bir kez hesaplanmakta ve bir kaynak-hedef düğüm çifti arasında k adet aktarımlı bir çözüm olup olmadığını (0-1) göstermektedir. Birinci yöntem en kısa yolu aktarım matrisleri ve uzaklık matrisini kullanarak hesaplarken ikinci yöntemde uzaklık matrisi ile birlikte düğüm etiketleme tekniği kullanılmıştır. Test sonuçlarına göre birinci yöntemin genel olarak daha hızlı çalıştığı söylenmiştir [18].

Pyrga ve ark. [29] zaman genişletme yaklaşımı ve zamana bağlı yaklaşım ile modelledikleri toplu ulaşım ağında iki yaklaşımın performanslarını karşılaştırmışlardır. Çalışma soncunda zamana bağlı yaklaşımın açık bir şekilde daha yüksek performansla çalıştığı ancak karmaşık ağları modellemede zaman genişletme yaklaşımının daha güçlü olduğu söylenmiştir. Ayrıca çalışmada Dijkstra algoritması çok etiketli (Multicriteria-Label) olarak uygulanmıştır. Böylece çözümler aktarım sayısı ve varış zamanına göre optimize edilerek Dijkstra algoritmasının çok kriterli bir versiyonu elde edilmiştir (Multicriteria Label Setting - MLS) [18, 37].

Jariyasunant ve ark. [38], hatlar arası optimal aktarım noktalarını önceden hesapladıkları çözüm yöntemiyle, problemin k adet en kısa yol çözümlerini hesaplamış ve en kötü durumda sorgu süresinin 3 saniyeden az sürdüğü görülmüştür. Ayrıca optimal aktarım noktalarının hesaplandığı önişlem adımının genellikle 1 dakikadan daha az zaman aldığı belirlenmiştir [18].

Kumari ve Geethanjali [39], toplu ulaşım ağı üzerinde en kısa yol problemi çözüm yöntemleri üzerine yapılan çalışmaları inceleyerek bir literatür incelemesi

yapmıştır. Standart en kısa yol ve k en kısa yol yöntemlerinin statik ve dinamik ağlardaki uygulamalarının incelendiği çalışmada rastlantısal kaynak-hedef düğümleri üzerinden elde edilen bulguların gerçek uygulamalarda çoğu kez elde edilemediği ifade edilmiştir [18].

Zhang ve ark. [40], ulaşım ağını özel (yürüme, bisiklet ve otomobil) ve genel (toplu ulaşım hatları) olarak iki alt ağa ayırarak problemi, bu alt ağların aktarım bağlantıları ile oluşturulmuş ağ üzerinden süper-ağ olarak adlandırdıkları teknik ile çözmüşlerdir. Tekniğin Eindhoven toplu ulaşmı ağı üzerinde uygulama sonuçlarını tartışarak çift-yönlü arama ve sezgisel tekniklerin sorgu performansını önemli ölçüde artırdığını söylemişlerdir [18].

Delling ve ark. [28] toplu ulaşım ağı verilerini tek boyutlu diziler şeklinde işleyerek yeni bir veri yapısı önermişlerdir. Bu yeni veri yapısı sayesinde optimizasyon işleminde Dijkstra gibi graf algoritmalarının performansını olumsuz etkileyen öncelik sırası kuyruk işlemlerine ihtiyaç duyulmadığından, sorgu sürelerinde önemli derecede iyileşme sağlanmıştır. Çalışmada ayrıca yeni veri yapısına uygun dinamik programlama temeline dayanan yeni bir algoritma (RAPTOR) önerilmiştir. Problemin varış zamanı ve aktarım sayısına göre en uygun çözümlerini hesaplayan algoritma, günümüzde bu alada geliştirilen en yüksek performanslı ve en çok kullanılan algoritmalar arasındadır [18].

Artigues ve ark. [41], toplu ulaşmı ağında en kısa yol çözümlerini, seyahat süresi ve aktarım sayısını minimize edecek şekilde deterministik olmayan ağ ağırlıkları üzerinden hesaplayabilen yaklaşım önermişlerdir. Topolojik etiketleme tekniğine dayanan bu yaklaşımla çift-yönlü ve sezgisel yöntemlerle birleştirilerek çözümler üretilmiş ve yöntemlerin performanslarını karşılaştırılmıştır [18].

Dibbelt ve ark. [31], Delling ve ark. [28] tarafından geliştirilen RAPTOR algoritmasına benzer bir şekilde, toplu ulaşım ağında zaman çizelgesi üzerinden çözüm üreten yeni yaklaşım geliştirmiştir. Bağlantı Tarama Algoritması (Connection Scan Algorithm - CSA) olarak adlandırdıkları algoritmada çözüm hesaplamaları, RAPTOR algoritmasından farklı olarak duraklar üzerinden değil; bağlantılar üzerinden yapılmaktadır. Bağlantıları başlangıç durağından erişilebilirliğine göre etiketleyen algoritma, böylece varış zamanı kriterinin yanı sıra aktarım sayısını da optimize ederek problemi RAPTOR gibi çok kriterli olarak çözmektedir [18].

Bast ve Storandt [1], durak çiftleri arasında zaman bağımsız olarak kullanılabilir tüm güzergâhları bir örüntü (pattern) olarak ele almış, hat ve durak bazında bulanık (fuzzy) olarak hesapladıkları örüntü benzerliklerine göre grupladıkları güzergahlar üzerinden çözümler üretmişlerdir (transfer patterns). Önişlem adımlı bu yaklaşım, önişlem zamanının yüksek olmasına karşın, çok kriterli optimizasyonda çok büyük boyuttaki toplu ulaşım ağında optimum çözümü literatürdeki en düşük

sorgu süresinde hesaplayabilmektedir [18].

2.4.3 En Kısa Yol Probleminde Algoritma Seçimi

Uygulamamızda zamandan bağımsız olan kısmında yani durakların birbiryle olan ilişkilerini bulmak için geçiş düğümleri algoritması kullanıldı. Aynen algoritmada olduğu durakların biribirine D mesafe olan keşimlerinde bir tablo oluşturuldu. Daha bu kesişim tablosu yine aynı algoritma ile türetilerek daha çok aktarmaya sahip olan hat keşisimleri tabloları eldi edildi. Daha sonra tariflere bağlı olarak hatların duraklarıda ne zaman olacağı bilgisinde oluşan varış zamanı tabloları oluşturuldu. Zamanandan bağımsız oluşturulan tablolardan alınan sonuçlara zaman kriteri girebilmek için yine geçiş düğümleri algoritması ile tüm bulunan cevaplar t süre ilersine kadar varış zamanı olan hatlar şeklinde sorgulanarak zamana göre uyan sonuçlar döndürülmüştür.

3 Metot ve Deneysel Sonuçlar

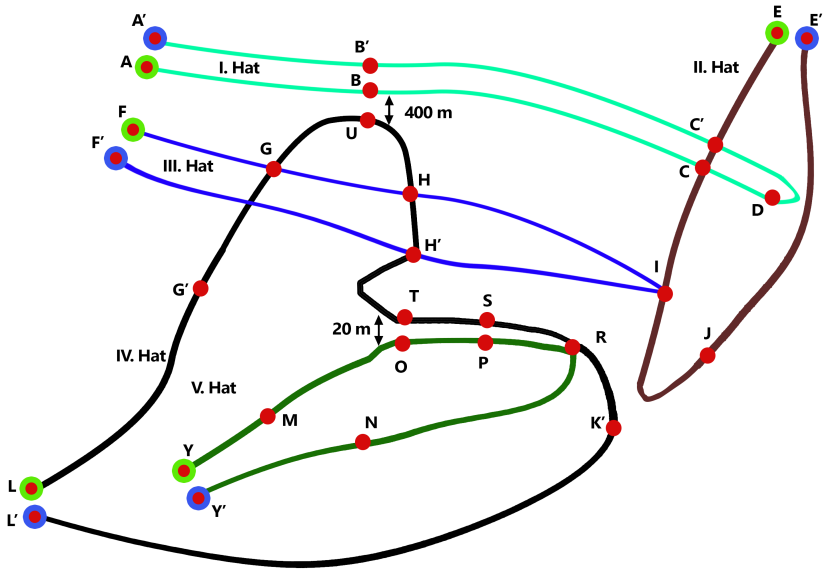
3.1 Geliştirilen Algoritmanın Küçük Bir Örnek Üzerinden Anlatılması

Örnek çalışmamızda geliştirdiğimiz algoritmayı beş hattan oluşan toplu ulaşım sistemi üzerinden izah etmek istiyoruz. Örnek hattı kendimiz kurgulayarak Şekil 3.1'de gösterdik. Harita üzerinde bulunan her hat, farklı renklerde gösterilmektedir. Hatlar üzerinde bulunan kırmızı noktalar ara durakları, yeşil konturlu olanlar başlangıç durakları ve mavi konturlu olanlar ise bitiş duraklarını temsil etmektedir. Harita üzerinde bazı duraklar arasında yazan mesafeler kesişen ya da yakın durakların durumunu göstermek için verilmiştir. Mesafe bilgisi verilmeyen yerler 1000 metrenin üzerinde olduğu için yazılmamıştır. Sistemin çalışması için hat bilgileri ve tarifeler tablolarına ihtiyaç vardır. Tablo 3.1 Hat Bilgileri Tablosu'nda hat adı, durak adı, enlem-boylam bilgisi, durak sırası ve önceki durak ile arasındaki geçen süre bilgisi bilinmesi gerekmektedir. Tablo 3.2 Tarifeler Tablosu'nda ise yine hat adı, tarife zamanı ve tarifenin hafta içi ya da hafta sonu bilgisinin bilinmesi gerekir.

Tablo 3.1'deki hat bilgileri incelenirse, Şekil 3.1 üzerindeki hatlar, durak adları ve durak sırası bilgileri görülmektedir. Durak sırası bilgisi hattın başlangıcını bir kabul ederek sıra ile yazılmıştır. Enlem-boylam ve önceki durak ile arasındaki geçen süre bilgisi kurgusal verilerdir. Algoritmanın ilerleyen evrelerinde bu bilgiler geçmiş zaman verilerinden yola çıkılarak hesaplanmıştır. Örneğin Şekil 3.1'de I. Hat olarak etiketlenmiş turkuaz renkli hat A' Durağı'ndan başlayarak A Durağı'nda sonlanmaktadır. Bu bilgi Tablo 3.1'in ilk dört satırda hat adı, durak adı, enlem-boylam, durak sırası ve önceki durak ile arasındaki geçen süre bilgisi verilmiştir. A' durağı başlangıç durağı olduğundan Tablo 3.1 Hat Bilgileri Tablosu'nun ilk satırında önceki durak ile arasındaki süre bilgisi sıfır dakika olarak girilmiştir.

Tablo 3.2 Tarifeler Tablosu da incelenirse hat bilgileri yine harita üzerindeki bilgi-

lerden tarife ve hafta içi/sonu bilgisi örnek olarak girilmiştir. Örneğin ilk kayıtta I. Hat'a ait hafta içi saat 08:00'da hareket edeceğinin bilgisi girilmiştir.



Şekil 3.1: Örnek olarak kurgulanmış bir toplu ulaşım haritası

Yukarıda verilen örnek toplu ulaşım haritasının Tablo 3.1 Hat Bilgileri Tablosu ve Tablo 3.2 Tarifeler Tablosu aşağıdaki gibi olduğu bilinmektedir. Bütün veriler hatlar tablosunda bulunmaktadır.

Hat Adı	**Durak Adı**	**Enlem Boylam**	**Durak Sırası**	**Önceki Durak ile Arasındaki Süre**
I. Hat	A'	37.871418, 32.490522	1	0 dk
I. Hat	B'	37.870393, 32.501228	2	10 dk
I. Hat	C'	37.870648, 32.508365	3	5 dk
I. Hat	D	37.871418, 32.490522	4	10 dk
I. Hat	C	37.870393, 32.501528	5	10 dk
I. Hat	B	37.870648, 32.508565	6	5 dk
I. Hat	A	37.870476, 32.516543	7	15 dk
II. Hat	E	37.868513, 32.468787	1	0 dk
II. Hat	C'	37.868686, 32.469219	2	10 dk
II. Hat	C	37.866466, 32.460027	3	5 dk
II. Hat	I	37.864162, 32.483386	4	15 dk

II. Hat	J	37.864246, 32.452674	5	10 dk
II. Hat	E'	37.868513,32.468487	6	10 dk
III. Hat	F	37.866295, 32.500470	1	0 dk
III. Hat	G	37.870648 ,32.508365	2	5 dk
III. Hat	H	37.864162, 32.483386	3	10 dk
III. Hat	I	37.860575, 32.473763	4	15 dk
III. Hat	H'	37.864162, 32.483486	5	5 dk
III. Hat	F'	37.864162, 32.583386	6	10 dk
IV. Hat	L	37.862880, 32.445645	1	0 dk
IV. Hat	G'	37.868686, 32.469219	2	10 dk
IV. Hat	G	37.868586, 32.465219	3	10 dk
IV. Hat	U	37.860062, 32.439157	4	5 dk
IV. Hat	H	37.866466, 32.460027	5	15 dk
IV. Hat	H'	37.864162, 32.483486	6	15 dk
IV. Hat	T	37.859124, 32.436561	7	10 dk
IV. Hat	S	37.865442, 32.430721	8	10 dk
IV. Hat	R	37.869710, 32.437642	9	10 dk
IV. Hat	K'	37.869510, 32.437542	10	15 dk
IV. Hat	L'	37.869410, 32.437342	11	10 dk
V. Hat	Y	37.870564, 32.446293	1	0 dk
V. Hat	M	37.871417, 32.451809	2	10 dk
V. Hat	O	37.872353, 32.466518	3	15 dk
V. Hat	P	37.876961, 32.473658	4	10 dk
V. Hat	R	37.869710, 32.437642	5	10 dk
V. Hat	N	37.874489, 32.457974	6	5 dk
V. Hat	Y'	37.870364, 32.443293	7	10 dk

Tablo 3.1: Hat Bilgileri Tablosu

Hat	**Tarife**	**Hafta içi/sonu**
I. Hat	8:00	Hafta içi
I. Hat	12:00	Hafta içi
I. Hat	17:00	Hafta içi
I. Hat	9:00	Hafta sonu
I. Hat	13:00	Hafta sonu
I. Hat	18:00	Hafta sonu
II. Hat	8:30	Hafta içi
II. Hat	12:30	Hafta içi
II. Hat	17:30	Hafta içi

II. Hat	9:30	Hafta sonu
II. Hat	13:30	Hafta sonu
II. Hat	18:30	Hafta sonu
III. Hat	7:00	Hafta içi
III. Hat	18:00	Hafta içi
III. Hat	7:15	Hafta sonu
III. Hat	18:15	Hafta sonu
IV. Hat	7:30	Hafta içi
IV. Hat	18:30	Hafta içi
IV. Hat	8:30	Hafta sonu
IV. Hat	19:30	Hafta sonu
V. Hat	9:45	Hafta içi
V. Hat	13:45	Hafta içi
V. Hat	17:45	Hafta içi
V. Hat	10:45	Hafta sonu
V. Hat	14:45	Hafta sonu
V. Hat	18:45	Hafta sonu

Tablo 3.2: Tarifeler Tablosu

İkincil tablolar bu kurgusal harita üzerinden oluşturulmuştur. Yukarıdaki tablolarda verilen bilgiler doğrultusunda 2-hat için Hat İlişki, 3-hat için Hat İlişki, n-hat için Hat İlişki ve Varış Zamanı tabloları oluşturulur.

3.1.1 2-Hat için Hat İlişki Tablosu'nun Oluşturulması

Tüm hatların bilgilerinin bulunduğu Tablo 3.1'deki duraklar referans alınarak kendi ile iç içe döngüye sokulur. Birinci döngüdeki hat adına **K** ve ikinci döngüdeki hat adına **M** diyelim. Döngü içerisinde hatların aynı olan durakları ve 600 metreye kadar yakın olan durakları varsa bulunur. **K** ve **M** değerleri aynı olduğunda döngüde sırada hat varsa devam edilir yoksa döngü bitirilir. Kesişen hatların adları, durakları, durak sıra numaraları ve mesafe bilgisi tabloya yazılır. İki durak arası 200 metreden az olan duraklar karşılıklı duraklar olduklarını göstermektedir. Bu şekilde kesişen duraklar için mesafe değeri 0 olarak kabul edilir. 0 olarak kabul edilmesini sebebini bir örnekle açıklayabiliriz. Örneğin *V.* Hat ve *IV.* Hat'tın kaşılıklı şekilde kesişen 2 durak kümesi ve ortak olan bir durağı vardır. Bunlar T Durağı ile O Durağı ve S Durağı ile P Durağı'dır. Ortak olan durağı da R Durağı'dır. *V.* Hat ile başlayıp *IV.* Hat ile yoluna devam edecek olan yolcunun O

Durağı'nda inip T Durağı'na geçmesi ya da P Durağı'nda inip S Durağı'na geçmesi veya iki hattın da durağı olan R Durağı'nda inip yine aynı duraktan diğer hatta binip devam etme seçeneği bulunmaktadır. Zaman açısından O Durağı'nda inip 20 metre yürüyüp T Durağı'ndan geçecek olan IV. Hat'ta binip devam etmesi en uygun olanıdır. Bu sebepledir ki R Durağı'na kadar gidip yolculuk süresini uzatmak yerine iki hattın ilk kesişen duraklarında aktarma yapmak daha mantıklı olan seçimlerdendir. Tablo 3.3 ilk satır I. Hat ile II. Hat'tın kesişen durağı C ve mesafe değeri 0 olarak girilmiştir. 2-hat için hat ilişki tablosunun oluşturulması için uygulanan algoritmanın akış diagramı Şekil 3.2'de verilmiştir.

başlangıç Hattı			**Varış Hattı**			**Mesafe**
Hat Adı	**Durak Adı**	**Durak Sırası**	**Hat Adı**	**Durak Adı**	**Durak Sırası**	**metre**
I. Hat	C'	3	II. Hat	C'	2	0
I. Hat	C	5	II. Hat	C	5	0
I. Hat	B	2	IV. Hat	U	3	400
II. Hat	C'	2	I. Hat	C'	3	0
II. Hat	C	3	I. Hat	C	5	0
II. Hat	I	4	III. Hat	I	4	0
III. Hat	I	4	II. Hat	I	4	0
III. Hat	G	2	IV. Hat	G	3	0
III. Hat	H	3	IV. Hat	H	5	0
III. Hat	H'	5	IV. Hat	H'	6	0
IV. Hat	U	4	I. Hat	B	6	400
IV. Hat	G	2	III. Hat	G	2	0
IV. Hat	H	5	III. Hat	H	3	0
IV. Hat	H'	6	III. Hat	H'	5	0
IV. Hat	T	7	V. Hat	O	3	0
IV. Hat	S	8	V. Hat	P	4	0
IV. Hat	R	9	V. Hat	R	5	0
V. Hat	O	3	IV. Hat	T	7	0
V. Hat	P	4	IV. Hat	S	8	0
V. Hat	R	5	IV. Hat	R	9	0

Tablo 3.3: 2-hat için Hat İlişki Tablosu

3.1.2 3-Hat için Hat İlişki Tablosu'nun Oluşturulması

Tablo 3.3 kendisi ile iç içe döngüye sokulur. Birinci döngüdeki başlangıç hat adına **A**, varış hat adına **B** ve ikinci döngüdeki başlangıç hat adı **C**, varış hattı adına **D** diyelim. **B** ve **C** değerleri aynı ise istenilen noktaya A hattı ile başlayarak **C** hattı ile aktarma yapılır ve **D** hattını kulanarak istenilen noktaya gidilebilir. Örneğin Tablo 3.3 birinci döngü **A** değeri I. Hat, B değeri II. Hat ve ikinci döngüde içerisinde **C** değeri II. Hat olan satırlar bulunur. Bulunan satırlar bize I. Hat ile başlanıp II. Hat ile aktarma yapılıp gidilebilecek olan hatların listesini verir. Böylece üç hat ve iki aktarma ile istenilen hedefe gidilebilir. Tablo 3.4'deki birinci satırda başlangıç hattı I. Hat, varış hattı III. Hat ve bu döngü yapısına göre aktarma hattı II. Hat olarak girilmiştir. Bu işlem 2-hat için Hat İlişki Tablosu ile 2-hat için Hat İlişki Tablosu'nun kesişim operasyonudur. 3-hat için Hat İlişki Tablosu'nun oluşturulmasını sağlayan algoritmanın akış diagramı Şekil 3.3'de verilmiştir.

Başlangıç Hattı	Varış Hattı	Aktarma Hattı
I. Hat	III. Hat	II. Hat
I. Hat	III. Hat	IV. Hat
I. Hat	V. Hat	IV. Hat
II. Hat	IV. Hat	I. Hat
II. Hat	IV. Hat	III. Hat
III. Hat	I. Hat	II. Hat
III. Hat	I. Hat	IV. Hat
III. Hat	V. Hat	IV. Hat
IV. Hat	II. Hat	I. Hat
IV. Hat	II. Hat	III. Hat
V. Hat	I. Hat	IV. Hat
V. Hat	III. Hat	IV. Hat

Tablo 3.4: 3-hat için Hat İlişki Tablosu

3.1.3 n-Hat için Hat İlişki Tablosu'nun Oluşturulması

3 hat ve 2 aktarma ile hedefimize gidemiyor olabiliriz. Daha fazla hat ve aktarmaya ihtiyacımız olabilir. Bunun için Tablo 3.3 ile Tablo 3.4 döngüye sokulur. Yine aynı Tablo 3.4'ün oluşturulmasında kullandığımız mantığı kullanarak yapabiliriz. Birinci döngüye Tablo 3.4'ü ve ikinci döngüye Tablo 3.3'ü koyabiliriz. Birinci döngüdeki başlangıç hat adına **A**, varış hat adına **B** ve ikinci döngüdeki başlangıç hat adı **C**, varış hattı adına **D** diyelim. **B** ve **C** değerleri aynı ise istenilen noktaya

A Hat'tı ile başlayarak **C** Hat'tı ile aktarma yapılır ve **D** Hat'tını kulanarak istenilen noktaya gidilebilir. Örneğin birinci döngüde **A** değeri I. Hat, **B** değeri III. Hat ve ikinci döngüde içerisinde **C** değeri III. Hat olan satırlar bulunur. Bulunan satırlar bize I. Hat ile başlanıp ilk aktarmada II. Hat'tı kullanıp daha sonra III. Hat ile tekrar aktarma yapılıp gidilebilecek olan hatların listesini verir. Tablo 3.5'deki ikinci satırda başlangıç hattı I. Hat varış hattı IV. Hat ve aktarma hattı III. Hat olarak girilmiştir. Burada I. Hat'tan III. Hat'ta gidebilmek için Tablo 3.4'deki bilgiden yararlanarak II. Hat ile aktama yapmamız gerekmektedir. Yani yolculukta kullanılacak hatlar sırasıyla I. Hat - II. Hat - III. Hat - IV. Hat şeklindedir. Bu işlem formülasyonu şöyledir. n, aktarmada istenilen hat sayısını temsil etmektedir. $n-1$-hat için Hat İlişki Tablosu ile 2-hat için Hat İlişki Tablosu'nun kesişimi operasyonudur. n-hat için hat ilişki tablosunun oluşturulması diagramı Şekil 3.4'de verilmiştir.

Başlangıç Hattı	**Varış Hattı**	**Aktarma Hattı**
I. Hat	II. Hat	III. Hat
I. Hat	IV. Hat	III. Hat
I. Hat	IV. Hat	V. Hat
II. Hat	I. Hat	IV. Hat
II. Hat	III. Hat	IV. Hat
III. Hat	II. Hat	I. Hat
III. Hat	IV. Hat	I. Hat
III. Hat	IV. Hat	V. Hat
IV. Hat	I. Hat	II. Hat
IV. Hat	III. Hat	II. Hat
V. Hat	IV. Hat	I. Hat
V. Hat	II. Hat	III. Hat

Tablo 3.5: 4-hat için hat ilişki tablosu

3.1.4 Varış Zamanı Tablosunun Oluşturulması

Tüm hatların tarife bilgilerinin bulunduğu Tablo 3.2'deki tarife ve hat bilgileri tablosundaki iki durak arası geçen zaman bilgisi alınır. Tarife alanı hattın sefere kaçta çıkacağını bildirmektedir. Tarife bilgisi başlangıç olarak kabul edilir ve iki durak arası geçen süre eklenerek hattın bilinen duraklarına o tarifeye göre varış zamanları bulunur. Varış zamanını algoritma koşarken de bulubiliriz. Ancak bu işlem sorgu zamanını uzamasına sebep olur. Sorgu zamanını en aza indirmek için hatların duraklarında ne zaman olacaklarını daha öncesinden bilinmesi zaman kazandıracağından Tablo 3.6'nın oluşturulması gerekmektedir. Örneğin ikinci satında

I. Hat'tın hafta içi saat 08:00 tarifesine göre B durağı'nda saat 08:10 olduğu bilgisi girilmiştir. Varış zamanı tablosunun oluşturulması için kullanılan algoritmanın akış diagramı Şekil 3.5'de verilmiştir.

Şekil 3.5: Varış zamanı hesaplama algoritması akış diagramı

Hat	**Tarife**	**Durak Adı**	**Durak Sırası**	**Hafta içi/sonu**	**Varış Zamanı**
I. Hat	8:00	A'	1	H.İ..	8:00
I. Hat	8:00	B'	2	H.İ..	8:10
I. Hat	8:00	C'	3	H.İ..	8:15
I. Hat	8:00	D	4	H.İ..	8:25
I. Hat	8:00	C	5	H.İ..	8:35

Tablo 3.6: Varış Zamanı Tablosu. Tablonun devamı eklerdeki Tablo 5.1'dedir.

3.1.5 Algoritmanın Detayları

Algoritmanın çalışması şöyledir. İlk olarak sisteme başlangıç ve varış noktalarının enlem/boylam bilgileri ve zaman bilgisi (ileri zaman planlaması istendiği durumlar için) kullanıcıdan alınır. Algoritma çalışırken gerekli olan hat bilgileri, 2-hat için Hat İlişki, n-Hat için Hat İlişki Tarifeler, Varış Zamanı tabloları daha öncesinden hazırlanmıştır ve bu tablolar kullanılacaktır. Başlangıç ve varış noktaları merkezde olacak şekilde oluşturulan K metre yarıçapındaki çemberler içerisinde kalan durak aranır. Eğer bu çemberler içerisinde kalan durak ya da duraklar yoksa, toplu taşıma araçları ile çözüm üretilemeyeceği sonucuna varılır. Eğer çemberler içerisinde durak ya da duraklar bulunursa başlangıç ve varış noktalarında bulunan duraklardan geçen hatlar bulunur. Başlangıç ve varış noktalarından geçen hatlardan aynı olanların olup olmadığına bakılır. Yani başlangıç ve varış hatları kümelerinin kesişimi bulunmuş olunur. Bulduğumuz bu kesişim tablosu bize istenilen bu noktalar arasında tek hat ile seyahat edilebileceği bilgisini almış oluruz. Bu bilgiden sonra istenilen zamana göre seyahat edilip edilemeyeceği kontrol edilir. Tablo 3.2'den bu hatların tarife bilgisine bakılır. İstenilen hareket zamanının L dakika sonrasına kadar seferi olan hatlar varsa listelenir ve MC kümesine eklenir. Eğer çözüm yoksa aktarmalı çözümler bulmak için bir sonraki adıma geçilir.

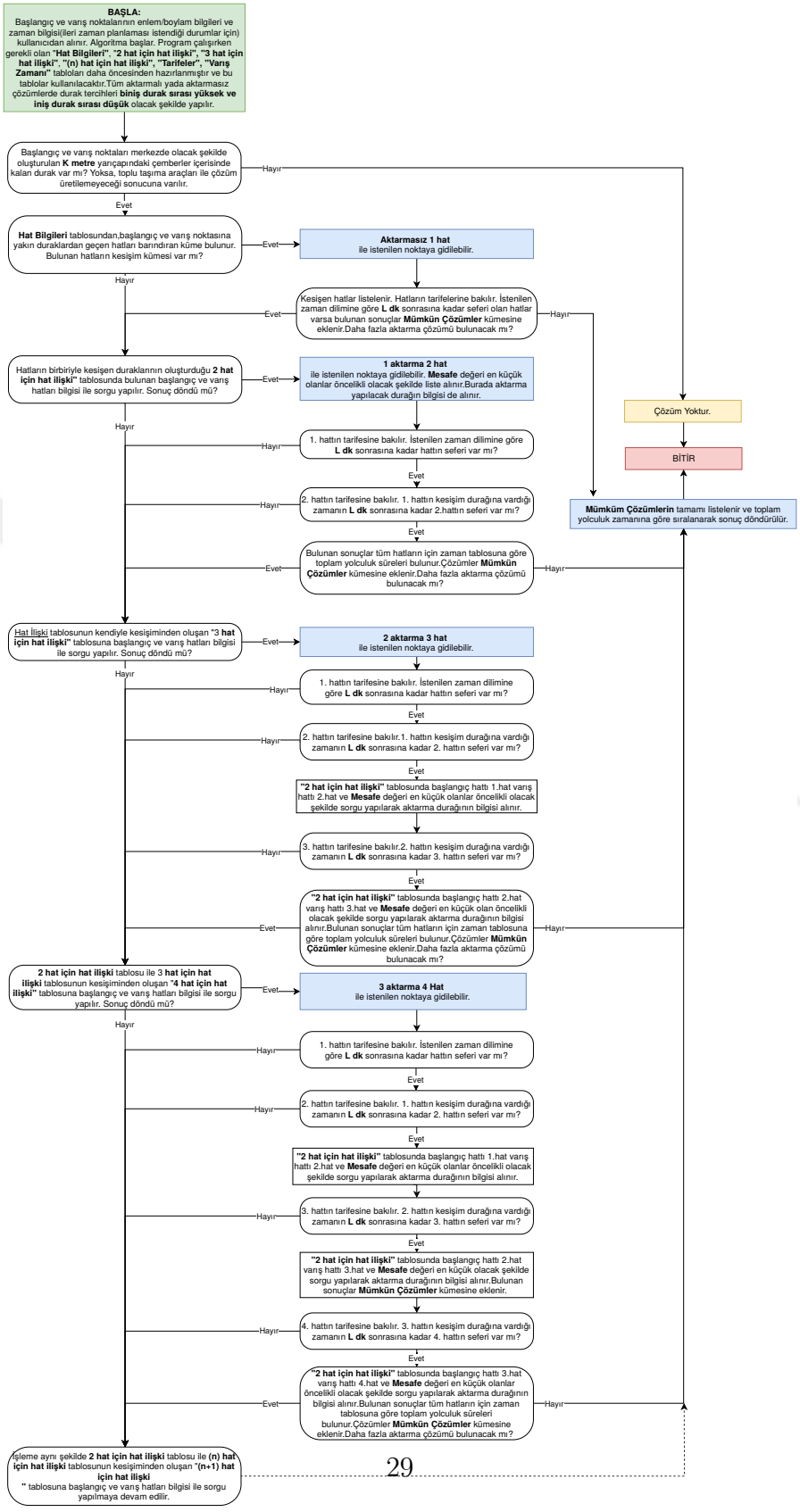
Aktarmalı çözümlerin ilk adımı olarak bir aktarma ve iki hat kullanarak istenilen hedefe gidilip gidilemeyeceğine bakılır. Bunun için bulduğumuz başlangıç ve varış noktalarından geçen hatlar Tablo 3.3'e girilerek mesafe değeri en küçükten başlayacak şekilde sorgu çekilir. Eğer sonuç bulunamazsa iki aktarma üç hat ile istenilen hedefe gidilip gidilemeyeceği bilgisini bulmak için bir sonraki adıma geçilir. Sonuç dönerse bir aktarma iki hat ile seyahat edilebileceği bilgisini almış oluruz. Bu bilgiden sonra istenilen zamana göre seyahat edilip edilemeyeceği kontrol edilir. Tablo 3.2'den başlangıç hattının tarife bilgisine bakılır. İstenilen hareket zamanının L dakika sonrasına kadar seferinin olup olmadığına bakılır. Eğer sefer yoksa iki aktarma üç hat ile istenilen hedefe gidilip gidilemeyeceği bilgisini bulmak için bir sonraki adıma geçilir. Eğer sefer varsa bu hattın bulunan seferinin Tablo 3.6'dan iki hattın kesişim noktası yani aktarma yapılacak olan durağa ne zaman varacağını bilgisi bulunur. Daha sonra varış hattının bu varış zamanı bilgisinden L dakika sonrasına kadar seferinin olup olmadığına bakılır. Eğer sefer yoksa iki aktarma üç hat ile istenilen hedefe gidilip gidilemeyeceği bilgisini bulmak için bir sonraki adıma geçilir. Sefer varsa başlangıç, varış hatları ve hareket zamanları, aktarma yapılacak durak, varış zamanları ve toplam yolculuk süresi bilgi ile MC kümesine eklenir. Eğer daha fazla aktarma ile gitmek isteniyorsa bir sonraki adıma geçilir.

İki aktarma üç hat ile istenilen noktalar arasında seyahatin mümkün olup olmadığını bakılır. Bunun için bulduğumuz başlangıç ve varış noktalarından geçen

hatlar Tablo 3.4'den sorgu çekilir. Eğer sonuç dönmezse üç aktarma dört hat ile istenilen hedefe gidilip gidilemeyeceği bilgisini bulmak için bir sonraki adıma geçilir. Sonuç dönerse istenilen bu noktalar arasında iki aktarma üç hat ile seyahat edilebileceği bilgisini almış oluruz. Bulunan bu sonuç bize başlangıç ve varış hattı bilinen hatların hangi aktarma hatlarını kullanacaklarının bilgisini verir. Başlangıç hattına birinci hat, aktarma hattına ikinci hat ve varış hattına da üçüncü hat diyelim. Başlangıç hattı birinci hat ve varış hattını da ikinci hat olarak kabul ederek ve mesafe değeri en küçük olandan başlayacak şekilde Tablo 3.3'den sorgu çekilir ve bu iki hattın aktarma durağı bilgisini buluruz. Daha sonra istenilen zamana göre seyahat edilip edilemeyeceği kontrol edilir. Tablo 3.2'den birinci hattın tarife bilgisine bakılır. İstenilen hareket zamanının L dakika sonrasına kadar seferinin olup olmadığına bakılır. Eğer sefer yoksa üç aktarma dört hat ile istenilen hedefe gidilip gidilemeyeceği bilgisini bulmak için bir sonraki adıma geçilir. Eğer sefer varsa bu hattın bulunan seferinin Tablo 3.6'dan aktarma yapılacak durağa ne zaman varacağı bilgisi bulunur. Daha sonra ikinci hattın bu varış zamanı bilgisinden L dakika sonrasına kadar seferinin olup olmadığına bakılır. Eğer sefer yoksa üç aktarma dört hat ile istenilen hedefe gidilip gidilemeyeceği bilgisini bulmak için bir sonraki adıma geçilir. Eğer seferi varsa başlangıç hattı ikinci hattı ve varış hattı üçüncü hat olarak kabul edilir ve mesafe değeri en küçük olandan başlayacak şekilde Tablo 3.3'den sorgu çekilir ve bu iki hattın aktarma durağı bilgisini buluruz.

