KOCAELİ ÜNİVERSİTESİ * FEN BİLİMLERİ ENSTİTÜSÜ

GEZGİN SATICI PROBLEMİ İÇİN VERİ MADENCİLİĞİ TABANLI SEZGİSEL BİR YAKLAŞIM

YÜKSEK LİSANS TEZİ

Semiye GÖNÜLOL

Ana Bilim Dalı: Endüstri Mühendisliği

Danışman: Prof. Dr. Alpaslan FIĞLALI

KOCAELİ,2009

KOCAELİ ÜNİVERSİTESİ * FEN BİLİMLERİ ENSTİTÜSÜ

GEZGİN SATICI PROBLEMİ İÇİN VERİ MADENCİLİĞİ TABANLI SEZGİSEL BİR YAKLAŞIM

YÜKSEK LİSANS TEZİ

Semiye GÖNÜLOL

Tezin Enstitüye Verildiği Tarih: 5 Aralık 2008

Tezin Savunulduğu Tarih: 14 Ocak 2009

Tez Danışmanı	Üye	Üye
Prof.Dr.Alpaslan FIĞLALI	Prof.Dr.Coşkun ÖZKAN	Doç.Dr.Ayhan DEMİRİZ
(A.Fıglalı)	(.....)	(A.Demiriz)

KOCAELİ, 2009

ÖNSÖZ VE TEŞEKKÜR

Veritabanı teknolojisindeki hızlı gelişmenin doğurduğu veri sayısındaki artışa rağmen, elde edilen anlamlı bilgi sayısında azalma olmaktadır. Veri Madenciliği(V.M) bu soruna cevap veren, büyük ölçekli veriler içinde saklı kalmış anlamlı bilgiye ulaşmak için, sahip olunan verileri analiz ederek, yorumlama sürecidir. Gezgin Satıcı Problemi (G.S.P.) ise, belirlenen sayıda şehri, her şehre yalnız bir kez uğramak şartıyla gezecek ve başlangıç şehrine geri dönecek minimum yolu bulma olarak tanımlanır. G.S.P., optimizasyon problemlerinden üzerinde en geniş çalışılmış ve en zor problemlerinden bir tanesidir. Söz konusu problemi çözmek amacıyla birçok algoritma ve sezgisel yaklaşım geliştirilmiştir. Bu çalışma da literatürde iyi bilinen 15 test problemi için veri madenciliği tabanlı yeni bir yaklaşım denenmiştir. Bulunan sonuçlara yerel arama yapılarak çözüm performansları arttırılmaya çalışılmıştır.

Yüksek lisans tez çalışmalarım süresince değerli zamanını benden esirgemeyen, bilgi ve tecrübesi ile her konuda bana yön gösteren ve yardımcı olan danışman hocam Kocaeli Üniversitesi Endüstri Mühendisliği Bölüm Başkanı Prof. Dr. Alpaslan FIĞLALI'ya emeklerinden dolayı teşekkür ederim.

Tezimdeki algoritmanın Matlab programlama dilinde geliştirilmesi aşamasında, benim için zaman ayırıp yardımlarını esirgemeyen Arş. Gör. Ümit TERZİ'ye ve Ahmet CİHAN'a teşekkür ederim.

Ayrıca TÜBİTAK'a teşekkürü bir borç bilirim.

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SEMBOLLER

d	:i ile j şehri arasındaki mesafe, (br.)
n	:Şehir sayısı, (adet)
S	:standart sapma, (br.)
Z	:maliyet,
$\bar{x}$	:örnek ortalaması, (br.)

Alt İndisler

ij	:i. ve j. şehir

Kısaltmalar

G.A.	: Genetik Algoritma
G.S.P.	: Gezgin Satıcı Problemi
S.G.S.P.	: Simetrik Gezgin Satıcı Problemi
S.T.S.P.	: Symetric Travelling Salesman Problem
T.S.P.	: Travelling Salesman Problem
V.M.	: Veri Madenciliği
V.M.T.Y.	: Veri Madenciliği Tabanlı Yaklaşım
Y.A.	: Yerel arama

GEZGİN SATICI PROBLEMİ İÇİN VERİ MADENCİLİĞİ TABANLI SEZGİSEL BİR YAKLAŞIM

Semiye GÖNÜLOL

Anahtar Kelimeler: Veri Madenciliği(V.M), Gezgin Satıcı Problemi(G.S.P.), Yerel Arama

Özet: Bu tez çalışmasında veri madenciliği yardımıyla, birçok alanda yaygın olarak kullanılan Gezgin Satıcı Probleminin (G.S.P.) çözümü üzerinde durulmuştur. Bu çalışma da amaç, veri madenciliğinin, G.S.P. üzerinde nasıl performans göstereceğini araştırmaktır. Bu kapsamda Simetrik Gezgin Satıcı Probleminde(S.G.S.P.), literatürde iyi bilinen bazı test problemi için uygun parametreler kullanılarak, rassal üretilen verilerle veri madenciliği yaklaşımı denenmiştir. Bulunan en iyi çözümler, bu çözümlerin süreleri ve optimumdan sapmaları belirlenmiştir. Veri madenciliği yaklaşımı ile bulunan en iyi çözümlere yerel arama uygulanarak çözüm performansı arttırılmaya çalışılmıştır. Elde edilen sonuçlar, ilgili problemin optimum sonuçları ile karşılaştırılmıştır. Veri madenciliği yaklaşımının, 300 şehirden az şehirli tüm gezgin satıcı problemlerinde oldukça iyi sonuçlar verdiği görülmüştür.

A HEURISTIC APPROACH BASED ON DATA MINING FOR TRAVELLING SALESMAN PROBLEM

Semiye GÖNÜLOL

Keywords: Data Mining, Travelling Salesman Problem, Local Search.

Abstract: This thesis observed to solve travelling salesman problem with data mining method.

This work has been made to examine;how data mining methods will performance to solve T.S.Ps. Symmetric Travelling Sallesman Problems(S.T.S.P) has been tried to answered with data mining approach using random data. Suitable parameters were used on well known test problems. The best solutions which implemented local search, were found by data mining approach. Thus, the performance of the results are improved. The gathered results from our study, are compared with the optimum results of the related problem. It is seen that data mining approach gives good results at all travelling salesman problems which has less than 300 cities.

1.GİRİŞ

Dünyadaki veri miktarının her 20 ayda ikiye katlandığı tahmin edilmektedir (Javovic ve diğerleri, 2002). Veri tabanlarında ve veri ambarlarında biriken bu kadar verinin bilgiye nasıl dönüştürüleceği önemli bir sorun haline gelmiştir. Bilgisayar uygulamalarının yaygınlaşması ve kullanışlı veri toplama araçlarının gelişmesiyle, veri tabanlarında sürekli ve büyük miktarda veri toplanmış ve halen de toplanmaya devam etmektedir. Böylelikle işlenmediği sürece kıymetsiz gibi görünen veri yığınları oluşmaktadır. Bu veri yığınlarını, içlerinde altın madenleri bulunan dağlara benzetmek mümkündür. Bu madenlere ulaşmak için kullanılan yöntem ise, temelinde istatistik uygulamaları yatan "Veri Madenciliği"dir.

Veritabanı teknolojisindeki hızlı gelişmenin doğurduğu veri sayısındaki artışa rağmen, elde edilen anlamlı bilgi sayısında azalma olmaktadır. Veri Madenciliği(V.M) bu soruna cevap veren, büyük ölçekli veriler içinde saklı kalmış anlamlı bilgiye ulaşmak için, sahip olunan verileri analiz ederek, yorumlama sürecidir.

Hızla gelişen teknolojik donanımlara rağmen günümüzde hala çözüm süresi uzun olan karmaşık problem mevcuttur. G.S.P. , serimdeki bütün düğümlere bir gezgin tarafından yalnızca bir kez uğramayı sağlayan en kısa yolun belirlenmesini amaçlayan bir en iyileme problemidir Gezgin satıcı problemi kolay karakterize edilmesine rağmen çözümü çok zordur. G.S.P. , NP-zor problem sınıfına girmektedir. Bu nedenle çözmek için yeterli bir algoritmanın geliştirilmesi çok zordur. Bununla beraber sadeliğinden (basitliğinden) ötürü G.S.P., bu sınıfta üzerinde en çok çalışılan problemlerden biridir. Bu problemin en iyi çözümlerini bulabilmek için günümüze kadar farklı yaklaşımlar içeren optimizasyon modelleri ve çeşitli yöntemler geliştirilmiştir. Bunlardan bazıları tavlama benzetimi, tabu arama, yapay sinir ağları, genetik algoritma v.s. dir. Ama çözüm yöntemleri genellikle üstel çözüm süresi gerektirmektedir. Bu nedenle G.S.P.'nin en iyi çözümünün etkin olarak

bulunması büyük önem taşımaktadır. Problem çözme süresinin kısaltılmasının yanı sıra en iyi çözümün elde edilmesi için veri madenciliği ile geliştirilen algoritmanın rekabet edebilir bir seçenek olup olmadığı ve performansını gözlemlemek amacıyla bu konuda çalışılmıştır. Bu amaçla:

İkinci bölümde, veri madenciliğinin tarihsel gelişimine, tercih edilme nedenlerine, uygulama alanlarında karşılaşılan problemlere, veri madenciliği yazılımlarına, sürecine ve tekniklerine değinilmektedir.