Daha sonra üçüncü hattın bir önceki bulunan varış zamanına göre seyahat edilip edilemeyeceği kontrol edilir. Tablo 3.2'den üçüncü hattın tarife bilgisine bakılır. İstenilen hareket zamanının L dakika sonrasına kadar seferinin olup olmadığına bakılır. Eğer sefer yoksa üç aktarma dört hat ile istenilen hedefe gidilip gidilemeyeceği bilgisini bulmak için bir sonraki adıma geçilir. Eğer sefer varsa başlangıç, varış ve aktarma hatları ve hareket zamanları, aktarma yapılacak duraklar, varış zamanları ve toplam yolculuk süresi bilgisi ile MC kümesine eklenir. Eğer daha fazla aktarma ile gitmek isteniyorsa bir sonraki adıma geçilir.

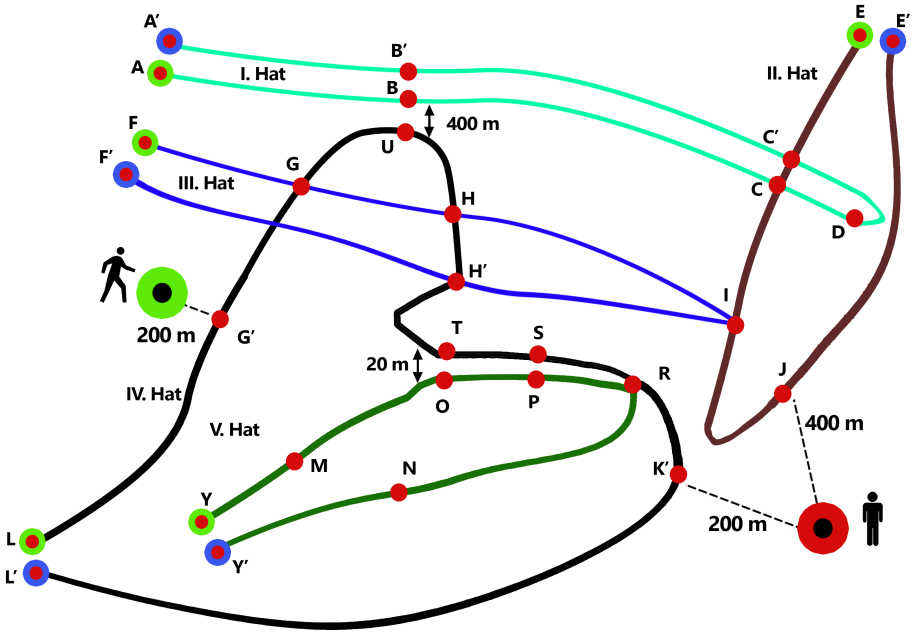
n aktarma $n+1$ hat ile istenilen noktalar arasında seyahatin mümkün olup olmadığını bakılır. Bunun için yukarıdaki bütün işlemler tekrar edilir. mümkün çözümler kümesine eklenen bütün çözümler toplam yolculuk sürelerine en az zamandan en uzun zamana göre sıralanır ve bilgiler sonuç olarak döndürülür. Algoritmanın akış diagramı Şekil 3.6'da verilmiştir.



Şekil 3.6: Önerdiğimiz algoritmanın akış diagramı

3.1.6 Algoritmanın Örnek Problemler Üzerinde Açıklamalı Uygulaması

3.1.6.1 Birinci Örnek



Şekil 3.7: Örnek kurgusal ulaşım haritası

Şekil 3.1 üzerinde Şekil 3.7'deki gibi herhangi iki nokta seçtik. Şimdi bu noktalar arasında tüm seyahat çözümlerini bulalım. Bu örnekte ve ardışık örneklerde K değerini 600 metre, L değerini 100 dk ve yolculuk başlangıç saati 07:00 olarak kabul edelim. Şekil 3.1'de iki nokta belirledik. Bu iki nokta üzerinden çözümlerimizi bulalım. İçerisi siyah dışında yeşil kontur olan nokta seyahat başlangıç noktası ve yine içerisi siyah dışarısında kırmızı kontur olan noktada varılmak istenilen noktayı temsil etmektedir. İlk olarak bu noktaları merkez nokta olacak şekilde 600 metre yarıçapında çemberler çizelim ve bu çemberler içersinde kalan durakları tespit edelim. Başlangıç noktasında oluşturulan çember içerisinde G' durağı ve varış noktasında oluşturulan çember içinde ise K' ve J durakları kalmaktadır. Bu duraklardan geçen hatları Tablo 3.1'den yararlanarak tesip edelim. G' ve K'

duraklarından IV. Hat ve J durağından ise II. Hat geçmektedir. Bu bilgiler ışığında olabilecek güzargah çözümlerimiz şöyledir. Başlangıç ve varış duraklarımızdan olan G' ve K' duraklarında da IV. Hat geçmektedir yani istenilen noktaya IV. Hat ile tek kullanarak gidilebilir. Diğer olabilecek çözümler ise şöyledir. İlk olarak IV. Hat'ta binip ardından varış hattı olarak II. Hat ile varmak isteğimiz noktaya gidebiliriz. İlk çözümümüz ile başlayalım.

Çözüm 1: İlk olarak bu durakların sıralarını Tablo 3.1'i kullanarak öğreniyoruz. Buna göre G' Durağı IV. Hat'ın ikinci durağı, K' Durağı IV. Hat'ın onununcu durağıdır. Durak sıralarını hattımızın gitmek istediğimiz yönde olup olmadığını anlamak ve daha kısa süreli yolculuk için kullacağız. G' Durağı K' Durağı'ndan daha önceki bir durak olduğu için seyahat edebilmemiz mümkündür.

Hat Adı	**Durak Adı**	**Enlem Boylam**	**Durak Sırası**	**Önceki Durak ile Arasındaki Süre**
IV. Hat	L	37.862880, 32.445645	1	0 dk
IV. Hat	G'	37.868686, 32.469219	2	10 dk
IV. Hat	G	37.868586, 32.465219	3	10 dk
IV. Hat	U	37.860062, 32.439157	4	5 dk
IV. Hat	H	37.866466, 32.460027	5	15 dk
IV. Hat	H'	37.864162, 32.483486	6	15 dk
IV. Hat	T	37.859124, 32.436561	7	10 dk
IV. Hat	S	37.865442, 32.430721	8	10 dk
IV. Hat	R	37.869710, 32.437642	9	10 dk
IV. Hat	K'	37.869510, 32.437542	10	15 dk
IV. Hat	L'	37.869410, 32.437342	11	10 dk

Tablo 3.7: IV. Hat'a ait durak bilgileri tablosu

Yolcunun ilk olarak 200 metre yürüyerek G' Durağı'na gitmesi gerekmektedir. Tablo 3.2'den yolculuğumuzun başlama saatinin 90 dakika ilerisine kadar IV. Hat'ın seferinin olup olmadığına bakıyoruz. Tablo 3.2'ye göre IV. Hat'ın saat 07:30'da seferi bulunmaktadır.

Hat	**Tarife**	**Hafta içi/sonu**
IV. Hat	7:30	Hafta içi
IV. Hat	18:30	Hafta içi
IV. Hat	8:30	Hafta sonu
IV. Hat	19:30	Hafta sonu

Tablo 3.8: IV. Hat'ın tarifeleri tablosu

Buna sefer göre hattın G' durağında ne zaman olacağını öğrenmek için Tablo 3.6' ya bakıyoruz. Hattımız saat 07:40'da G' durağında olmaktadır. 07:40'da IV. Hattı binen yolcumuz saat 09:10'da K' durağında olmaktadır. Saat 07:40'ta başlayan otobüs yolculuğu 09:10'da sona ermektedir. 200 metre yürüyerek istenilen noktaya ulaşılabilir. Otobüs ile toplam yolculuk süresi 1 saat 30 dakikadır. MC kümesine eklenir.

Hat	**Tarife**	**Durak Adı**	**Durak Sırası**	**Hafta içi/sonu**	**Varış Zamanı**
IV. Hat	7:30	L	1	H.İ.	7:30
IV. Hat	7:30	G'	2	H.İ.	7:40
IV. Hat	7:30	G	3	H.İ.	7:50
IV. Hat	7:30	U	4	H.İ.	7:55
IV. Hat	7:30	H	5	H.İ.	8:10
IV. Hat	7:30	H'	6	H.İ.	8:25
IV. Hat	7:30	T	7	H.İ.	8:35
IV. Hat	7:30	S	8	H.İ.	8:45
IV. Hat	7:30	R	9	H.İ.	8:55
IV. Hat	7:30	K'	10	H.İ.	9:10
IV. Hat	7:30	L'	11	H.İ.	9:20

Tablo 3.9: IV. Hat'ın varış zamanı tablosu

Çözüm 2: İkinci çözümümüzün başlangıç hattı IV. Hat ve varış hattı II. Hat idi. Bu hatların birbiriyle direk kesişen durakları olup olmadığına bakmak için Tablo 3.3' den bu bilgilerle sorgumuzu yapıyoruz. Sorgu sonucuna bu iki hattın direk kesişen durakları bulunmamaktadır. Bir sonraki aşamaya başlangıç ve varış hattı bu iki hattı olan aktarma hatlarını bulmak için Tablo 3.4'e bakıyoruz. Sorgumuza göre ilk olarak IV. Hat daha sonra I. Hat ve son olarak da II. Hat kullanılarak ve yine ilk IV. Hat daha sonra III. Hat ve son olarak da II. Hat kullarak istenilen noktalar arası seyahat edilebilir.

Başlangıç Hattı	Varış Hattı	Aktarma Hattı
III. Hat	I. Hat	II. Hat
III. Hat	I. Hat	IV. Hat
III. Hat	V. Hat	IV. Hat
IV. Hat	II. Hat	I. Hat
IV. Hat	II. Hat	III. Hat
V. Hat	I. Hat	IV. Hat
V. Hat	III. Hat	IV. Hat

Tablo 3.10: IV. Hat ile II. Hat ile biten aktarmalar tablosu

İlk olarak 200 metre yürüyerek G' Durağı'na giderek seyahate başlıyoruz. İkinci çözümümüzün ilk aşamasını IV. Hat ile I. Hat Tablo 3.3'dan sorgulayarak başlıyoruz. Sorguya göre IV. Hat'ın G' Durağı'ndan binip U Durağı'nda inip daha sonra yürüyerek I. Hat'ın B Durağı'ndan tekrar binilmesi gerekmektedir.

başlangıç Hattı			**Varış Hattı**			**Mesafe**
Hat Adı	**Durak Adı**	**Durak Sırası**	**Hat Adı**	**Durak Adı**	**Durak Sırası**	
IV. Hat	U	4	I. Hat	B	6	400 m
IV. Hat	G	2	III. Hat	G	2	0
IV. Hat	H	5	III. Hat	H	3	0
IV. Hat	H'	6	III. Hat	H'	5	0
IV. Hat	T	7	V. Hat	O	3	0
IV. Hat	S	8	V. Hat	P	4	0
IV. Hat	R	9	V. Hat	R	5	0

Tablo 3.11: IV. Hat ile başlayan I. Hat biten hat ilişki tablosu

Daha sonra I. Hat ile II. Hat'tı Tablo 3.3'den sorguluyoruz. İki hattın kesişen durakları C ve C' duraklarıdır.

başlangıç Hattı			**Varış Hattı**			**Mesafe**
Hat Adı	**Durak Adı**	**Durak Sırası**	**Hat Adı**	**Durak Adı**	**Durak Sırası**	
I. Hat	C'	3	II. Hat	C'	2	0
I. Hat	C	5	II. Hat	C	5	0
I. Hat	B	2	IV. Hat	U	3	400 m
II. Hat	C'	2	I. Hat	C'	3	0
II. Hat	C	3	I. Hat	C	5	0
II. Hat	I	4	III. Hat	I	4	0

Tablo 3.12: I. Hat ile başlayan II. Hat biten hat ilişki tablosu

I. Hat'a göre kişinin B Durağı'nda binip C ya da C' Durağı'nda inmesi gerekmektedir. C ve C' duraklarını durak sıraları ve B Durağı'nın durak sırasından küçük olduğundan aracımızın gitmek isteğimiz yöne değil tam tersi istikamete gittiğinin bilgisini alıyoruz.Bu sebepten bu çözüm geçerli çözüm olamaktan çıkmaktadır.

Hat Adı	Durak Adı	Enlem Boylam	Durak Sırası	Önceki Durak ile Arasındaki Süre
I. Hat	A'	37.871418, 32.490522	1	0 dk
I. Hat	B'	37.870393, 32.501228	2	10 dk
I. Hat	C'	37.870648, 32.508365	3	5 dk
I. Hat	D	37.871418, 32.490522	4	10 dk
I. Hat	C	37.870393, 32.501528	5	10 dk
I. Hat	B	37.870648, 32.508565	6	5 dk
I. Hat	A	37.870476, 32.516543	7	15 dk

Tablo 3.13: I. Hat'a ait durak bilgileri tablosu

Bir diğer alternatif olan sırayla IV. Hat, III. Hat ve II. Hat çözümünü inceleyelim. Bunun için ilk olarak başlangıç hattı olarak IV. Hat ve varış hattı olarak III. Hat'ı kabul ederek Tablo 3.3'den sorgulayalım. Bu hatlar G, G' ve H durakların kesişmektedirler. IV. Hat'a göre durak sırası en küçük olan G Durağı'nda inip yine aynı duraktan III. Hat ile devam edilir.

Başlangıç Hattı			Varış Hattı			Mesafe
Hat Adı	Durak Adı	Durak Sırası	Hat Adı	Durak Adı	Durak Sırası	
IV. Hat	U	4	I. Hat	B	6	400 m
IV. Hat	G	2	III. Hat	G	2	0
IV. Hat	H	5	III. Hat	H	3	0
IV. Hat	H'	6	III. Hat	H'	5	0
IV. Hat	T	7	V. Hat	O	3	0
IV. Hat	S	8	V. Hat	P	4	0
IV. Hat	R	9	V. Hat	R	5	0

Tablo 3.14: IV. Hat ile başlayan III. Hat biten hat ilişki tablosu

Daha sonra III. Hat ile başlayan II. Hat ile biten hat ilişkilerine bakılır. I durağında inip yine I durağından II. Hat'a binip J durağında inerek yolculuk gerçekleşebilir.

başlangıç Hattı			Varış Hattı			Mesafe
Hat Adı	Durak Adı	Durak Sırası	Hat Adı	Durak Adı	Durak Sırası	
III. Hat	I	4	II. Hat	I	4	0
III. Hat	G	2	IV. Hat	G	3	0
III. Hat	H	3	IV. Hat	H	5	0

IV. Hat	U	4	I. Hat	B	6	400 m
IV. Hat	G	2	III. Hat	G	2	0
IV. Hat	H	5	III. Hat	H	3	0
IV. Hat	H'	6	III. Hat	H'	5	0

Tablo 3.15: III. Hat ile başlayan II. Hat biten hat ilişki tablosu

Zamana göre tarifelere bakılır. İlk olarak saat 7:30'da sefere başlayan ve saat 7:40'da G' Durağı'ndan geçen IV. Hat'a binip G Durağı'nda inilmesi gerekir. Saat 7:00'da sefere başlayan ve saat 08:05'de G Durağı'ndan geçecek olan III. Hat'a binilir ve I Durağı'nda inilir. Daha sonra saat 8:30'da sefere başlayan ve saat 09:00'da I durağından geçecek olan II. Hat'a binilir ve J Durağı'nda inilir. Yolculuk saat 09:10'da sona erer.

Hat	**Tarife**	**Hafta içi/sonu**
II. Hat	8:30	Hafta içi
II. Hat	12:30	Hafta içi
II. Hat	17:30	Hafta içi
III. Hat	8:00	Hafta içi
III. Hat	18:00	Hafta içi
IV. Hat	7:30	Hafta içi
IV. Hat	18:30	Hafta içi

Tablo 3.16: IV. Hat, III. Hat ve II. Hat'tın tarifeleri tablosu

Hat	**Tarife**	**Durak Adı**	**Durak Sırası**	**Hafta içi/sonu**	**Varış Zamanı**
II. Hat	8:30	E	1	H.İ.	8:30
II. Hat	8:30	C'	2	H.İ.	8:40
II. Hat	8:30	C	3	H.İ.	8:45
II. Hat	8:30	I	4	H.İ.	9:00 (3)
II. Hat	8:30	J	5	H.İ.	9:10 (4)
II. Hat	8:30	E'	6	H.İ.	9:20
III. Hat	8:00	F	1	H.İ.	8:00
III. Hat	8:00	G	2	H.İ.	8:05 (2)
III. Hat	8:00	H	3	H.İ.	8:15
III. Hat	8:00	I	4	H.İ.	8:30
III. Hat	8:00	H'	5	H.İ.	8:35

III. Hat	8:00	F'	6	H.İ.	8:45
IV. Hat	7:30	L	1	H.İ.	7:30
IV. Hat	7:30	G'	2	H.İ.	7:40 (1)
IV. Hat	7:30	G	3	H.İ.	7:50
IV. Hat	7:30	U	4	H.İ.	7:55
IV. Hat	7:30	H	5	H.İ.	8:10
IV. Hat	7:30	H'	6	H.İ.	8:25
IV. Hat	7:30	T	7	H.İ.	8:35
IV. Hat	7:30	S	8	H.İ.	8:45
IV. Hat	7:30	R	9	H.İ.	8:55
IV. Hat	7:30	K'	10	H.İ.	9:10
IV. Hat	7:30	L'	11	H.İ.	9:20

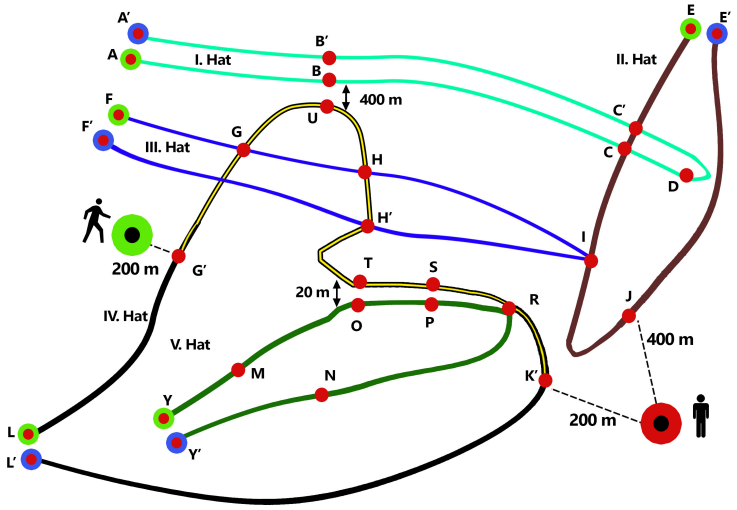
Tablo 3.17: Varış Zamanı Tablosu

Daha sonra 400 metre yürüyerek istenilen noktaya ulaşılabilir. Otobüs ile toplam yolculuk süresi 1 saat 30 dakikadır. Mümkün çözümler kümesine eklenir. Bulduğumuz bu MC en kısa yolculuk süresine göre sıralanarak kullanıcıya bilgi olarak sonulur.

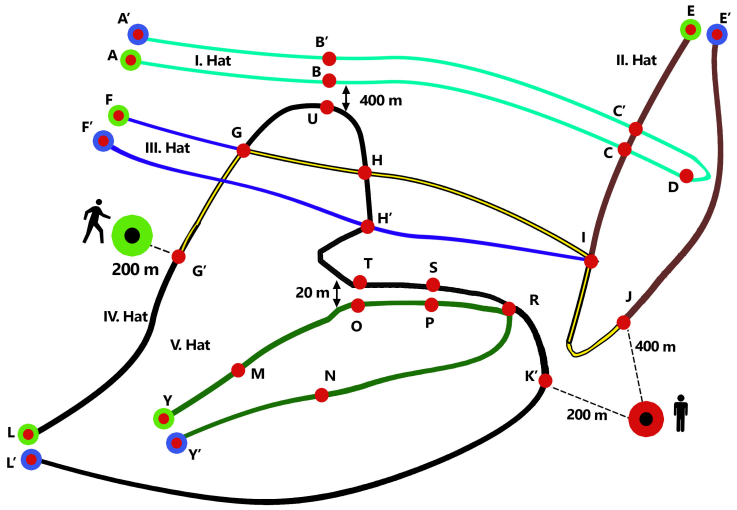
Çözüm	**Süre**	**Güzergah Bilgisi**
1. çözüm	90 dk	200m → IV. Hat → 200m
2. çözüm	90 dk	200m → IV. Hat → III. Hat → II. Hat→ 200m

Tablo 3.18: Birinci örnek problemin çözümleri tablosu

Şekil 3.8' de Şekil 3.9'de çözümün güzergahları harita üzeri için sarı ve dışı siyah konturlu olarak gösterilmiştir.

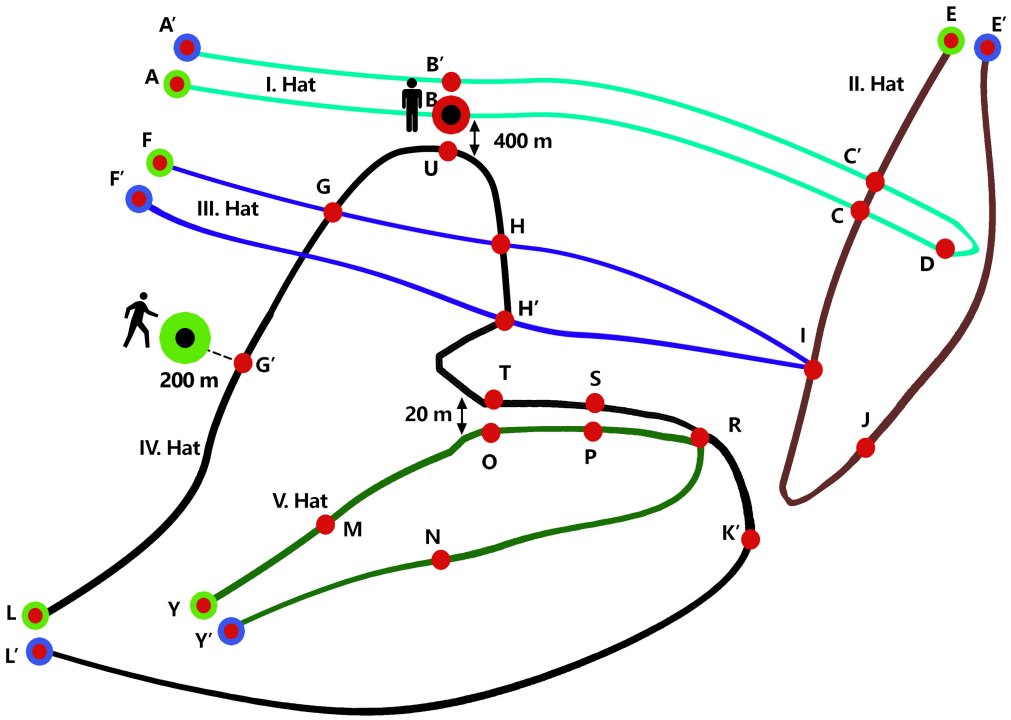


Şekil 3.8: Birinci çözümün harita üzerinde içi sarı ve dışı siyah konturlu çizgilerle verilmiştir



Şekil 3.9: İkinci çözümün harita üzerinde içi sarı ve dışı siyah konturlu çizgilerle verilmiştir

3.1.6.2 İkinci Örnek



Şekil 3.10: İkinci örnek problem haritası

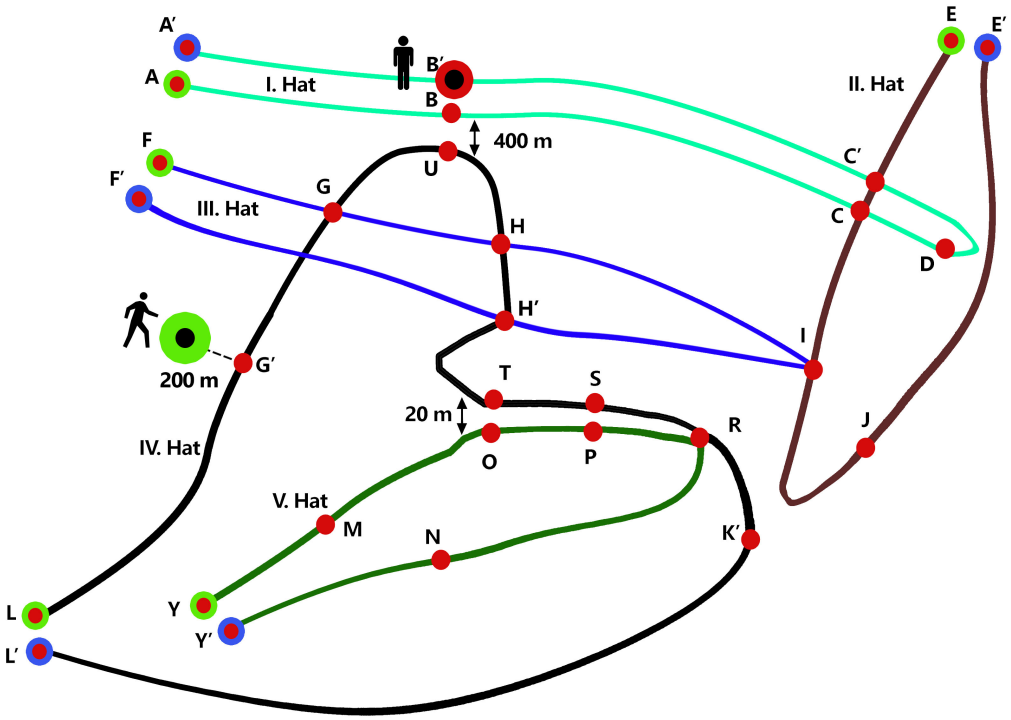
İlk olarak başlangıç ve varış noktarı merkezde olacak şekilde 600 metre yarı çapında çember çiziyoruz. Çember içinde kalan duraklar başlangıç noktasındaki çemberin içinde G' ve varış noktasındaki çemberin içinde B ve U Durakları kalmaktadır. G' ve U Durakları'ndan IV. hat ve B durağından II. hat geçmektedir. Bu durumda hat çözümleri IV. Hat ve IV. hat - II. Hat'ları sırasıyla kullanılabilir. U ve B Durakları kesişen duraklar olduğu için iki çözümden biri kullanılır.

Çözüm 1 : Saat 07:40'ta IV. Hat'a binilir ve 07:55'te U Durağı'nda inilir ve 400 metre yürünerek istenilen noktaya varılır.

Çözüm	Süre	Güzergah Bilgisi
1. çözüm	15 dk	200m → IV. Hat → 400m

Tablo 3.19: İkinci örnek problemin çözümleri tablosu

3.1.6.3 Üçüncü Örnek



Şekil 3.11: Üçüncü örnek problem haritası

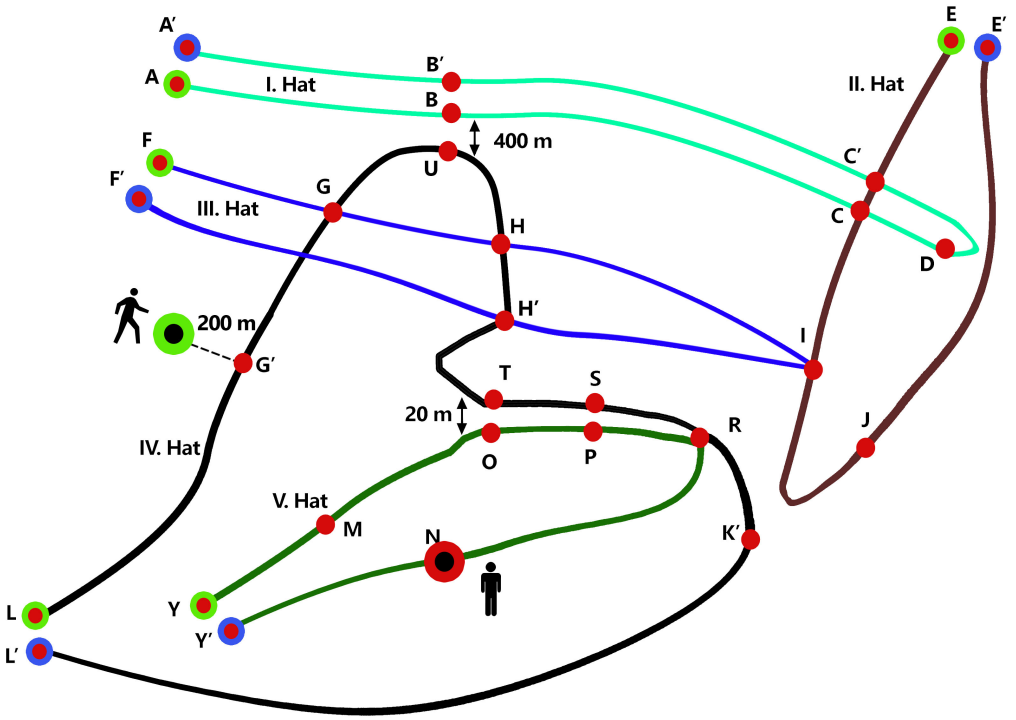
İlk olarak başlangıç ve varış noktarı merkezde olacak şekilde 600 metre yarı çapında çember çiziyoruz. Çember içinde kalan duraklar başlangıç noktasındaki çemberin içinde G' ve varış noktasındaki çemberin içinde B' Durağı kalmaktadır. G' Durağı'ndan IV. hat ve B Durağı'ndan II. hat geçmektedir. Bu durumda IV. hat - II. Hat'ları sırasıyla kullanılarak istenilen noktalara ulaşılabilir.

Çözüm 1: IV. Hat ve II. Hat sadece B Durağı'nda kesişmektedir. Gitmek istediğimiz B' Durağı durak sırasına göre daha önce olduğu için bu iki nokta için çözüm yoktur.

Çözüm	Süre	Güzergah Bilgisi
1. çözüm	- dk	Çözüm yoktur.

Tablo 3.20: Üçüncü örnek problemin çözümleri tablosu

3.1.6.4 Dördüncü Örnek



Şekil 3.12: Dördüncü örnek problem haritası

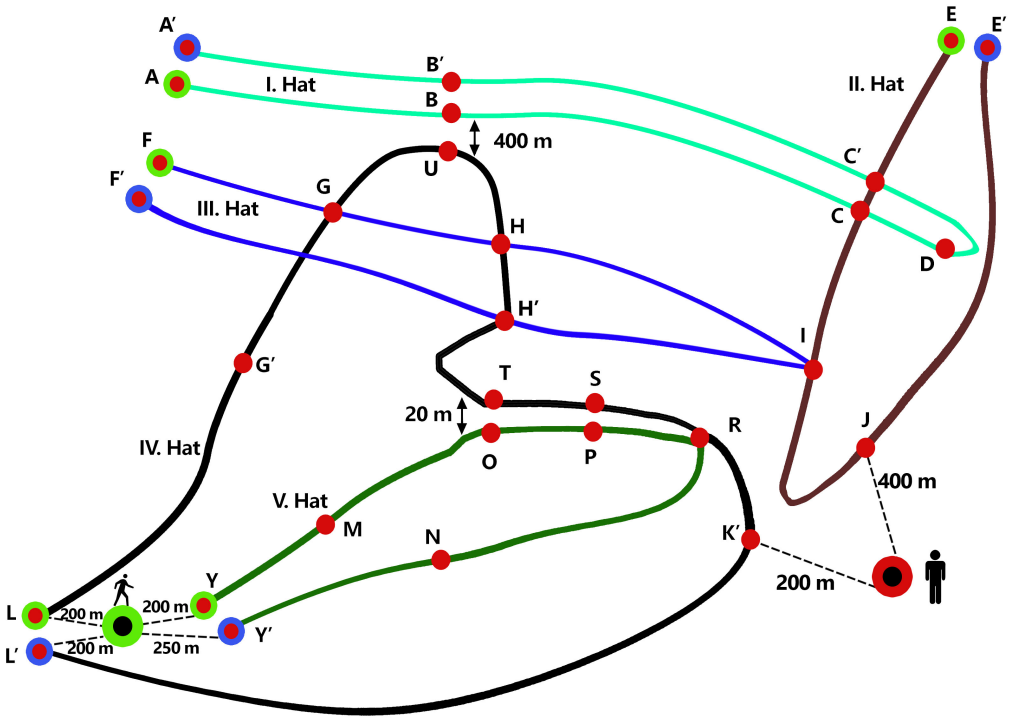
İlk olarak başlangıç ve varış noktarı merkezde olacak şekilde 600 metre yarı çapında çember çiziyoruz. Çember içinde kalan duraklar başlangıç noktasındaki çemberin içinde G' ve varış noktasındaki çemberin içinde N Durağı kalmaktadır. G' Durağı'ndan IV. hat ve N Durağı'ndan V. hat geçmektedir. Bu durumda IV. hat - V. Hat'ları sırasıyla kullanılarak istenilen noktalara ulaşılabilir. IV. Hat ile V. Hat sırasıyla $T-O$, $S-P$, R duraklarında kesişmektedir. Durak sırası en küçük olan bizim için ilk çözüm olacağın T ve O Durakları'ndan aktarma yapabiliriz.

Çözüm 1: Saat 07:40'ta IV. Hat ile G' Durağı'ndan binilir ve saat 08:35'te T Durağı'nda olabiliriz. 100 dk ilerisine V. Hat'ın seferlerine bakalır. Saat 09:45'te seferine başlayan V. Hat saat 10:10'da O Durağı'nda olmaktadır. O Durağı'ndan binilir ve saat 10:35'te N Durağı'nda inilerek yolculuk tamamlanır. Toplam yolculuk süresi 185 dakikadır.

Çözüm	Süre	Güzergah Bilgisi
1. çözüm	185 dk	200m→ IV. Hat → V. Hat

Tablo 3.21: Dördüncü örnek problemin çözümleri tablosu

3.1.6.5 Beşinci Örnek



Şekil 3.13: Beşinci örnek problem haritası

İlk olarak başlangıç ve varış noktarı merkezde olacak şekilde 600 metre yarı çapında çember çiziyoruz. Çember içinde kalan duraklar; başlangıç noktasındaki çemberin içinde L, L', Y, Y' ve varış noktasındaki çemberin içinde K' ve J Durakları kalmaktadır. L, L', Y, Y' Durakları'ndan IV. Hat ve V. Hat, K', J Durakları'ndan IV. hat ve II. Hat geçmektedir. Bu durumda IV. Hat, V. Hat-IV. Hat ve V. Hat-II. Hat'ları sırasıyla kullanılarak istenilen noktalara ulaşılabilir. L' ve Y' Durakları hatları son durakları olduğu için bu duraklar bizim çözümlerimiz için kullanılamaz. Bu bilgi duraklarını kontrol ederek edindik.

Çözüm 1: Saat 07:30'da IV. Hat ile L Durağı'ndan binilir ve saat 09:10'da K' Durağı'nda olabiliriz. Toplam yolculuk süresi 100 dakikadır.

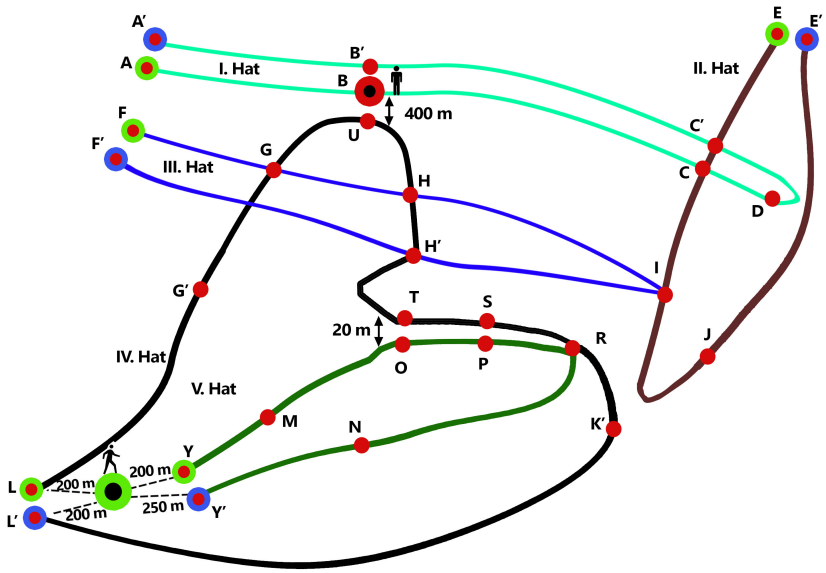
Çözüm 2: V. Hat ve IV. Hat'ları kullanarak ulaşabileceğimiz çözüm. Bu hatlar sırasıyla $O-T$, $P-S$ ve R Durakları'nda kesişmektedir. Bu duraklardan ilk kesişen $O-T$ Durakları seçilir. Saat 09:45'te Y Durağı'dan binilir ve saat 10:10'da O Durağı'nda inilir. Bu saatten 100 dakika ilerisine kadar IV. Hat'ın sefer olup olmadığına bakılır. IV. Hat'tın seferi olmadığından çözüm geçersiz bir çözümdür.

Çözüm 3: V. Hat ve II. Hat'ları kullanarak ulaşabileceğimiz çözüm. Bu hatlar direk kesişmemektedir. 3 hat kesişim tablosunda da çözümü olmadığından çözüm yoktur.

Çözüm	Süre	Güzergah Bilgisi
1. çözüm	100 dk	IV. Hat → 200m

Tablo 3.22: Beşinci örnek problemin çözümleri tablosu

3.1.6.6 Altıncı Örnek



Şekil 3.14: Altıncı örnek problem haritası

İlk olarak başlangıç ve varış noktarı merkezde olacak şekilde 600 metre yarı çapında çember çiziyoruz. Çember içinde kalan duraklar; başlangıç noktasındaki çemberin içinde L, L', Y, Y' ve varış noktasındaki çemberin içinde B Durakları kalmaktadır. L, L', Y, Y' Durakları'ndan IV. Hat ve V. Hat, B Durağı'ndan II. hat geçmektedir. Bu durumda IV. Hat-II. Hat, V. Hat-II. Hat'ları sırasıyla kullanılarak istenilen noktalara ulaşılabilir. L' ve Y' durakları, hatların son durakları olduğu için bu duraklar bizim çözümlerimiz için kullanılamaz. Bu bilgi duraklarını kontrol ederek edindik.

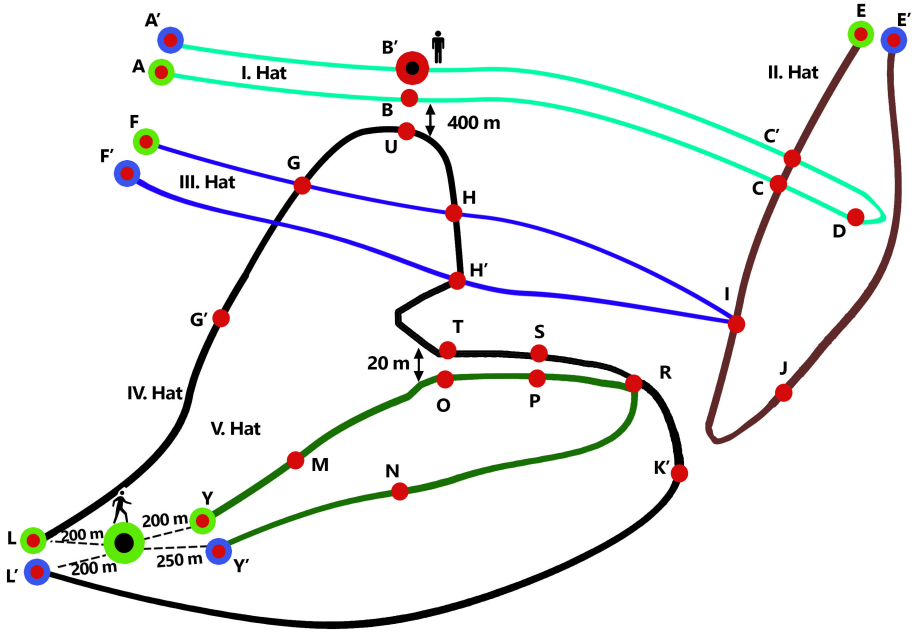
Çözüm 1: IV. Hat ve II. Hat'ları kullanarak ulaşabileceğimiz çözüm. Bu iki hattın birbiriyle kesişen durakları olmadığı için aktarma hattıyla gidibilecek alternetiflere bakalır. I. Hat ile aktarma yapılarak istenilen yere gidilebilir. Saat 07:30' da IV. Hat'a binilir ve saat 07:55'de U Durağı'nda inilir. B Durağı'na geçilir. Ancak I. Hat ve II. Hat'ların kesiştiği C ve C' Duraklarında kesiştiğinden B Durağı'nın durak sırası bu iki duraktan sonra olduğu için çözüm buradan ileri gidememektedir ve çözüm yoktur.

Çözüm 2 : V. Hat ve II. Hat'ları kullanarak ulaşabileceğimiz çözüm. Bu hatlar direk birbiriyle kesişmemektedir. Aktarmalı çözümlerde alternatifler aranır. Diğer aktarmalı çözümlerde de cevap bulunamadığı için çözüm yoktur.

Çözüm	**Süre**	**Güzergah Bilgisi**
1. çözüm	- dk	Çözüm yoktur.

Tablo 3.23: Altıncı örnek problemin çözümleri tablosu

3.1.6.7 Yedinci Örnek



Şekil 3.15: Yedinci örnek problem haritası

İlk olarak başlangıç ve varış noktarı merkezde olacak şekilde 600 metre yarı çapında çember çiziyoruz. Çember içinde kalan duraklar; başlangıç noktasındaki çemberin içinde L, L', Y, Y' ve varış noktasındaki çemberin içinde B' Durakları kalmaktadır. L, L', Y, Y' Durakları'ndan IV. Hat ve V. Hat, B' Durağı'ndan II. hat geçmektedir. Bu durumda IV. Hat-II. Hat, V. Hat-II. Hat'ları sırasıyla kullanılarak istenilen noktalara ulaşılabilir. L' ve Y' Durakları hatları son durakları olduğu için bu duraklar bizim çözümlerimiz için kullanılamaz. Bu bilgi duraklarını kontrol ederek edindik.

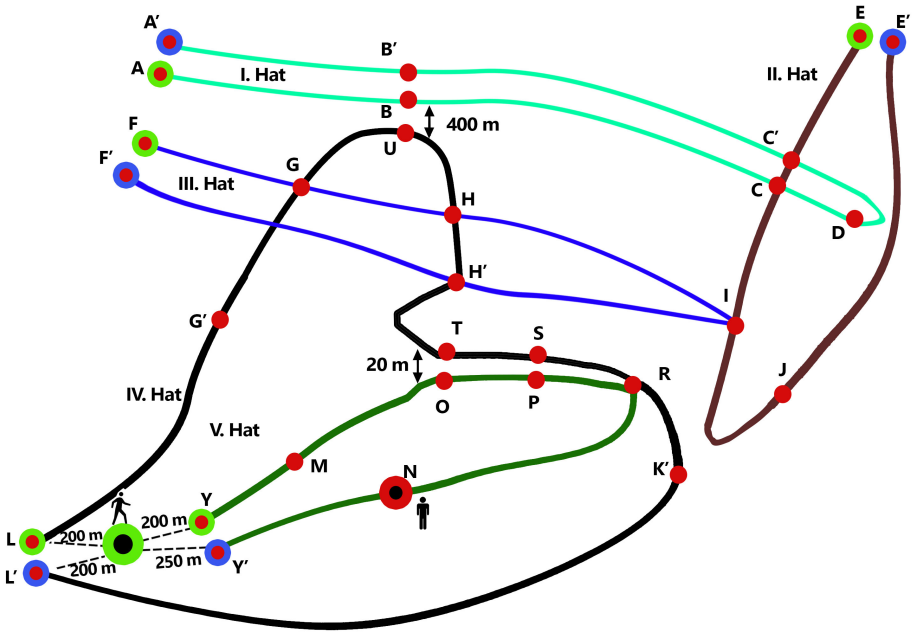
Çözüm 1: IV. Hat ve II. Hat'ları kullanarak ulaşabileceğimiz çözüm. Bu iki hattın biribiryle kesişen durakları olmadığı için aktarma hattıyla gidibilecek alternetiflere bakalır. I. Hat ile aktarma yapılarak istenilen yere gidilebilir. Saat 07:30'da IV. Hat'a binilir ve saat 07:55'de U Durağı'nda inilir.B Durağı'na geçilir. Ancak I. Hat ve II. Hat'ların kesiştiği C ve C' Duraklarında kesiştiğinden B Durağı'nın durak sırası bu iki duraktan sonra olduğu için çözüm buradan ileri gidememektedir ve çözüm yoktur.

Çözüm 2: V. Hat ve II. Hat'ları kullanarak ulaşabileceğimiz çözüm. Bu hatlar direk birbiriyle kesişmemektedir. Aktarmalı çözümlerde alternatifler aranır. Diğer aktarmalı çözümlerde de cevap bulunamadığı için çözüm yoktur.

Çözüm	Süre	Güzergah Bilgisi
1. çözüm	- dk	Çözüm yoktur.

Tablo 3.24: Yedinci örnek problemin çözümleri tablosu

3.1.6.8 Sekizinci Örnek



Şekil 3.16: Sekizinci örnek problem haritası

İlk olarak başlangıç ve varış noktarı merkezde olacak şekilde 600 metre yarı çapında çember çiziyoruz. Çember içinde kalan duraklar; başlangıç noktasındaki çemberin içinde L, L', Y, Y' ve varış noktasındaki çemberin içinde N Durakları kalmaktadır. L, L', Y, Y' Durakları'ndan IV. Hat ve V. Hat, N Durağı'ndan V. hat geçmektedir. Bu durumda IV. Hat-V. Hat, V. Hat'ları sırasıyla kullanılarak istenilen noktalara

ulaşılabilir. L' ve Y' Durakları hatları son durakları olduğu için bu duraklar bizim çözümlerimiz için kullanılamaz. Bu bilgi duraklarını kontrol ederek edindik.

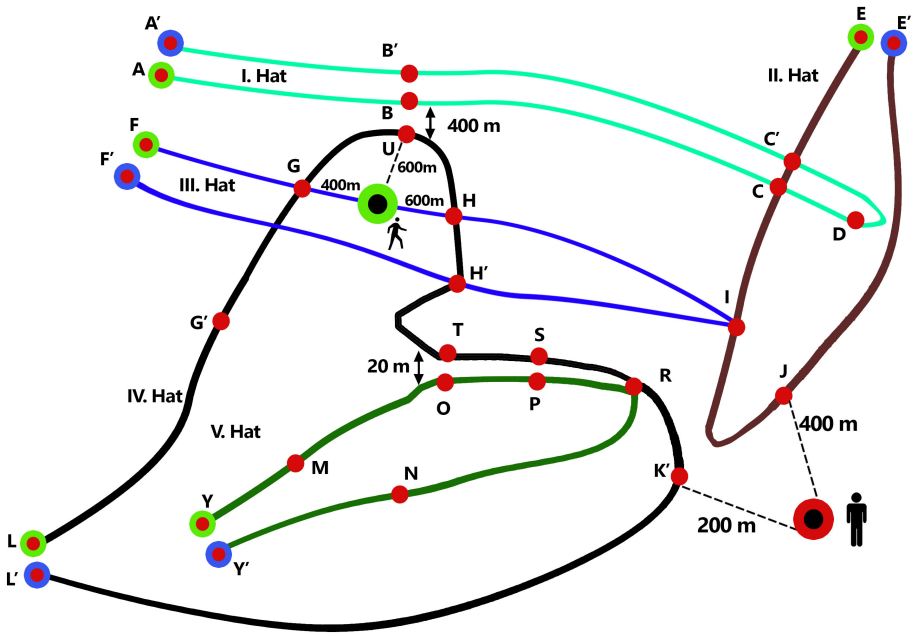
Çözüm 1: Saat 07:30'da IV. Hat ile L Durağı'ndan binilir ve saat 08:35'te T Durağı'nda olabiliriz. 100 dk ilerisine V. Hat'tın seferlerine bakalır. Saat 09:45'te seferine başlayan V. Hat saat 10:10'da O Durağı'nda olmaktadır. O Durağı'ndan binilir ve saat 10:35'te N Durağı'nda inilerek yolculuk tamamlanır. Toplam yolculuk süresi 185 dakikadır.

Çözüm 2: Sadece V. Hat kullanarak ulaşabileceğimiz çözüm. Saat 09:45'te Y Durağı'ndan binilir ve saat 10:35'te N Durağı'nda inilerek istenilen noktaya 50 dakikada varılabilir.

Çözüm	**Süre**	**Güzergah Bilgisi**
1. çözüm	185 dk	200m$\rightarrow$ IV. Hat $\rightarrow$ V. Hat
2. çözüm	50 dk	200m$\rightarrow$ V. Hat

Tablo 3.25: Sekizinci örnek problemin çözümleri tablosu

3.1.6.9 Dokuzuncu Örnek



Şekil 3.17: Dokuzuncu örnek problem haritası

İlk olarak başlangıç ve varış noktarı merkezde olacak şekilde 600 metre yarı çapında çember çiziyoruz. Çember içinde kalan duraklar; başlangıç noktasındaki çemberin içinde G, U, H ve varış noktasındaki çemberin içinde K ve J Durakları kalmaktadır. G, U, H Durakları'ndan IV. Hat-III. Hat ve K' ve J Durağı'ndan IV. Hat-II. Hat'lar geçmektedir. Bu durumda III. Hat-IV. Hat, IV. Hat-II. Hat, IV. Hat ve III. Hat-II. Hat şeklinde istenilen noktaya gidilebilir.

Çözüm 1 : III. Hat ve IV. Hat'ları kullanarak ulaşabileceğimiz çözüm. En az yürüyerek ulaşabileceğimiz G Durağı'na gidilir. Saat 07:05'te G Durağı'ndan binilir ve iki hat H ve H' Durakları'ndan kesişmektedir. Durak sırasına göre en küçük olan H Durağı'nda inilir. Saat 08:10'da H Durağı'ndan geçecek olan IV. Hat'a binilir ve saat 09:10'da K' Durağı'nda inilir. 200 metre yürüyerek istenilen hedefe ulaşılır. Toplam yolculuk süresi 125 dakikadır.

Çözüm 2 : IV. Hat ve II. Hat'ları kullanarak ulaşabileceğimiz çözüm. Bu hatlar direk birbiriyle kesişmemektedir. Aktarmalı çözümlerde alternatifler aranır. IV.

Hat-I. Hat-II. Hat'ları ya da IV. Hat-III. Hat-II. Hat'larını kullanarak istenilen noktaya gidilebilir. IV. Hat ve I. Hat U ve B Durakları'nda kesişmektedir. Ancak I. Hat ve II. Hat'tın kesiştiği durak sırası B Durağı'ndan küçük olduğu için çözüm yoktur. Diğer alternatif te G ve H Durakları III. Hat'tında durakları olduğu için çözümü kullanmak sadece IV. Hat'a ait olan U Durağı'ndan saat 07:55'te binilir ve saat 08:10'da H Durağı'nda inilir. Saat 08:10'un 90 dakika sonrasına kadar III. Hat'ın seferi olmadığı için çözüm yoktur.

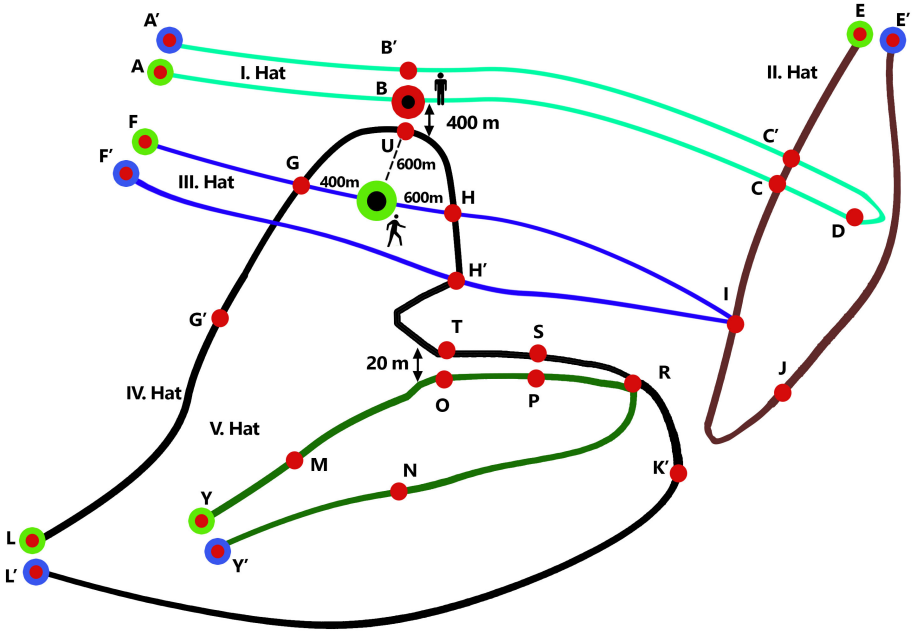
Çözüm 3 : Sadece IV. Hat kullanarak ulaşabileceğimiz çözüm. En az yürüyerek ulaşabileceğimiz G Durağı'na gidilir. Saat 07:05'te G Durağı'ndan binilir ve saat 09:10'da K' Durağı'nda inilerek istenilen noktaya 125 dakikada varılabilir.

Çözüm 4 : III. Hat ve II. Hat'ları kullanarak ulaşabileceğimiz çözüm. Saat 08:10'da H Durağı'ndan III. Hat'a binilir ve saat 09:10'da K' Durağı'nda inilerek istenilen noktaya 60 dk'da varılabilir.

Çözüm	**Süre**	**Güzergah Bilgisi**
1. çözüm	125 dk	400m→ III. Hat → IV. Hat→ 200m
2. çözüm	- dk	Çözüm yoktur.
3. çözüm	125 dk	400m→ IV. Hat→ 200m
4. çözüm	60 dk	600m→ III. Hat → II. Hat→ 400m

Tablo 3.26: Dokuzuncu örnek problemin çözümleri tablosu

3.1.6.10 Onuncu Örnek



Şekil 3.18: Onuncu örnek problem haritası

İlk olarak başlangıç ve varış noktarı merkezde olacak şekilde 600 metre yarı çapında çember çiziyoruz. Çember içinde kalan duraklar; başlangıç noktasındaki çemberin içinde G, U, H ve varış noktasındaki çemberin içinde B Durağı kalmaktadır. G, U, H Durakları'ndan IV. Hat-III. Hat ve B Durağı'ndan I. Hat geçmektedir. Bu durumda III. Hat-I. Hat, IV. Hat-I. Hat şeklinde istenilen noktaya gidilebilir.

Çözüm 1 : III. Hat ve I. Hat'ları kullanarak ulaşabileceğimiz çözüm. Bu hatlar direk birbiriyle kesişmemektedir. Aktarmalı çözümlerde alternatifler aranır. III. Hat-II. Hat-I. Hat'ları ya da III. Hat-IV. Hat-I. Hat'larını kullanarak istenilen noktaya gidilebilir. İlk alternatif III. Hat-II. Hat-I. Hat şeklinde olan çözümü inceleyelim. III. Hat ve II. Hat I Durağı'nda kesişmektedir. Ancak II. Hat ile I. Hat'ın kesiştiği Cve C' duraklarının durak sırası I Durağı'nın durak sırasından küçük olduğu için çözüm yoktur. III. Hat-IV. Hat-I. Hat şekildeki çözümü inceleyelim. III. Hat ile başlanıldığında inilecek olan H Durağı IV. Hat ile I. Hat'tın kesişimi olan U Durağı'nın durak sırasından büyük olduğu için çözüm yoktur.

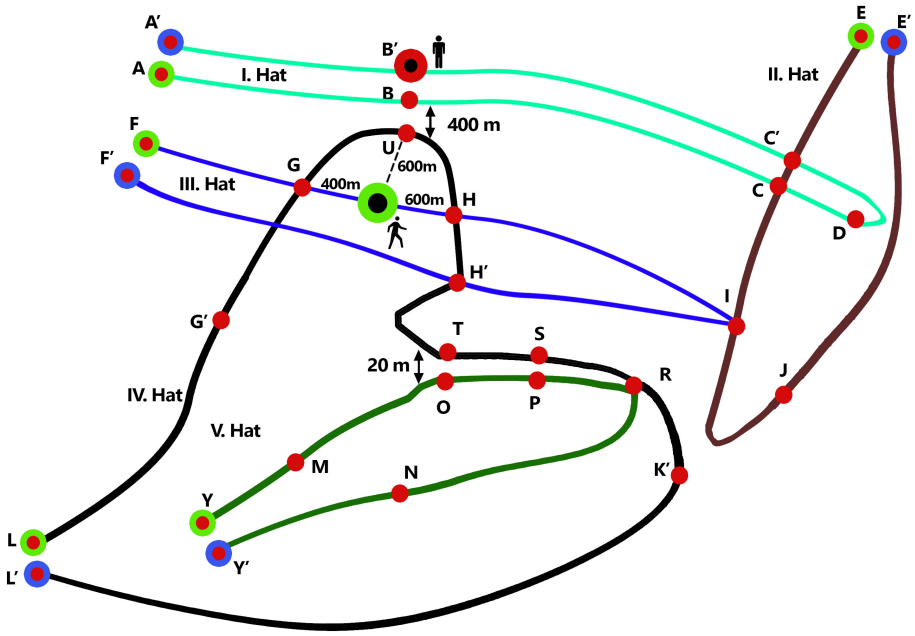
Çözüm 2 : IV. Hat ve I. Hat'ları kullanarak ulaşabileceğimiz çözüm. En az yürüyerek ulaşabileceğimiz G Durağı'na gidilir. Saat 07:50'te G Durağı'ndan binilir ve saat 07:55'te U Durağı'nda inilir. 400 metre yürüyerek 5 dk içerisinden istenilen noktaya gidilir.

Çözüm 3 : Sadece IV. Hat kullanarak ulaşabileceğimiz çözüm. En az yürüyerek ulaşabileceğimiz G Durağı'na gidilir. Saat 07:05'te G Durağı'ndan binilir ve saat 09:10'da K' Durağı'nda inilerek istenilen noktaya 125 dk da varılabilir.

Çözüm	**Süre**	**Güzergah Bilgisi**
1. çözüm	5 dk	400m→ III. Hat → IV. Hat→I. Hat→ 400m
2. çözüm	- dk	Çözüm yoktur.
3. çözüm	125 dk	400m→ IV. Hat → I. Hat→ 400m

Tablo 3.27: Onuncu örnek problemin çözümleri tablosu

3.1.6.11 Onbirinci Örnek



Şekil 3.19: Onbirinci örnek problem haritası

İlk olarak başlangıç ve varış noktarı merkezde olacak şekilde 600 metre yarı çapında çember çiziyoruz. Çember içinde kalan duraklar; başlangıç noktasındaki çemberin içinde G, U, H ve varış noktasındaki çemberin içinde B Durağı kalmaktadır. G, U, H Durakları'ndan IV. Hat-III. Hat ve B' Durağı'ndan I. Hat geçmektedir. Bu durumda III. Hat-I. Hat, IV. Hat-I. Hat şeklinde istenilen noktaya gidilebilir.

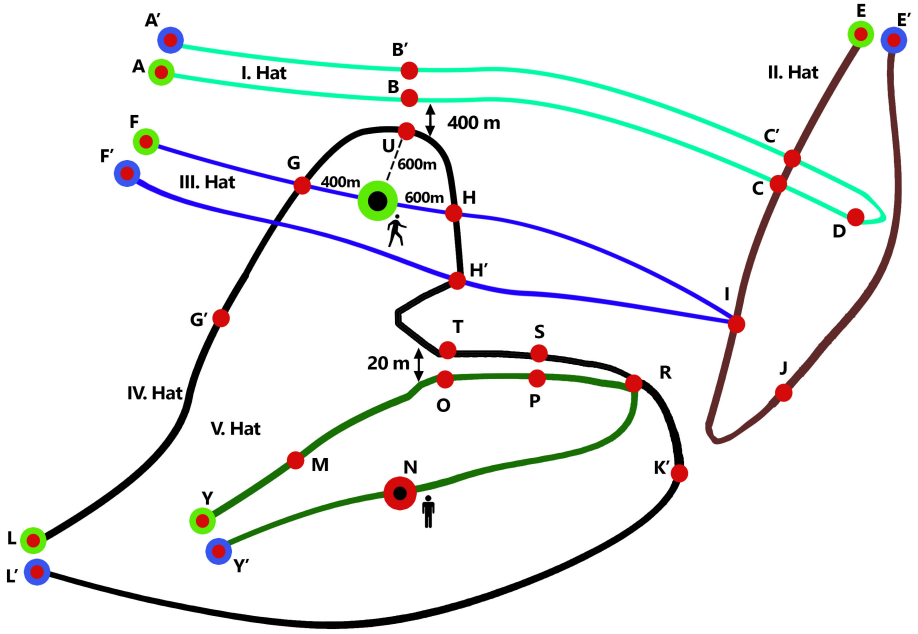
Çözüm 1 : III. Hat ve I. Hat'ları kullanarak ulaşabileceğimiz çözüm. Bu hatlar direk birbiriyle kesişmemektedir. Aktarmalı çözümlerde alternatifler aranır. III. Hat-II. Hat-I. Hat'ları ya da III. Hat-IV. Hat-I. Hat'larını kullanarak istenilen noktaya gidilebilir. İlk alternatif III. Hat-II. Hat-I. Hat şeklinde olan çözümü inceleyelim. III. Hat ve II. Hat I durağında kesişmektedir. Ancak II. Hat ile I. Hat'ın kesiştiği C ve C' Durakları'nın durak sırası I Durağı'nın durak sırasından küçük olduğu için çözüm yoktur. III. Hat-IV. Hat-I. Hat şekildeki çözümü inceleyelim. IV. Hat ve I. Hat B Durağı'nda kesişmektedir ancak inilmek istenen B' Durağı'nın durak sırası daha küçük olduğu için çözüm yoktur.

Çözüm 2 : IV. Hat ve I. Hat'ları kullanarak ulaşabileceğimiz çözüm. IV. Hat ve I. Hat B Durağı'nda kesişmektedir ancak inilmek istenen B' Durağı'nın durak sırası daha küçük olduğu için çözüm yoktur.

Çözüm	**Süre**	**Güzergah Bilgisi**
1. çözüm	- dk	Çözüm yoktur.

Tablo 3.28: Onbirinci örnek problemin çözümleri tablosu

3.1.6.12 Onikinci Örnek



Şekil 3.20: Onikinci örnek problem haritası

İlk olarak başlangıç ve varış noktarı merkezde olacak şekilde 600 metre yarı çapında çember çiziyoruz. Çember içinde kalan duraklar; başlangıç noktasındaki çemberin içinde G, U, H ve varış noktasındaki çemberin içinde N Durağı kalmaktadır. G, U, H Durakları'ndan IV. Hat-III. Hat ve N Durağı'ndan V. Hat geçmektedir. Bu durumda III. Hat-V. Hat, IV. Hat-V. Hat şeklinde istenilen noktaya gidilebilir.

Çözüm 1 : III. Hat ve V. Hat'ları kullanarak ulaşabileceğimiz çözüm. Bu hatlar direk birbiriyle kesişmemektedir. Aktarmalı çözümlerde alternatifler aranır. III. Hat-IV. Hat-V. Hat'larını kullanarak istenilen noktaya gidilebilir. 400 metre yürüyerek saat 07:05'te G Durağı'ndan geçecek olan III. Hat'a binilir ve saat 07:15'te H Durağı'nda inilir. Saat 08:10'da bu duraktan geçecek olan IV. Hat'a binilir. IV. Hat ve V. Hat'ın durak sırana göre en küçük olan saat 08:35'te T Durağı'nda inilir ve 20 metre yürüyerek iki hattın kesişim durağı olan O Durağı'na gidilir. Saat 10:10'da bu duraktan geçecek olan V. Hat'a binilir. Saat 10:35'te N Durağı'nda

inilir. Toplam yolculuk süresi 210 dakikadır.

Çözüm 2 : IV. Hat ve V. Hat'ları kullanarak ulaşabileceğimiz çözüm. 400 metre yürüyerek saat 07:50'te G Durağı'ndan geçecek olan IV. Hat'a binilir. IV. Hat ve V. Hat'ın durak sırana göre en küçük olan saat 08:35'te T Durağı'nda inilir ve 20 metre yürüyerek iki hattın kesişim durağı olan O Durağı'na gidilir. Saat 10:10'da bu duraktan geçecek olan V. Hat'a binilir. Saat 10:35'te N Durağı'nda inilir. Toplam yolculuk süresi 165 dakikadır.

Çözüm	**Süre**	**Güzergah Bilgisi**
1. çözüm	210 dk	400m→ III. Hat→ IV. Hat →20m→ V. Hat
1. çözüm	165 dk	400m→ IV. Hat→ 20m→ V. Hat

Tablo 3.29: Onikinci örnek problemin çözümleri tablosu

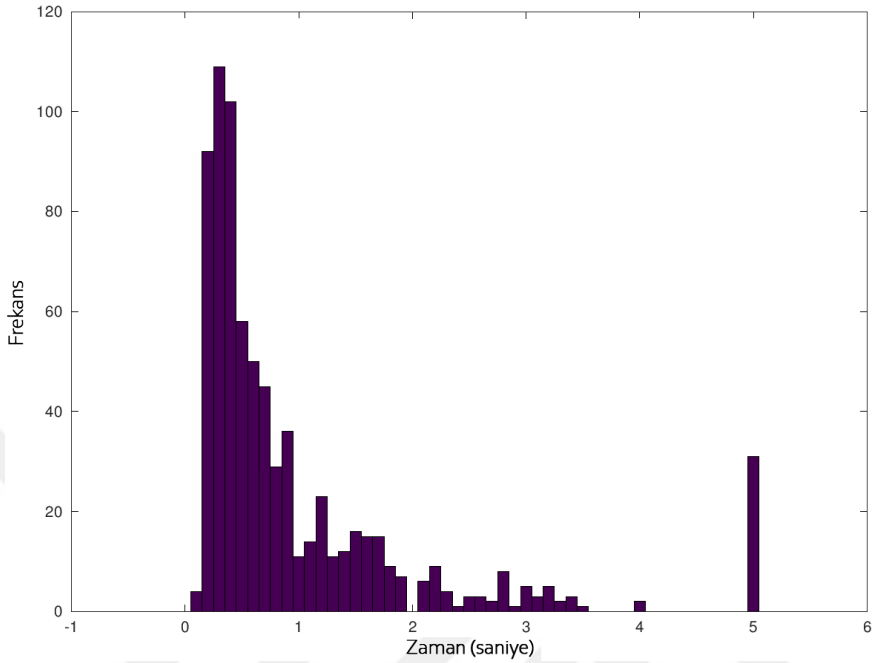
3.2 Sistemde Kullanılan Platformlar

Sistemin tasarımında Delphi, C Sharp, Php, Jquery ve Ajax dilleri kullanıldı. Sistemdeki ikincil tabloların oluşturulmasında Delphi, oluşturulan bu tablolardan sorgu işleminin yapılıp cevap dönülecek olan kısımda C Sharp ve dönülen cevapların web ortamında harita üzerine aktarılıp detaylı bilgi verilecek olan kısımda da PhP, Jquery ve Ajax dilleri kullanıldı. Sistemi koşturduğumuz PC'ye ait bilgiler şöyledir. İşlemcisi Intel Xeon CPU E7-4850 2.00 GHz, RAM'ı 8 GB, harddiski 250 GB ve işletim sistemi de Windows Server 2012 64 bittir. Veritabanı olarak Microsoft SQL Server 2016 kullanılmıştır. Bu özellikler doğrultusunda 5000 duraklı, 830 araçlı ve 420 hatlı birincil tabloları (Tablo 3.1 ve Tablo 3.2) hazır olan bir toplu ulaşım sistemi için algoritmanın koşma süresi yaklaşık 45 dakikadır. Bu süre sonunda tüm ikincil tablolar oluşturulabilmektedir. Toplu taşıma sisteminde olası değişikliklerde algoritmanın yeniden koşturulması gerekmektedir.

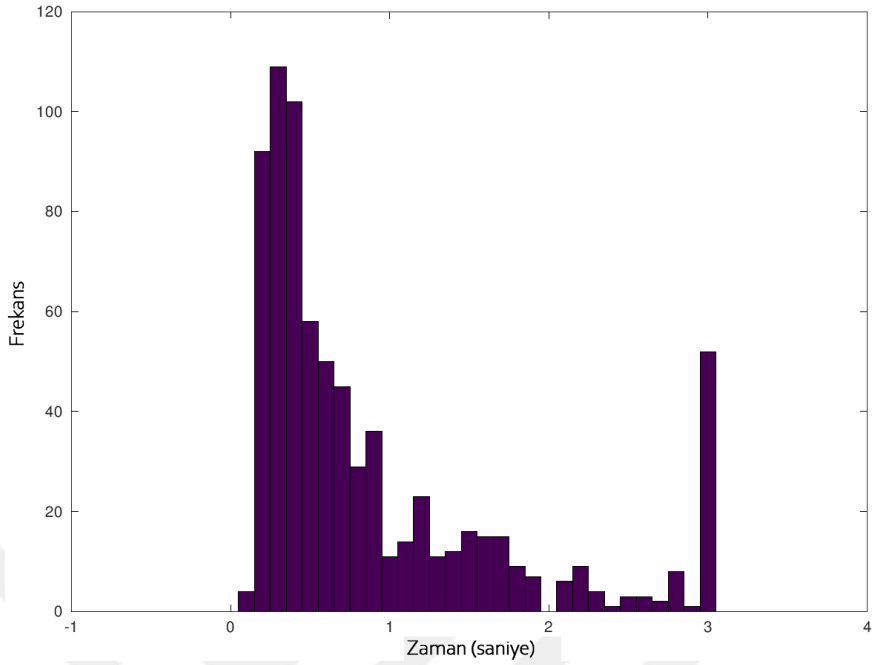
3.3 Sistemin Hesaplama Süreleri ve Grafikleri

Algoritmada yapılan yaklaşık 31.000 gerçek sorgu verilerine göre hat bazlı bir çözüm için ortalama hesaplama süreleri grafikleri çıkarılmıştır. x Ekseni etiketi ortalama hesaplama süresini ve y Ekseni etiketi işlem sayısını temsil etmektedir. x ekseninde hesaplama aralığı 0,1 olarak ayarlanmıştır. Bir hat için hesaplama süresi grafiği Şekil 3.21'de, iki hat için hesaplama süresi grafiği Şekil 3.22'de, üç hat için hesaplama süresi grafiği Şekil 3.23'de, dört hat için hesaplama süresi grafiği

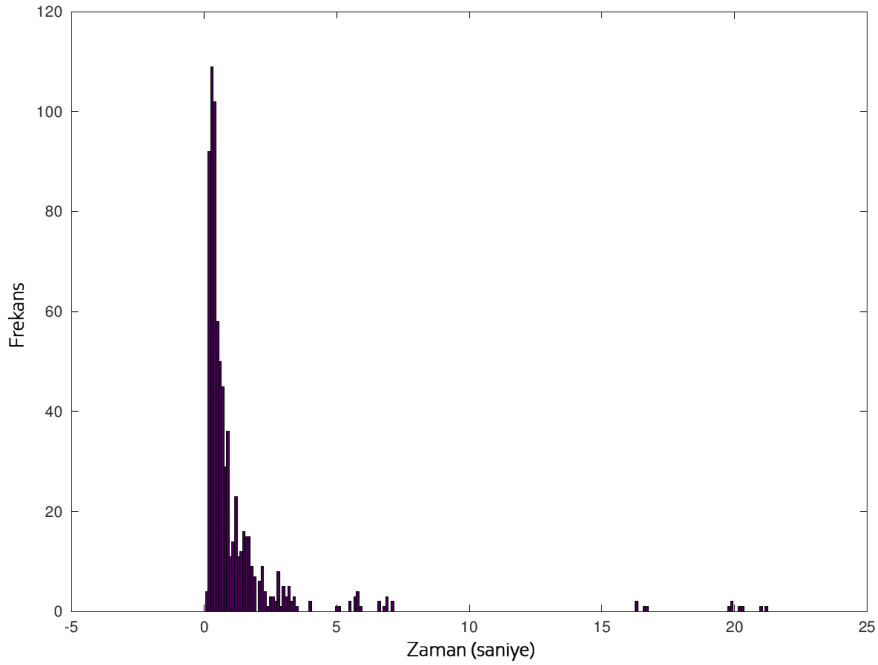
Şekil 3.24'de verilmiştir. Tek hat, iki hat, üç hat ve dört için ortalama hesaplama süresi ve ortalama varyansları Tablo 3.30'de verilmiştir.