Üçüncü bölümde, Euler ve Hamilton turlu problemlerinden, Gezgin Satıcı Probleminin(G.S.P.) temel özelliklerinden, tarihsel gelişiminden, uygulamalarından, G.S.P. için geliştirilen çözüm yöntemlerinden bahsedilmektedir.

Dördüncü bölümde, G.S.P. için veri madenciliği tabanlı sezgisel yeni yaklaşım anlatılmaktadır. Bu amaçla Matlab programlama dilinde kodlanan eşik hesaplama, rassal tur oluşturma, kural ilişki matrisi, tur uzunluğu hesaplama, yerel arama açıklanmakta ve bu kodlar seçilen 15 test problemine uygulanarak, sonuçları ve süreleri tablolarda verilmektedir.

Sonuç ve öneriler bölümünde ise, önerilen veri madenciliği tabanlı sezgisel yaklaşımın etkinliği irdelenerek, olası geliştirmeler üzerinde durulmaktadır. Yapılan programın kodu ve bilgisayar çıktısı ekler bölümünde verilmektedir.

2.VERİ MADENCİLİĞİ

2.1. Veri Madenciliği Tanımı

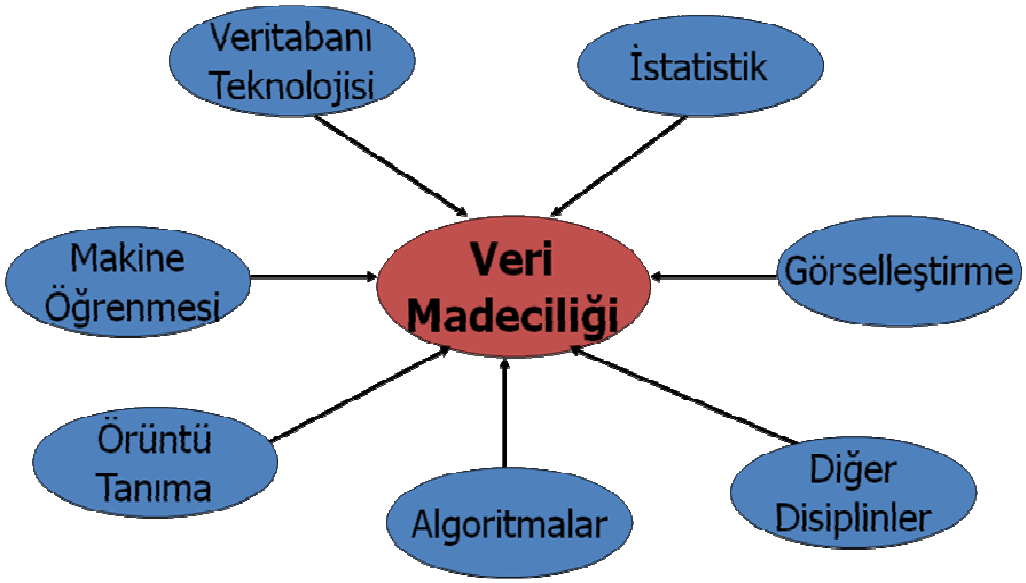
Veri madenciliğinin amacı veri yığınından bilgi elde etmektir. Elde edilen bu bilginin bazı "ideal" özelliklere sahip olması gerekir. Bu "ideal" özellikler bilginin doğru, anlaşılır ve ilginç olmasıdır. İlginçlikten kastedilen, keşfedilen bilginin kullanıcı için yeni, şaşırtıcı ve kullanışlı olmasıdır.(Freitas, 2002)

Veri madenciliği adını ve popülaritesini, saklanan verilerin bir "dağa" benzetilmesinden ve bu dağın içinde çok değerli taşların olmasından alır. Sorun ise, bu yığın içerisinde değersiz kayaların da bulunmasıdır ve değerli olanlara ulaşabilmek için değersizlerin ayıklanması gerektiğidir (Berson ve diğerleri, 2000). Ayrıca veri madenciliğinin etkili kullanımı ile projelerde maliyetler azaltılıp, gelirler artırılabilir (Javovic ve diğerleri, 2002).

Veri madenciliği büyük hacimli dallardaki örüntüleri araştıran matematiksel algoritmaları kullanır. Veri madenciliği hipotezleri keşfeder, sonuçları birleştirmek için insan yeteneğini kullanır (Davis,1999).

Veri madenciliği; çok çeşitli verilere dayanarak daha önce keşfedilmemiş bilgileri ortaya çıkarmak, karar vermek ve eylem planını gerçekleştirmek için kullanma sürecidir (Swift, 2001). Bu noktada kendi başına bir çözüm değil çözüme ulaşmak için verilecek karar sürecini destekleyen, problemi çözmek için gerekli olan bilgileri sağlamaya yarayan bir araçtır.

Veri madenciliği istatistiğin, yapay zekânın ve veri tabanlarında bilgi keşfinin karışımıdır(Pregibon,1996). Veri madenciliği şekil 2.1' de de görüldüğü üzere birçok teknolojinin bileşiminden oluşmaktadır.



Şekil 2.1:Veri madenciliğinde kullanılan teknolojiler

Sonuç olarak veri madenciliği, önceden bilinmeyen ilişkilerin bulunması için bugünün endüstrisinde yaratılan büyük miktarlardaki veriyi analiz eden bir yoldur. Yüksek güçlü bilgisayarlara ve gereken yazılımlara kolay ve düşük fiyatlarla ulaşılabilmesi bu teknolojinin işlemesini olanaklı kılmıştır. İnternet ise birçok noktadaki verinin toplanmasını sağlamaktadır (Akbulut, 2006).

Veri madenciliği tanımlarda öne çıkan noktalar şunlardır: Veri Madenciliği;

- Büyük ve karmaşık verilerle çalışır.
- Her türlü veriyi kullanarak çözümler üretebilir.
- İstatistik, yapay zekâ, makine öğrenmesi, veri tabanlarında bilgi keşfi, bilgisayar bilimi vb. gibi disiplinlerden faydalanır.
- Daha önceden bilinmeyen, doğrulanabilir, etkinleştirilebilir enformasyon arar.
- Otomatik veya yarı otomatik olarak çalışan çözüm araçları kullanır.
- Birçok endüstride kullanılmaktadır.
- Sorunlara göre değişen çözüm araçları vardır.
- Hızla büyümekte olan bir sektördür.

2.2. Veri Madenciliği Tarihsel Gelişimi

1960'lar da veri toplama sistemleri, enformasyon teknolojisi kullanılmaktaydı. 1960'lardan sonra gözle görülür gelişmeler meydana gelmiştir. 1970'ler de ise ilişkisel veri tabanları kullanılmaya başlanmış, 1980'lerde ise ilişkisel veri tabanları popüler olmaya başlamıştır. 1990 ve 2000'lerde çoklu ortam veritabanları ve web veritabanları, veri ambarlama ile veri madenciliğinin oluşması süreci devam etmektedir.

Tablo 2.1:Veri madenciliği gelişimi (Aldana, 2000)

Gelişim Adımları	**Cevaplanan Karar Problemi**	**Kullanılabilen Teknolojiler**	**Ürün Sağlayıcıları**	**Karakteristikler**
Veri toplama (1960'lar)	"Benim toplam karım geçen yılda ne kadar arttı?"	Bilgisayarlar, Teypler, Diskler	IBM,CDC	Geriye dönük,statik veri dağıtımı
Veri Erişimi (1980'ler)	"İngiltere'de geçen mart ayında birim satışları ne kadardı?"	İlişkisel veritabanları, SQL,ODBC.	Oracle,Sybase, Informix IBM,Microsoft,	Kayıt düzeyinde geriye dönük dinamik veri dağıtımı.
Veri Ambarlama ve Karar Destek Sistemleri (1990'lar)	"İngiltere'de geçen mart ayında birim satışları ne kadardı?"	OLAP,Çok boyutlu veritabanı sistemleri, Veri ambarları	Pilot,comshare,arbor, Cognos,microstrategy	Çoklu düzeylerde, geriye dönük dinamik veri dağıtımı
Veri Madenciliği (Bugün)	"Gelecek ay Boston'da ki birim satışlar muhtemelen ne olabilir, niçin?"	İleri düzeyde algoritmalar, çok işlemcili bilgisayarlar, büyük veritabanları	Pilot,Lockheed,IBM, SGI,SPSS Clementine, SAS,Microsoft v.s.	Geleceğe dönük,proaktif, enformasyon dağıtımı

Tablo 2.1 incelenirse 1960'lı yıllar ve öncesinde başlayan veri toplama ve veritabanı yaratma çalışmaları ilkel dosyalama işlemlerinden ibaretti. Cevaplanabilen karar problemleri ayrıntıya girmeden belirli bir döneme ilişkin problemler iken, kullanabilen teknolojiler bilgisayarlar, teypler ve diskler olmuştur. Bu yıllarda ürün sağlayıcı firmalar IBM ve CDC' dir ve geriye dönük, statik veri dağıtımı yapılmıştır. Bu yıllarda sadece geriye dönük aranılan veriye ulaşılırken bu verilerden enformasyon elde edilmediği görülmektedir (Bilen, 2004).