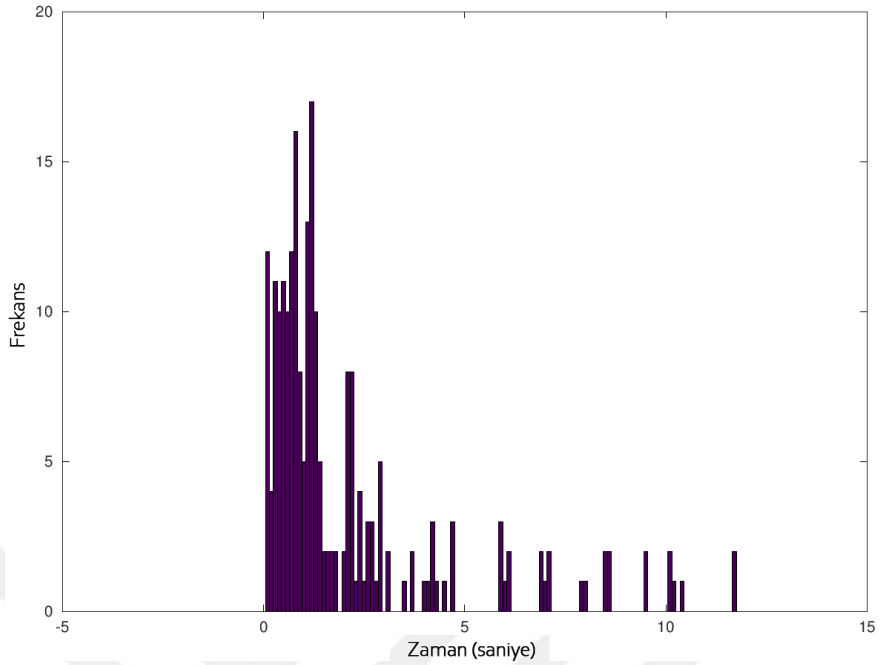
Şekil 3.21: Tek hat için hesaplama süresi histogramı



Şekil 3.22: İki hat için hesaplama süresi histogramı



Şekil 3.23: Üç hat için hesaplama süresi histogramı



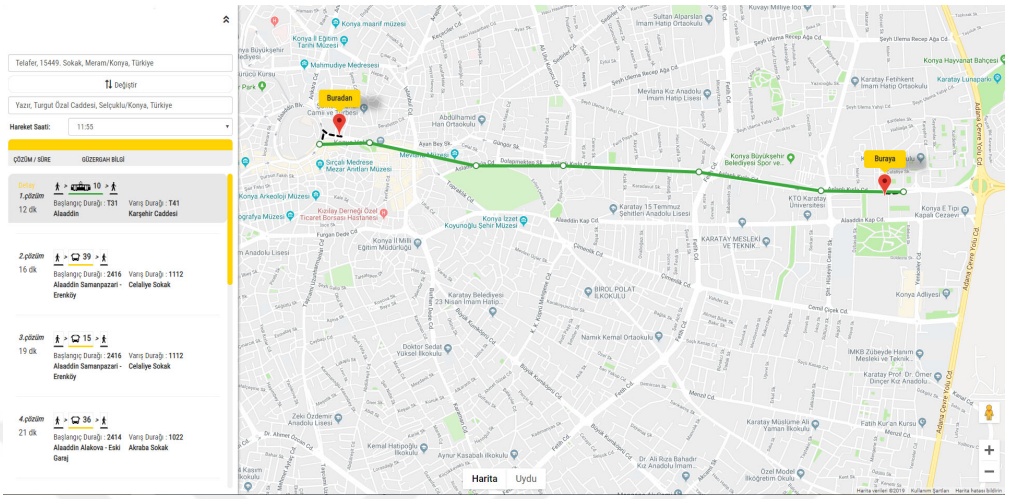
Şekil 3.24: Dört hat için hesaplama süresi histogramı

Hat Sayısı	**Ortalama Hesaplama Süresi**	**Ortalama Varyans**
Bir hat	0.46782 sn	0.14019
İki hat	0.25097 sn	0.10443
Üç hat	1.2252 sn	5.9864
Dört hat	2.0786 sn	5.9603

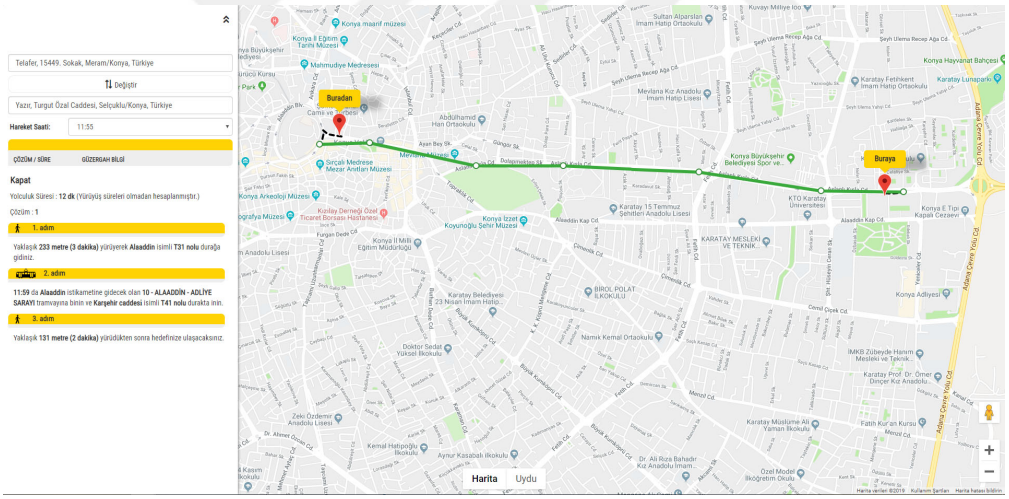
Tablo 3.30: Hat sayılarına göre ortalama hesaplama süresi ve ortalama varyanslar

Yukarıdaki grafikler ve bilgiler algoritmanın çalışma hızı hakkında detaylı bilgi vermektedir. Algoritmamız ile sorgu tabanlı güzergah planamada çok kısa sürelerde sistemimizin cevap verdiği görülmektedir.

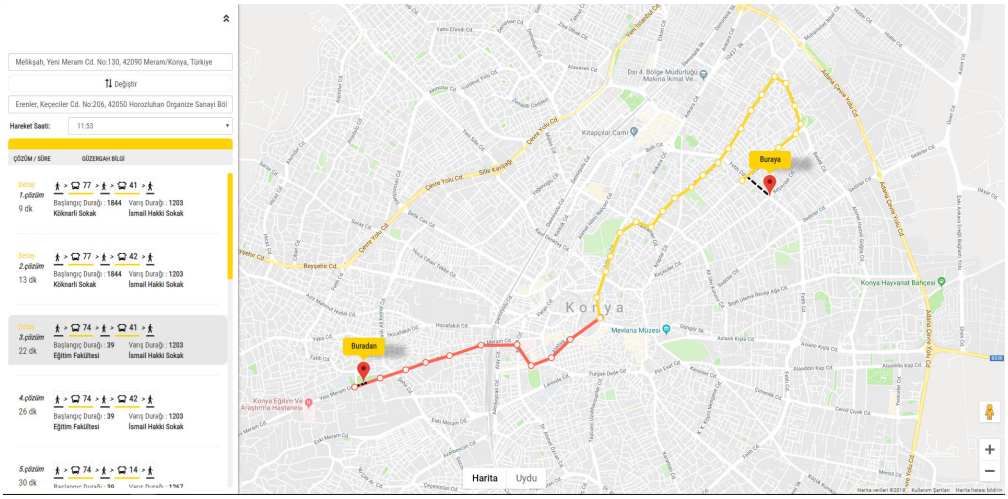
3.4 Gerçek Uygulamadan Sonuçlar



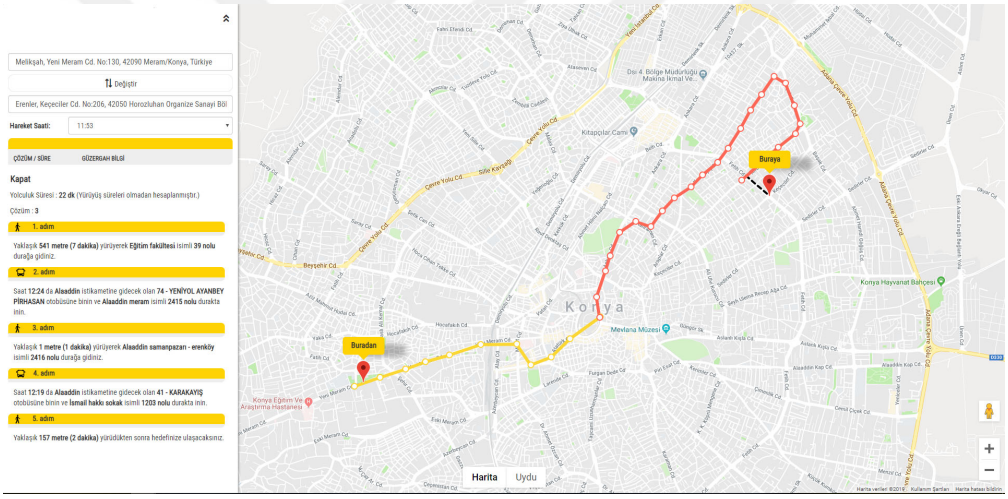
Şekil 3.25: Tek hat kullanarak kullanılan ulaşım çözümü örneği



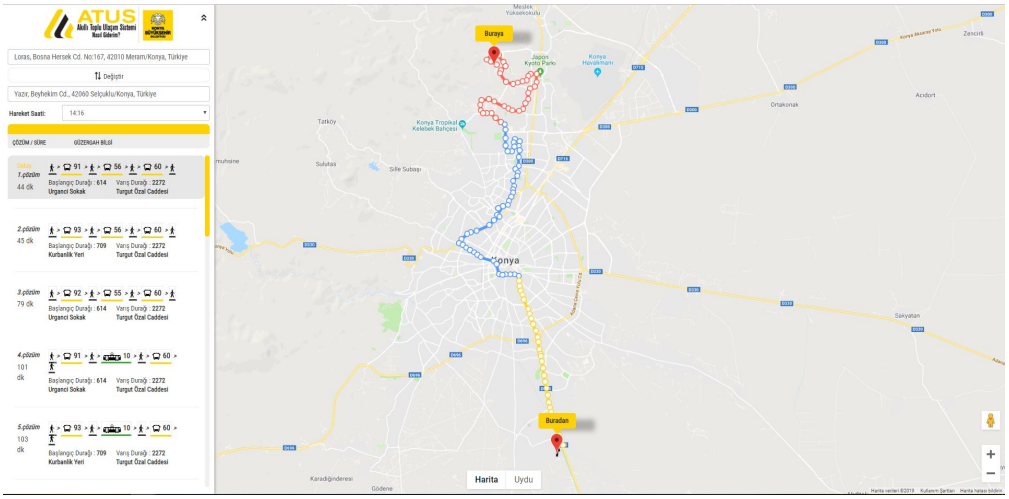
Şekil 3.26: Tek hat kullanarak kullanılan ulaşım çözümü örneği detayı



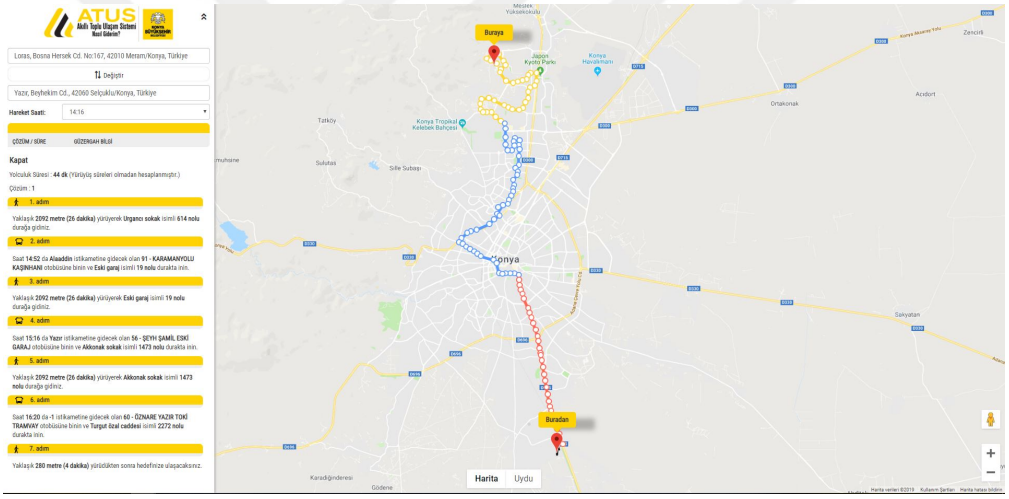
Şekil 3.27: İki hat kullanarak kullanılan ulaşım çözümü örneği



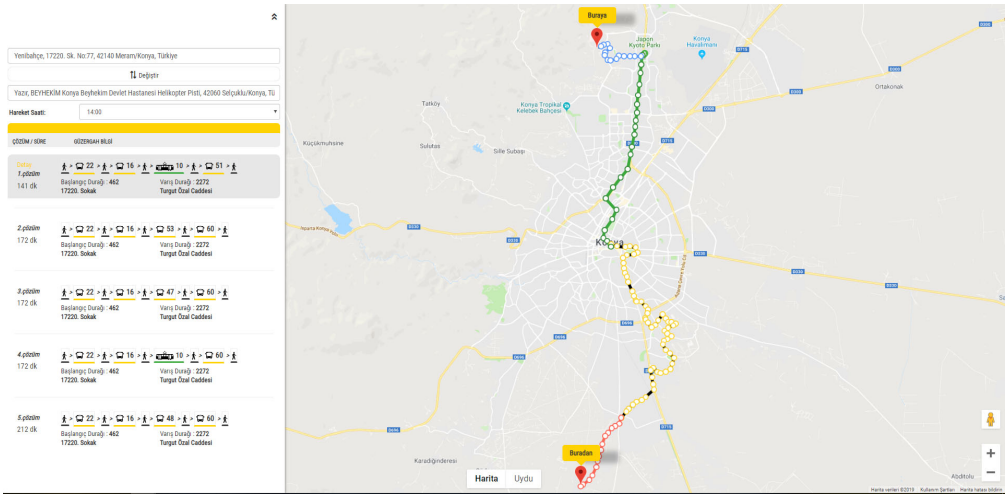
Şekil 3.28: İki hat kullanarak kullanılan ulaşım çözümü örneği detayı



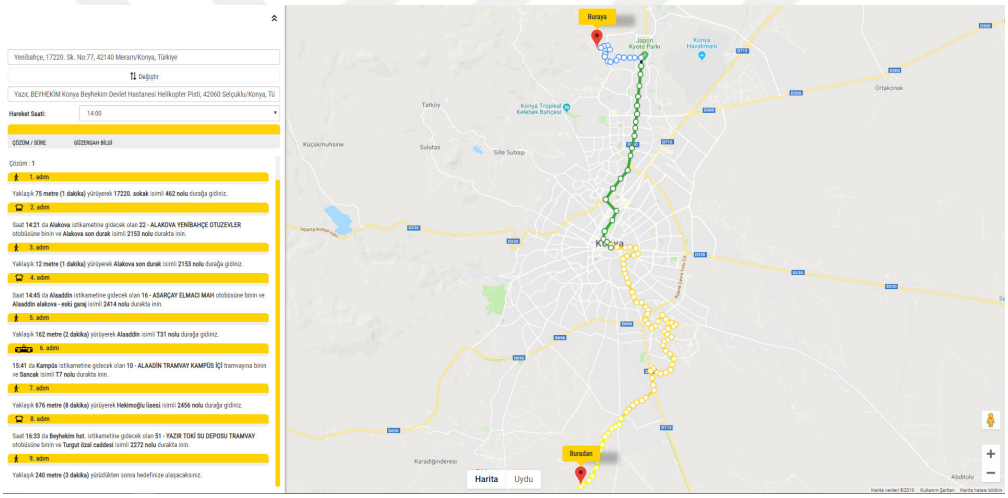
Şekil 3.29: Üç hat kullanarak kullanılan ulaşım çözümü örneği



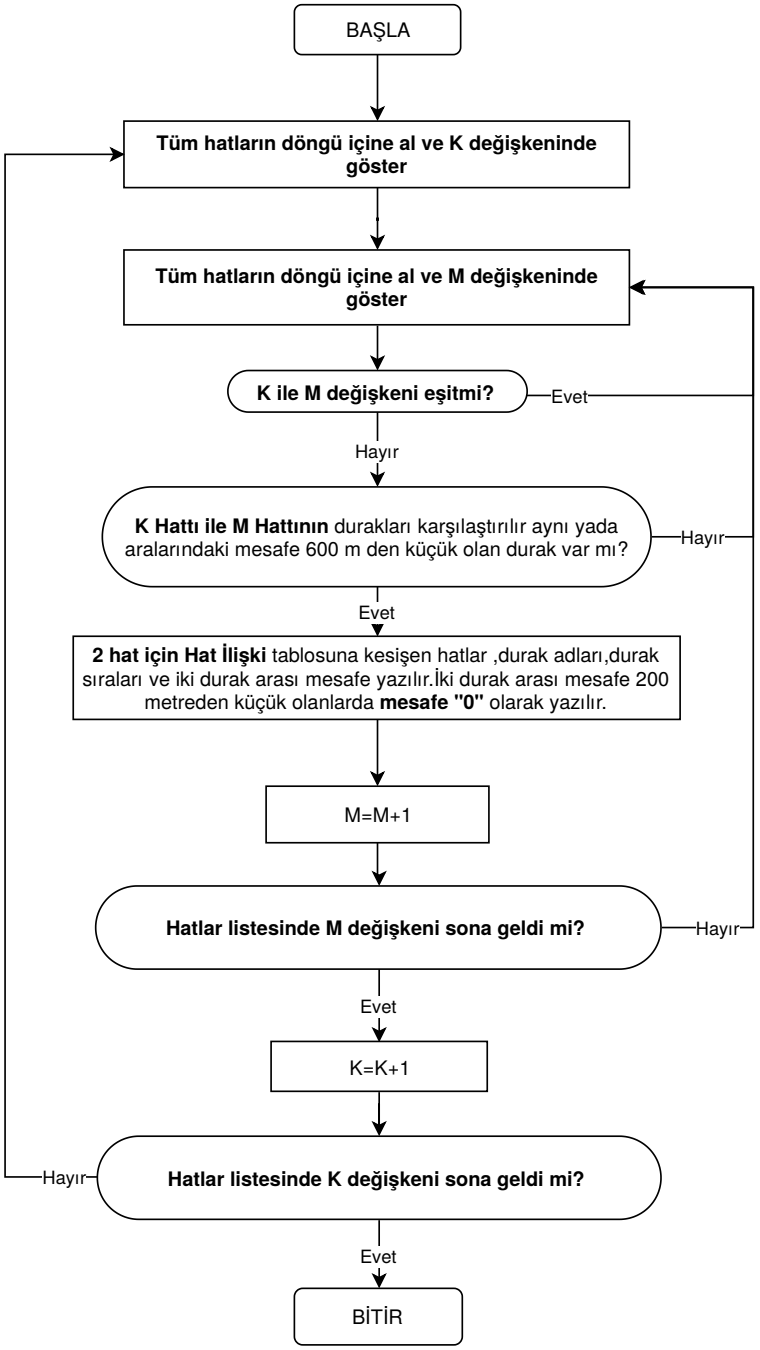
Şekil 3.30: Üç hat kullanarak kullanılan ulaşım çözümü örneği detayı



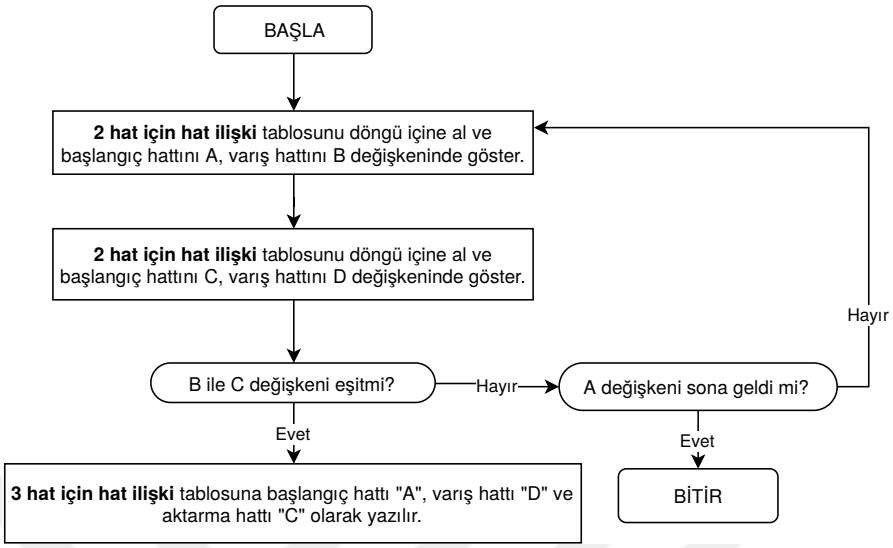
Şekil 3.31: Dört hat kullanarak kullanılan ulaşım çözümü örneği



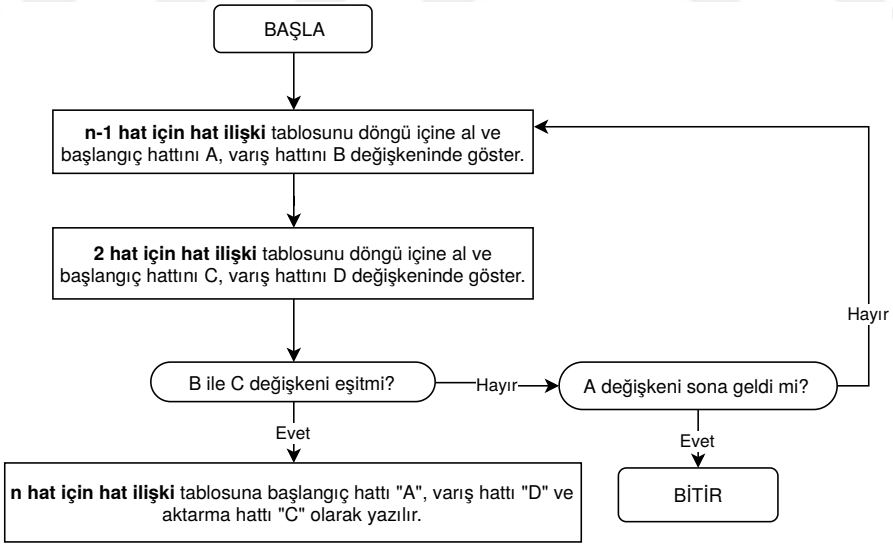
Şekil 3.32: Dört hat kullanarak kullanılan ulaşım çözümü örneği detayı



Şekil 3.2: 2-hat için hat ilişki diagramı



Şekil 3.3: 3-hat için Hat İlişki Diagramı



Şekil 3.4: n-hat için hat ilişki diagramı

4 Sonuçlar

Günümüzde şehirlerin hızla büyümesi ile şehir içerisinde bir noktadan bir noktaya seyahatler artık giderek karmaşık bir hal almaktadır. Şehir içi toplu taşıma ağlarında farklı türden birçok seyahat aracı vardır. Örneğin, otobüs, dolmuş, tramvay, vapur, gibi birçok toplu taşıma aracı şehir içi toplu taşıma kullanılmaktadır. Tabi ki birçok toplu taşıma alternatifinin güzellikleri olduğu gibi yapının büyüklüğüyle orantılı karmaşıklıkta ortaya çıkmaktadır. Şehir içinde bir noktadan başka bir noktaya bu araçlardan birini ya da birden fazlasını kullanarak ulaşabiliriz. Aradığımız ulaşım da en hızlı, en az yürüyerek, en az aktarmayla istenilen noktaya varabilmeyi hedefleyebiliriz. Ancak bizim için en iyi ulaşım çözümünün nasıl olduğunu bulmak gerçekten bilinmesi en zor konulardandır. Özellikler toplu taşıma araçları kullanılırken hangi saatte istenilen durakta geçeceği, hangi durakta aktarma yapılacağı, hangi durakta inilmesi gerektiği ya da istediğimiz noktaya ne zaman varacağımızı bilmek çok zor ve karmaşıktır.

Çalışmamda tüm bu soruların çözümünü ele aldık. Bu tarz problemler graf teorisinin modellemeleriyle çözüme kavuşabilmektedir. Bu konuyla geliştirilen algoritmalar incelenmiştir ve çalışmamıza en uygun olan, en kolay ve hızlı sonuç veren geçiş düğümleri algoritması seçilmiştir. Geçiş düğümleri algoritması ile kodlama yapılmış ve uygulama geliştirilmiştir. Uygulamanın sağladığı faydalar şöyledir: İlk olarak kullanıcı istediği iki noktada kendisine uygun çözümün ne olduğunu görebilmektedir. İstediği ulaşım planında hangi duraklarda ne zaman olacağını ya da hangi duraklarda ne zaman ineceğini ve toplam yolculuk süresinin ne olacağını ve gitmek istediği noktaya ne zaman varabileceğimi bilmesi en büyük faydalarındadır. Uygulamanı toplum bazında sağladığı faydalar ise şöyledir. Kişi toplu taşıma araçlarıyla verimli taşıma yapabildiği için toplu taşıma araçlarında oluşan yoğunluk en az inmektedir. Toplu taşımanın karmaşıklığından dolayı ya da zamandan dolayı toplu taşıma kullamaktan vazgeçen özel araçları olan kişilerin oluşturacağı şehir trafiği ve sarf edilecek olan yakıt ve zamandan da tasarruf edilecektir. Uygulamanın düzgün bir şekilde çalışabilmesi için durak koordinatlarının, hat bilgilerinin, tarife bilgilerinin değiştikçe düzenli olarak güncellemesi gerekmektedir. Uygulama ilerleyen

süreçte çok modlu olarak yani yürüme, bisiklet, elektrikli mobil araçlar gibi senaryolar da eklenerek geliştirilebilir. Böylelikle alternatif ulaşım metodları da kullanılarak daha verimli planlama sonuçları elde edilebilir. Gene ileriki çalışmalarda bakılabilecek bir diğer husus da vasıtaların ulaşma zamanındaki belirsizlikleri de işin içine katarak stokastik bileşenlerin varlığı durumlarındaki optimal çözümlerin ortaya çıkarılması konusudur.

5 Ekler

Hat	Tarife	Durak Adı	Durak Sırası	Hafta içi/sonu	Varış Zamanı
I. Hat	8:00	A'	1	H.İ.	8:00
I. Hat	8:00	B'	2	H.İ.	8:10
I. Hat	8:00	C'	3	H.İ.	8:15
I. Hat	8:00	D	4	H.İ.	8:25
I. Hat	8:00	C	5	H.İ.	8:35
I. Hat	8:00	B	6	H.İ.	8:40
I. Hat	8:00	A	7	H.İ.	8:55
I. Hat	12:00	A'	1	H.İ.	12:00
I. Hat	12:00	B'	2	H.İ.	12:10
I. Hat	12:00	C'	3	H.İ.	12:15
I. Hat	12:00	D	4	H.İ.	12:25
I. Hat	12:00	C	5	H.İ.	12:35
I. Hat	12:00	B	6	H.İ.	12:40
I. Hat	12:00	A	7	H.İ.	12:55
I. Hat	17:00	A'	1	H.İ.	17:00
I. Hat	17:00	B'	2	H.İ.	17:10
I. Hat	17:00	C'	3	H.İ.	17:15
I. Hat	17:00	D	4	H.İ.	17:25
I. Hat	17:00	C	5	H.İ.	17:35
I. Hat	17:00	B	6	H.İ.	17:40
I. Hat	17:00	A	7	H.İ.	17:55
I. Hat	9:00	A'	1	H.S.	9:00
I. Hat	9:00	B'	2	H.S.	9:10
I. Hat	9:00	C'	3	H.S.	9:15
I. Hat	9:00	D	4	H.S.	9:25

I. Hat	9:00	C	5	H.S.	9:35
I. Hat	9:00	B	6	H.S.	9:40
I. Hat	9:00	A	7	H.S.	9:55
I. Hat	13:00	A'	1	H.S.	13:00
I. Hat	13:00	B'	2	H.S.	13:10
I. Hat	13:00	C'	3	H.S.	13:15
I. Hat	13:00	D	4	H.S.	13:25
I. Hat	13:00	C	5	H.S.	13:35
I. Hat	13:00	B	6	H.S.	13:40
I. Hat	13:00	A	7	H.S.	13:55
I. Hat	18:00	A'	1	H.S.	18:00
I. Hat	18:00	B'	2	H.S.	18:10
I. Hat	18:00	C'	3	H.S.	18:15
I. Hat	18:00	D	4	H.S.	18:25
I. Hat	18:00	C	5	H.S.	18:35
I. Hat	18:00	B	6	H.S.	18:40
I. Hat	18:00	A	7	H.S.	18:55
II. Hat	8:30	E	1	H.İ.	8:30
II. Hat	8:30	C'	2	H.İ.	8:40
II. Hat	8:30	C	3	H.İ.	8:45
II. Hat	8:30	I	4	H.İ.	9:00
II. Hat	8:30	J	5	H.İ.	9:10
II. Hat	8:30	E'	6	H.İ.	9:20
II. Hat	12:30	E	1	H.İ.	12:30
II. Hat	12:30	C'	2	H.İ.	12:40
II. Hat	12:30	C	3	H.İ.	12:45
II. Hat	12:30	I	4	H.İ.	13:00
II. Hat	12:30	J	5	H.İ.	13:10
II. Hat	12:30	E'	6	H.İ.	13:20
II. Hat	17:30	E	1	H.İ.	17:30
II. Hat	17:30	C'	2	H.İ.	17:40
II. Hat	17:30	C	3	H.İ.	17:45
II. Hat	17:30	I	4	H.İ.	18:00
II. Hat	17:30	J	5	H.İ.	18:10
II. Hat	17:30	E'	6	H.İ.	18:20
II. Hat	9:30	E	1	H.S.	9:30
II. Hat	9:30	C'	2	H.S.	9:40

II. Hat	9:30	C	3	H.S.	9:45
II. Hat	9:30	I	4	H.S.	10:00
II. Hat	9:30	J	5	H.S.	10:10
II. Hat	9:30	E'	6	H.S.	10:20
II. Hat	13:30	E	1	H.S.	13:30
II. Hat	13:30	C'	2	H.S.	13:40
II. Hat	13:30	C	3	H.S.	13:45
II. Hat	13:30	I	4	H.S.	14:00
II. Hat	13:30	J	5	H.S.	14:10
II. Hat	13:30	E'	6	H.S.	14:20
II. Hat	18:30	E	1	H.S.	18:30
II. Hat	18:30	C'	2	H.S.	18:40
II. Hat	18:30	C	3	H.S.	18:45
II. Hat	18:30	I	4	H.S.	19:00
II. Hat	18:30	J	5	H.S.	19:10
II. Hat	18:30	E'	6	H.S.	19:20
III. Hat	7:00	F	1	H.İ.	7:00
III. Hat	7:00	G	2	H.İ.	7:05
III. Hat	7:00	H	3	H.İ.	7:15
III. Hat	7:00	I	4	H.İ.	7:30
III. Hat	7:00	H'	5	H.İ.	7:35
III. Hat	7:00	F'	6	H.İ.	7:45
III. Hat	18:00	F	1	H.İ.	18:00
III. Hat	18:00	G	2	H.İ.	18:05
III. Hat	18:00	H	3	H.İ.	18:15
III. Hat	18:00	I	4	H.İ.	18:30
III. Hat	18:00	H'	5	H.İ.	18:35
III. Hat	18:00	F'	6	H.İ.	18:45
III. Hat	8:00	F	1	H.S.	8:00
III. Hat	8:00	G	2	H.S.	8:05
III. Hat	8:00	H	3	H.S.	8:15
III. Hat	8:00	I	4	H.S.	8:30
III. Hat	8:00	H'	5	H.S.	8:35
III. Hat	8:00	F'	6	H.S.	8:45
III. Hat	19:00	F	1	H.S.	19:00
III. Hat	19:00	G	2	H.S.	19:05
III. Hat	19:00	H	3	H.S.	19:15
III. Hat	19:00	I	4	H.S.	19:30

III. Hat	19:00	H'	5	H.S.	19:35
III. Hat	19:00	F'	6	H.S.	19:45
IV. Hat	7:30	L	1	H.İ.	7:30
IV. Hat	7:30	G'	2	H.İ.	7:40
IV. Hat	7:30	G	3	H.İ.	7:50
IV. Hat	7:30	U	4	H.İ.	7:55
IV. Hat	7:30	H	5	H.İ.	8:10
IV. Hat	7:30	H'	6	H.İ.	8:25
IV. Hat	7:30	T	7	H.İ.	8:35
IV. Hat	7:30	S	8	H.İ.	8:45
IV. Hat	7:30	R	9	H.İ.	8:55
IV. Hat	7:30	K'	10	H.İ.	9:10
IV. Hat	7:30	L'	11	H.İ.	9:20
IV. Hat	18:30	L	1	H.İ.	18:30
IV. Hat	18:30	G'	2	H.İ.	18:40
IV. Hat	18:30	G	3	H.İ.	18:50
IV. Hat	18:30	U	4	H.İ.	18:55
IV. Hat	18:30	H	5	H.İ.	19:10
IV. Hat	18:30	H'	6	H.İ.	19:25
IV. Hat	18:30	T	7	H.İ.	19:35
IV. Hat	18:30	S	8	H.İ.	19:45
IV. Hat	18:30	R	9	H.İ.	19:55
IV. Hat	18:30	K'	10	H.İ.	20:10
IV. Hat	18:30	L'	11	H.İ.	20:20
IV. Hat	8:30	L	1	H.S.	8:30
IV. Hat	8:30	G'	2	H.S.	8:40
IV. Hat	8:30	G	3	H.S.	8:50
IV. Hat	8:30	U	4	H.S.	8:55
IV. Hat	8:30	H	5	H.S.	9:10
IV. Hat	8:30	H'	6	H.S.	9:25
IV. Hat	8:30	T	7	H.S.	9:35
IV. Hat	8:30	S	8	H.S.	9:45
IV. Hat	8:30	R	9	H.S.	9:55
IV. Hat	8:30	K'	10	H.S.	10:10
IV. Hat	8:30	L'	11	H.S.	10:20
IV. Hat	19:30	L	1	H.S.	19:30
IV. Hat	19:30	G'	2	H.S.	19:40

IV. Hat	19:30	G	3	H.S.	19:50
IV. Hat	19:30	U	4	H.S.	19:55
IV. Hat	19:30	H	5	H.S.	20:10
IV. Hat	19:30	H'	6	H.S.	20:25
IV. Hat	19:30	T	7	H.S.	20:35
IV. Hat	19:30	S	8	H.S.	20:45
IV. Hat	19:30	R	9	H.S.	20:55
IV. Hat	19:30	K'	10	H.S.	21:10
IV. Hat	19:30	L'	11	H.S.	21:20
V. Hat	9:45	Y	1	H.İ.	9:45
V. Hat	9:45	M	2	H.İ.	9:55
V. Hat	9:45	O	3	H.İ.	10:10
V. Hat	9:45	P	4	H.İ.	10:20
V. Hat	9:45	R	5	H.İ.	10:30
V. Hat	9:45	N	6	H.İ.	10:35
V. Hat	9:45	Y'	7	H.İ.	10:45
V. Hat	13:45	Y	1	H.İ.	13:45
V. Hat	13:45	M	2	H.İ.	13:55
V. Hat	13:45	O	3	H.İ.	14:10
V. Hat	13:45	P	4	H.İ.	14:20
V. Hat	13:45	R	5	H.İ.	14:30
V. Hat	13:45	N	6	H.İ.	14:35
V. Hat	13:45	Y'	7	H.İ.	14:45
V. Hat	17:45	Y	1	H.İ.	17:45
V. Hat	17:45	M	2	H.İ.	17:55
V. Hat	17:45	O	3	H.İ.	18:10
V. Hat	17:45	P	4	H.İ.	18:20
V. Hat	17:45	R	5	H.İ.	18:30
V. Hat	17:45	N	6	H.İ.	18:35
V. Hat	17:45	Y'	7	H.İ.	18:45
V. Hat	10:45	Y	1	H.S.	10:45
V. Hat	10:45	M	2	H.S.	10:55
V. Hat	10:45	O	3	H.S.	11:10
V. Hat	10:45	P	4	H.S.	11:20
V. Hat	10:45	R	5	H.S.	11:30
V. Hat	10:45	N	6	H.S.	11:35
V. Hat	10:45	Y'	7	H.S.	11:45

V. Hat	14:45	Y	1	H.S.	14:45
V. Hat	14:45	M	2	H.S.	14:55
V. Hat	14:45	O	3	H.S.	15:10
V. Hat	14:45	P	4	H.S.	15:20
V. Hat	14:45	R	5	H.S.	15:30
V. Hat	14:45	N	6	H.S.	15:35
V. Hat	14:45	Y'	7	H.S.	15:45
V. Hat	18:45	Y	1	H.S.	18:45
V. Hat	18:45	M	2	H.S.	18:55
V. Hat	18:45	O	3	H.S.	19:10
V. Hat	18:45	P	4	H.S.	19:20
V. Hat	18:45	R	5	H.S.	19:30
V. Hat	18:45	N	6	H.S.	19:35
V. Hat	18:45	Y'	7	H.S.	19:45

Tablo 5.1: Varış zamanı tablosu

```
1 fonsiyonlar.komutcalistirmysql(MyQuery_ Hat,'select * from alt_ Hatlar');
2 fonsiyonlar.komutcalistirmysql(MyQuery_ Hat2,'select * from alt_ Hatlar');
```

Kod 5.1: Verilerin çekilmesi

```
1  fonsiyonlar.komutcalistirmysql(MyQuery_ Hat,'select * from alt_ Hatlar');
2  fonsiyonlar.komutcalistirmysql(MyQuery_ Hat2,'select * from alt_ Hatlar');
3
4
5    for x:=1 to MyQuery_ Hat2.RecordCount do
6    begin
7      MyQuery_ Hat.First;
8      for y:=1 to MyQuery_ Hat.RecordCount do
9      begin
10       if MyQuery_ Hat.FieldByName('ANA_ Hat_NO').AsString<>MyQuery_ Hat2.
       ↪ FieldByName('ANA_ Hat_NO').AsString then
11       begin
12
13       sorgu :='insert into  Hat_iliski(ANA_ Hat_NO1,ALT_ Hat_NO1,DURAK_NO1,
       ↪ Hat1_DURAK_SIRA,ANA_ Hat_NO2,ALT_ Hat_NO2,DURAK_NO2,
       ↪ Hat2_DURAK_SIRA,MESAFE) '+
14             //'SELECT  h.ANA_ Hat_NO AS ANA_ Hat_NO1,h.ALT_ Hat_NO AS
```

```
    ↪ ALT_ Hat_NO_1,du.DURAK_NO AS DURAK_NO1,d.ANA_ Hat_NO as
    ↪ ANA_ Hat_NO2, '+
15          //'d.ALT_ Hat_NO AS ALT_ Hat_NO_2,d.DURAK_NO AS DURAK_NO2,
    ↪ ROUND(aractakip.distance(du.ENLEM, du.BOYLAM, d.ENLEM, d.
    ↪ BOYLAM)*1000,0) as MESAFE '+
16          'SELECT h.ANA_ Hat_NO AS ANA_ Hat_NO1,h.ALT_ Hat_NO AS
    ↪ ALT_ Hat_NO_1,du.DURAK_NO AS DURAK_NO1,h.SIRA as
    ↪ Hat1_DURAK_SIRA,d.ANA_ Hat_NO as ANA_ Hat_NO2, '+
17          'd.ALT_ Hat_NO AS ALT_ Hat_NO_2,d.DURAK_NO AS DURAK_NO2,d.
    ↪ SIRA as Hat2_DURAK_SIRA,ROUND(aractakip.distance(du.ENLEM, du.
    ↪ BOYLAM, d.ENLEM, d.BOYLAM)*1000,0) as MESAFE '+
18          'FROM Hat_durak_sira h '+
19          'join duraklar du on h.DURAK_NO = du.DURAK_NO '+
20
21          'cross join '+
22
23          ' ( '+
24          'select Hat_durak_sira.ANA_ Hat_NO, Hat_durak_sira.ALT_ Hat_NO,
    ↪ Hat_durak_sira.DURAK_NO, DURAK_ADI, ISTIKAMET, ENLEM,
    ↪ BOYLAM,SIRA '+
25          'FROM Hat_durak_sira '+
26          'join duraklar du on Hat_durak_sira.DURAK_NO = du.DURAK_NO '+
27          'where ANA_ Hat_NO='''+MyQuery_ Hat.fieldbyname('ANA_ Hat_NO').
    ↪ AsString+''' and ALT_ Hat_NO='''+MyQuery_ Hat.fieldbyname('ALT_
    ↪ Hat_NO').AsString+''' '+
28          ' ) d '+
29
30          ' where h.ANA_ Hat_NO='''+MyQuery_ Hat2.fieldbyname('ANA_ Hat_NO
    ↪ ').AsString+''' and h.ALT_ Hat_NO='''+MyQuery_ Hat2.fieldbyname('ALT_
    ↪  Hat_NO').AsString+''' '+
31          ' and aractakip.distance(du.ENLEM, du.BOYLAM, d.ENLEM, d.BOYLAM)
    ↪ *1000 <600 '+
32          ' order by aractakip.distance(du.ENLEM, du.BOYLAM, d.ENLEM, d.
    ↪ BOYLAM) limit 100 ';
33          komutcalistirmysql(qr_temp,sorgu);
34          Memo1.Lines.Add(MyQuery_ Hat.fieldbyname('ANA_ Hat_NO').AsString+'-'+
    ↪ MyQuery_ Hat.fieldbyname('ALT_ Hat_NO').AsString+'-'+MyQuery_ Hat2.
    ↪ fieldbyname('ANA_ Hat_NO').AsString+'-'+MyQuery_ Hat2.fieldbyname('
    ↪ ALT_ Hat_NO').AsString);
35          end;
36          MyQuery_ Hat.Next;
37        end;
38        MyQuery_ Hat2.Next;
```

Kod 5.2: 2 Hat için Hat ilişki tablosunun oluşturulması

```
1
2  komutcalistirmysql(qr_temp,'select * from ana_ Hatlar order by ANA_ Hat_NO ');
3      komutcalistirmysql(MyQuery_ Hat,'select * from ana_ Hatlar order by ANA_
   ↪ Hat_NO ');
4
5      onceki Hat  := '0';
6      oncekialt Hat:= '0';
7      onceki_durak:= '0';
8
9      while not qr_temp.Eof do
10     begin
11
12         while not MyQuery_ Hat.Eof do
13         begin
14          Self.Refresh;
15          if qr_temp.fieldbyname('ANA_ Hat_NO').AsString<>MyQuery_ Hat.
   ↪ fieldbyname('ANA_ Hat_NO').AsString then
16          begin
17
18            komutcalistirmysql(MyQuery_ Hat2,'select * from  Hat_iliski where ANA_
   ↪  Hat_NO1='''+qr_temp.fieldbyname('ANA_ Hat_NO').AsString+''' and ANA_
   ↪   Hat_NO2='''+MyQuery_ Hat.fieldbyname('ANA_ Hat_NO').AsString+''' ');
19
20            if MyQuery_ Hat2.RecordCount=0 then
21            begin
22            sorgu := 'insert into  Hat_iliski_3 Hat ( CIKIS_ Hat,VARIS_ Hat,
   ↪ AKTARMA_ Hat,SORGU_TIPI)  '+
23                     ' select distinct h.ANA_ Hat_NO1 as CIKIS_ Hat ,hi.ANA_
   ↪ Hat_NO1 as VARIS_ Hat, h.ANA_ Hat_NO2 as AKTARMA_ Hat,1 '+
24                     ' from  Hat_iliski h '+
25                     ' join  Hat_iliski hi on h.ANA_ Hat_NO2 = hi.ANA_ Hat_NO2 '+
26                     ' where h.ANA_ Hat_NO1 ='+qr_temp.fieldbyname('ANA_
   ↪ Hat_NO').AsString+'
27                     ' and hi.ANA_ Hat_NO1 ='+ MyQuery_ Hat.fieldbyname('ANA_
   ↪ Hat_NO').AsString+' ';
28
29            komutcalistirmysql(MyQuery_ Hat2,sorgu);
30            end;
31
```

```
32            end;
33
34             MyQuery_ Hat.Next;
35           end;
36       MyQuery_ Hat.First;
37       qr_temp.Next;
38       end;
```

Kod 5.3: 3 Hat için Hat ilişki tablosunun oluşturulması

```
1  komutcalistirmysql(MyQuery2,'select * from durak_tarife_varis  limit 1');
2      komutcalistirmysql(qr_temp,'select * from tarifeler  order by ANA_ Hat_NO,ALT_
       ↪  Hat_NO');
3
4
5      while not qr_temp.Eof do
6      begin
7      komutcalistirmysql(MyQuery_ Hat,'SELECT * FROM  Hat_durak_sira h where
       ↪ h.ANA_ Hat_NO='+qr_temp.fieldbyname('ANA_ Hat_NO').AsString+' and h
       ↪ .ALT_ Hat_NO='+qr_temp.fieldbyname('ALT_ Hat_NO').AsString+' order by
       ↪  SIRA');
8
9      onceki Hat  := '0';
10     oncekialt Hat:= '0';
11     onceki_durak:= '0';
12     onceki_durak_saat_HI:='00:00:00';
13     onceki_durak_saat_HI_cumartesi:='00:00:00';
14
15       while not MyQuery_ Hat.Eof do
16       begin
17
18           if qr_temp.FieldByName('ZAMAN_DILIMI').AsString='HAFTA.IČ.I' then
19           begin
20
21           saat:=onceki_durak_saat_HI;
22           if Copy(saat,1,1) ='0' then
23           saat:=Copy(saat,2,1)
24           else
25           saat:=Copy(saat,1,2);
26
27           komutcalistirmysql(MyQuery_ Hat2 , 'select SURE from
       ↪ durak_sure_ortalama where BASLANGIC_DURAK_NO ='''+onceki_durak+'''
```

```
↪ and BITIS_DURAK_NO ='''+MyQuery_ Hat.FieldByName('DURAK_NO').
↪ AsString+''' and ZAMAN_ARALIGI ='''+saat+'-HI'+''' ');
28
29          if onceki_durak='0' then
30          begin
31          onceki_durak_saat_HI:=qr_temp.fieldbyname('CIKIS_ZAMANI').AsString;
32
33          MyQuery2.Insert;
34          MyQuery2.FieldByName('ANA_ Hat_NO').AsString  := qr_temp.
↪ fieldbyname('ANA_ Hat_NO').AsString;
35          MyQuery2.FieldByName('ALT_ Hat_NO').AsString  := qr_temp.
↪ fieldbyname('ALT_ Hat_NO').AsString;
36          MyQuery2.FieldByName('TARIFE').AsString       := qr_temp.fieldbyname('
↪ CIKIS_ZAMANI').AsString;
37          MyQuery2.FieldByName('ZAMAN_DILIMI').AsString := 'HI';
38          MyQuery2.FieldByName('DURAK_NO').AsString     := MyQuery_ Hat.
↪ FieldByName('DURAK_NO').AsString;
39          MyQuery2.FieldByName('DURAK_SIRA').AsString   := MyQuery_ Hat.
↪ FieldByName('SIRA').AsString;
40          MyQuery2.FieldByName('SAAT').AsString   := qr_temp.fieldbyname('
↪ CIKIS_ZAMANI').AsString;
41          MyQuery2.FieldByName('HESAPLAMA_YONTEMI').AsString   := '0';
42          MyQuery2.Post;
43          MyQuery2.Insert;
44          MyQuery2.FieldByName('ANA_ Hat_NO').AsString  := qr_temp.
↪ fieldbyname('ANA_ Hat_NO').AsString;
45          MyQuery2.FieldByName('ALT_ Hat_NO').AsString  := qr_temp.
↪ fieldbyname('ALT_ Hat_NO').AsString;
46          MyQuery2.FieldByName('TARIFE').AsString       := qr_temp.fieldbyname('
↪ CIKIS_ZAMANI').AsString;
47          MyQuery2.FieldByName('ZAMAN_DILIMI').AsString := 'C';
48          MyQuery2.FieldByName('DURAK_NO').AsString     := MyQuery_ Hat.
↪ FieldByName('DURAK_NO').AsString;
49          MyQuery2.FieldByName('DURAK_SIRA').AsString   := MyQuery_ Hat.
↪ FieldByName('SIRA').AsString;
50          MyQuery2.FieldByName('SAAT').AsString   := qr_temp.fieldbyname('
↪ CIKIS_ZAMANI').AsString;
51          MyQuery2.FieldByName('HESAPLAMA_YONTEMI').AsString   := '0';
52          MyQuery2.Post;
53
54
55          end;
56
```

```
57
58          if MyQuery_ Hat2.RecordCount<>0 then
59          begin
60
61            if MyQuery_ Hat.RecNo=1 then
62            begin
63            saatHesap := StrToTime(qr_temp.fieldbyname('CIKIS_ZAMANI').
    ↪ AsString);
64            saatHesap := IncSecond(saatHesap,MyQuery_ Hat2.fieldbyname('SURE').
    ↪ asinteger);
65            onceki_durak_saat_HI:=TimeToStr(saathesap);
66            end
67            else
68            begin
69            saatHesap:=IncSecond(StrToTime(onceki_durak_saat_HI),MyQuery_ Hat2
    ↪ .fieldbyname('SURE').asinteger);
70            end;
71
72          MyQuery2.Insert;
73          MyQuery2.FieldByName('ANA_ Hat_NO').AsString   := qr_temp.
    ↪ fieldbyname('ANA_ Hat_NO').AsString;
74          MyQuery2.FieldByName('ALT_ Hat_NO').AsString   := qr_temp.
    ↪ fieldbyname('ALT_ Hat_NO').AsString;
75          MyQuery2.FieldByName('TARIFE').AsString       := qr_temp.fieldbyname('
    ↪ CIKIS_ZAMANI').AsString;
76          MyQuery2.FieldByName('ZAMAN_DILIMI').AsString := 'HI';
77          MyQuery2.FieldByName('DURAK_NO').AsString     := MyQuery_ Hat.
    ↪ FieldByName('DURAK_NO').AsString;
78          MyQuery2.FieldByName('DURAK_SIRA').AsString   := MyQuery_ Hat.
    ↪ FieldByName('SIRA').AsString;
79          MyQuery2.FieldByName('SAAT').AsString   := TimeToStr(saatHesap);
80          MyQuery2.FieldByName('HESAPLAMA_YONTEMI').AsString   := '0';
81          MyQuery2.Post;
82          onceki_durak_saat_HI:=MyQuery2.FieldByName('SAAT').AsString;
83          end
84          else
85          begin
86          MyQuery2.Insert;
87          MyQuery2.FieldByName('ANA_ Hat_NO').AsString   := qr_temp.
    ↪ fieldbyname('ANA_ Hat_NO').AsString;
88          MyQuery2.FieldByName('ALT_ Hat_NO').AsString   := qr_temp.
    ↪ fieldbyname('ALT_ Hat_NO').AsString;
89          MyQuery2.FieldByName('TARIFE').AsString       := qr_temp.fieldbyname('
```

```
    ↪ CIKIS_ZAMANI').AsString;
90          MyQuery2.FieldByName('ZAMAN_DILIMI').AsString := 'HI';
91          MyQuery2.FieldByName('DURAK_NO').AsString     := MyQuery_ Hat.
    ↪ FieldByName('DURAK_NO').AsString;
92          MyQuery2.FieldByName('DURAK_SIRA').AsString   := MyQuery_ Hat.
    ↪ FieldByName('SIRA').AsString;
93          MyQuery2.FieldByName('SAAT').AsString   := TimeToStr(incsecond(
    ↪ strtotime(onceki_durak_saat_HI),50));
94          MyQuery2.FieldByName('HESAPLAMA_YONTEMI').AsString   :='-1';
95          MyQuery2.Post;
96          onceki_durak_saat_HI_cumartesi:=MyQuery2.FieldByName('SAAT').
    ↪ AsString;
97          end;
98
99           komutcalistirmysql(MyQuery_ Hat2 , 'select SURE from
    ↪ durak_sure_ortalama where BASLANGIC_DURAK_NO ='''+onceki_durak+'''
    ↪ and BITIS_DURAK_NO ='''+MyQuery_ Hat.FieldByName('DURAK_NO').
    ↪ AsString+''' and ZAMAN_ARALIGI ='''+saat+'-C'+''' ');
100
101         if onceki_durak='0' then
102         begin
103         onceki_durak_saat_HI_cumartesi:=qr_temp.fieldbyname('CIKIS_ZAMANI').
    ↪ AsString;
104         end;
105
106
107         if MyQuery_ Hat2.RecordCount<>0 then
108         begin
109
110           if MyQuery_ Hat.RecNo=1 then
111           begin
112           saatHesap := StrToTime(qr_temp.fieldbyname('CIKIS_ZAMANI').
    ↪ AsString);
113           saatHesap := IncSecond(saatHesap,MyQuery_ Hat2.fieldbyname('SURE').
    ↪ asinteger);
114           onceki_durak_saat_HI_cumartesi:=TimeToStr(saathesap);
115           end
116           else
117           begin
118           saatHesap:=IncSecond(StrToTime(onceki_durak_saat_HI_cumartesi),
    ↪ MyQuery_ Hat2.fieldbyname('SURE').asinteger);
119           end;
120
```

```
121             MyQuery2.Insert;
122             MyQuery2.FieldByName('ANA_ Hat_NO').AsString   := qr_temp.
    ↪ fieldbyname('ANA_ Hat_NO').AsString;
123             MyQuery2.FieldByName('ALT_ Hat_NO').AsString   := qr_temp.
    ↪ fieldbyname('ALT_ Hat_NO').AsString;
124             MyQuery2.FieldByName('TARIFE').AsString        := qr_temp.fieldbyname('
    ↪ CIKIS_ZAMANI').AsString;
125             MyQuery2.FieldByName('ZAMAN_DILIMI').AsString := 'C';
126             MyQuery2.FieldByName('DURAK_NO').AsString      := MyQuery_ Hat.
    ↪ FieldByName('DURAK_NO').AsString;
127             MyQuery2.FieldByName('DURAK_SIRA').AsString   := MyQuery_ Hat.
    ↪ FieldByName('SIRA').AsString;
128             MyQuery2.FieldByName('SAAT').AsString   := TimeToStr(saatHesap);
129             MyQuery2.Post;
130             onceki_durak_saat_HI_cumartesi:=MyQuery2.FieldByName('SAAT').
    ↪ AsString;
131
132             end
133             else
134             begin
135             MyQuery2.Insert;
136             MyQuery2.FieldByName('ANA_ Hat_NO').AsString   := qr_temp.
    ↪ fieldbyname('ANA_ Hat_NO').AsString;
137             MyQuery2.FieldByName('ALT_ Hat_NO').AsString   := qr_temp.
    ↪ fieldbyname('ALT_ Hat_NO').AsString;
138             MyQuery2.FieldByName('TARIFE').AsString        := qr_temp.fieldbyname('
    ↪ CIKIS_ZAMANI').AsString;
139             MyQuery2.FieldByName('ZAMAN_DILIMI').AsString := 'C';
140             MyQuery2.FieldByName('DURAK_NO').AsString      := MyQuery_ Hat.
    ↪ FieldByName('DURAK_NO').AsString;
141             MyQuery2.FieldByName('DURAK_SIRA').AsString   := MyQuery_ Hat.
    ↪ FieldByName('SIRA').AsString;
142             MyQuery2.FieldByName('SAAT').AsString   := TimeToStr(incsecond(
    ↪ strtotime(onceki_durak_saat_HI_cumartesi),50));
143             MyQuery2.FieldByName('HESAPLAMA_YONTEMI').AsString   :='-1';
144
145             MyQuery2.Post;
146             onceki_durak_saat_HI_cumartesi:=MyQuery2.FieldByName('SAAT').
    ↪ AsString;
147
148             end;
149
150
```

```
151         onceki_durak := MyQuery_ Hat.FieldByName('DURAK_NO').AsString ;
152         onceki Hat    := MyQuery_ Hat.FieldByName('ANA_ Hat_NO').AsString;
153         oncekialt Hat := MyQuery_ Hat.FieldByName('ALT_ Hat_NO').AsString;
154         end;
155
156
157
158       if qr_temp.FieldByName('ZAMAN_DILIMI').AsString='PAZAR' then
159       begin
160          saat:=onceki_durak_saat_HI;
161          if Copy(saat,1,1) ='0' then
162          saat:=Copy(saat,2,1)
163          else
164          saat:=Copy(saat,1,2);
165
166          komutcalistirmysql(MyQuery_ Hat2 , 'select SURE from
    ↪ durak_sure_ortalama where BASLANGIC_DURAK_NO ='''+onceki_durak+'''
    ↪ and BITIS_DURAK_NO ='''+MyQuery_ Hat.FieldByName('DURAK_NO').
    ↪ AsString+''' and ZAMAN_ARALIGI ='''+saat+'-P'+''' ');
167
168           if onceki_durak='0' then
169           begin
170           onceki_durak_saat_HI:=qr_temp.fieldbyname('CIKIS_ZAMANI').AsString;
171
172           MyQuery2.Insert;
173           MyQuery2.FieldByName('ANA_ Hat_NO').AsString   := qr_temp.
    ↪ fieldbyname('ANA_ Hat_NO').AsString;
174           MyQuery2.FieldByName('ALT_ Hat_NO').AsString   := qr_temp.
    ↪ fieldbyname('ALT_ Hat_NO').AsString;
175           MyQuery2.FieldByName('TARIFE').AsString        := qr_temp.fieldbyname('
    ↪ CIKIS_ZAMANI').AsString;
176           MyQuery2.FieldByName('ZAMAN_DILIMI').AsString := 'P';
177           MyQuery2.FieldByName('DURAK_NO').AsString      := MyQuery_ Hat.
    ↪ FieldByName('DURAK_NO').AsString;
178           MyQuery2.FieldByName('DURAK_SIRA').AsString   := MyQuery_ Hat.
    ↪ FieldByName('SIRA').AsString;
179           MyQuery2.FieldByName('SAAT').AsString   := qr_temp.fieldbyname('
    ↪ CIKIS_ZAMANI').AsString;
180           MyQuery2.FieldByName('HESAPLAMA_YONTEMI').AsString   := '0';
181
182           MyQuery2.Post;
183
184           end;
```

```
185
186
187             if MyQuery_ Hat2.RecordCount<>0 then
188             begin
189
190               if MyQuery_ Hat.RecNo=1 then
191               begin
192               saatHesap := StrToTime(qr_temp.fieldbyname('CIKIS_ZAMANI').
    ↪ AsString);
193               saatHesap := IncSecond(saatHesap,MyQuery_ Hat2.fieldbyname('SURE').
    ↪ asinteger);
194               onceki_durak_saat_HI:=TimeToStr(saathesap);
195               end
196               else
197               begin
198               saatHesap:=IncSecond(StrToTime(onceki_durak_saat_HI),MyQuery_ Hat2
    ↪ .fieldbyname('SURE').asinteger);
199               end;
200
201             MyQuery2.Insert;
202             MyQuery2.FieldByName('ANA_ Hat_NO').AsString   := qr_temp.
    ↪ fieldbyname('ANA_ Hat_NO').AsString;
203             MyQuery2.FieldByName('ALT_ Hat_NO').AsString   := qr_temp.
    ↪ fieldbyname('ALT_ Hat_NO').AsString;
204               MyQuery2.FieldByName('TARIFE').AsString       := qr_temp.fieldbyname('
    ↪ CIKIS_ZAMANI').AsString;
205             MyQuery2.FieldByName('ZAMAN_DILIMI').AsString := 'P';
206             MyQuery2.FieldByName('DURAK_NO').AsString     := MyQuery_ Hat.
    ↪ FieldByName('DURAK_NO').AsString;
207             MyQuery2.FieldByName('DURAK_SIRA').AsString   := MyQuery_ Hat.
    ↪ FieldByName('SIRA').AsString;
208             MyQuery2.FieldByName('SAAT').AsString   := TimeToStr(saatHesap);
209             MyQuery2.FieldByName('HESAPLAMA_YONTEMI').AsString   := '0';
210             MyQuery2.Post;
211             onceki_durak_saat_HI:=MyQuery2.FieldByName('SAAT').AsString;
212             end
213             else
214             begin
215             MyQuery2.Insert;
216             MyQuery2.FieldByName('ANA_ Hat_NO').AsString   := qr_temp.
    ↪ fieldbyname('ANA_ Hat_NO').AsString;
217             MyQuery2.FieldByName('ALT_ Hat_NO').AsString   := qr_temp.
    ↪ fieldbyname('ALT_ Hat_NO').AsString;
```

```
218             MyQuery2.FieldByName('TARIFE').AsString       := qr_temp.fieldbyname('
    ↪ CIKIS_ZAMANI').AsString;
219             MyQuery2.FieldByName('ZAMAN_DILIMI').AsString := 'P';
220             MyQuery2.FieldByName('DURAK_NO').AsString     := MyQuery_ Hat.
    ↪ FieldByName('DURAK_NO').AsString;
221             MyQuery2.FieldByName('DURAK_SIRA').AsString   := MyQuery_ Hat.
    ↪ FieldByName('SIRA').AsString;
222             MyQuery2.FieldByName('SAAT').AsString   := TimeToStr(incsecond(
    ↪ strtotime(onceki_durak_saat_HI),50));
223             MyQuery2.FieldByName('HESAPLAMA_YONTEMI').AsString   :='-1';
224             MyQuery2.Post;
225             onceki_durak_saat_HI_cumartesi:=MyQuery2.FieldByName('SAAT').
    ↪ AsString;
226             end;
227
228          onceki_durak := MyQuery_ Hat.FieldByName('DURAK_NO').AsString ;
229          onceki Hat    := MyQuery_ Hat.FieldByName('ANA_ Hat_NO').AsString;
230          oncekialt Hat := MyQuery_ Hat.FieldByName('ALT_ Hat_NO').AsString;
231
232
233        end;
234              MyQuery_ Hat.Next;
235
236
237      end;
238        qr_temp.Next;
239   end;
```