1980'li yıllara geçildiğinde statik veri dağıtımında dinamik veri dağıtımına geçildiği görülmektedir. İlişkisel veri tabanlarının oluşumu, SQL ve ODBC ile veri kaynaklarına ulaşım bu yıllarda gerçekleşmiştir. Ürün sağlayıcılardaki artışta dikkat çekicidir (Bilen, 2004).

1990'lı yıllarda veri toplama ve saklamadaki gelişimin sonucu olarak veri ambarları oluşturulmaya başlanmış ve karar destek sistemleri devreye girmiştir. Dinamik veri dağıtımı çoklu düzeylerde yapılmaya başlanmıştır. OLAP ve çok boyutlu veri tabanları göze batan değişimleridir (Bilen,2004).

Bugün ise veri madenciliği tam anlamı ile yapılmaya başlanmış olup geriye dönük yapılabilen veri dağıtımına ek olarak ileriye dönük tahminlere imkan veren proaktif enformasyon dağıtımı da yapılmaya başlanmıştır. Burada da ön plana çıkan değişim enformasyon dağıtımı olmuştur. 1960'lı yıllarda yalnızca istenilen verinin elde edilmesiyle sonuçlanan işlemler artık şimdi geleceğe dönük tahminler ve bu tahminlerin nedenlerinin açıklanmasına dönüşen işlemlere dönüşmüştür (Bilen,2004).

2.3. Veri Madenciliği'nin Tercih Edilme Nedenleri

Veri madenciliğine ilginin artması aşağıdaki faktörlerle açıklanabilir;

1980'ler de şirketler, müşterileri, rakipleri, ürünleri ile ilgili verilerden oluşan veri tabanları oluşturulmuşlardır. Bu veri tabanları potansiyel altın madeni gibidir. Sayısı milyonları geçen bu veriler gizli bilgiler içerirler ve bunlara kolaylıkla SQL veri tabanı sorgulama dili ya da başka yüzeysel sorgulama dilleri kullanılarak ulaşılabilir. SQL sadece bir sorgulama dilidir ve önceden bilinen sınırlamalar altında bilgileri bulmaya yardım eder (Uysal,1994). Veri madenciliği algoritmaları tipik olarak, veri tabanının alt gruplarında ya da uygun kümelerde belirginleşir. Birçok durumda, tekrarlanabilen SQL sorguları kullanılır ve ortalama sonuçlar elde edilir. Bunu elle yapmak mümkündür fakat oldukça yorucu ve uzun süren bir iştir.

Bilgisayarlarda ağ kullanımı gelişmeye devam etmektedir. Bu durumda veri tabanı ile bağlantı kurmak kolaylaşır. Böylece demografik verili dosya ile müşteri dosyası arasında bağlantı kurulabilir ve belirli popülasyon gruplarının kimliklerinin belirlenmesi sağlanabilir.

Son birkaç yılda makine öğrenimi teknikleri oldukça gelişmiştir. Sinir ağları, genetik algoritmalara ve diğer basit uygulanabilir öğrenme teknikleri veri tabanlarıyla ilginç bağlantılar kurmayı kolaylaştırır.

Müşteri ile hizmet veren arasındaki ilişki, kişisel bilgileri hizmet verenin masasındaki bilgisayardan merkezi bilgi sistemlerine gönderir. Pazarlamacılar ve sigorta da bu yeni kazanılan teknikleri kullanmak ister (Yalçıntaş, 2003).

2.4. Veri Madenciliği Uygulama Alanları

Günümüzde veri madenciliği farklı alanlarda uygulanmaya başlanmıştır. Bu uygulama alanları aşağıda verilmiştir.

Pazarlama:

- Müşteri segmentasyonunda,
- Müşterilerin demografik özellikleri arasındaki bağlantıların kurulmasında,
- Çeşitli pazarlama kampanyalarında,
- Mevcut müşterilerin elde tutulması için geliştirilecek pazarlama stratejilerinin oluşturulmasında,
- Çapraz satış analizlerinde,
- Müşteri değerlemesinde,
- Müşteri ilişkileri yönetiminde,
- Çeşitli müşteri analizlerinde,
- Satış tahminlerinde,
- Hile yoluyla suç işleyen müşterilerin saptanmasında
- Kaybedilen müşterilerin geri kazanılmasında
- Kaybedilebilecek urumda olan müşterilerin tespitinde

- Sepet analizleri yardımı ile marketlerde ürünlerin raflara dağılımının belirlenmesinde.

Bankacılık:

- Farklı finansal göstergeler arasındaki gizli korelasyonlarının bulunmasında,
- Kredi kartı dolandırıcılıklarının tespitinde
- Müşteri segmentasyonunda,
- Kredi taleplerinin değerlendirilmesinde,
- Usulsüzlük tespitinde,
- Risk analizlerinde,
- Risk yönetiminde,
- Stok tahmininde,
- Kar analizinde,
- Portföy yönetiminde.

Sigortacılık:

- Yeni poliçe talep edecek müşterilerin tahmin edilmesinde,
- Sigorta dolandırıcılıklarının tespitinde,
- Riskli müşteri tipinin belirlenmesinde.

Perakendecilik:

- Satış noktası veri analizlerinde,
- Alış-veriş sepeti analizlerinde,
- Tedarik ve mağaza yerleşim optimizasyonunda.

Borsa:

- Hisse senedi fiyat tahmininde,
- Genel piyasa analizlerinde,
- Alım-satım stratejilerinin optimizasyonunda.

Telekomünikasyon:

- Kalite ve iyileştirme analizlerinde,

- Hisse tespitlerinde,
- Hatların yoğunluk tahminlerinde,
- İletişim desenlerinin belirlenmesinde,
- Kaynakların daha iyi kullanılmasında,
- Servis kalitesinin arttırılmasında.

Sağlık ve İlaç:

- Test sonuçlarının tahmininde,
- Ürün geliştirmede,
- Tıbbi teşhiste,
- Tedavi sürecinin belirlenmesinde,
- DNA içerisinde genlerin sıralarının belirlenmesinde,
- Protein analizlerinin yapılmasında,
- Hastalık haritalarının hazırlanmasında,
- Hastalık tanılarında,
- Sağlık politikalarına yön verilmesinde.

Endüstri:

- Kalite kontrol analizlerinde
- Lojistikte,
- Üretim süreçlerinin optimizasyonunda.

Bilim ve Mühendislik:

- Ampirik veriler üzerinde modeller kurarak bilimsel ve teknik problemlerin çözümlenmesinde.

Web hizmetleri:

- Elektronik ticaret yapan firmalar için müşteri davranışlarının belirlenmesinde,
- Web sitesini ziyaret eden kullanıcının daha önceki davranışlarına göre yönlendirilmesinde,
- Web sitesi güvenliğinin sağlanmasında,
- Kullanıcı davranışlarına göre web sitesinin yenilenmesinde,

- Kullanıcı profilinin belirlenmesinde.

Aşağıdaki şekil 2.2 'de veri madenciliğinin sektörler bazında kullanımına ilişkin bir araştırmanın sonuçları yer almaktadır (Kdnugget,2005). Bu şekilde, araştırmaya katılan toplam 421 şirketin 51 adedinin bankacılık alanında veri madenciliğinin kullandığı görülmektedir.

Son 3 yıl içinde veri madenciliğinin uygulandığı alanlar	
Bankacılık (51)	12%
Bioteknoloji / Genetik (11)	3%
Kredi skorlama (35)	8%
CRM (52)	12%
Doğrudan pazarlama (34)	8%
e-Ticaret (11)	3%
Eğlence/ Müzik (4)	1%
Sahtekarlık tespiti (31)	7%
Şans oyunu (2)	0,01 %
Kamu uygulamaları (12)	3%
Sigortacılık (24)	6%
Yatırım / Hisse senedi (5)	1%
Junk email / Anti-spam (5)	1%
Sağlık/ İK (15)	4%
İmalat (19)	5%
Tıp/ Farmakoloji (12)	3%
Perakende (25)	6%
Bilim (17)	4%
Güvenlik / Anti-terörizm(5)	1%
Telekomünikasyon (23)	5%
Seyahat (8)	2%
Web (9)	2%
Diğer (11)	3%

Şekil 2.2: Veri Madenciliğinin Uygulandığı Alanlar(Kdnugget, 2005).

2.5. Veri Madenciliği Uygulamalarında Karşılaşılan Problemler

Veri madenciliği girdi olarak kullanılacak ham veriyi veritabanlarından alır. Bu da veri tabanlarının dinamik, eksiksiz, geniş ve net veri içermemesi durumunda sorunlar doğurur (Aydoğan, 2003).

Diğer sorunlar da verinin konu ile uyumsuzluğundan, veri tabanlarındaki bilgilerin, veri eklendikçe ya da silindikçe değişebilmesinden doğabilir. Veri tabanlarındaki eksik bilgi ve bu yanlışlardan dolayı veri madenciliği amacına tam olarak ulaşmayabilir. Bu bilgi yanlışlığı, ölçüm hatalarından, ya da öznel yaklaşımdan olabilir (Akbulut, 2006).