Kod 5.4: Varış zamanı tablosunun oluşturulması

```
1  NASIL G.IDER.IM
2
3
4  public DataSet NasilGiderim6(String cikisKoordinat, String varisKoordinat, String
       ↪ yolculukBaslamaSaati)
5          {
6              string sessionid = UnixDatetimeAl().ToString() + "_" + cikisKoordinat + "_
       ↪ " + varisKoordinat;
7
8              string runningTime;
9              Stopwatch sw = new Stopwatch();
10             sw.Start();
```

```
11
12
13          string Yolculuk_ Hat1, Yolculuk_ Hat1_Ad, Yolculuk_Alt Hat1,
    ↪ Yolculuk_Alt Hat1_Ad, Yolculuk_ Hat1_baslama_durak, Yolculuk_
    ↪ Hat1_baslama_durak_adi, Yolculuk_ Hat1_baslama_durak_yon, Yolculuk_
    ↪ Hat1_bitis_durak, Yolculuk_ Hat1_bitis_durak_adi, Yolculuk_
    ↪ Hat1_baslama_durak_sira, Yolculuk_ Hat1_bitis_durak_sira, Yolculuk_
    ↪ Hat1_baslama_saat, Yolculuk_ Hat1_bitis_saat;
14          string Yolculuk_ Hat2, Yolculuk_ Hat2_Ad, Yolculuk_Alt Hat2,
    ↪ Yolculuk_Alt Hat2_Ad, Yolculuk_ Hat2_baslama_durak, Yolculuk_
    ↪ Hat2_baslama_durak_adi, Yolculuk_ Hat2_baslama_durak_yon, Yolculuk_
    ↪ Hat2_bitis_durak, Yolculuk_ Hat2_bitis_durak_adi, Yolculuk_
    ↪ Hat2_baslama_durak_sira, Yolculuk_ Hat2_bitis_durak_sira, Yolculuk_
    ↪ Hat2_baslama_saat, Yolculuk_ Hat2_bitis_saat;
15          string Yolculuk_ Hat3, Yolculuk_ Hat3_Ad, Yolculuk_Alt Hat3,
    ↪ Yolculuk_Alt Hat3_Ad, Yolculuk_ Hat3_baslama_durak, Yolculuk_
    ↪ Hat3_baslama_durak_adi, Yolculuk_ Hat3_baslama_durak_yon, Yolculuk_
    ↪ Hat3_bitis_durak, Yolculuk_ Hat3_bitis_durak_adi, Yolculuk_
    ↪ Hat3_baslama_durak_sira, Yolculuk_ Hat3_bitis_durak_sira, Yolculuk_
    ↪ Hat3_baslama_saat, Yolculuk_ Hat3_bitis_saat;
16          string Yolculuk_ Hat4, Yolculuk_ Hat4_Ad, Yolculuk_Alt Hat4,
    ↪ Yolculuk_Alt Hat4_Ad, Yolculuk_ Hat4_baslama_durak, Yolculuk_
    ↪ Hat4_baslama_durak_adi, Yolculuk_ Hat4_baslama_durak_yon, Yolculuk_
    ↪ Hat4_bitis_durak, Yolculuk_ Hat4_bitis_durak_adi, Yolculuk_
    ↪ Hat4_baslama_durak_sira, Yolculuk_ Hat4_bitis_durak_sira, Yolculuk_
    ↪ Hat4_baslama_saat, Yolculuk_ Hat4_bitis_saat;
17
18          string Yurume1_cikis_enlem_boylam, Yurume2_cikis_enlem_boylam,
    ↪ Yurume3_cikis_enlem_boylam, Yurume4_cikis_enlem_boylam,
    ↪ Yurume5_cikis_enlem_boylam;
19          string Yurume1_varis_enlem_boylam, Yurume2_varis_enlem_boylam,
    ↪ Yurume3_varis_enlem_boylam, Yurume4_varis_enlem_boylam,
    ↪ Yurume5_varis_enlem_boylam;
20
21          string cikisEnlem, cikisBoylam, varisEnlem, varisBoylam;
22          string hesaplamaGun;
23          string sorgu;
24          fnks_getSistemZamani();
25          string zamanDilimi = sistemZamani.Hour.ToString() + "-";
26          if (sistemZamani.DayOfWeek == DayOfWeek.Saturday)
27          {
28              hesaplamaGun = "C";
29          }
```

```
30          else if (sistemZamani.DayOfWeek == DayOfWeek.Sunday)
31          {
32              hesaplamaGun = "P";
33          }
34          else
35          {
36              hesaplamaGun = "HI";
37          }
38
39          string alt  Hatbul;
40          if (cikisKoordinat == "")
41          {
42              cikisEnlem = "37.7508417";//4 aktarma
43              cikisBoylam = "32.4853000";//4 aktarma
44
45              varisEnlem = "37.9815417"; // 4 l"U aktarma
46              varisBoylam = "32.5077250"; // 4 l"U aktarma
47          }
48          else
49          {
50              string[] t = cikisKoordinat.Split(',');
51              cikisEnlem = t[0];
52              cikisBoylam = t[1].Trim();
53              t = varisKoordinat.Split(',');
54              varisEnlem = t[0].Trim();
55              varisBoylam = t[1].Trim();
56          }
57
58
59          DateTime zamantamami, zamantekadim, t1;
60          zamantamami = DateTime.Now;
61          zamantekadim = DateTime.Now;
62          TimeSpan gecenZaman, gt;
63
64          string cikisAna  Hat, varisAna  Hat;
65          string cikisAlt  Hat, varisAlt  Hat, cikisIkinciDurak, varisBirinciDurak;
66
67          zamantamami = DateTime.Now;
68
69          int CikisDurakSirasi;
70          String CikisIstikamet;
71          int VarisDurakSirasi;
72          String VarisIstikamet;
```

```
73          int bul = 0;
74          int cozum = 0;
75          DataSet   Hat_iliski = new DataSet();
76          DataSet temp = new DataSet();
77          DataSet temp2 = new DataSet();
78          DataSet aktarma  Hatti = new DataSet();
79          DataSet Dorduncu_  Hat = new DataSet();
80
81          int say = 0;
82          int hesaplasay = 0;
83
84          int gelisSuresiSn;
85
86          String Sonuc = "1";
87          int SonucInt = -100;
88
89          DataSet donecekDs = new DataSet();
90          donecekDs.Tables.Add("Table");
91          donecekDs.Tables[0].Columns.Add("SESSION", typeof(String));
92          donecekDs.Tables[0].Columns.Add("COZUM", typeof(String));
93          donecekDs.Tables[0].Columns.Add("ADIM", typeof(String));
94          donecekDs.Tables[0].Columns.Add("ARAC_TIPI", typeof(String));
95          donecekDs.Tables[0].Columns.Add("BASLAMA_DURAK", typeof(String));
96          donecekDs.Tables[0].Columns.Add("BASLAMA_DURAK_AD", typeof(String
    ↪ ));
97          donecekDs.Tables[0].Columns.Add("BASLAMA_DURAK_YON", typeof(
    ↪ String));
98          donecekDs.Tables[0].Columns.Add("BASLAMA_DURAK_SIRASI", typeof(
    ↪ String));
99          donecekDs.Tables[0].Columns.Add("BITIS_DURAK", typeof(String));
100         donecekDs.Tables[0].Columns.Add("BITIS_DURAK_AD", typeof(String));
101         donecekDs.Tables[0].Columns.Add("BITIS_DURAK_SIRASI", typeof(String))
    ↪ ;
102         donecekDs.Tables[0].Columns.Add("   Hat_NO", typeof(String));
103         donecekDs.Tables[0].Columns.Add("   Hat_ADI", typeof(String));
104         donecekDs.Tables[0].Columns.Add("ALT_  Hat_NO", typeof(String));
105         donecekDs.Tables[0].Columns.Add("BASLAMA_SAATI", typeof(String));
106         donecekDs.Tables[0].Columns.Add("BITIS_SAATI", typeof(String));
107         donecekDs.Tables[0].Columns.Add("MESAFE", typeof(String));
108
109
110
111         DataSet aktarmaDs = new DataSet();
```

```
112         aktarmaDs.Tables.Add("Table");
113         aktarmaDs.Tables[0].Columns.Add("ANA_ Hat1", typeof(String));
114         aktarmaDs.Tables[0].Columns.Add("ANA_ Hat2", typeof(String));
115
116         DataSet hesaplaDs = new DataSet();
117         hesaplaDs.Tables.Add("Table");
118         hesaplaDs.Tables[0].Columns.Add("AKTARIM", typeof(String));
119         hesaplaDs.Tables[0].Columns.Add("BASLAMA_NOKTASI", typeof(String));
120         hesaplaDs.Tables[0].Columns.Add("BITIS_NOKTASI", typeof(String));
121         hesaplaDs.Tables[0].Columns.Add(" Hat_NO", typeof(int));
122         hesaplaDs.Tables[0].Columns.Add("ALT_ Hat_NO", typeof(String));
123         hesaplaDs.Tables[0].Columns.Add("SURE", typeof(String));
124         hesaplaDs.Tables[0].Columns.Add("BASLAMA_SAATI", typeof(String));
125         hesaplaDs.Tables[0].Columns.Add("BITIS_SAATI", typeof(String));
126
127
128         DataSet cikis Hatlar = new DataSet();
129         cikis Hatlar.Tables.Add("Table");
130         cikis Hatlar.Tables[0].Columns.Add("ANA_ Hat_NO", typeof(String));
131         cikis Hatlar.Tables[0].Columns.Add("ALT_ Hat_NO", typeof(String));
132         cikis Hatlar.Tables[0].Columns.Add("DURAK_NO", typeof(String));
133         cikis Hatlar.Tables[0].Columns.Add("DURAK_SIRASI", typeof(String));
134         cikis Hatlar.Tables[0].Columns.Add("SAAT", typeof(String));
135
136         DataSet varis Hatlar = new DataSet();
137         varis Hatlar.Tables.Add("Table");
138         varis Hatlar.Tables[0].Columns.Add("ANA_ Hat_NO", typeof(String));
139         varis Hatlar.Tables[0].Columns.Add("ALT_ Hat_NO", typeof(String));
140         varis Hatlar.Tables[0].Columns.Add("DURAK_NO", typeof(String));
141         varis Hatlar.Tables[0].Columns.Add("DURAK_SIRASI", typeof(String));
142
143         t1 = DateTime.Now;
144
145         DataSet cikisDuraklar = getKoordinataEnYakinDuraklar(cikisEnlem,
    ↪ cikisBoylam, EnlemBoylaminKacMetreYakinindaDurakAranacak, 0);
146         DataSet varisDuraklar = getKoordinataEnYakinDuraklar(varisEnlem,
    ↪ varisBoylam, EnlemBoylaminKacMetreYakinindaDurakAranacak, 0);
147         String cikisDuraklarStr = "(";
148
149         if (cikisDuraklar.Tables[0].Rows.Count == 0)
150         {
151             cikisDuraklar = getKoordinataEnYakinDuraklar(cikisEnlem, cikisBoylam,
    ↪ 800, 0);
```

```
152
153             if (cikisDuraklar.Tables[0].Rows.Count == 0)
154                 Sonuc = "Čikiş durağı bulunamadi";
155         }
156
157         for (int i = 0; i < cikisDuraklar.Tables[0].Rows.Count; i++)
158         {
159             cikisDuraklarStr += "" + cikisDuraklar.Tables[0].Rows[i]["DURAK_NO"]
    ↪  + "A"";
160             cikisDuraklarStr += ",";
161             cikisDuraklarStr += "" + cikisDuraklar.Tables[0].Rows[i]["DURAK_NO"]
    ↪  + "D"";
162             if (i < cikisDuraklar.Tables[0].Rows.Count - 1) cikisDuraklarStr += ",";
163         }
164         cikisDuraklarStr += ")";
165
166         if (varisDuraklar.Tables[0].Rows.Count == 0)
167         {
168             varisDuraklar = getKoordinataEnYakinDuraklar(varisEnlem, varisBoylam,
    ↪  800, 0);
169             if (varisDuraklar.Tables[0].Rows.Count == 0)
170                 Sonuc += "Varis durağı bulunamadi";
171         }
172
173         String varisDuraklarStr = "(";
174         for (int i = 0; i < varisDuraklar.Tables[0].Rows.Count; i++)
175         {
176             varisDuraklarStr += "" + varisDuraklar.Tables[0].Rows[i]["DURAK_NO
    ↪ "] + "A"";
177             varisDuraklarStr += ",";
178             varisDuraklarStr += "" + varisDuraklar.Tables[0].Rows[i]["DURAK_NO
    ↪ "] + "D"";
179             if (i < varisDuraklar.Tables[0].Rows.Count - 1) varisDuraklarStr += ",";
180         }
181         varisDuraklarStr += ")";
182
183         int COZUM_SAYISI = 0;
184         int ADIM_SAYISI = 0;
185         string ARAC_TURU = "";
186
187         if (Sonuc == "1")
188         {
189             string tekBoyutluSorgu = "";
```

```
190             tekBoyutluSorgu += "SELECT DISTINCT ";
191             tekBoyutluSorgu += "S1.ANA_ Hat_NO, ";
192             tekBoyutluSorgu += "S1.ALT_ Hat_NO ";// +",";
193             tekBoyutluSorgu += "from  Hat_durak_sira S1,  Hat_durak_sira S2,ANA_
    ↪  HatLAR ANA,ALT_ HatLAR ALT,DURAKLAR D1,DURAKLAR D2 ";
194             tekBoyutluSorgu += "where S1.DURAK_NO in " + cikisDuraklarStr + "
    ↪ ";
195             tekBoyutluSorgu += "and S2.DURAK_NO in " + varisDuraklarStr + " ";
196             tekBoyutluSorgu += "and S1.ANA_ Hat_NO = S2.ANA_ Hat_NO ";
197             tekBoyutluSorgu += "AND S1.ALT_ Hat_NO = S2.ALT_ Hat_NO ";
198             tekBoyutluSorgu += "AND S1.SIRA<S2.SIRA ";
199             tekBoyutluSorgu += "AND S1.ANA_ Hat_NO = ANA.ANA_ Hat_NO ";
200             tekBoyutluSorgu += "AND S1.ALT_ Hat_NO = ALT.ALT_ Hat_NO ";
201             tekBoyutluSorgu += "AND ANA.ANA_ Hat_NO = ALT.ANA_ Hat_NO
    ↪ ";
202             tekBoyutluSorgu += "AND S1.DURAK_NO=D1.DURAK_NO ";
203             tekBoyutluSorgu += "AND S2.DURAK_NO=D2.DURAK_NO ";
204             DataSet ds = new DataSet();
205             myDataAdapter.SelectCommand = new MySqlCommand(
    ↪ tekBoyutluSorgu, myConn);
206             myDataAdapter.Fill(ds);
207             connectionKapat();
208
209             string alt Hatlar;
210             alt Hatlar = "";
211
212             if (ds.Tables[0].Rows.Count != 0)
213             {
214                 string ana Hat = ds.Tables[0].Rows[0]["ANA_ Hat_NO"].ToString();
215                 alt Hatlar = ds.Tables[0].Rows[0]["ALT_ Hat_NO"].ToString() + ",";
216
217                 for (int i = 0; i < ds.Tables[0].Rows.Count; i++)
218                 {
219                     zamantekadim = DateTime.Now;
220
221                     if (ds.Tables[0].Rows[i]["ANA_ Hat_NO"].ToString() != ana Hat || i
    ↪  == ds.Tables[0].Rows.Count - 1)
222                     {
223                         alt Hatlar = alt Hatlar.Substring(0, alt Hatlar.Length - 1);
224
225                         sorgu = "SELECT s1.DURAK_NO,s1.SIRA ";
226                         sorgu += " from  Hat_durak_sira S1,DURAKLAR D1 where S1
    ↪ .DURAK_NO=D1.DURAK_NO ";
```

```
227                     sorgu += " and S1.ANA_ Hat_NO=" + ana Hat + " and S1.
    ↪ ALT_ Hat_NO in (" + alt Hatlar + ")";
228                     sorgu += " order by distanceyeni('" + cikisEnlem + "','" +
    ↪ cikisBoylam + "' ,d1.ENLEM ,d1.BOYLAM) limit 1";
229                     temp.Clear();
230                     myDataAdapter.SelectCommand = new MySqlCommand(sorgu,
    ↪ myConn);
231                     myDataAdapter.Fill(temp);
232                     connectionKapat();
233
234                     string CIKIS_DURAK_NO = temp.Tables[0].Rows[0]["
    ↪ DURAK_NO"].ToString();
235                     string CIKIS_DURAK_SIRA = temp.Tables[0].Rows[0]["SIRA"].
    ↪ ToString();
236
237                     sorgu = "SELECT s1.DURAK_NO,s1.SIRA ";
238                     sorgu += " from Hat_durak_sira S1,DURAKLAR D1 where S1
    ↪ .DURAK_NO=D1.DURAK_NO ";
239                     sorgu += " and S1.ANA_ Hat_NO=" + ana Hat + " and S1.
    ↪ ALT_ Hat_NO in (" + alt Hatlar + ")";
240                     sorgu += " order by distanceyeni('" + varisEnlem + "','" +
    ↪ varisBoylam + "' ,d1.ENLEM ,d1.BOYLAM) limit 1";
241                     temp2.Clear();
242                     myDataAdapter.SelectCommand = new MySqlCommand(sorgu,
    ↪ myConn);
243                     myDataAdapter.Fill(temp2);
244                     connectionKapat();
245
246                     string VARIS_DURAK_NO = temp2.Tables[0].Rows[0]["
    ↪ DURAK_NO"].ToString();
247                     string VARIS_DURAK_SIRA = temp2.Tables[0].Rows[0]["SIRA
    ↪ "].ToString();
248
249                     int donen;
250                     donen = DuraklariSec2(CIKIS_DURAK_NO,
    ↪ VARIS_DURAK_NO, ana Hat, alt Hatlar, out CIKIS_DURAK_NO, out
    ↪ CIKIS_DURAK_SIRA, out Yolculuk_ Hat1_baslama_durak_yon, out
    ↪ VARIS_DURAK_NO, out VARIS_DURAK_SIRA);
251
252                     if (donen == 1)
253                     {
254                         gelisSuresiSn =
    ↪ fnks_AracinDuragaGelmeSaatiVeHedefDuragaGitmeSaati(ana Hat, alt Hatlar
```

```
↪ , CIKIS_DURAK_NO, VARIS_DURAK_NO, yolculukBaslamaSaati,
↪ hesaplamaGun, out Yolculuk_ Hat1_baslama_saat, out Yolculuk_
↪ Hat1_bitis_saat, out alt Hatbul);
255
256                     if (gelisSuresiSn != −1)
257                     {
258                         cozum++;
259                         gecenZaman = DateTime.Now − zamantekadim;
260                         COZUM_SAYISI++;
261
262                         if (ana Hat == "10") { ARAC_TURU = "T"; } else {
↪ ARAC_TURU = "O"; }
263
264                         Yolculuk_ Hat1 = ana Hat;
265                         Yolculuk_Alt Hat1 = alt Hatbul;
266                         Yolculuk_ Hat1_baslama_durak = CIKIS_DURAK_NO;
267                         Yolculuk_ Hat1_bitis_durak = VARIS_DURAK_NO;
268                         Yurume1_cikis_enlem_boylam = cikisEnlem + ";" +
↪ cikisBoylam;
269                         Yurume1_varis_enlem_boylam = durakenlemboylam(
↪ Yolculuk_ Hat1_baslama_durak, out Yolculuk_ Hat1_baslama_durak_adi);
270                         Yurume2_cikis_enlem_boylam = durakenlemboylam(
↪ Yolculuk_ Hat1_bitis_durak, out Yolculuk_ Hat1_bitis_durak_adi);
271                         Yurume2_varis_enlem_boylam = varisEnlem + ";" +
↪ varisBoylam;
272
273                         donecekDs.Tables[0].Rows.Add(
274                             sessionid,
275                             COZUM_SAYISI.ToString(),
276                             "3",
277                             "Y|" + ARAC_TURU + "|Y",
278                             Yurume1_cikis_enlem_boylam + "|" + Yolculuk_
↪ Hat1_baslama_durak + "|" + Yurume2_cikis_enlem_boylam,//
↪ durakenlemboylam(Yolculuk_ Hat1_bitis_durak),
279                             "0|" + Yolculuk_ Hat1_baslama_durak_adi + "|0",
280                             "0|" + Yolculuk_ Hat1_baslama_durak_yon + "|0",
281                             "0|" + CIKIS_DURAK_SIRA + "|0",
282                             Yurume1_varis_enlem_boylam + "|" + Yolculuk_
↪ Hat1_bitis_durak + "|" + varisEnlem + ";" + varisBoylam,
283                             "0|" + Yolculuk_ Hat1_bitis_durak_adi + "|0",
284                             "0|" + VARIS_DURAK_SIRA + "|0",
285                             "0|" + Yolculuk_ Hat1 + "|0",
286                             "0|" + fnks_ HatAdi(Yolculuk_ Hat1, Yolculuk_Alt
```

```
↪ Hat1) + "|0",
                                "0|" + Yolculuk_Alt  Hat1 + "|0",
                                "0|" + Yolculuk_  Hat1_baslama_saat + "|0",
                                "0|" + Yolculuk_  Hat1_bitis_saat + "|0",
                                mesafehesapla(Yurume1_cikis_enlem_boylam,
↪ Yurume1_varis_enlem_boylam) + "|0|" + mesafehesapla(
↪ Yurume2_cikis_enlem_boylam, Yurume2_varis_enlem_boylam)
                                );
                        }
                    }
                    alt  Hatlar = ds.Tables[0].Rows[i]["ALT_  Hat_NO"].ToString() +
↪  ",";
                    ana  Hat = ds.Tables[0].Rows[i]["ANA_  Hat_NO"].ToString();

                }
                else
                {
                    if (String.Compare(ds.Tables[0].Rows[i]["ALT_  Hat_NO"].
↪ ToString(), alt  Hatlar) == 1)
                        alt  Hatlar += ds.Tables[0].Rows[i]["ALT_  Hat_NO"].
↪ ToString() + ",";
                }
            }

        }
        else
            Sonuc = "tek vasitali   Hat bulunamadi";

        string sorgutopla = "";
        if (donecekDs.Tables[0].Rows.Count == 0)
        {

            String sorguIliski;
            string varis  Hatlartmp = "";
            string ana  Hat, alt  Hat, durak, duraksira;
            string durak2, duraksira2;

            ana  Hat = "";
            alt  Hat = "";
            duraksira = "0";
            duraksira2 = "0";
            durak = "";
            durak2 = "";
```

```
324
325
326             DataSet dss = new DataSet();
327             dss = fnks_ HatOptimize3(cikisEnlem, cikisBoylam,
    ↪ EnlemBoylaminKacMetreYakinindaDurakAranacak, hesaplamaGun,
    ↪ yolculukBaslamaSaati);
328
329             DataView view = new DataView(dss.Tables[0]);
330             view.Sort = "ANA_ Hat_NO,ALT_ Hat_NO ASC, DURAK_SIRASI
    ↪ DESC";
331             string vvvv = "";
332             string ddd = "";
333
334             foreach (DataRow row in view.ToTable().Rows)
335             {
336                 cikis Hatlar.Tables[0].Rows.Add(row["ANA_ Hat_NO"], row["ALT_
    ↪  Hat_NO"], row["DURAK_NO"], row["DURAK_SIRASI"], row["SAAT"]);
337             }
338
339             varis Hatlar = fnks_ HatOptimize2(varisEnlem, varisBoylam,
    ↪ EnlemBoylaminKacMetreYakinindaDurakAranacak);
340
341             for (int g = 0; g < varis Hatlar.Tables[0].Rows.Count; g++)
342             {
343                 fnks_get HatDurakSira(varis Hatlar.Tables[0].Rows[g]["
    ↪ DURAK_NO"].ToString(), varis Hatlar.Tables[0].Rows[g]["ANA_ Hat_NO"].
    ↪ ToString(), varis Hatlar.Tables[0].Rows[g]["ALT_ Hat_NO"].ToString(), out
    ↪ VarisDurakSirasi, out VarisIstikamet);
344                 varis Hatlartmp += varis Hatlar.Tables[0].Rows[g]["ANA_
    ↪ Hat_NO"].ToString() + "-" + varis Hatlar.Tables[0].Rows[g]["ALT_ Hat_NO
    ↪ "].ToString() + "-" + varis Hatlar.Tables[0].Rows[g]["DURAK_SIRASI"].
    ↪ ToString() + "-" + varis Hatlar.Tables[0].Rows[g]["DURAK_NO"].ToString()
    ↪  + " | ";
345             }
346
347             varis Hatlartmp = "";
348
349             for (int i = 0; i < cikis Hatlar.Tables[0].Rows.Count; i++)
350             {
351
352                 vvvv += cikis Hatlar.Tables[0].Rows[i]["ANA_ Hat_NO"].ToString
    ↪ () + "-" + cikis Hatlar.Tables[0].Rows[i]["ALT_ Hat_NO"].ToString() + "-"
    ↪  + cikis Hatlar.Tables[0].Rows[i]["DURAK_SIRASI"].ToString() + "    ////
```

```
↪ ";
353                 if (cozum >= MAX_COZUM_SAYISI) break;
354                 for (int x = 0; x < varis_Hatlar.Tables[0].Rows.Count; x++)
355                 {
356                     if (cozum >= MAX_COZUM_SAYISI) break;
357
358                     zamantekadim = DateTime.Now;
359                     t1 = DateTime.Now;
360
361                     cikis_Hatlar.Tables[0].Rows[i]["ANA_Hat_NO"].ToString(), cikis
↪  Hatlar.Tables[0].Rows[i]["ALT_Hat_NO"].ToString(), out CikisDurakSirasi,
↪ out CikisIstikamet);
362                     CikisDurakSirasi = int.Parse(cikis_Hatlar.Tables[0].Rows[i]["
↪ DURAK_SIRASI"].ToString());
363                     fnks_get_HatDurakSira(varis_Hatlar.Tables[0].Rows[x]["
↪ DURAK_NO"].ToString(), varis_Hatlar.Tables[0].Rows[x]["ANA_Hat_NO"].
↪ ToString(), varis_Hatlar.Tables[0].Rows[x]["ALT_Hat_NO"].ToString(), out
↪ VarisDurakSirasi, out VarisIstikamet);
364
365                     sorguIliski = "select h.ANA_Hat_NO1,h.ALT_Hat_NO1,h.
↪ DURAK_NO1,h.Hat1_DURAK_SIRA,h.ANA_Hat_NO2,h.ALT_Hat_NO2,h.
↪ DURAK_NO2,h.Hat2_DURAK_SIRA,h.MESAFE2,(h.Hat1_DURAK_SIRA
↪ -0)+(300-h.Hat2_DURAK_SIRA) HS ";
366                     sorguIliski += "from Hat_iliski h where ";
367                     sorguIliski += "h.ANA_Hat_NO1=" + cikis_Hatlar.Tables[0].
↪ Rows[i]["ANA_Hat_NO"] + " and ";
368                     sorguIliski += "h.ALT_Hat_NO1=" + cikis_Hatlar.Tables[0].
↪ Rows[i]["ALT_Hat_NO"] + " and ";
369                     sorguIliski += "h.ANA_Hat_NO2=" + varis_Hatlar.Tables[0].
↪ Rows[x]["ANA_Hat_NO"] + " and ";
370                     sorguIliski += "h.ALT_Hat_NO2 in (" + varis_Hatlar.Tables[0].
↪ Rows[x]["ALT_Hat_NO"] + ") and ";
371                     sorguIliski += "h.Hat1_DURAK_SIRA>" + CikisDurakSirasi
↪ + " and ";
372                     sorguIliski += "h.Hat2_DURAK_SIRA<" + VarisDurakSirasi;
373                     sorguIliski += " order by h.MESAFE2, HS limit 1";
374
375                     if (cikis_Hatlar.Tables[0].Rows[i]["ANA_Hat_NO"].ToString()
↪ == "7" && varis_Hatlar.Tables[0].Rows[x]["ANA_Hat_NO"].ToString() ==
↪ "56")
376                     {
377                         int aaa = 1;
378                     }
```

```
379
380                     sorgutopla = sorguIliski + "; ";
381
382                     varis_Hatlartmp += varis_Hatlar.Tables[0].Rows[x]["ANA_
    ↪ Hat_NO"].ToString() + "-" + varis_Hatlar.Tables[0].Rows[x]["ALT_ Hat_NO
    ↪ "].ToString() + "-" + VarisDurakSirasi + " | ";
383
384                     Hat_iliski = new DataSet();
385                     myDataAdapter.SelectCommand = new MySqlCommand(
    ↪ sorguIliski, myConn);
386                     myDataAdapter.Fill( Hat_iliski);
387
388                     gt = DateTime.Now - t1;
389                     string zamanhesapla;
390                     zamanhesapla = gt.Milliseconds.ToString();
391
392                     connectionKapat();
393
394                     if ( Hat_iliski.Tables[0].Rows.Count > 0)
395                     {
396                         bul = 0;
397
398                         for (int ix = 0; ix < aktarmaDs.Tables[0].Rows.Count; ix++)
399                         {
400
401                             if (aktarmaDs.Tables[0].Rows[ix]["ANA_ Hat1"].ToString()
    ↪ == Hat_iliski.Tables[0].Rows[0]["ANA_ Hat_NO1"].ToString() + "-" +
    ↪ Hat_iliski.Tables[0].Rows[0]["ANA_ Hat_NO2"].ToString())
402                             {
403                                 bul = 1;
404                                 break;
405                             }
406                         }
407
408                         if (bul == 0)
409                         {
410                             aktarmaDs.Tables[0].Rows.Add( Hat_iliski.Tables[0].Rows
    ↪ [0]["ANA_ Hat_NO1"].ToString() + "-" + Hat_iliski.Tables[0].Rows[0]["
    ↪ ANA_ Hat_NO2"].ToString());
411                             cikisAna_Hat = Hat_iliski.Tables[0].Rows[0]["ANA_
    ↪ Hat_NO1"].ToString();
412                             varisAna_Hat = Hat_iliski.Tables[0].Rows[0]["ANA_
    ↪ Hat_NO2"].ToString();
```

```
413                         cikisAlt_Hat =  Hat_iliski.Tables[0].Rows[0]["ALT_
    ↪ Hat_NO1"].ToString();
414                         varisAlt_Hat = varis_Hatlar.Tables[0].Rows[x]["ALT_
    ↪ Hat_NO"].ToString();// birden fazla alt_Hat var "once hangisi gelecek belli
    ↪ değil //  Hat_iliski.Tables[0].Rows[0]["ALT_ Hat_NO2"].ToString();
415                         cikisIkinciDurak =  Hat_iliski.Tables[0].Rows[0]["
    ↪ DURAK_NO1"].ToString();
416                         varisBirinciDurak =  Hat_iliski.Tables[0].Rows[0]["
    ↪ DURAK_NO2"].ToString();
417
418                         ddd += varisAna_Hat + "-" + varisAlt_Hat + "  //// 
    ↪ ";
419
420                         SonucInt = fnks_aracDurakSaatDahaOnceHesaplandimi(
    ↪ hesaplaDs, cikisAna_Hat, cikisAlt_Hat, cikis_Hatlar.Tables[0].Rows[i]["
    ↪ DURAK_NO"].ToString(), cikisIkinciDurak, out Yolculuk_ Hat1_baslama_saat,
    ↪  out Yolculuk_ Hat1_bitis_saat, out alt_Hatbul);
421                         if (SonucInt == -1)
422                         {
423                             SonucInt =
    ↪ fnks_AracinDuragaGelmeSaatiVeHedefDuragaGitmeSaati(cikisAna_Hat,
    ↪ cikisAlt_Hat, cikis_Hatlar.Tables[0].Rows[i]["DURAK_NO"].ToString(),
    ↪ cikisIkinciDurak, yolculukBaslamaSaati, hesaplamaGun, out Yolculuk_
    ↪ Hat1_baslama_saat, out Yolculuk_ Hat1_bitis_saat, out alt_Hatbul);
424                         }
425
426                         if (SonucInt == 1)
427                         {
428                             hesaplaDs.Tables[0].Rows.Add("0", cikis_Hatlar.Tables
    ↪ [0].Rows[i]["DURAK_NO"].ToString(),  Hat_iliski.Tables[0].Rows[0]["
    ↪ DURAK_NO1"].ToString(),  Hat_iliski.Tables[0].Rows[0]["ANA_ Hat_NO1"].
    ↪ ToString(),  Hat_iliski.Tables[0].Rows[0]["ALT_ Hat_NO1"].ToString(), "
    ↪ hesapla_SURE", Yolculuk_ Hat1_baslama_saat, Yolculuk_ Hat1_bitis_saat);
429                             SonucInt = fnks_aracDurakSaatDahaOnceHesaplandimi(
    ↪ hesaplaDs, varisAna_Hat, varisAlt_Hat, varisBirinciDurak, varis_Hatlar.
    ↪ Tables[0].Rows[x]["DURAK_NO"].ToString(), out Yolculuk_
    ↪ Hat2_baslama_saat, out Yolculuk_ Hat2_bitis_saat, out alt_Hatbul);
430                             if (SonucInt == -1)
431                             {
432                                 SonucInt =
    ↪ fnks_AracinDuragaGelmeSaatiVeHedefDuragaGitmeSaati(varisAna_Hat,
    ↪ varisAlt_Hat, varisBirinciDurak, varis_Hatlar.Tables[0].Rows[x]["DURAK_NO
    ↪ "].ToString(), Yolculuk_ Hat1_bitis_saat, hesaplamaGun, out Yolculuk_
```

```
↪ Hat2_baslama_saat, out Yolculuk_ Hat2_bitis_saat, out alt Hatbul);
433                             }
434
435                             if (SonucInt == 1)
436                             {
437                                 cozum++;
438                                 ADIM_SAYISI++;
439                                 hesaplaDs.Tables[0].Rows.Add("1", varisBirinciDurak
↪ , varis Hatlar.Tables[0].Rows[x]["DURAK_NO"].ToString(), varisAna Hat, alt
↪  Hatbul, "hesapla_SURE", Yolculuk_ Hat2_baslama_saat, Yolculuk_
↪ Hat2_bitis_saat);
440                                 gecenZaman = DateTime.Now − zamantekadim;
441                                 string ARAC_TURU2;
442
443
444
445                                 if ( Hat_iliski.Tables[0].Rows[0]["ANA_ Hat_NO1"].
↪ ToString() == "10") { ARAC_TURU = "T"; } else { ARAC_TURU = "O"; }
446                                 if ( Hat_iliski.Tables[0].Rows[0]["ANA_ Hat_NO2"].
↪ ToString() == "10") { ARAC_TURU2 = "T"; } else { ARAC_TURU2 = "O";
↪  }
447
448                                 Yolculuk_ Hat1 = Hat_iliski.Tables[0].Rows[0]["
↪ ANA_ Hat_NO1"].ToString();
449                                 Yolculuk_Alt Hat1 = Hat_iliski.Tables[0].Rows[0]["
↪ ALT_ Hat_NO1"].ToString();
450                                 Yolculuk_ Hat1_baslama_durak = cikis Hatlar.
↪ Tables[0].Rows[i]["DURAK_NO"].ToString();
451                                 Yolculuk_ Hat1_bitis_durak = Hat_iliski.Tables[0].
↪ Rows[0]["DURAK_NO1"].ToString(); ;
452
453                                 int donen;
454                                 donen = DuraklariSec2(Yolculuk_
↪ Hat1_baslama_durak, Yolculuk_ Hat1_bitis_durak, Yolculuk_ Hat1,
↪ Yolculuk_Alt Hat1, out Yolculuk_ Hat1_baslama_durak, out Yolculuk_
↪ Hat1_baslama_durak_sira, out Yolculuk_ Hat1_baslama_durak_yon, out
↪ Yolculuk_ Hat1_bitis_durak, out Yolculuk_ Hat1_bitis_durak_sira);
455
456                                 Yolculuk_ Hat2 = Hat_iliski.Tables[0].Rows[0]["
↪ ANA_ Hat_NO2"].ToString();
457                                 Yolculuk_Alt Hat2 = alt Hatbul;
458                                 Yolculuk_ Hat2_baslama_durak = Hat_iliski.Tables
↪ [0].Rows[0]["DURAK_NO2"].ToString();
```

```
459                                 Yolculuk_ Hat2_bitis_durak = varis Hatlar.Tables[0].
    ↪ Rows[x]["DURAK_NO"].ToString();
460
461                                 donen = DuraklariSec2(Yolculuk_
    ↪ Hat2_baslama_durak, Yolculuk_ Hat2_bitis_durak, Yolculuk_ Hat2,
    ↪ Yolculuk_Alt Hat2, out Yolculuk_ Hat2_baslama_durak, out Yolculuk_
    ↪ Hat2_baslama_durak_sira, out Yolculuk_ Hat2_baslama_durak_yon, out
    ↪ Yolculuk_ Hat2_bitis_durak, out Yolculuk_ Hat2_bitis_durak_sira);
462
463                                 Yurume1_cikis_enlem_boylam = cikisEnlem + ";" +
    ↪ cikisBoylam;
464                                 Yurume1_varis_enlem_boylam = durakenlemboylam(
    ↪ Yolculuk_ Hat1_baslama_durak, out Yolculuk_ Hat1_baslama_durak_adi);
465                                 Yurume2_cikis_enlem_boylam = durakenlemboylam(
    ↪ Yolculuk_ Hat1_bitis_durak, out Yolculuk_ Hat1_bitis_durak_adi);
466                                 Yurume2_varis_enlem_boylam = durakenlemboylam(
    ↪ Yolculuk_ Hat2_baslama_durak, out Yolculuk_ Hat2_baslama_durak_adi);
467                                 Yurume3_cikis_enlem_boylam = durakenlemboylam(
    ↪ Yolculuk_ Hat2_bitis_durak, out Yolculuk_ Hat2_bitis_durak_adi);
468                                 Yurume3_varis_enlem_boylam = varisEnlem + ";" +
    ↪ varisBoylam;
469
470                                 donecekDs.Tables[0].Rows.Add(
471                                     sessionid,
472                                     cozum.ToString(),
473                                     "5",
474                                     "Y|" + ARAC_TURU + "|Y|" + ARAC_TURU2
    ↪ + "|Y",
475                                      Yurume1_cikis_enlem_boylam + "|" + Yolculuk_
    ↪ Hat1_baslama_durak + "|" + Yurume2_cikis_enlem_boylam + "|" + Yolculuk_
    ↪  Hat2_baslama_durak + "|" + Yurume3_cikis_enlem_boylam,
476
477                                      "0|" + Yolculuk_ Hat1_baslama_durak_adi +
    ↪ "|0|" + Yolculuk_ Hat2_baslama_durak_adi + "|0",
478                                      "0|" + Yolculuk_ Hat1_baslama_durak_yon +
    ↪ "|0|" + Yolculuk_ Hat2_baslama_durak_yon + "|0",
479                                      "0|" + Yolculuk_ Hat1_baslama_durak_sira +
    ↪ "|0|" + Yolculuk_ Hat2_baslama_durak_sira + "|0",
480
481                                     Yurume1_varis_enlem_boylam + "|" + Yolculuk_
    ↪ Hat1_bitis_durak + "|" + Yurume2_varis_enlem_boylam + "|" + Yolculuk_
    ↪ Hat2_bitis_durak + "|" + Yurume3_varis_enlem_boylam,
482
```

```
483                                     "0|" + Yolculuk_ Hat1_bitis_durak_adi + "|0|" +
    ↪ Yolculuk_ Hat2_bitis_durak_adi + "|0",
484
485                                     "0|" + Yolculuk_ Hat1_bitis_durak_sira + "|0|" +
    ↪  Yolculuk_ Hat2_bitis_durak_sira + "|0",
486
487                                     "0|" + Yolculuk_ Hat1 + "|0|" + Yolculuk_ Hat2
    ↪  + "|0",
488
489                                     "0|" + fnks_ HatAdi(Yolculuk_ Hat1,
    ↪ Yolculuk_Alt Hat1) + "|0|" + fnks_ HatAdi(Yolculuk_ Hat2, Yolculuk_Alt
    ↪ Hat2) + "|0",
490
491                                     "0|" + Yolculuk_Alt Hat1 + "|0|" + Yolculuk_Alt
    ↪  Hat2 + "|0",
492
493                                     "0|" + Yolculuk_ Hat1_baslama_saat + "|0|" +
    ↪ Yolculuk_ Hat2_baslama_saat + "|0",
494
495                                     "0|" + Yolculuk_ Hat1_bitis_saat + "|0|" +
    ↪ Yolculuk_ Hat2_bitis_saat + "|0",
496
497                                     mesafehesapla(Yurume1_cikis_enlem_boylam,
    ↪ Yurume1_varis_enlem_boylam) + "|0|" + mesafehesapla(
    ↪ Yurume2_cikis_enlem_boylam, Yurume2_varis_enlem_boylam)
498                                     );
499
500                                 if (cozum > MAX_COZUM_SAYISI) break;
501                             }
502                             else
503                             {
504                                 hesaplaDs.Tables[0].Rows.Add("0", varisBirinciDurak
    ↪ , varis Hatlar.Tables[0].Rows[x]["DURAK_NO"].ToString(), varisAna Hat, alt
    ↪  Hatbul, "-1", "00:00:00", "00:00:00");
505                             }
506
507                         }
508                         else
509                         {
510                             hesaplaDs.Tables[0].Rows.Add("0", cikis Hatlar.Tables
    ↪ [0].Rows[i]["DURAK_NO"].ToString(), cikisIkinciDurak, cikisAna Hat,
    ↪ cikisAlt Hat, "-1", "00:00:00", "00:00:00");
511                         }
```

```
512
513                         }
514
515                     }
516
517                 }
518             }
519
520         }
521
522
523
524
525
526         int sayb = 0;
527
528         string ch = "", vh = "";
529
530         if (donecekDs.Tables[0].Rows.Count == 0)
531         {
532
533             for (int i = 0; i < cikis_Hatlar.Tables[0].Rows.Count; i++)
534             {
535                 ch += cikis_Hatlar.Tables[0].Rows[i]["ANA_Hat_NO"].ToString()
    ↪ + "-" + cikis_Hatlar.Tables[0].Rows[i]["ALT_Hat_NO"].ToString() + "-" +
    ↪  cikis_Hatlar.Tables[0].Rows[i]["DURAK_SIRASI"].ToString() + "    //// ";
536                 for (int x = 0; x < varis_Hatlar.Tables[0].Rows.Count; x++)
537                 {
538                     if (cozum > MAX_COZUM_SAYISI) break;
539
540                     vh += varis_Hatlar.Tables[0].Rows[x]["ANA_Hat_NO"].
    ↪ ToString() + "-" + varis_Hatlar.Tables[0].Rows[x]["ALT_Hat_NO"].ToString
    ↪ () + "-" + varis_Hatlar.Tables[0].Rows[x]["DURAK_SIRASI"].ToString() + "
    ↪     //// ";
541                     t1 = DateTime.Now;
542                     fnks_getHatDurakSira(cikis_Hatlar.Tables[0].Rows[i]["
    ↪ DURAK_NO"].ToString(), cikis_Hatlar.Tables[0].Rows[i]["ANA_Hat_NO"].
    ↪ ToString(), cikis_Hatlar.Tables[0].Rows[i]["ALT_Hat_NO"].ToString(), out
    ↪ CikisDurakSirasi, out CikisIstikamet);
543
544                     sorgu = "select * from  Hat_iliski_3 Hat h where CIKIS_Hat
    ↪ =" + cikis_Hatlar.Tables[0].Rows[i]["ANA_Hat_NO"].ToString() + " and
    ↪ VARIS_Hat = " + varis_Hatlar.Tables[0].Rows[x]["ANA_Hat_NO"].
```

```
    ↪ ToString();
545
546                 DataSet Hat_iliski3 Hat = new DataSet();
547                 myDataAdapter.SelectCommand = new MySqlCommand(sorgu,
    ↪ myConn);
548                 myDataAdapter.Fill( Hat_iliski3 Hat);
549
550                 sayb += Hat_iliski3 Hat.Tables[0].Rows.Count;
551
552                 for (int id = 0; id <= Hat_iliski3 Hat.Tables[0].Rows.Count -
    ↪ 1; id++) // "Uçl"U aktarma
553                 {
554                     if (cozum > MAX_COZUM_SAYISI) break;
555
556                     zamantekadim = DateTime.Now;
557                     sorgu = "select distinct t.ALT_ Hat_NO2 as ALT_ Hat_NO
    ↪ from Hat_iliski t where t.ANA_ Hat_NO1=" + Hat_iliski3 Hat.Tables[0].
    ↪ Rows[id]["VARIS_ Hat"] + " and t.ANA_ Hat_NO2=" + Hat_iliski3 Hat.
    ↪ Tables[0].Rows[id]["AKTARMA_ Hat"];
558                     Hat_iliski.Clear();
559                     myDataAdapter.SelectCommand = new MySqlCommand(
    ↪ sorgu, myConn);
560                     myDataAdapter.Fill( Hat_iliski);
561
562                     string stralt Hat = "";
563                     for (int b = 0; b < Hat_iliski.Tables[0].Rows.Count; b++)
564                     {
565                         stralt Hat += Hat_iliski.Tables[0].Rows[b]["ALT_
    ↪ Hat_NO"] + ",";
566                     }
567
568                     if ( Hat_iliski.Tables[0].Rows.Count == 0)
569                         continue;
570
571                     stralt Hat = stralt Hat.Substring(0, stralt Hat.Length - 1);
572
573                     sorgu = "select h.ANA_ Hat_NO1,h.ALT_ Hat_NO1,h.
    ↪ DURAK_NO1,h. Hat1_DURAK_SIRA,h.ANA_ Hat_NO2,h.ALT_ Hat_NO2,h.
    ↪ DURAK_NO2,h. Hat2_DURAK_SIRA,h.MESAFE2,(h. Hat1_DURAK_SIRA
    ↪ -0)+(300-h. Hat2_DURAK_SIRA) HS ";
574                     sorgu += "from Hat_iliski h where ";
575                     sorgu += "h.ANA_ Hat_NO1=" + cikis Hatlar.Tables[0].
    ↪ Rows[i]["ANA_ Hat_NO"] + " and ";
```

```
576                     sorgu += "h.ALT_ Hat_NO1=" + cikis Hatlar.Tables[0].
    ↪ Rows[i]["ALT_ Hat_NO"] + " and ";
577                     sorgu += "h.ANA_ Hat_NO2=" + Hat_iliski3 Hat.Tables
    ↪ [0].Rows[id]["AKTARMA_ Hat"] + " and ";
578                     sorgu += "h.ALT_ Hat_NO2 in (" + stralt Hat + ") and ";
579                     sorgu += "h. Hat1_DURAK_SIRA>" + CikisDurakSirasi;
580                     sorgu += " order by h.MESAFE2, HS limit 2";
581
582                      Hat_iliski.Clear();
583
584                     myDataAdapter.SelectCommand = new MySqlCommand(
    ↪ sorgu, myConn);
585                     myDataAdapter.Fill( Hat_iliski);
586
587
588                     if ( Hat_iliski.Tables[0].Rows.Count > 0)
589                     {
590                        stralt Hat = Hat_iliski.Tables[0].Rows[0]["ALT_
    ↪ Hat_NO2"].ToString() + ",";
591                        for (int b = 0; b < Hat_iliski.Tables[0].Rows.Count; b++)
592                        {
593                           if (b != 0 && Hat_iliski.Tables[0].Rows[b]["ALT_
    ↪ Hat_NO2"] != Hat_iliski.Tables[0].Rows[b - 1]["ALT_ Hat_NO2"])
594                              stralt Hat += Hat_iliski.Tables[0].Rows[b]["ALT_
    ↪ Hat_NO2"].ToString() + ",";
595                        }
596
597                        stralt Hat = stralt Hat.Substring(0, stralt Hat.Length −
    ↪ 1);
598
599                        say++;
600                        bul = 0;
601
602                        string buyukduraksira = Hat_iliski.Tables[0].Rows[0]["
    ↪ Hat2_DURAK_SIRA"].ToString();
603                        string kucukduraksira = "−1";
604                        if ( Hat_iliski.Tables[0].Rows.Count > 2)
605                        {
606                           if (int.Parse( Hat_iliski.Tables[0].Rows[0]["
    ↪ Hat2_DURAK_SIRA"].ToString()) < int.Parse( Hat_iliski.Tables[0].Rows[1]["
    ↪ Hat2_DURAK_SIRA"].ToString()))
607                           {
608                              buyukduraksira = Hat_iliski.Tables[0].Rows[1]["
```

```
↪ Hat2_DURAK_SIRA"].ToString();
609                             kucukduraksira =   Hat_iliski.Tables[0].Rows[0]["
↪ Hat2_DURAK_SIRA"].ToString();
610                         }
611                         else
612                         {
613                             buyukduraksira =   Hat_iliski.Tables[0].Rows[0]["
↪ Hat2_DURAK_SIRA"].ToString();
614                             kucukduraksira =   Hat_iliski.Tables[0].Rows[1]["
↪ Hat2_DURAK_SIRA"].ToString();
615                         }
616                     }
617
618
619                     sorgu = "select h.ANA_ Hat_NO1,h.ALT_ Hat_NO1,h.
↪ DURAK_NO1,h. Hat1_DURAK_SIRA,h.ANA_ Hat_NO2,h.ALT_ Hat_NO2,h.
↪ DURAK_NO2,h. Hat2_DURAK_SIRA,h.MESAFE2,(h. Hat1_DURAK_SIRA
↪ -0)+(300-h. Hat2_DURAK_SIRA) HS ";
620                     sorgu += "from   Hat_iliski h where ";
621                     sorgu += "h.ANA_ Hat_NO1=" +   Hat_iliski.Tables[0].
↪ Rows[0]["ANA_ Hat_NO2"] + " and ";
622                     sorgu += "h.ALT_ Hat_NO1 in (" + stralt  Hat + " ) and
↪  ";
623                     sorgu += "h.ANA_ Hat_NO2=" +   Hat_iliski3  Hat.
↪ Tables[0].Rows[id]["VARIS_ Hat"] + " and ";
624                     sorgu += "h.ALT_ Hat_NO2 in (" + stralt  Hat + ") and
↪ ";
625                     sorgu += "h. Hat1_DURAK_SIRA>" + buyukduraksira
↪ + " ";
626                     sorgu += " order by h.MESAFE2, HS limit 2";
627
628                     temp.Clear();
629                     myDataAdapter.SelectCommand = new MySqlCommand(
↪ sorgu, myConn);
630                     myDataAdapter.Fill(temp);
631
632                     if (temp.Tables[0].Rows.Count == 0)
633                     {
634
635                         if (kucukduraksira != "-1")
636                         {
637                             sorgu = "select h.ANA_ Hat_NO1,h.ALT_ Hat_NO1
↪ ,h.DURAK_NO1,h. Hat1_DURAK_SIRA,h.ANA_ Hat_NO2,h.ALT_ Hat_NO2
```

```
    ↪ ,h.DURAK_NO2,h. Hat2_DURAK_SIRA,h.MESAFE2,(h.
    ↪ Hat1_DURAK_SIRA−0)+(300−h. Hat2_DURAK_SIRA) HS ";
638                             sorgu += "from  Hat_iliski h where ";
639                             sorgu += "h.ANA_ Hat_NO1=" +  Hat_iliski.
    ↪ Tables[0].Rows[0]["ANA_ Hat_NO2"] + " and ";
640                             sorgu += "h.ALT_ Hat_NO1 in (" + stralt  Hat + "
    ↪  ) and ";
641                             sorgu += "h.ANA_ Hat_NO2=" +  Hat_iliski3  Hat
    ↪ .Tables[0].Rows[id]["VARIS_ Hat"] + " and ";
642                             sorgu += "h. Hat1_DURAK_SIRA>" +
    ↪ kucukduraksira + " ";
643                             sorgu += " order by h.MESAFE2, HS limit 2";
644                             temp.Clear();
645                             myDataAdapter.SelectCommand = new
    ↪ MySqlCommand(sorgu, myConn);
646                             myDataAdapter.Fill(temp);
647                         }
648                     }
649
650
651                     if (temp.Tables[0].Rows.Count > 0)
652                     { fnks_AracinDuragaGelmeSaatiVeHedefDuragaGitmeSaati
    ↪ (  Hat_iliski.Tables[0].Rows[0]["ANA_ Hat_NO1"].ToString().ToString(), cikis
    ↪ Hatlar.Tables[0].Rows[i]["ALT_ Hat_NO"].ToString(), cikis  Hatlar.Tables[0].
    ↪ Rows[i]["DURAK_NO"].ToString(),  Hat_iliski.Tables[0].Rows[0]["
    ↪ DURAK_NO1"].ToString(), yolculukBaslamaSaati, hesaplamaGun, out
    ↪ Yolculuk_ Hat1_baslama_saat, out Yolculuk_ Hat1_bitis_saat, out alt  Hatbul);
653
654
655                         for (int f = 0; f <  Hat_iliski.Tables[0].Rows.Count; f
    ↪ ++)
656                         {
657                             for (int y = 0; y < temp.Tables[0].Rows.Count; y++)
658                             {
659                                 bul = 0;
660                                 for (int ix = 0; ix < aktarmaDs.Tables[0].Rows.
    ↪ Count; ix++)
661                                 {
662                                     if (aktarmaDs.Tables[0].Rows[ix]["ANA_ Hat1
    ↪ "].ToString() ==  Hat_iliski.Tables[0].Rows[0]["ANA_ Hat_NO1"].ToString()
    ↪ + "−" +  Hat_iliski.Tables[0].Rows[f]["ANA_ Hat_NO2"].ToString() + "−" +
    ↪  temp.Tables[0].Rows[y]["ANA_ Hat_NO2"].ToString())
663                                     {
```

```
664                                     bul = 1;
665                                     break;
666                                 }
667                             }
668
669                             if (bul == 0)
670                             {
671
672                                 sorgu = "select h.ANA_ Hat_NO1,h.ALT_
    ↪ Hat_NO1,h.DURAK_NO1,h. Hat1_DURAK_SIRA,h.ANA_ Hat_NO2,h.ALT_
    ↪ Hat_NO2,h.DURAK_NO2,h. Hat2_DURAK_SIRA,h.MESAFE2,(h.
    ↪ Hat1_DURAK_SIRA-0)+(300-h. Hat2_DURAK_SIRA) HS ";
673                                 sorgu += "from Hat_iliski h where ";
674                                 sorgu += "h.ANA_ Hat_NO1=" + Hat_iliski
    ↪ .Tables[0].Rows[f]["ANA_ Hat_NO2"] + " and ";
675                                 sorgu += "h.ALT_ Hat_NO1 =" + Hat_iliski
    ↪ .Tables[0].Rows[f]["ALT_ Hat_NO2"] + " and ";
676                                 sorgu += "h.ANA_ Hat_NO2=" + temp.
    ↪ Tables[0].Rows[y]["ANA_ Hat_NO2"] + " and ";
677                                 sorgu += "h.ALT_ Hat_NO2 in (" + temp.
    ↪ Tables[0].Rows[y]["ALT_ Hat_NO2"] + ") and ";
678                                 sorgu += "h. Hat1_DURAK_SIRA>" +
    ↪ Hat_iliski.Tables[0].Rows[f][" Hat2_DURAK_SIRA"] + " ";
679                                 sorgu += " and h. Hat1_DURAK_SIRA<" +
    ↪  temp.Tables[0].Rows[y][" Hat2_DURAK_SIRA"] + " ";
680                                 sorgu += " and h. Hat2_DURAK_SIRA<" +
    ↪  varis Hatlar.Tables[0].Rows[x]["DURAK_SIRASI"] + " ";
681                                 sorgu += " order by h.MESAFE2, HS limit
    ↪ 2";
682
683                                 aktarma Hatti.Clear();
684                                 myDataAdapter.SelectCommand = new
    ↪ MySqlCommand(sorgu, myConn);
685                                 myDataAdapter.Fill(aktarma Hatti);
686
687                                 if (aktarma Hatti.Tables[0].Rows.Count > 0)
688                                 {
689
690                                     SonucInt =
    ↪ fnks_AracinDuragaGelmeSaatiVeHedefDuragaGitmeSaati(aktarma Hatti.
    ↪ Tables[0].Rows[0]["ANA_ Hat_NO1"].ToString(), aktarma Hatti.Tables[0].
    ↪ Rows[0]["ALT_ Hat_NO1"].ToString(), Hat_iliski.Tables[0].Rows[f]["
    ↪ DURAK_NO2"].ToString(), aktarma Hatti.Tables[0].Rows[0]["DURAK_NO1
```

```
↪ "].ToString(), Yolculuk_ Hat1_bitis_saat, hesaplamaGun, out Yolculuk_
↪ Hat2_baslama_saat, out Yolculuk_ Hat2_bitis_saat, out alt Hatbul);
691                                     if (SonucInt == -1)
692                                     { continue; }
693
694                                     SonucInt =
↪ fnks_AracinDuragaGelmeSaatiVeHedefDuragaGitmeSaati(aktarma Hatti.
↪ Tables[0].Rows[0]["ANA_ Hat_NO2"].ToString(), aktarma Hatti.Tables[0].
↪ Rows[0]["ALT_ Hat_NO2"].ToString(), aktarma Hatti.Tables[0].Rows[0]["
↪ DURAK_NO2"].ToString(), varis Hatlar.Tables[0].Rows[x]["DURAK_NO"].
↪ ToString(), Yolculuk_ Hat2_bitis_saat, hesaplamaGun, out Yolculuk_
↪ Hat3_baslama_saat, out Yolculuk_ Hat3_bitis_saat, out alt Hatbul);
695                                     if (SonucInt == -1)
696                                     { continue; }
697
698                                     gecenZaman = DateTime.Now -
↪ zamantamami;
699
700                                     aktarmaDs.Tables[0].Rows.Add( Hat_iliski.
↪ Tables[0].Rows[0]["ANA_ Hat_NO1"].ToString() + "-" + aktarma Hatti.
↪ Tables[0].Rows[0]["ANA_ Hat_NO1"].ToString() + "-" + aktarma Hatti.
↪ Tables[0].Rows[0]["ANA_ Hat_NO2"].ToString());
701
702                                     string CIKIS_DURAK_NO2,
↪ VARIS_DURAK_NO2, CIKIS_DURAK_NO3, VARIS_DURAK_NO3,
↪ ARAC_TURU2, ARAC_TURU3;
703
704                                     string CIKIS_DURAK_NO1 = cikis Hatlar.
↪ Tables[0].Rows[i]["DURAK_NO"].ToString();
705                                     string VARIS_DURAK_NO1 = Hat_iliski.
↪ Tables[0].Rows[0]["DURAK_NO1"].ToString();
706
707                                     if ( Hat_iliski.Tables[0].Rows[0]["ANA_
↪ Hat_NO1"].ToString() == "10") { ARAC_TURU = "T"; } else {
↪ ARAC_TURU = "O"; }
708                                     CIKIS_DURAK_NO2 = Hat_iliski.Tables
↪ [0].Rows[f]["DURAK_NO2"].ToString();
709                                     VARIS_DURAK_NO2 = aktarma Hatti.
↪ Tables[0].Rows[0]["DURAK_NO1"].ToString();
710
711                                     if (aktarma Hatti.Tables[0].Rows[0]["ANA_
↪ Hat_NO1"].ToString() == "10") { ARAC_TURU2 = "T"; } else {
↪ ARAC_TURU2 = "O"; }
```

```
712                                         CIKIS_DURAK_NO3 = aktarma_Hatti.
    ↪ Tables[0].Rows[0]["DURAK_NO2"].ToString();
713                                         VARIS_DURAK_NO3 = varis_Hatlar.
    ↪ Tables[0].Rows[x]["DURAK_NO"].ToString();
714
715                                         if (aktarma_Hatti.Tables[0].Rows[0]["ANA_
    ↪  Hat_NO2"].ToString() == "10") { ARAC_TURU3 = "T"; } else {
    ↪ ARAC_TURU3 = "O"; }
716
717                                         cozum++;
718                                         int Hat_durak_sira_baslangic1,
    ↪ Hat_durak_sira_baslangic2, Hat_durak_sira_baslangic3;
719                                         int Hat_durak_sira_bitis1,
    ↪ Hat_durak_sira_bitis2, Hat_durak_sira_bitis3;
720                                         string yon;
721                                         Yolculuk_Hat1 = Hat_iliski.Tables[0].
    ↪ Rows[0]["ANA_Hat_NO1"].ToString();
722                                         Yolculuk_Alt_Hat1 = cikis_Hatlar.Tables
    ↪ [0].Rows[i]["ALT_Hat_NO"].ToString();
723                                         Yolculuk_Hat1_baslama_durak =
    ↪ CIKIS_DURAK_NO1;
724                                         Yolculuk_Hat1_bitis_durak =
    ↪ VARIS_DURAK_NO1;
725                                         DuraklariSec2(CIKIS_DURAK_NO1,
    ↪ VARIS_DURAK_NO1, Yolculuk_Hat1, Yolculuk_Alt_Hat1, out
    ↪ CIKIS_DURAK_NO1, out Yolculuk_Hat1_baslama_durak_sira, out Yolculuk_
    ↪ Hat1_baslama_durak_yon, out VARIS_DURAK_NO1, out Yolculuk_
    ↪ Hat1_bitis_durak_sira);
726                                         Yolculuk_Hat2 = aktarma_Hatti.Tables[0].
    ↪ Rows[0]["ANA_Hat_NO1"].ToString();
727                                         Yolculuk_Alt_Hat2 = aktarma_Hatti.
    ↪ Tables[0].Rows[0]["ALT_Hat_NO1"].ToString();
728                                         Yolculuk_Hat2_baslama_durak =
    ↪ CIKIS_DURAK_NO2;
729                                         Yolculuk_Hat2_bitis_durak =
    ↪ VARIS_DURAK_NO2;
730                                         DuraklariSec2(CIKIS_DURAK_NO2,
    ↪ VARIS_DURAK_NO2, Yolculuk_Hat2, Yolculuk_Alt_Hat2, out
    ↪ CIKIS_DURAK_NO2, out Yolculuk_Hat2_baslama_durak_sira, out Yolculuk_
    ↪ Hat2_baslama_durak_yon, out VARIS_DURAK_NO2, out Yolculuk_
    ↪ Hat2_bitis_durak_sira);
731                                         Yolculuk_Hat3 = aktarma_Hatti.Tables[0].
    ↪ Rows[0]["ANA_Hat_NO2"].ToString();
```

```
732                                     Yolculuk_Alt  Hat3 = aktarma  Hatti.
    ↪ Tables[0].Rows[0]["ALT_ Hat_NO2"].ToString();
733                                     Yolculuk_ Hat3_baslama_durak =
    ↪ CIKIS_DURAK_NO3;
734                                     Yolculuk_ Hat3_bitis_durak =
    ↪ VARIS_DURAK_NO3;
735                                     DuraklariSec2(CIKIS_DURAK_NO3,
    ↪ VARIS_DURAK_NO3, Yolculuk_ Hat3, Yolculuk_Alt  Hat3, out
    ↪ CIKIS_DURAK_NO3, out Yolculuk_ Hat3_baslama_durak_sira, out Yolculuk_
    ↪ Hat3_baslama_durak_yon, out VARIS_DURAK_NO3, out Yolculuk_
    ↪ Hat3_bitis_durak_sira);
736                                     Yurume1_cikis_enlem_boylam = cikisEnlem
    ↪ + ";" + cikisBoylam;
737                                     Yurume1_varis_enlem_boylam =
    ↪ durakenlemboylam(Yolculuk_ Hat1_baslama_durak, out Yolculuk_
    ↪ Hat1_baslama_durak_adi);
738                                     Yurume2_cikis_enlem_boylam =
    ↪ durakenlemboylam(Yolculuk_ Hat1_bitis_durak, out Yolculuk_
    ↪ Hat1_bitis_durak_adi);
739                                     Yurume2_varis_enlem_boylam =
    ↪ durakenlemboylam(Yolculuk_ Hat2_baslama_durak, out Yolculuk_
    ↪ Hat2_baslama_durak_adi);
740                                     Yurume3_cikis_enlem_boylam =
    ↪ durakenlemboylam(Yolculuk_ Hat2_bitis_durak, out Yolculuk_
    ↪ Hat2_bitis_durak_adi);
741                                     Yurume3_varis_enlem_boylam =
    ↪ durakenlemboylam(Yolculuk_ Hat3_baslama_durak, out Yolculuk_
    ↪ Hat3_baslama_durak_adi);
742                                     Yurume4_cikis_enlem_boylam =
    ↪ durakenlemboylam(Yolculuk_ Hat3_bitis_durak, out Yolculuk_
    ↪ Hat3_bitis_durak_adi);
743                                     Yurume4_varis_enlem_boylam = varisEnlem
    ↪ + ";" + varisBoylam;
744
745                                     donecekDs.Tables[0].Rows.Add(
746                                         sessionid,
747                                         cozum.ToString(),
748                                         "7",
749                                         "Y|" + ARAC_TURU + "|Y|" +
    ↪ ARAC_TURU2 + "|Y|" + ARAC_TURU3 + "|Y",
750                                         Yurume1_cikis_enlem_boylam + "|" +
    ↪ Yolculuk_ Hat1_baslama_durak + "|" + Yurume2_cikis_enlem_boylam + "|" +
    ↪  Yolculuk_ Hat2_baslama_durak + "|" + Yurume3_cikis_enlem_boylam + "|"
```

```
↪ + Yolculuk_ Hat3_baslama_durak + "|" + Yurume4_cikis_enlem_boylam,
751                                         "0|" + Yolculuk_
↪ Hat1_baslama_durak_adi + "|0|" + Yolculuk_ Hat2_baslama_durak_adi + "|0|"
↪  + Yolculuk_ Hat3_baslama_durak_adi + "|0",
752                                         "0|" + Yolculuk_
↪ Hat1_baslama_durak_yon + "|0|" + Yolculuk_ Hat2_baslama_durak_yon +
↪ "|0|" + Yolculuk_ Hat3_baslama_durak_yon + "|0",
753                                         "0|" + Yolculuk_
↪ Hat1_baslama_durak_sira + "|0|" + Yolculuk_ Hat2_baslama_durak_sira +
↪ "|0|" + Yolculuk_ Hat3_baslama_durak_sira + "|0",
754                                         Yurume1_varis_enlem_boylam + "|" +
↪ Yolculuk_ Hat1_bitis_durak + "|" + Yurume2_varis_enlem_boylam + "|" +
↪ Yolculuk_ Hat2_bitis_durak + "|" + Yurume3_varis_enlem_boylam + "|" +
↪ Yolculuk_ Hat3_bitis_durak + "|" + Yurume4_varis_enlem_boylam,
755                                         "0|" + Yolculuk_ Hat1_bitis_durak_adi +
↪  "|0|" + Yolculuk_ Hat2_bitis_durak_adi + "|0|" + Yolculuk_
↪ Hat3_bitis_durak_adi + "|0",
756                                         "0|" + Yolculuk_ Hat1_bitis_durak_sira
↪ + "|0|" + Yolculuk_ Hat2_bitis_durak_sira + "|0|" + Yolculuk_
↪ Hat3_bitis_durak_sira + "|0",
757                                         "0|" + Yolculuk_ Hat1 + "|0|" +
↪ Yolculuk_ Hat2 + "|0|" + Yolculuk_ Hat3 + "|0",
758                                         "0|" + fnks_ HatAdi(Yolculuk_ Hat1,
↪ Yolculuk_Alt Hat1) + "|0|" + fnks_ HatAdi(Yolculuk_ Hat2, Yolculuk_Alt
↪ Hat2) + "|0|" + fnks_ HatAdi(Yolculuk_ Hat3, Yolculuk_Alt Hat3) + "|0",
759                                         "0|" + Yolculuk_Alt Hat1 + "|0|" +
↪ Yolculuk_Alt Hat2 + "|0|" + Yolculuk_Alt Hat3 + "|0",
760
761                                         "0|" + Yolculuk_ Hat1_baslama_saat +
↪ "|0|" + Yolculuk_ Hat2_baslama_saat + "|0|" + Yolculuk_ Hat3_baslama_saat
↪  + "|0",
762                                         "0|" + Yolculuk_ Hat1_bitis_saat + "|0|"
↪  + Yolculuk_ Hat2_bitis_saat + "|0|" + Yolculuk_ Hat3_bitis_saat + "|0",
763                                         mesafehesapla(
↪ Yurume1_cikis_enlem_boylam, Yurume1_varis_enlem_boylam) + "|0|" +
↪ mesafehesapla(Yurume2_cikis_enlem_boylam, Yurume2_varis_enlem_boylam) +
↪ "|0|" + mesafehesapla(Yurume3_cikis_enlem_boylam,
↪ Yurume3_varis_enlem_boylam)
764                                         );
765
766                                     if (cozum > MAX_COZUM_SAYISI) break;
767                                 }
768                             }
```

```
769                                 }
770                             }
771                         }
772                     }
773                 }
774             }
775         }
776     }
777
778
779     string DURAK_sira = "";
780     string durakno_2 = "";
781     sayb = 0;
782
783     ch = "";
784     vh = "";
785
786     if (donecekDs.Tables[0].Rows.Count == 0)
787     {
788         for (int i = 0; i < cikis  Hatlar.Tables[0].Rows.Count; i++)
789         {
790             if (cozum > MAX_COZUM_SAYISI) break;
791
792             ch += cikis  Hatlar.Tables[0].Rows[i]["ANA_  Hat_NO"].ToString()
↪ + "-" + cikis  Hatlar.Tables[0].Rows[i]["ALT_  Hat_NO"].ToString() + "-" +
↪  cikis  Hatlar.Tables[0].Rows[i]["DURAK_SIRASI"].ToString() + "    //// ";
793             for (int x = 0; x < varis  Hatlar.Tables[0].Rows.Count; x++)
794             {
795                 vh += varis  Hatlar.Tables[0].Rows[x]["ANA_  Hat_NO"].
↪ ToString() + ",";// +varis  Hatlar.Tables[0].Rows[x]["ALT_  Hat_NO"].
↪ ToString() + "-" + varis  Hatlar.Tables[0].Rows[x]["DURAK_SIRASI"].
↪ ToString() + "    //// ";
796             }
797
798             vh = vh.Substring(0, vh.Length - 1);
799             t1 = DateTime.Now;
800             fnks_get  HatDurakSira(cikis  Hatlar.Tables[0].Rows[i]["DURAK_NO
↪ "].ToString(), cikis  Hatlar.Tables[0].Rows[i]["ANA_  Hat_NO"].ToString(),
↪ cikis  Hatlar.Tables[0].Rows[i]["ALT_  Hat_NO"].ToString(), out
↪ CikisDurakSirasi, out CikisIstikamet);
801
802             sorgu = "select * from   Hat_iliski_3  Hat t where t.VARIS_  Hat in
↪ (" + vh + ") and t.CIKIS_  Hat in ";
```

```
803                 sorgu += "(select distinct h.ANA_ Hat_NO2 Hat from Hat_iliski
    ↪ h where h.ANA_ Hat_NO1 in (" + cikis Hatlar.Tables[0].Rows[i]["ANA_
    ↪ Hat_NO"].ToString() + ") order by CIKIS_ Hat,VARIS_ Hat)";
804                 DataSet Hat_iliski3 Hat = new DataSet();
805                 myDataAdapter.SelectCommand = new MySqlCommand(sorgu,
    ↪ myConn);
806                 myDataAdapter.Fill( Hat_iliski3 Hat);
807
808                 sayb += Hat_iliski3 Hat.Tables[0].Rows.Count;
809
810                 for (int id = 0; id <= Hat_iliski3 Hat.Tables[0].Rows.Count - 1;
    ↪ id++) // "Uçl"U aktarma
811                 {
812                     if (cozum > MAX_COZUM_SAYISI) break;
813
814                     zamantekadim = DateTime.Now;
815                     sorgu = "select distinct t.ALT_ Hat_NO1 as ALT_ Hat_NO from
    ↪ Hat_iliski t where t.ANA_ Hat_NO1=" + Hat_iliski3 Hat.Tables[0].Rows[
    ↪ id]["CIKIS_ Hat"] + " and t.ANA_ Hat_NO2=" + Hat_iliski3 Hat.Tables
    ↪ [0].Rows[id]["AKTARMA_ Hat"];
816                     Hat_iliski.Clear();
817                     myDataAdapter.SelectCommand = new MySqlCommand(sorgu,
    ↪ myConn);
818                     myDataAdapter.Fill( Hat_iliski);
819
820                     string stralt Hat = "";
821                     for (int b = 0; b < Hat_iliski.Tables[0].Rows.Count; b++)
822                     {
823                         stralt Hat += Hat_iliski.Tables[0].Rows[b]["ALT_ Hat_NO
    ↪ "] + ",";
824                     }
825
826                     if ( Hat_iliski.Tables[0].Rows.Count == 0)
827                         continue;
828
829                     stralt Hat = stralt Hat.Substring(0, stralt Hat.Length - 1);
830
831                     sorgu = "select h.ANA_ Hat_NO1,h.ALT_ Hat_NO1,h.
    ↪ DURAK_NO1,h. Hat1_DURAK_SIRA,h.ANA_ Hat_NO2,h.ALT_ Hat_NO2,h.
    ↪ DURAK_NO2,h. Hat2_DURAK_SIRA,h.MESAFE2,(h. Hat1_DURAK_SIRA
    ↪ -0)+(300-h. Hat2_DURAK_SIRA) HS ";
832                     sorgu += "from Hat_iliski h where ";
833                     sorgu += "h.ANA_ Hat_NO1=" + cikis Hatlar.Tables[0].Rows[i
```

```
↪ ]["ANA_ Hat_NO"] + " and ";
834                     sorgu += "h.ALT_ Hat_NO1=" + cikis Hatlar.Tables[0].Rows[i
↪ ]["ALT_ Hat_NO"] + " and ";
835                     sorgu += "h.ANA_ Hat_NO2=" + Hat_iliski3 Hat.Tables[0].
↪ Rows[id]["CIKIS_ Hat"] + " and ";
836                     sorgu += "h.ALT_ Hat_NO2 in (" + stralt Hat + ") and ";
837                     sorgu += "h. Hat1_DURAK_SIRA>" + CikisDurakSirasi;
838                     sorgu += " order by h.MESAFE2, HS limit 2";
839
840                      Hat_iliski.Clear();
841                     myDataAdapter.SelectCommand = new MySqlCommand(sorgu,
↪ myConn);
842                     myDataAdapter.Fill( Hat_iliski);
843
844                     if ( Hat_iliski.Tables[0].Rows.Count > 0)
845                     {
846
847                         DataView view = new DataView( Hat_iliski.Tables[0]);
848                         view.Sort = " Hat2_DURAK_SIRA DESC";
849
850                         stralt Hat = Hat_iliski.Tables[0].Rows[0]["ALT_ Hat_NO2"].
↪ ToString() + ",";
851                         for (int b = 0; b < Hat_iliski.Tables[0].Rows.Count; b++)
852                         {
853                             if (b != 0 && Hat_iliski.Tables[0].Rows[b]["ALT_
↪ Hat_NO2"] != Hat_iliski.Tables[0].Rows[b - 1]["ALT_ Hat_NO2"])
854                                 stralt Hat += Hat_iliski.Tables[0].Rows[b]["ALT_
↪ Hat_NO2"].ToString() + ",";
855                         }
856
857                         stralt Hat = stralt Hat.Substring(0, stralt Hat.Length - 1);
858
859                         foreach (DataRow row in view.ToTable().Rows)
860                         {
861
862                             sorgu = "select h.ANA_ Hat_NO1,h.ALT_ Hat_NO1,h.
↪ DURAK_NO1,h. Hat1_DURAK_SIRA,h.ANA_ Hat_NO2,h.ALT_ Hat_NO2,h.
↪ DURAK_NO2,h. Hat2_DURAK_SIRA,h.MESAFE2,(h. Hat1_DURAK_SIRA
↪ -0)+(300-h. Hat2_DURAK_SIRA) HS ";
863                             sorgu += "from Hat_iliski h where ";
864                             sorgu += "h.ANA_ Hat_NO1=" + Hat_iliski.Tables[0].
↪ Rows[0]["ANA_ Hat_NO2"] + " and ";
865                             sorgu += "h.ALT_ Hat_NO1 in (" + stralt Hat + " ) and
```

```
↪ ";
866                         sorgu += "h.ANA_ Hat_NO2=" +  Hat_iliski3 Hat.
↪ Tables[0].Rows[id]["AKTARMA_ Hat"] + " and ";
867                         sorgu += "h.ALT_ Hat_NO2 in (" + stralt Hat + ") and
↪ ";
868                         sorgu += "h. Hat1_DURAK_SIRA>" + row["
↪ Hat2_DURAK_SIRA"] + " ";
869                         sorgu += " order by h.MESAFE2, HS limit 2";
870
871                         temp.Clear();
872                         myDataAdapter.SelectCommand = new MySqlCommand(
↪ sorgu, myConn);
873                         myDataAdapter.Fill(temp);
874
875                         if (temp.Tables[0].Rows.Count > 0)
876                         {
877                             DURAK_sira = row[" Hat2_DURAK_SIRA"].ToString()
↪ ;
878                             durakno_2 = row["DURAK_NO2"].ToString();
879                             break;
880                         }
881                     }
882
883                     say++;
884                     bul = 0;
885                     if (temp.Tables[0].Rows.Count > 0)
886                     {  fnks_AracinDuragaGelmeSaatiVeHedefDuragaGitmeSaati(
↪ cikis Hatlar.Tables[0].Rows[i]["ANA_ Hat_NO"].ToString(), cikis Hatlar.
↪ Tables[0].Rows[i]["ALT_ Hat_NO"].ToString(), cikis Hatlar.Tables[0].Rows[i]["
↪ DURAK_NO"].ToString(),  Hat_iliski.Tables[0].Rows[0]["DURAK_NO1"].
↪ ToString(), yolculukBaslamaSaati, hesaplamaGun, out Yolculuk_
↪ Hat1_bitis_saat, out Yolculuk_ Hat1_bitis_saat, out alt Hatbul);
887
888                         for (int f = 0; f <  Hat_iliski.Tables[0].Rows.Count; f++)
889                         {
890
891                             for (int y = 0; y < temp.Tables[0].Rows.Count; y++)
892                             {
893                                 if (cozum > MAX_COZUM_SAYISI) break;
894
895
896
897                                 sorgu = "select h.ANA_ Hat_NO1,h.ALT_ Hat_NO1
```

```
    ↪ ,h.DURAK_NO1,h. Hat1_DURAK_SIRA,h.ANA_ Hat_NO2,h.ALT_ Hat_NO2
    ↪ ,h.DURAK_NO2,h. Hat2_DURAK_SIRA,h.MESAFE2,(h.
    ↪ Hat1_DURAK_SIRA−0)+(300−h. Hat2_DURAK_SIRA) HS ";
898                             sorgu += "from  Hat_iliski h where ";
899                             sorgu += "h.ANA_ Hat_NO1=" +  Hat_iliski.
    ↪ Tables[0].Rows[f]["ANA_ Hat_NO2"] + " and ";
900                             sorgu += "h.ALT_ Hat_NO1 =" +  Hat_iliski.
    ↪ Tables[0].Rows[f]["ALT_ Hat_NO2"] + "  and ";
901                             sorgu += "h.ANA_ Hat_NO2=" + temp.Tables[0].
    ↪ Rows[y]["ANA_ Hat_NO2"] + " and ";
902                             sorgu += "h.ALT_ Hat_NO2 in (" + temp.Tables[0].
    ↪ Rows[y]["ALT_ Hat_NO2"] + ") and ";
903                             sorgu += "h. Hat1_DURAK_SIRA>" +
    ↪ DURAK_sira + " ";
904                             sorgu += " and h. Hat1_DURAK_SIRA<=" +
    ↪ temp.Tables[0].Rows[y]["  Hat1_DURAK_SIRA"] + " ";
905                             sorgu += " order by h.MESAFE2, HS limit 2";
906
907                             aktarma  Hatti.Clear();
908                             myDataAdapter.SelectCommand = new
    ↪ MySqlCommand(sorgu, myConn);
909                             myDataAdapter.Fill(aktarma  Hatti);
910
911                             if (aktarma  Hatti.Tables[0].Rows.Count > 0)
912                             {
913                                 for (int c = 0; c < aktarma  Hatti.Tables[0].Rows.
    ↪ Count; c++)
914                                 {
915                                     temp2.Clear();
916                                     sorgu = "SELECT ANA_ Hat_NO,ALT_
    ↪ Hat_NO,SIRA,S1.DURAK_NO ";
917                                     sorgu += " from  Hat_durak_sira S1,
    ↪ DURAKLAR D1 where  S1.DURAK_NO=D1.DURAK_NO ";
918                                     sorgu += " and S1.ANA_ Hat_NO=" +
    ↪ Hat_iliski3  Hat.Tables[0].Rows[id]["VARIS_ Hat"].ToString() + " ";
919                                     sorgu += " order by distanceyeni('" +
    ↪ varisEnlem + "','" + varisBoylam + "' ,d1.ENLEM ,d1.BOYLAM)  limit 2";
920                                     myDataAdapter.SelectCommand = new
    ↪ MySqlCommand(sorgu, myConn);
921                                     myDataAdapter.Fill(temp2);
922
923                                     for (int m = 0; m < temp2.Tables[0].Rows.
    ↪ Count; m++)
```

```
924                                     {
925                                         sorgu = "select h.ANA_ Hat_NO1,h.ALT_
    ↪ Hat_NO1,h.DURAK_NO1,h. Hat1_DURAK_SIRA,h.ANA_ Hat_NO2,h.ALT_
    ↪ Hat_NO2,h.DURAK_NO2,h. Hat2_DURAK_SIRA,h.MESAFE2,(h.
    ↪ Hat1_DURAK_SIRA-0)+(300-h. Hat2_DURAK_SIRA) HS ";
926                                         sorgu += "from Hat_iliski h where ";
927                                         sorgu += "h.ANA_ Hat_NO1=" +
    ↪ aktarma Hatti.Tables[0].Rows[c]["ANA_ Hat_NO2"] + " and ";
928                                         sorgu += "h.ALT_ Hat_NO1 =" +
    ↪ aktarma Hatti.Tables[0].Rows[c]["ALT_ Hat_NO2"] + " and ";
929                                         sorgu += "h.ANA_ Hat_NO2=" + temp2.
    ↪ Tables[0].Rows[m]["ANA_ Hat_NO"] + " and ";
930                                         sorgu += "h. Hat1_DURAK_SIRA>" +
    ↪ aktarma Hatti.Tables[0].Rows[c][" Hat2_DURAK_SIRA"] + " ";
931                                         sorgu += " and h. Hat2_DURAK_SIRA
    ↪ <=" + temp2.Tables[0].Rows[m]["SIRA"] + " ";
932                                         sorgu += " order by h.MESAFE2, HS limit
    ↪ 2";
933                                         Dorduncu_ Hat.Clear();
934                                         myDataAdapter.SelectCommand = new
    ↪ MySqlCommand(sorgu, myConn);
935                                         myDataAdapter.Fill(Dorduncu_ Hat);
936
937                                         if (Dorduncu_ Hat.Tables[0].Rows.Count >
    ↪ 0)
938                                         {
939                                             bul = 0;
940                                             for (int ix = 0; ix < aktarmaDs.Tables
    ↪ [0].Rows.Count; ix++)
941                                             {
942                                                 if (aktarmaDs.Tables[0].Rows[ix]["
    ↪ ANA_ Hat1"].ToString() == cikis Hatlar.Tables[0].Rows[i]["ANA_ Hat_NO
    ↪ "].ToString() + "-" + Hat_iliski.Tables[0].Rows[0]["ANA_ Hat_NO2"].
    ↪ ToString() + "-" + aktarma Hatti.Tables[0].Rows[c]["ANA_ Hat_NO2"].
    ↪ ToString() + "-" + Dorduncu_ Hat.Tables[0].Rows[0]["ANA_ Hat_NO2"].
    ↪ ToString())
943                                                 {
944                                                     bul = 1;
945                                                     break;
946                                                 }
947                                             }
948
949                                             if (bul == 0)
```

```
950                                         {
951                                             cozum++;
952                                             SonucInt =
    ↪ fnks_AracinDuragaGelmeSaatiVeHedefDuragaGitmeSaati(aktarma_Hatti.
    ↪ Tables[0].Rows[c]["ANA_Hat_NO1"].ToString().ToString(), aktarma_Hatti.
    ↪ Tables[0].Rows[c]["ALT_Hat_NO1"].ToString(), durakno_2, aktarma_Hatti.
    ↪ Tables[0].Rows[c]["DURAK_NO1"].ToString(), Yolculuk_Hat1_bitis_saat,
    ↪ hesaplamaGun, out Yolculuk_Hat2_baslama_saat, out Yolculuk_
    ↪ Hat2_bitis_saat, out alt_Hatbul);
953
954                                             if (SonucInt == -1)
955                                             { continue; }
956
957
958                                             SonucInt =
    ↪ fnks_AracinDuragaGelmeSaatiVeHedefDuragaGitmeSaati(aktarma_Hatti.
    ↪ Tables[0].Rows[c]["ANA_Hat_NO2"].ToString(), aktarma_Hatti.Tables[0].
    ↪ Rows[c]["ALT_Hat_NO2"].ToString(), aktarma_Hatti.Tables[0].Rows[0]["
    ↪ DURAK_NO2"].ToString(), Dorduncu_Hat.Tables[0].Rows[0]["DURAK_NO1
    ↪ "].ToString(), Yolculuk_Hat2_bitis_saat, hesaplamaGun, out Yolculuk_
    ↪ Hat3_baslama_saat, out Yolculuk_Hat3_bitis_saat, out alt_Hatbul);
959                                             if (SonucInt == -1)
960                                             { continue; }
961
962                                             SonucInt =
    ↪ fnks_AracinDuragaGelmeSaatiVeHedefDuragaGitmeSaati(Dorduncu_Hat.
    ↪ Tables[0].Rows[0]["ANA_Hat_NO2"].ToString(), Dorduncu_Hat.Tables[0].
    ↪ Rows[0]["ALT_Hat_NO2"].ToString(), Dorduncu_Hat.Tables[0].Rows[0]["
    ↪ DURAK_NO2"].ToString(), temp2.Tables[0].Rows[m]["DURAK_NO"].ToString
    ↪ (), Yolculuk_Hat3_bitis_saat, hesaplamaGun, out Yolculuk_
    ↪ Hat4_baslama_saat, out Yolculuk_Hat4_bitis_saat, out alt_Hatbul);
963                                             if (SonucInt == -1)
964                                             { continue; }
965
966                                             gecenZaman = DateTime.Now -
    ↪ zamantamami;
967                                              aktarmaDs.Tables[0].Rows.Add(cikis
    ↪  Hatlar.Tables[0].Rows[i]["ANA_Hat_NO"].ToString() + "-" +  Hat_iliski.
    ↪ Tables[0].Rows[0]["ANA_Hat_NO2"].ToString() + "-" + aktarma_Hatti.
    ↪ Tables[0].Rows[c]["ANA_Hat_NO2"].ToString() + "-" + Dorduncu_Hat.
    ↪ Tables[0].Rows[0]["ANA_Hat_NO2"].ToString());
968
969                                             string CIKIS_DURAK_NO2,
```

```
↪ VARIS_DURAK_NO2, CIKIS_DURAK_NO3, VARIS_DURAK_NO3,
↪ CIKIS_DURAK_NO4, VARIS_DURAK_NO4, ARAC_TURU1, ARAC_TURU2,
↪ ARAC_TURU3, ARAC_TURU4;
970
971                                         string CIKIS_DURAK_NO1 = cikis
↪ Hatlar.Tables[0].Rows[i]["DURAK_NO"].ToString();
972                                         string VARIS_DURAK_NO1 =
↪ Hat_iliski.Tables[0].Rows[0]["DURAK_NO1"].ToString();
973
974                                         if ( Hat_iliski.Tables[0].Rows[0]["
↪ ANA_ Hat_NO1"].ToString() == "10") { ARAC_TURU1 = "T"; } else {
↪ ARAC_TURU1 = "O"; }
975                                         CIKIS_DURAK_NO2 = durakno_2;
976                                         VARIS_DURAK_NO2 = temp.Tables
↪ [0].Rows[0]["DURAK_NO1"].ToString();
977
978                                         if ( Hat_iliski.Tables[0].Rows[0]["
↪ ANA_ Hat_NO2"].ToString() == "10") { ARAC_TURU2 = "T"; } else {
↪ ARAC_TURU2 = "O"; }
979                                         CIKIS_DURAK_NO3 = temp.Tables
↪ [0].Rows[y]["DURAK_NO2"].ToString();
980                                         VARIS_DURAK_NO3 = Dorduncu_
↪ Hat.Tables[0].Rows[0]["DURAK_NO1"].ToString();
981
982                                         if (aktarma Hatti.Tables[0].Rows[c]["
↪ ANA_ Hat_NO2"].ToString() == "10") { ARAC_TURU3 = "T"; } else {
↪ ARAC_TURU3 = "O"; }
983                                         CIKIS_DURAK_NO4 = Dorduncu_
↪ Hat.Tables[0].Rows[0]["DURAK_NO2"].ToString();
984                                         VARIS_DURAK_NO4 = temp2.Tables
↪ [0].Rows[m]["DURAK_NO"].ToString();
985
986                                         if (Dorduncu_ Hat.Tables[0].Rows
↪ [0]["ANA_ Hat_NO2"].ToString() == "10") { ARAC_TURU4 = "T"; } else {
↪ ARAC_TURU4 = "O"; }
987
988                                         Yolculuk_ Hat1 = Hat_iliski.Tables
↪ [0].Rows[0]["ANA_ Hat_NO1"].ToString();
989                                         Yolculuk_Alt Hat1 = cikis Hatlar.
↪ Tables[0].Rows[i]["ALT_ Hat_NO"].ToString();
990                                         Yolculuk_ Hat1_baslama_durak =
↪ CIKIS_DURAK_NO1;
991                                         Yolculuk_ Hat1_bitis_durak =
```

```
↪ VARIS_DURAK_NO1;
992                                                 DuraklariSec2(CIKIS_DURAK_NO1,
↪ VARIS_DURAK_NO1, Yolculuk_ Hat1, Yolculuk_Alt Hat1, out
↪ CIKIS_DURAK_NO1, out Yolculuk_ Hat1_baslama_durak_sira, out Yolculuk_
↪ Hat1_baslama_durak_yon, out VARIS_DURAK_NO1, out Yolculuk_
↪ Hat1_bitis_durak_sira);
993
994                                                 Yolculuk_ Hat2 = Hat_iliski.Tables
↪ [0].Rows[0]["ANA_ Hat_NO2"].ToString();
995                                                 Yolculuk_Alt Hat2 = Hat_iliski.
↪ Tables[0].Rows[0]["ALT_ Hat_NO2"].ToString();
996                                                 Yolculuk_ Hat2_baslama_durak =
↪ CIKIS_DURAK_NO2;
997                                                 Yolculuk_ Hat2_bitis_durak =
↪ VARIS_DURAK_NO2;
998                                                 DuraklariSec2(CIKIS_DURAK_NO2,
↪ VARIS_DURAK_NO2, Yolculuk_ Hat2, Yolculuk_Alt Hat2, out
↪ CIKIS_DURAK_NO2, out Yolculuk_ Hat2_baslama_durak_sira, out Yolculuk_
↪ Hat2_baslama_durak_yon, out VARIS_DURAK_NO2, out Yolculuk_
↪ Hat2_bitis_durak_sira);
999
1000                                                Yolculuk_ Hat3 = aktarma Hatti.
↪ Tables[0].Rows[c]["ANA_ Hat_NO2"].ToString();
1001                                                Yolculuk_Alt Hat3 = aktarma Hatti.
↪ Tables[0].Rows[c]["ALT_ Hat_NO2"].ToString();
1002                                                Yolculuk_ Hat3_baslama_durak =
↪ CIKIS_DURAK_NO3;
1003                                                Yolculuk_ Hat3_bitis_durak =
↪ VARIS_DURAK_NO3;
1004                                                DuraklariSec2(CIKIS_DURAK_NO3,
↪ VARIS_DURAK_NO3, Yolculuk_ Hat3, Yolculuk_Alt Hat3, out
↪ CIKIS_DURAK_NO3, out Yolculuk_ Hat3_baslama_durak_sira, out Yolculuk_
↪ Hat3_baslama_durak_yon, out VARIS_DURAK_NO3, out Yolculuk_
↪ Hat3_bitis_durak_sira);
1005
1006                                                Yolculuk_ Hat4 = Dorduncu_ Hat.
↪ Tables[0].Rows[0]["ANA_ Hat_NO2"].ToString();
1007                                                Yolculuk_Alt Hat4 = Dorduncu_ Hat
↪ .Tables[0].Rows[0]["ALT_ Hat_NO2"].ToString();
1008                                                Yolculuk_ Hat4_baslama_durak =
↪ CIKIS_DURAK_NO4;
1009                                                Yolculuk_ Hat4_bitis_durak =
↪ VARIS_DURAK_NO4;
```

```
1010                                                DuraklariSec2(CIKIS_DURAK_NO4,
    ↪ VARIS_DURAK_NO4, Yolculuk_ Hat4, Yolculuk_Alt Hat4, out
    ↪ CIKIS_DURAK_NO4, out Yolculuk_ Hat4_baslama_durak_sira, out Yolculuk_
    ↪ Hat4_baslama_durak_yon, out VARIS_DURAK_NO4, out Yolculuk_
    ↪ Hat4_bitis_durak_sira);
1011
1012                                                Yurume1_cikis_enlem_boylam =
    ↪ cikisEnlem + ";" + cikisBoylam;
1013                                                Yurume1_varis_enlem_boylam =
    ↪ durakenlemboylam(Yolculuk_ Hat1_baslama_durak, out Yolculuk_
    ↪ Hat1_baslama_durak_adi);
1014                                                Yurume2_cikis_enlem_boylam =
    ↪ durakenlemboylam(Yolculuk_ Hat1_bitis_durak, out Yolculuk_
    ↪ Hat1_bitis_durak_adi);
1015                                                Yurume2_varis_enlem_boylam =
    ↪ durakenlemboylam(Yolculuk_ Hat2_baslama_durak, out Yolculuk_
    ↪ Hat2_baslama_durak_adi);
1016                                                Yurume3_cikis_enlem_boylam =
    ↪ durakenlemboylam(Yolculuk_ Hat2_bitis_durak, out Yolculuk_
    ↪ Hat2_bitis_durak_adi);
1017                                                Yurume3_varis_enlem_boylam =
    ↪ durakenlemboylam(Yolculuk_ Hat3_baslama_durak, out Yolculuk_
    ↪ Hat3_baslama_durak_adi);
1018                                                Yurume4_cikis_enlem_boylam =
    ↪ durakenlemboylam(Yolculuk_ Hat3_bitis_durak, out Yolculuk_
    ↪ Hat3_bitis_durak_adi);
1019                                                Yurume4_varis_enlem_boylam =
    ↪ durakenlemboylam(Yolculuk_ Hat4_baslama_durak, out Yolculuk_
    ↪ Hat4_baslama_durak_adi);
1020                                                Yurume5_cikis_enlem_boylam =
    ↪ durakenlemboylam(Yolculuk_ Hat4_bitis_durak, out Yolculuk_
    ↪ Hat4_bitis_durak_adi);
1021                                                Yurume5_varis_enlem_boylam =
    ↪ varisEnlem + ";" + varisBoylam;
1022
1023
1024                                                donecekDs.Tables[0].Rows.Add(
1025                                                    sessionid,
1026                                                    cozum.ToString(),
1027                                                    "9",
1028                                                    "Y|" + ARAC_TURU1 + "|Y|" +
    ↪ ARAC_TURU2 + "|Y|" + ARAC_TURU3 + "|Y|" + ARAC_TURU4 + "|Y",
1029                                                     cikisEnlem + ";" + cikisBoylam
```

```
↪ + "|" + Yolculuk_ Hat1_baslama_durak + "|" + Yurume2_cikis_enlem_boylam
↪  + "|" + Yolculuk_ Hat2_baslama_durak + "|" +
↪ Yurume3_cikis_enlem_boylam + "|" + Yolculuk_ Hat3_baslama_durak + "|" +
↪  Yurume4_cikis_enlem_boylam + "|" + Yolculuk_ Hat4_baslama_durak + "|"
↪ + Yurume5_cikis_enlem_boylam,
1030                                                    "0|" + Yolculuk_
↪ Hat1_baslama_durak_adi + "|0|" + Yolculuk_ Hat2_baslama_durak_adi + "|0|"
↪  + Yolculuk_ Hat3_baslama_durak_adi + "|0|" + Yolculuk_
↪ Hat4_baslama_durak_adi + "|0",
1031                                                    "0|" + Yolculuk_
↪ Hat1_baslama_durak_yon + "|0|" + Yolculuk_ Hat2_baslama_durak_yon +
↪ "|0|" + Yolculuk_ Hat3_baslama_durak_yon + "|0|" + Yolculuk_
↪ Hat4_baslama_durak_yon + "|0",
1032                                                    "0|" + Yolculuk_
↪ Hat1_baslama_durak_sira + "|0|" + Yolculuk_ Hat2_baslama_durak_sira +
↪ "|0|" + Yolculuk_ Hat3_baslama_durak_sira + "|0|" + Yolculuk_
↪ Hat4_baslama_durak_sira + "|0",
1033                                                    Yurume1_varis_enlem_boylam +
↪ "|" + Yolculuk_ Hat1_bitis_durak + "|" + Yurume2_varis_enlem_boylam + "|"
↪  + Yolculuk_ Hat2_bitis_durak + "|" + Yurume3_varis_enlem_boylam + "|" +
↪ Yolculuk_ Hat3_bitis_durak + "|" + Yurume4_varis_enlem_boylam + "|" +
↪ Yolculuk_ Hat4_bitis_durak + "|" + varisEnlem + ";" + varisBoylam,
1034                                                    "0|" + Yolculuk_
↪ Hat1_bitis_durak_adi + "|0|" + Yolculuk_ Hat2_bitis_durak_adi + "|0|" +
↪ Yolculuk_ Hat3_bitis_durak_adi + "|0|" + Yolculuk_ Hat4_bitis_durak_adi +
↪ "|0",
1035                                                    "0|" + Yolculuk_
↪ Hat1_bitis_durak_sira + "|0|" + Yolculuk_ Hat2_bitis_durak_sira + "|0|" +
↪ Yolculuk_ Hat3_bitis_durak_sira + "|0|" + Yolculuk_ Hat4_bitis_durak_sira +
↪ "|0",
1036                                                    "0|" + Yolculuk_ Hat1 + "|0|" +
↪ Yolculuk_ Hat2 + "|0|" + Yolculuk_ Hat3 + "|0|" + Yolculuk_ Hat4 + "|0",
1037                                                    "0|" + fnks_ HatAdi(Yolculuk_
↪ Hat1, Yolculuk_Alt Hat1) + "|0|" + fnks_ HatAdi(Yolculuk_ Hat2,
↪ Yolculuk_Alt Hat2) + "|0|" + fnks_ HatAdi(Yolculuk_ Hat3, Yolculuk_Alt
↪ Hat3) + "|0|" + fnks_ HatAdi(Yolculuk_ Hat4, Yolculuk_Alt Hat4) + "|0",
1038                                                    "0|" + Yolculuk_Alt Hat1 + "|0|"
↪ + Yolculuk_Alt Hat2 + "|0|" + Yolculuk_Alt Hat3 + "|0|" + Yolculuk_Alt
↪ Hat4 + "|0",
1039                                                    "0|" + Yolculuk_ Hat1_bitis_saat
↪ + "|0|" + Yolculuk_ Hat2_baslama_saat + "|0|" + Yolculuk_
↪ Hat3_baslama_saat + "|0|" + Yolculuk_ Hat4_baslama_saat + "|0",
1040                                                    "0|" + Yolculuk_ Hat1_bitis_saat
```

```
↪ + "|0|" + Yolculuk_ Hat2_bitis_saat + "|0|" + Yolculuk_ Hat3_bitis_saat +
↪ "|0|" + Yolculuk_ Hat4_bitis_saat + "|0",
1041                                                    mesafehesapla(
↪ Yurume1_cikis_enlem_boylam, Yurume1_varis_enlem_boylam) + "|0|" +
↪ mesafehesapla(Yurume2_cikis_enlem_boylam, Yurume2_varis_enlem_boylam) +
↪ "|0|" + mesafehesapla(Yurume3_cikis_enlem_boylam,
↪ Yurume3_varis_enlem_boylam) + "|0|" + mesafehesapla(
↪ Yurume4_cikis_enlem_boylam, Yurume4_varis_enlem_boylam) + "|0|" +
↪ mesafehesapla(Yurume5_cikis_enlem_boylam, Yurume5_varis_enlem_boylam)
1042                                                    );
1043
1044                                                    if (cozum > MAX_COZUM_SAYISI)
↪ break;
1045
1046                                                }
1047                                            }
1048                                        }
1049                                    }
1050                                }
1051                            }
1052                        }
1053                    }
1054                }
1055            }
1056
1057        }
1058    }
1059 }
1060
1061
1062
1063 runningTime = sw.Elapsed.TotalSeconds.ToString();
1064 sw.Reset();
1065
1066 string cozumx = "";
1067 for (int f = 0; f < donecekDs.Tables.Count; f++)
1068 {
1069
1070     for (int t = 0; t < donecekDs.Tables[f].Rows.Count; t++)
1071     {
1072         cozumx += "COZUM = " + donecekDs.Tables[f].Rows[t]["COZUM
↪ "].ToString() + " , \n" +
1073                   "ADIM = " + donecekDs.Tables[f].Rows[t]["ADIM"].ToString()
```

```
↪ + " , \n" +
1074                    "ARAC_TIPI = " + donecekDs.Tables[f].Rows[t]["ARAC_TIPI
↪ "].ToString() + " , \n" +
1075                    "BASLAMA_DURAK = " + donecekDs.Tables[f].Rows[t]["
↪ BASLAMA_DURAK"].ToString() + " , \n" +
1076                    "BASLAMA_DURAK_SIRASI = " + donecekDs.Tables[f].Rows
↪ [t]["BASLAMA_DURAK_SIRASI"].ToString() + " , \n" +
1077                    "BITIS_DURAK = " + donecekDs.Tables[f].Rows[t]["
↪ BITIS_DURAK"].ToString() + " , " +
1078                    "BITIS_DURAK_SIRASI = " + donecekDs.Tables[f].Rows[t]["
↪ BITIS_DURAK_SIRASI"].ToString() + " , \n" +
1079                    " Hat_NO = " + donecekDs.Tables[f].Rows[t][" Hat_NO"].
↪ ToString() + " , \n" +
1080                    "ALT_ Hat_NO = " + donecekDs.Tables[f].Rows[t]["ALT_
↪ Hat_NO"].ToString() + " , \n" +
1081                    "BASLAMA_SAATI = " + donecekDs.Tables[f].Rows[t]["
↪ BASLAMA_SAATI"].ToString() + " , \n" +
1082                    "BITIS_SAATI = " + donecekDs.Tables[f].Rows[t]["
↪ BITIS_SAATI"].ToString() + " , \n" +
1083                    "MESAFE = " + donecekDs.Tables[f].Rows[t]["MESAFE"].
↪ ToString() + " , \n" +
1084
1085                }
1086            }
1087            nasilgiderimlogedkle(cikisKoordinat, varisKoordinat, runningTime, cozumx,
↪ sessionid);
1088
1089            return donecekDs;
1090
1091        }
```