2.6. Veri Madenciliği Yazılımları

Farklı algoritmalara sahip ve farklı işletim sistemleri üzerinde çalışabilen birçok veri madenciliği yazılımı bulunmaktadır. Tablo 1.2. 'de bazı yazılım programları ve çalıştırabildikleri fonksiyonel özellikleri gösterilmektedir.

Tablo 2.2: Yazılım programları ve çalıştırabildikleri fonksiyonel özellikler

Ürün Adı	**IBM Intelligent Miner**	**Oracle Darwin**	**SAS Enterprise Miner**	**Angoss Knowledge Seeker**	**WEKA**	**SPSS Clementine**
Karar Ağacı	X	X	X	X	X	X
Sinir Ağları	X	X	X			X
Zaman Serileri	X		X			X
Tahmin	X	X	X	X	X	X
Kümeleme	X		X		X	X
Birliktelik	X		X		X	X
Görselleştirme	X	X	X	X	X	X

2.7. Veri Madenciliği Süreci

Ne kadar etkin olursa olsun, hiçbir veri madenciliği algoritmasının üzerinde inceleme yapılan işin ve verilerin özelliklerinin bilinmemesi durumunda fayda sağlanması mümkün değildir. Bu nedenle aşağıda tanımlanan tüm aşamalardan önce, iş ve veri özelliklerinin öğrenilmesi başarının ilk ve temel şartı olacaktır. Başarılı bir veri madenciliği projesinde izlenmesi gereken adımlar aşağıdadır:

1. Problemin tanımlanması,

2. Verilerin hazırlanması,
3. Modelin kurulması ve değerlendirilmesi,
4. Modelin kullanılması,
5. Modelin izlenmesi.

2.7.1. Problemin tanımlanması

Veri madenciliği çalışmalarında başarılı olmanın en önemli şartı, projenin hangi işletme amacı için yapılacağının ve elde edilecek sonuçların başarı düzeylerinin nasıl ölçüleceğinin tanımlanmasıdır. İlgili işletme amacı, işletme problemi üzerine odaklanmış ve açık bir dille ifade edilmiş olmalı. Ayrıca yanlış tahminlerde katlanılacak olan maliyetlere ve doğru tahminlerde kazanılacak faydalara ilişkin tahminlere de bu aşamada yer verilmelidir. Analistin, işletmede üretilen sayısal verilerin boyutlarını, proje için yeterlilik düzeyini ve iş süreçlerini iyi analiz etmesi gerekmektedir (Alataş ve Akın, 2004).

2.7.2. Verilerin hazırlanması

Veri madenciliğinin en önemli aşamalarından biri olan verinin hazırlanması aşaması, analistin toplam zaman ve enerjisinin %50 - %85 ini harcamasına neden olmaktadır (Piramuthu S.,1998). Bu aşamada firmanın mevcut bilgi sistemleri üzerinde ürettiği sayısal bilginin iyi analiz edilmesi, veriler ile mevcut iş problemi arasında ilişki olması gerektiği unutulmamalıdır. Proje kapsamında kullanılacak sayısal verilerin, hangi iş süreçleri ile yaratıldığı da bu veriler kullanılmadan analiz edilmelidir, bu sayede analist veri kalitesi hakkında fikir sahibi olabilir.

Verilerin hazırlanması aşaması kendi içerisinde toplama, birleştirme ve temizleme, dönüştürme adımlarından meydana gelmektedir (Alataş ve Akın, 2004).

2.7.3. Modelin kurulması ve değerlendirilmesi

Tanımlanan problem için en uygun modelin bulunabilmesi, olabildiğince çok sayıda modelin kurularak denenmesi ile mümkündür. Bu nedenle veri hazırlama ve model

kurma aşamaları, en iyi olduğu düşünülen modele varılıncaya kadar yenilenen bir süreçtir. Bir diğer önemli değerlendirme ölçütü ise modelin anlaşılabilir olmasıdır. Birçok işletme için uygulamasında ilgili kararın niçin verildiğinin yorumlanabilmesi çok daha büyük önem taşıyabilir.

2.7.4. Modelin kullanılması

Kurulan ve geçerliliği kabul edilen model doğrudan bir uygulama olabileceği gibi, bir başka uygulamanın alt parçası olarak kullanılabilir. Kurulan modeller risk analizi, kredi değerlendirme, dolandırıcılık tespiti gibi işletme uygulamalarında doğrudan kullanılabileceği gibi, promosyon planlaması simülasyonuna entegre edilebilir veya tahmini envanter düzeyleri yeniden sipariş noktasının altına düştüğünde, otomatik olarak sipariş verilmesini sağlayacak bir uygulamanın içine gömülebilir (Aydoğan, 2003).

2.7.5. Modelin izlenmesi

Zaman içerisinde bütün sistemlerin özelliklerinde ve dolayısıyla ürettikleri verilerde ortaya çıkan değişiklikler, kurulan modellerin sürekli olarak izlenmesini ve gerekiyorsa yeniden düzenlenmesini gerektirecektir. Tahmin edilen ve gözlenen değişkenler arasındaki farklılığı gösteren grafikler model sonuçlarının izlenmesinde kullanılan yararlı bir yöntemdir (Shearer, 2000).

2.8. Veri Madenciliği Teknikleri

Veri madenciliği tekniklerini işlevlerine göre 3 temel grupta toplanır:

- Sınıflama (Classification),
- Kümeleme (Clustering),
- Birliktelik kuralları ve sıralı örüntüler (Association rules and sequential patterns).

2.8.1. Sınıflama

Sınıflama, verinin önceden belirlenen çıktılara uygun olarak ayrıştırılmasını sağlayan bir tekniktir. Çıktılar, önceden bilindiği için sınıflama, veri kümesini denetimli olarak öğrenir. Örneğin; A finans hizmetleri şirketi; müşterilerinin yeni bir yatırım fırsatıyla ilgilenip ilgilenmediğini öğrenmek istemektedir. Daha önceden benzer bir ürün satmıştır ve geçmiş veriler hangi müşterilerin önceki teklife cevap verdiğini göstermektedir. Amaç; bu teklife cevap veren müşterilerin özelliklerini belirlemek, böylece pazarlama ve satış çalışmalarını daha etkin yürütmektir. Müşteri kayıtlarında müşterinin önceki teklife cevap verip vermediğini gösteren "evet"/ "hayır" şeklinde bir alan bulunur. Bu alan "hedef " ya da "bağımlı" değişken olarak adlandırılır. Amaç, müşterilerin diğer niteliklerinin (gelir düzeyi, iş türü, yaş, medeni durum, kaç yıldır müşteri olduğu, satın aldığı diğer ürün ve yatırım türleri) hedef değişken üzerindeki etkilerini analiz etmektir. Analizde yer alan diğer nitelikler "bağımsız" ya da " tahminci" değişken adını alır.

Sınıflama ve regresyon modellerinde kullanılan başlıca yöntemler,

- Karar Ağaçları (Decision Trees),
- Yapay Sinir Ağları (Artificial Neural Networks),
- Genetik Algoritmalar (Genetic Algorithms),
- K-En Yakın Komsu (K-Nearest Neighbor),
- Bellek Tabanlı Yöntemler (Memory Based Reasoning),
- Regresyondur (Akpınar, 2000).

2.8.1.1. Karar ağaçları

Karar ağaçları, yaygın olarak kullanılan sınıflama algoritmalarından biridir. Karar ağacı yapılarında, her düğüm bir nitelik üzerinde gerçekleştirilen testi, her dal bu testin çıktısını, her yaprak düğüm ise sınıfları temsil eder. En üstteki düğüm kök düğüm olarak adlandırılır. Karar ağaçları, kök düğümden yaprak düğüme doğru çalışır (Wei ve Chiu , 2002)

Geliştirilen karar ağacı algoritmalar içerisinde;

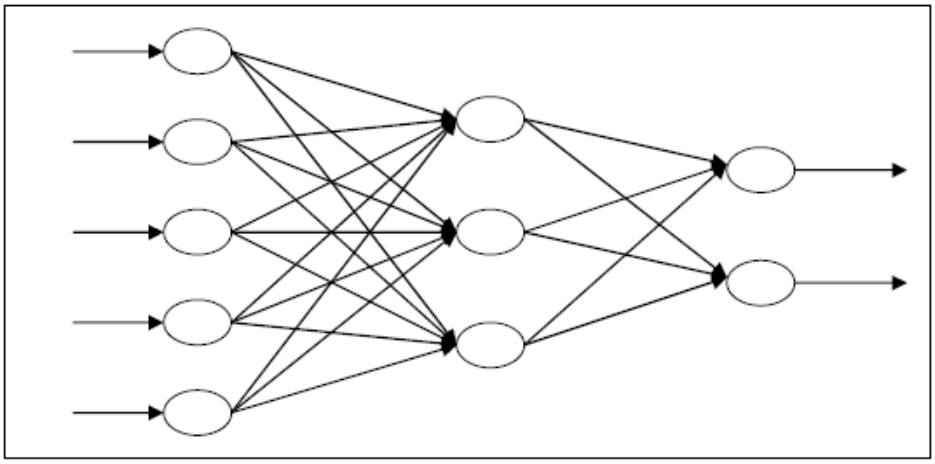
- CHAID (Chi- Squared Automatic Interaction Detector), C&RT (Classification and Regression Trees),
- ID3,
- Exhaustive CHAID,
- C4.5,
- MARS (Multivariate Adaptive Regression Splines),
- QUEST (Quick, Unbiased, Efficient Statistical Tree),
- C5.0,
- SLIQ (Supervised Learning in Quest),
- SPRINT (Scalable Parallelizable Induction of Decision Trees) başlıcalarıdır (Akpınar, 2000).