Kod 5.5: Nasıl giderim sistem

Kaynaklar

[1] Hannah Bast, Daniel Delling, Andrew Goldberg, Matthias Müller-Hannemann, Thomas Pajor, Peter Sanders, Dorothea Wagner, and Renato F Werneck. Route planning in transportation networks. In *Algorithm engineering*, pages 19–80. Springer, 2016.

[2] Joe Naoum-Sawaya, Randy Cogill, Bissan Ghaddar, Shravan Sajja, Robert Shorten, Nicole Taheri, Pierpaolo Tommasi, Rudi Verago, and Fabian Wirth. Stochastic optimization approach for the car placement problem in ridesharing systems. *Transportation Research Part B: Methodological*, 80:173–184, 2015.

[3] Chong Chen, Ye Xu, Jincheng Shang, and Gordon Huang. Alternatives of strategic environmental assessment for road traffic development planning—case of changchun city, china. *Chinese Geographical Science*, 19(1):25–36, 2009.

[4] Feifei Li, Dihan Cheng, Marios Hadjieleftheriou, George Kollios, and Shang-Hua Teng. On trip planning queries in spatial databases. In *International symposium on spatial and temporal databases*, pages 273–290. Springer, 2005.

[5] Ming Zhu, Xiao-Yang Liu, Meikang Qiu, Ruimin Shen, Wei Shu, and Min-You Wu. Traffic big data based path planning strategy in public vehicle systems. In *2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS)*, pages 1–2. IEEE, 2016.

[6] Sinem Bozkurt, Ahmet Yazici, and Kemal Keskin. A multicriteria route planning approach considering driver preferences. In *2012 IEEE International Conference on Vehicular Electronics and Safety (ICVES 2012)*, pages 324–328. IEEE, 2012.

[7] Elise D Miller, Hani S Mahmassani, and Athanasios Ziliaskopoulos. Path search techniques for transportation networks with time-dependent, stochastic arc costs. In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, volume 2, pages 1716–1721. IEEE, 1994.

[8] Yves Rochat and Éric D Taillard. Probabilistic diversification and intensification in local search for vehicle routing. *Journal of heuristics*, 1(1):147–167, 1995.

[9] Dieter Jungnickel and D Jungnickel. *Graphs, networks and algorithms*. Springer, 2005.

[10] Ronald Cohn Jesse Russell. "graph theory",translation by janne tamminen. *Ruohonen K.*, 1, 2013.

[11] J. Russell and R. Cohn. *Dijkstra's Algorithm*. Book on Demand, 2012.

[12] Wei Zeng and Richard L Church. Finding shortest paths on real road networks: the case for a. *International journal of geographical information science*, 23(4):531–543, 2009.

[13] Akhil Kaushik et al. Extended bellman ford algorithm with optimized time of computation. In *Proceedings of International Conference on ICT for Sustainable Development*, pages 241–247. Springer, 2016.

[14] Baruch Awerbuch, Amotz Bar-Noy, and Madan Gopal. Approximate distributed bellman-ford algorithms. *IEEE Transactions on Communications*, 42(8):2515–2517, 1994.

[15] ROBERT W. FLOYD. Algorithm 97 shortest path. *2nd Edition*, (3), 1995.

[16] Andrew V Goldberg and Chris Harrelson. Computing the shortest path: A search meets graph theory. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 156–165. Society for Industrial and Applied Mathematics, 2005.

[17] Andrew Goldberg and Renato Werneck. Computing point-to-point shortest paths from external memory, March 2 2006. US Patent App. 11/115,558.

[18] Ali AKAY. Toplu taşıma ağları üzerinde güzergah optimizasyonu problemine en az aktarım odaklı bir çözüm yaklaşımı. pages 8–44, 2015.

[19] Liang Zhao, Tatsuya Ohshima, and Hiroshi Nagamochi. A* algorithm for the time-dependent shortest path problem. In *WAAC08: The 11th Japan-Korea Joint Workshop on Algorithms and Computation*, 2008.

[20] Giacomo Nannicini, Daniel Delling, Dominik Schultes, and Leo Liberti. Bidirectional a* search on time-dependent road networks. *Networks*, 59(2):240–251, 2012.

[21] Frank Schulz, Dorothea Wagner, and Karsten Weihe. Dijkstra's algorithm on-line: an empirical case study from public railroad transport. *Journal of Experimental Algorithmics (JEA)*, 5:12, 2000.

[22] Dorothea Wagner, Thomas Willhalm, and Christos Zaroliagis. Dynamic shortest paths containers. *Electronic Notes in Theoretical Computer Science*, 92:65–84, 2004.

[23] Dorothea Wagner, Thomas Willhalm, and Christos Zaroliagis. Geometric containers for efficient shortest-path computation. *Journal of Experimental Algorithmics (JEA)*, 10:1–3, 2005.

[24] Moritz Hilger, Ekkehard Köhler, Rolf H Möhring, and Heiko Schilling. Fast point-to-point shortest path computations with arc-flags. *The Shortest Path Problem: Ninth DIMACS Implementation Challenge*, 74:41–72, 2009.

[25] Peter Sanders and Dominik Schultes. Highway hierarchies hasten exact shortest path queries. In *European Symposium on Algorithms*, pages 568–579. Springer, 2005.

[26] Foo Hee Meng, Y Lao, Leong Hon Wai, and Lau Hoong Chuin. A multi-criteria, multi-modal passenger route advisory system. In *Proceedings*, 1999.

[27] Matthias Müller-Hannemann, Frank Schulz, Dorothea Wagner, and Christos Zaroliagis. Timetable information: Models and algorithms. In *Algorithmic Methods for Railway Optimization*, pages 67–90. Springer, 2007.

[28] Daniel Delling, Peter Sanders, Dominik Schultes, and Dorothea Wagner. Engineering route planning algorithms. In *Algorithmics of large and complex networks*, pages 117–139. Springer, 2009.

[29] Evangelia Pyrga, Frank Schulz, Dorothea Wagner, and Christos Zaroliagis. Efficient models for timetable information in public transportation systems. *Journal of Experimental Algorithmics (JEA)*, 12:2–4, 2008.

[30] Daniel Delling, Thomas Pajor, and Renato F Werneck. Round-based public transit routing. *Transportation Science*, 49(3):591–604, 2014.

[31] Julian Dibbelt, Thomas Pajor, Ben Strasser, and Dorothea Wagner. Intriguingly simple and fast transit routing. In *International Symposium on Experimental Algorithms*, pages 43–54. Springer, 2013.

[32] Gerth Stølting Brodal and Riko Jacob. Time-dependent networks as models to achieve fast exact time-table queries. *Electronic Notes in Theoretical Computer Science*, 92:3–15, 2004.

[33] Chris Barrett, Keith Bisset, Martin Holzer, Goran Konjevod, Madhav Marathe, and Dorothea Wagner. Implementations of routing algorithms for transportation networks. In *DIMACS Workshop on Shortest-Path Challenge*, 2006.

[34] Maurizio Bielli, Azedine Boulmakoul, and Hicham Mouncif. Object modeling and path computation for multimodal travel systems. *European Journal of Operational Research*, 175(3):1705–1730, 2006.

[35] Gernot Veit Batz, Robert Geisberger, and Peter Sanders. Time dependent contraction hierarchies-basic algorithmic ideas. Technical report, Citeseer, 2008.

[36] Jolanta Koszelew. Two methods of quasi-optimal routes generation in public transportation network. In *2008 7th Computer Information Systems and Industrial Management Applications*, pages 231–236. IEEE, 2008.

[37] Pierre Hansen. Bicriterion path problems. In *Multiple criteria decision making theory and application*, pages 109–127. Springer, 1980.

[38] Jerald Jariyasunant, Eric Mai, and Raja Sengupta. Algorithm for finding optimal paths in a public transit network with real-time data. *Transportation Research Record*, 2256(1):34–42, 2011.

[39] S Meena Kumari and N Geethanjali. A survey on shortest path routing algorithms for public transport travel. *Global Journal of Computer Science and Technology*, 9(5):73–76, 2010.

[40] Jianwei Zhang, Feixiong Liao, Theo Arentze, and Harry Timmermans. A multimodal transport network model for advanced traveler information systems. *Procedia-Social and Behavioral Sciences*, 20:313–322, 2011.

[41] Christian Artigues, Marie-José Huguet, Fallou Gueye, Frédéric Schettini, and Laurent Dezou. State-based accelerations and bidirectional search for bi-objective multi-modal shortest paths. *Transportation Research Part C: Emerging Technologies*, 27:233–259, 2013.