2.8.1.2. Yapay sinir ağları

Sinir ağları, insan beyninden esinlenilerek şekillendirilmiştir. İnsan beyni sinir hücresinden (neron) oluşur. Bu sinir hücreleri çok sayıda birleşme noktasına (synapse) sahiptir. Bir sinir hücresi diğer bir sinir hücresiyle bu birleşme noktaları yoluyla bağlantı kurar. Şekil 2.3'te de görüldüğü gibi insan beyninin karmaşık yapısı daha karmaşık öğrenme makinelerinin oluşturulmasında model olarak alınmış ve yapay sinir ağları olarak adlandırılmıştır. Yapay sinir ağları için farklı yapılar vardır ve bunların her biri verilen işleri yapmak için farklı yol ve öğrenme yöntemleri kullanırlar. Yapay sinir ağının, belirli bir işi yapmak için eğitildiği evreye şifreleme evresi, sınıflama ya da kestirim yapma evresine ise şifre çözme evresi adı verilir.

Yapay sinir ağları; sınıflama, kümeleme ve tahmin amaçları ile kolaylıkla kullanılabilecek genel amaçlı ve güçlü araçlardır. Ekonomik alanlardan tıbbi konulara, değerli müşterilerin belirlenmesi için yapılan kümeleme işlemlerinden kredi kartlarında sahtekârlıkların belirlenmesine kadar çok geniş bir alanda uygulanabilmektedir (Tantuğ, 2002). Sınıflama amaçlı kullanılan yapay sinir ağları, geri yayılım (backpropagation) algoritması ve RBF (Radial Basis Function) ağlarıdır.

Kümeleme amaçlı kullanılan yapay sinir ağları, 80'lerin basında Kohonen tarafından geliştirilen öz düzenlemeli haritadır.



Şekil 2.3:Yapay Sinir Ağı Örneği

Yapay sinir ağlarının V.M açısından kuvvetli yönleri şunlardır;

- Çok geniş açıdaki (spektrum) sorunların çözümünde kullanılabilirler,
- Çok karmaşık durumlarda dahi iyi sonuçlar üretmektedirler,
- Hem sayısal hem de kategorik veriler üzerinde işlem yapabilirler.

Yapay sinir ağlarının V.M açısından zayıf yönleri de şunlardır;

- 0 ile 1 arasında giriş verileri olması zorunludur,
- Ürettikleri sonuçların açıklamasını yapamazlar,
- Varılan sonucun olası en iyi sonuç olduğunun garantisi yoktur (Tantuğ, 2002).

2.8.1.3. Genetik algoritma

Genetik algoritma, Darwin tarafından geliştirilen "evrim teorisini"ne dayalıdır. Algoritma ilk olarak popülasyon adı verilen bir çözüm kümesi (öğrenme veri kümesi) ile başlatılır. Bir popülasyondan alınan sonuçlar bir öncekinden daha iyi olacağı beklenen yeni bir popülasyon oluşturmak için kullanılır. Evrim süreci (yeni popülasyonlar yaratma iterasyonu) tamamlandığında bağımlılık kuralları veya sınıf modelleri ortaya konmuş olur (Shah ve Kursak,2004).

2.8.1.4. K - en yakın komşu algoritması

Veri uzayında birbirine yakın olan aynı tip kayıtlar, birbirlerinin komşusu durumundadırlar. Bu anlayış doğrultusunda, çok kolay fakat güçlü olan k–en yakın komşu algoritması geliştirilmiştir. K-en yakın komşu algoritmasının temel felsefesi "komşunun yaptığını yaptır" dır. Belirli bir bireyin (kayıtın) davranışı (özelliğini) tahmin etmek istenirse, veri uzayında o bireye yakın olan örneğin 10 bireyin davranışına bakılabilir. Bu 10 komşunun davranışının ortalaması hesaplanır ve bu hesaplanan ortalama bireylerin tahmini olur. K-en yakın komşudaki k harfi araştırdığımız komşu sayısıdır. Örneğin, 5-en yakın komşuda, 5 komşuya bakılır (Adriaans ve Zantinge, 1996).

2.8.1.5. Bellek tabanlı yöntemler

İnsanlar kararlarını genellikle daha önce yaşadıkları deneyimlere göre verirler. Örneğin doktorlar bir hastayı incelerken, elde ettiği bulguları daha önce tedavi ettiği benzer hastalığa yakalanmış hastalar üzerindeki deneyimlerini kullanarak değerlendirirler. Bellek tabanlı yöntemler de benzer şekilde deneyimleri kullanmaktadır. Bu yöntemlerde, bilinen kayıtların bulunduğu bir veritabanı oluşturulur ve sistem yeni gelen bir kayda komşu olan diğer kayıtları belirler ve bu kayıtları kullanarak tahminde bulunur ya da bir sınıflama işlemi uygular. Bellek tabanlı yöntemlerin en önemli özelliği veriyi olduğu gibi kullanabilme yeteneğidir. Diğer V.M yöntemlerinin aksine bellek tabanlı yöntemler, kayıtların şekli (format) yerine sadece iki işlemin varlığı ile ilgilenir. Bu işlemler, iki kayıt arasındaki uzaklığı belirleyen bir uzaklık fonksiyonu ve komşu kayıtları işleyerek bir sonuç üreten kombinasyon fonksiyonudur (Tantuğ, 2002). Bellek tabanlı yöntemler sahtekârlık tespiti ve klinik işlemler gibi alanlarda kullanılmaktadır.

Bellek tabanlı yöntemler sahtekârlık tespiti ve klinik işlemler gibi alanlarda kullanılmaktadır.

Bellek tabanlı yöntemlerin güçlü olduğu noktalar:

- Kolayca anlaşılabilir sonuçlar üretir,
- Rastgele seçilen, hatta birbiri ile ilgisiz olabilen verilere bile uygulanabilir,
- Çözümleme alanlarının çok olduğu durumlarda dahi etkili olarak çalışabilir,
- Eğitim kümesinin oluşturulması basittir.

Bellek tabanlı yöntemlerin zayıf olduğu noktalar:

- Sınıflama ve tahmin işlemleri için kullanıldığında işlem maliyeti yüksektir,
- Eğitim kümesi için büyük miktarlarda alana ihtiyaç vardır,
- Üretilen sonuçlar; seçilen uzaklık fonksiyonuna, kombinasyon fonksiyonuna ve komşu sayısına doğrudan bağlıdır (Tantuğ, 2002).

2.8.1.6. Regresyon analizi

Regresyon analizi, bir ya da daha fazla bağımsız değişken ile bağımlı değişken arasındaki ilişkiyi matematiksel olarak modelleyen bir yöntemdir. Veri madenciliğinde yaygın olarak kullanılan regresyon modellerinden doğrusal regresyonda tahmin edilecek olan bağımlı değişken sürekli değer alırken; lojistik regresyonda bağımlı değişken kesikli bir değer almaktadır. Doğrusal regresyonda bağımlı değişkenin değeri; lojistik regresyonda ise bağımlı değişkenin alabileceği değerlerden birinin gerçekleşme olasılığı tahmin edilmektedir (Hui ve Jha, 2000)

2.8.2. Kümeleme

Kümeleme, verideki benzer kayıtların gruplandırılmasını sağlayan bir tekniktir. Kümeleme işlemi çoğunlukla bir başka V.M uygulaması için bir ilk işlem olarak kullanılır (Tantuğ, 2002). Kümelemede, genellikle K-ortalamalar algoritması ya da Kohonen şebekesi gibi istatistiksel yöntemler kullanılmaktadır. Hangi yöntem kullanılırsa kullanılsın süreç aynı şekilde işler. Her kayıt var olan kümelerle karşılaştırılır. Bir kayıt kendisine en yakın kümeye atanır ve bu kümeyi tanımlayan değeri değiştirir. Optimum çözüm bulununcaya kadar kayıtlar yeniden atanır ve küme merkezleri ayarlanır (Hui ve Jha, 2000). En yaygın kullanılan kümeleme algoritması “K ortalamalar algoritması” dır. K ortalamalar algoritması diğer

kümeleme teknikleri ile karşılaştırıldığında büyük veritabanlarının kümelenmesinde oldukça etkin bir algoritmadır. Yeni bir vaka ortaya çıktığında; algoritma tüm veriyi inceleyerek buna en çok benzeyen vakaların bir altkümesini oluşturur ve onları çıktıyı tahmin etmek için kullanır (Anand, 2003)

Bu algoritmanın adımları aşağıdadır:

- Veri seti rassal olarak k adet başlangıç kümesine ayrılır.
- Veri setinde yer alan örnekler; merkezi kendisine en yakın olan kümeye atanır.
- Her atamanın sonunda küme merkezi (ortalama) yeniden hesaplanır.
- Veri setindeki tüm örneklerin ataması yapılana kadar 2. ve 3. adımlar tekrarlanır.

Yeni bir vakanın ait olduğu kümeyi belirlemek için algoritma yeni vakanın öğrenme verisindeki her bir vakadan uzaklığını hesaplar. k değerinin ve uzaklık ölçüsünün modelin kalitesi üzerinde büyük etkisi vardır bu nedenle onları dikkatle seçmek çok önemlidir.

K ortalamalar algoritması oldukça etkin bir algoritma olmakla birlikte, sadece nümerik veri ile çalışır. Fakat veri madenciliği uygulamaları sıklıkla kategorik verileri de içermektedir. K ortalamalar algoritmasının geliştirilmesi ile elde edilen k modlar algoritması ise kategorik veriler üzerinde çalışabilen bir algoritmadır. K ortalamalar algoritmasında küme merkezleri, küme ortalaması alınarak hesaplanırken, k modlar algoritmasında küme merkezlerinin belirlenmesinde kümede en sık tekrarlanan değerler (mod) dikkate alınır (San ve diğ. , 2004).

2.8.3. Birliktelik kuralları ve ardışık zamanlı örüntüler

Birliktelik kuralları ve ardışık zamanlı örüntüleri birbirinden ayıran özellik zaman kavramının uygulamada olmasıdır. Belli bir dönem boyunca nesneler arasındaki birlikteliklerin incelenmesi "ardışık zamanlı örüntü çözümlemesi" olarak da isimlendirilir (Goebel ve Gruenwald, 1999).

Birliktelik kuralları; ticaret, mühendislik, fen ve sağlık sektörlerinin içinde bulunduğu birçok alanda uygulanmaktadır. Birliktelik kuralları, V.M araştırmalarında çok büyük yatırımlar yapılan, V.M'nin özel bir uygulama alanıdır. Birliktelik kuralları aynı işlem içinde çoğunlukla beraber görülen nesneleri içeren kurallardır. Birliktelik kurallarının bulunması ile pazar sepeti çözümlemesi yapılmaktadır. Pazar sepeti çözümlemesinde, nesneler müşteriler tarafından satın alınan ürünlerdir ve bir işlem (kayıt) ise birçok nesneyi içinde bulunduran tek bir satın almadır. Pazar sepeti çözümlemesinde sıklıkla beraber alınan nesneler üzerine çalışılır (Rushing, 1997). Bulunan kurallar ile nesnelerin birbiri ile nasıl ilişkili olduğu bilgisine ulaşılır.

Birliktelik analizi, Apriori, AprioriTid, MSApriori, AIS, STM, Sequence, GRI teknikleri ile yapılabilmektedir.

Bir alışveriş sırasında veya birbirini izleyen alışverişlerde müşterinin hangi mal veya hizmetleri satın alma eğiliminde olduğunun belirlenmesi, müşteriye daha fazla ürün satma yollarından birisidir (Han ve Kamber, 2001).

Birliktelik analizi, bir veri kümesindeki kayıtlar arasındaki bağlantıları arayan denetimsiz (unsupervised) veri madenciliği şeklidir. Birliktelik analizi çoğu zaman perakende sektöründe süpermarket müşterilerinin satın alma davranışlarını ortaya koymak için kullanıldığından "pazar sepeti analizi" olarak da adlandırılır (Bland, 2002).

Birliktelik kurallarına ait örnekler aşağıda yer almaktadır:

- Müşteriler bira satın aldıklarında %75 olasılıkla çocuk bezi de satın alırlar.
- Düşük yağlı peynir ve yağsız süt alan müşteriler %85 olasılıkla diyet süt alırlar.

Ardışık analiz ise birbiriyle ilişkisi olan ancak birbirini izleyen dönemlerde gerçekleşen ilişkilerin tanımlanmasında kullanılır. Aşağıda ardışık analize ait örnekler yer almaktadır.

- Çadır alan müşterilerin %10'u bir ay içerisinde sırt çantası almaktadır.
- A hissesi %15 artarsa üç gün içinde B hissesi %60 olasılıkla artacaktır.

3. GEZGİN SATICI PROBLEMİ

Gezgin Satıcı Problemi, en önemli algoritma problemlerinden biridir. Problem şu şekildedir:

- Bir seyyar satıcı var,
- Bu satıcı, mallarını *n* şehirde satmak istiyor,
- Öte yandan, mantıklı bir şekilde, bu satıcı bu şehirleri mümkün olan en kısa şekilde ve her bir şehre maksimum bir kere uğrayarak turlamak istiyor

Problemin amacı, satıcıya bu en kısa yolu sunabilmektir. Basit bir şekilde:

- İlk şehirde, satıcının n değişik şehir arasında seçim hakkı vardır
- İkinci şehirde, satıcının (n-1*)* değişik şehir arasında seçim hakkı vardır v.s.

Dolayısıyla, sonuç olarak satıcının (n-1)!/*2* değişik tur arasından seçim hakkı olacaktır. Bu, 100 şehirlik bir tur için bile (9,33* (10 ^157)) değişik tur etmektedir (http://tr.wikipedia.org/wiki).

3.1. Tur Belirleme Problemlerinin Sınıflandırılması

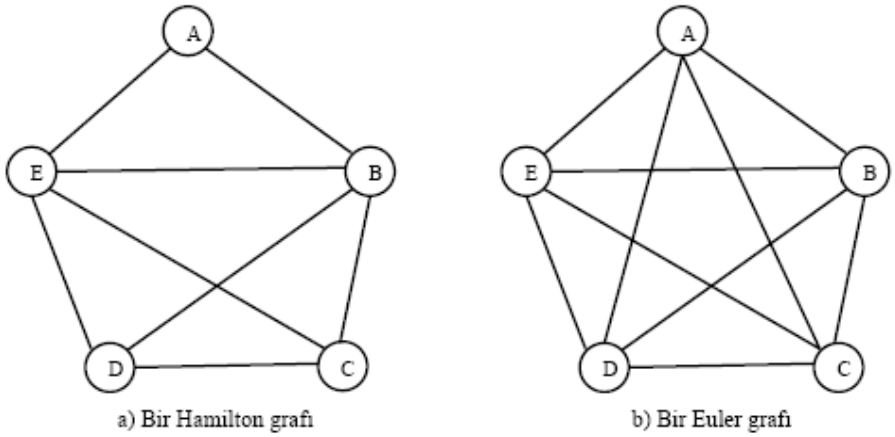
Tur Belirleme problemlerine dair çeşitli sınıflandırmalar vardır (Bodin ve Golden,1981). Ama genel olarak tur belirleme problemleri iki başlıkta incelenmektedir (Current ve Marsh,1993);

- Euler turlu problemler(Ayrıtlar için tur belirleme)
- Hamilton turlu problemler(Düğümler için tur belirleme).

3.1.1. Euler turlu problemler

Euler turu, serim kuramının da kurucusu olarak kabul edilen İsviçreli matematikçi Leonhard Euler(1707-1783) tarafından Königsberg köprüleri üzerinde tasarlanan bir problemin çözümü için tanımlanmıştır (Minieka,1978). Königsberg köprüleri problemi, eski doğu Prusya topraklarında kalan Königsberg kentinin (bugünkü adı Kaliningrad) halkı tarafından, kentin içinden geçen Pregel nehri üzerindeki 7 köprüden geçiş yapmaya dair Pazar eğlencesi olarak tanımlanmış bir oyundur. Oyundaki temel soru şudur: Acaba her köprüden yalnızca bir kere geçmek suretiyle, bir yerden başlayıp tekrar aynı yere dönülmesini sağlayacak yol var mıdır?(Gondran, 1984)

Euler grafı(Euler's graph); graf üzerinde kenarların tekrarlanmadığı kapalı bir dolaşımın yapılabileceği grafa Euler Grafı(Euler's Graph) denir. Euler çevrim, graf üzerinde her kenarı kapsayan, kapalı bir gezi yapılabilecek bir yolun bulunmasıdır. Şekil 3.2' de bu grafa bir örnek verilmiştir (Bayzan, 2005).



Şekil 3.2: Hamilton ve euler grafı

Euler, bu problemi inceleyerek ilk kez bir problemin düğüm ve ayrıtlarından oluşan bir serim olarak tanımlanmasını sağlamış ve 1736 yılında yaptığı bir çalışma ile "Königsberg Köprüleri Problemi" terimini kullanarak, problemin tanımlandığı şekliyle bir çözümünün olamayacağını ispatlamıştır. Buna göre Euler turu, bir başlangıçtan yola çıkarak serimdeki bütün ayrıtlardan sadece bir kez geçip yine başlangıca dönen yoldur. Her serimde Euler turunun olması gerekmez. Euler, bunun

koşullarını da incelemiş ve eğer serimdeki bütün düğümlerin dereceleri (düğüme bağlı ayrıt sayısı) çift sayı ise serimde Euler turunun olacağını, aksi halde olamayacağını göstermiştir.

3.1.2. Hamilton turlu problemler

Hamilton turu İrlandalı matematikçi William Rowan Hamilton (1805–1865) 1859 yılında tanımlanan bir matematiksel problem sonucu ortaya çıkmıştır (Gondran ve Minoux, 1984). Problem şöyle tanımlanmaktadır: Dünya yüzeyine dağılmış 20 tane şehir seçilerek bu şehirlere bir gezi düzenlenecektir. Amaç, her şehre yalnız bir kez uğramak ve geziye başlanan şehre geri dönmektir. Bu şartlar altında amaca uygun olarak hangi yollar izlenmelidir ve bu yollar nasıl belirlenmelidir?

Hamilton turu ile Euler turu arasındaki en önemli fark, problemin tanımlandığı ağlardır. Hamilton turu; uğranacak yerlerin düğüm, düğümleri birbirine bağlayan yolların da kenar olarak tanımlandığı ağlarda kullanılırken; Euler turu ise; kavşakların düğüm, uğranılması düşünülen yerlerin ise kenarlar üzerinde tanımlandığı graflarda kullanılır (Sipahioğlu A., 1996). Diğer bir ifadeyle, Euler turunda uğranacak yerler kenarlar üzerinde tanımlanır ve istenilen yere uğrayabilmek için graftaki her kenardan mutlaka ve yalnızca bir kez geçilmesi istenir. Hamilton turunda ise, uğranılması düşünülen yerler düğümler üzerinde tanımlanır ve graftaki her düğümden mutlaka ve yalnızca bir kez geçilmesi istenir. Bu sebeple, Euler turlu problemlere, kenarları gezecek gezgin için tur belirleme problemi; Hamilton turlu problemlere de, düğümleri gezecek gezgin için tur belirleme problemi demek doğru bir ifade olur. (Bayzan, 2005). Tez kapsamında Hamilton turlu problem olan Gezgin satıcı problemi(G.S.P.) esas alınmıştır.

Hamilton grafı (Hamilton's Graph); her düğümden yalnızca bir kez geçilmesi esasına dayanan ve kapalı bir dolaşım gerçekleştirilebilen grafa Hamilton Grafı (Hamilton's Graph) denir. Hamilton çevrim, graftaki her düğümü içine alan kapalı bir yolun olmasıdır. Gezgin satıcı problemi en az maliyetli Hamilton çevrim bulunmasına dayanan bir problem türüdür. Şekil 3.2' de örnek bir Hamilton graf gösterilmiştir.(Bayzan, 2005)

Hamilton turunun söz konusu olduğu en temel problem, Gezgin Satıcı Problemidir (G.S.P.) (Travelling Salesman Problem (T.S.P)). G.S.P., yöneylem araştırmasının en çok ilgi görmüş problemlerinden biridir ve bu ismi ilk olarak kimin kullandığı kesin olarak bilinememektedir (Lawler ve diğ.,1985). Aslında konuya olan ilgi Euler ile başlamış ve pek çok bilim adamı tarafından da incelenmiştir, ama problemle ilgili ilk ve en derli toplu çalışma 1954 yılında Dantzig,Fulkerson ve Johnson tarafından yayınlanan "Solution of a large-scale traveling salesman problem" isimli çalışmadır(Hoffman ve diğ., 1985). Bu çalışmada G.S.P. açıkça ortaya konmakta ve en iyi çözümü bulmak için tamsayılı doğrusal karar modeli önerilmektedir.

G.S.P., hem pek çok uygulama alanı olan bir problem olması nedeniyle, hem de farklı pek çok problemin temelini oluşturması nedeniyle oldukça önemlidir. Hamilton turlu diğer problemlerin tümü aslında G.S.P.'den türemiş problemlerdir. Bunların arasında m-Gezgin Satıcı Problemi(m-G.S.P.) ve Araç Turu Belirleme Problemi (ATBP), üzerinde çok çalışılmış diğer Hamilton turlu problemlerdir. Hamilton turlu problemler, sadece bu çalışmada aktarıldığı kadar değildir. İncelenen sistemin bileşenlerindeki özel durumlardan yararlanılarak sürekli yeni problem türleri tanımlanmaktadır. Dikkati çeken nokta, yeni tanımlanan her problemin öncekilerden daha karmaşık olduğu ve çözüm sürecinin daha da zorlaştığıdır. Ama gittikçe daha özel durumların ve daha yeni yaklaşımların probleme katıldığı gözlenmektedir. Bir başka önemli noktada bilinen pek çok yöntemin bu problemlerin çözümünde kullanılmaya çalışıldığıdır. Örneğin en kısa yol, en küçük kapsayan ağaç, dal-sınır tekniği, dinamik programlama, küme kapsama ve ayrıştırma, tavlama benzetimi, modelleri, ayrıştırma ve atama problemleri, genetik algoritma, yapay sinir ağları, karınca kolonileri gibi yöntemler sıkça kullanılmaktadır.

Gelecekte hem daha karmaşık ve gerçek hayat problemlerindeki özel durumları içeren problem türlerinin geliştirileceğini, hem de yeni çözüm yöntemleri türetileceğini söylemek mümkündür. Geliştirilecek yeni yaklaşımlar varolan problemlerin daha büyük boyutlarda çözülmesine olanak tanıyacaktır.

3.2. Gezgin Satıcı Probleminin Temel Özellikleri

Gezgin Satıcı Problemi (G.S.P.), belirlenen sayıda şehri, her şehre yalnız bir kez uğramak şartı ile gezecek, başlangıç şehrine geri dönecek minimum yolu bulma olarak tanımlanır. G.S.P., optimizasyon problemlerinden üzerinde en geniş çalışılmış ve en zor problemlerinden bir tanesidir. Optimizasyon problemlerinde çok değişkenli fonksiyonların en küçüklenmesine ya da en büyüklenmesine yönelik etkin yöntemlerin araştırılması ile ilişkilidir. Söz konusu problemi çözmek amacıyla geliştirilen algoritmalar ve sezgisel yaklaşımlar, şu soruya cevap üretmeye çalışmaktadır; n adet şehir ve her bir şehir arasındaki mesafelerin verildiği bir durumda, her bir şehrin sadece bir kez ziyaret edildiği ve tekrar başlangıç noktasına dönüldüğü en kısa tur nasıl oluşturulabilir?

n şehirli bir G.S.P.'de şehirler i ile j şehirleri arasındaki mesafeyi dij ile gösterecek olursak, eğer her dij = dji ise problem simetrik gezgin satıcı problemi (S.G.S.P.) olarak adlandırılır ve aşağıdaki gibi formüle edilebilir.

$$\text{Min } Z = \sum_{\forall ij, i \neq j} d_{ij} X_{ij} \tag{3.1}$$

$$\sum_{\substack{j=1 \\ j \neq i}}^{n} X_{ij} = 1, \qquad \text{i=1,….,n} \tag{3.2}$$

$$\sum_{\substack{i=1 \\ i \neq j}}^{n} X_{ij} = 1, \qquad \text{j=1,….,n} \tag{3.3}$$

$$\sum_{\substack{ni,nj \in S \\ i \neq j}} X_{ij} \leq |S| - 1 \qquad \forall S \subset N; \quad 2 \leq |S| \leq n-2 \tag{3.4}$$

$$X_{ij} = 0 - 1 \qquad \forall i, j = 1,....,n; \quad i \neq j \tag{3.5}$$

Yukarıdaki kısıtlardan eşitlik (3.2) ve eşitlik (3.3) derece kısıtlarıdır. Bunlar her şehre sadece bir kere girilmesini ve her şehirden sadece bir defa çıkılmasını sağlar. Eşitlik (3.4) ise alt tur eleme kısıtıdır. Bu kısıt S bağlı düğümler alt kümesinde sadece bir tane tur olmasını sağlar. Eşitlik (3.5) ise *Xij* 0-1 değişkenini ifade eder.

Burada eğer *Xij*= 1 ise, gezgin satıcı i şehrinden j şehrine gidiyor, *Xij* =0, ise şehirlerarasındaki mesafe gidilmiyor demektir (Eiselt ve Sandblom, 2000).

G.S.P., şehir sayısının (n) artması ile birlikte, olası tur sayısının şehir sayısına bağlı olarak artış göstermesi nedeniyle kolaylıkla çözülemez hale gelebilmektedir. Gezgin satıcı probleminin çözümüne yönelik yapılan birçok çalışma olmuştur, fakat şimdiye kadar düğüm sayısı arttıkça işlem zamanının üstel olarak artmadığı bir çözüm algoritması gösterilmemiştir (Bauk and Kova, 2004). Problem çözüm uzayının üstel olarak artmasından dolayı NP-Complete problem sınıfına girmektedir. Bu ifade, makul büyüklükteki bir problem için ele alınması gereken çok sayıda varsayımın varlığından dolayı, optimum bir çözüm aramanın (hesaplama maliyetlerinin çok yüksek olması nedeniyle) mümkün olmadığı problemler için kullanılmaktadır.

Bugüne kadar çözülen en büyük gezgin satıcı problemi 24,978 noktalıdır ve İsveç'te yerleşimi olan her nokta için çözülmüştür. Bu çözüm, Intel Xeon 2.8 ghz bir işlemcinin 92 yılına denk bir sürede yapılmıştır (öte yandan, 96 bilgisayarlı bir ağ üzerinde çözüldüğünden çözülmesi 3 yıl sürmüştür). Şu anda çözülmeye çalışılan en büyük problem Dünya üzerinde kayıtlı yerleşim olan her nokta için en kısa yolun ne olduğudur. Bu problem 1,904,711 şehir içermektedir (http://tr.wikipedia.org/wiki/).

3.3. Gezgin Satıcı Probleminin Tarihsel Gelişimi

Lawler, Hassler Whitney'in 1934 yılında Princeton Üniversitesinde yaptığı seminer konuşmasında, G.S.P. hakkında yapılmış olan çalışmaların sonuçları ve konunun kapsamı hakkında detaylı bir bilgi verdiğini bildirmektedir.

Dantzig, (1954), özel olarak seçtiği 49 şehrin yol uzunluklarını atlas üzerinden elde edip, bu şehirler arasında optimal Hamilton tur uzunluğunu hesaplamıştır. G.S.P.'nin özlü bir açıklaması ile birlikte konunun ortaya çıkışına yönelik bazı tarihsel bilgilerde verilmiştir.

Christofides (1971), kendisinden önce G.S.P.'nin optimal çözümü için önerilen minumum yayılan ağaç ve atama problemine dayalı alt sınır hesaplamalarına ek

olarak, yeni bir alt sınır algoritması vermiştir. 14 test problemi için bu yeni alt sınır algoritması uygulanmış ve simetrik durumda optimal değerin %4.7, asimetrik durumda ise optimal değerin sadece %3.8 altında alt sınır değerleri elde edilmiştir.
Lenstra ve Rinnooy Kan (1975), G.S.P.'nin bilgisayar bağlantısı, taşıt güzergâhı atama, kümeleme uygulamalarını açıklayıp, problemleri G.S.P.'ye göre formüle etmişlerdir.

Golden (1980), G.S.P. için önerilen çok sayıda sezgisel yöntemi 3 grupta toplayıp, büyüklükleri 25 ile 100 düğüm arasında değişen 8 test problemi ile bu sezgisel yöntemlerin performanslarını karşılaştırmışlardır. Bu deneysel çalışma sonucunda, bir karma yöntem yardımıyla optimal değerin %2-3'ü dahilinde bir çözüm elde edilebileceği kararına varılmıştır. Karma yöntem tekrarlı olarak uygulandığında ise optimal değerin %1-2'si dahilinde bir G.S.P. turu bulunabilmektedir.

Laporte (1987), şehirleri n kümeye en az bir kez, her düğümden en çok bir kez geçen ve n küme arasında bir Hamilton çevrimi arayan bu G.S.P. genellemesi için tam algoritma vermişlerdir. Problem, bir tamsayılı doğrusal programa göre formüle edilip, daha sonra programın gevşetilmesi ile dal ve sınır algoritması ile çözülmüştür. Bu algoritma kullanılarak 100 düğüm ve 8 kümeye kadar olan problemler optimal olarak çözülmüştür.

Arthur ve Frendewey (1988), G.S.P.'nin çözümünde kullanılan yöntemlerin performanslarının araştırılmasında güçlüğün nedenini, optimal tur uzunluğu bilinen büyük problemlerde (n>100) eksik oluşuna bağlamıştır. Bu nedenle optimal tur uzunluğu bilinen simetrik ve Öklit test problemleri için uzaklık matrisleri oluşturan bir algoritma vermişlerdir. Bu algoritma ile oluşturulan problemlerin rastgele oluşturulan problemler kadar zor olduğu belirtilmektedir.

Ong ve Huang (1989), birim alandaki n nokta arasında beklenen tur uzunluğunun, asimtotik olarak n'nin karekökü ile orantılı olduğunu seçilen sezgisel yöntemler için bir deneysel çalışmayla göstermişlerdir. Her bir sezgisel yöntem yardımıyla bulunan tur uzunluğuna n'nin karekökünün bir doğrusal fonksiyonu yardımıyla yaklaşılabilmektedir. Model parametreleri en küçük kareler yöntemiyle tahmin

edilmiştir. G.S.P. için iyi bir sezgisel yöntemin, n değeri arttığında da aynı özelliğe sahip olacağı bildirilmiştir.

Laporte (1992), G.S.P.' nin çözümüne ilişkin o yıla kadarki bazı tam ve yaklaşık algoritmaları tanıtmıştır.

Belmore ve Nemhmelauser (1996), problemi anlattıktan sonra konuyla ilgili temel birkaç teorem vermişlerdir. Burada çözüm yöntemleri; tur kurma, tur ilerleme ve alt tur eleme olmak üzere 3 gruba ayrılmıştır. Alt tur eleme yönteminde, uzaklık matrisinin atama problemine göre çözüldüğü, eğer bulunan çözüm tek bir tur içermiyor ise dal ve sınır yönteminin kullanılacağından bahsedilmektedir.

Dorigo vd. çalışmalarında (Dorigo, 1997) Karınca Kolonisi Sistemi (K.K.S) kullanarak bir algoritma yardımıyla G.S.P.' yi çözmeye çalışmışlardır. Karınca adı verilen işbirlikçi etmenler G.S.P. için iyi çözümler bulmaya çalışmaktadırlar. Feromen adı verilen bir maddeyle doğrudan olmayan bir iletişim yapan karıncalar bu maddeyi yolların üzerinde saklayarak en çok geçilen, en iyi yolun bulunmasında yardımcı olmaktadırlar. Çalışmada K.K.S deneyler yardımıyla anlaşılmaya çalışılmaktadır.

Obitko (Obitko, 1999) genetik algoritmaları tanıtmak için hazırladığı çalışmasında gezgin satıcı probleminin genetik algoritmalarla çözümünü de örneklemiştir. Bu çalışma genetik algoritma eğitiminde kullanılmak üzere bilgilendirici bir çalışmadır. Örnekler Java Appletlerinin kullanımıyla gerçekleştirilmiştir.

3.4. Gezgin Satıcı Probleminin Uygulamaları

G.S.P.'nin n müşteri veya şehir arasında en kısa Hamilton çevriminin bulunmasına dayalı çeşitli uygulamaları bulunmaktadır. Buna ek olarak G.S.P. , ilişkisiz gibi görünen bazı problemlerde G.S.P.'ye göre formüle edilerek çözülmesine yardımcı olmaktadır.

G.S.P. uygulamalarına şu örnekler verilebilir. Yol planlama (uçak, otobüs, dağıtım kamyonları, bilgisayar ağları, posta taşıyıcıları, vb.), iş sıralanmasında (n adet iş, tek makinede ardışık olarak gerçekleştirilebilir), bilgisayar bağlantısında, bilgisayarların veya diğer elektronik sistemlerin tasarımlarında ve baskı devre kartlarındaki delgi işlemi sırasının belirlenmesi gibi birçok alanda kullanılmaktadır.

3.5. G.S.P. için Geliştirilen Çözüm Yöntemleri

Yöneylem araştırmasındaki ve bilgisayar dünyasındaki her gelişme, bu problemlere yansımış ve yeni tekniklerin geliştirilmesine yol açmıştır. G.S.P. için geliştirilen yöntemleri 3 alt başlıkta incelenmiştir. Bunlar; kesin çözüm yöntemleri, geleneksel sezgisel yöntemler ve meta sezgisel yöntemlerdir.

3.5.1.Kesin Çözüm Yöntemleri

Geliştirilen kesin çözüm yöntemleri, problemi genişleterek veya ayrıştırarak bilinen yöntemlere çeviren ve çözen yöntemlerden dal-sınır yöntemleri, matematiksel programlamaya dayalı dinamik programlama yaklaşımları bu gruba girer (Sipahioğlu A., 1996).

3.5.1.1. Dal-sınır algoritması

Dal-sınır algoritması, G.S.P.'nin genişletilerek çözülmesi ve iyi alt sınırlar türetilerek dallandırmalarla problemin tamsayılı çözümünün belirlenmesi mantığına dayanır. Genişletme genellikle problemin, atama problemi, eşleme problemi, örten ağaç problemi veya en kısa yol n yol problemine çevrilmesi ile yapılmaktadır. (Ulusoy ve Tovya,1983).

3.5.1.2.Dinamik programlama yaklaşımları

Dinamik programlama, yöneylem araştırmasında kullanılan optimizasyon yöntemlerinden birisidir. Optimizasyonda amaç, mevcut kısıtlayıcı koşullar altında, eldeki sorunla ilgili en iyi karara varmaktır.

Biri diğerini izleyen ve karşılıklı etkileri olan bir dizi kararın bütünüyle ele alındığı problemler için geliştirilen karar modelleri ve bunların çözümleri "Dinamik Programlama" başlığı altında incelenir. Öte yandan incelenen problemin biri diğeriyle ilişkili alt problemlere ayrılabilme özelliğini taşıması ya da bir problem için geliştirilen karar modelinin, birbirine bağlı karar modelleri haline dönüştürülmesi, dinamik programlama uygulaması için yeterli olmaktadır.

Bazı ekonomik değişme ve gelişmeler, gelecek dönem için önceden yapılan planları geçersiz kılabilir. Bu durumda yeni bir planlamaya gereksinim vardır ya da önceki plan güncelleştirilmelidir. Koşullar bir zaman sürecinde değişiyorsa ve bunların alınan kararlara etkisi önemli ise, dinamik programlama modellerine gereksinim vardır(Sezen K.,1998).

3.5.2.Geleneksel Sezgisel Yöntemler

G.S.P.' de geleneksel sezgisel yöntemlere;

- Clarke ve Wright algoritması,
- İyileştirme değiştirme yöntemi,
- Önce tur belirle sonra grupla algoritması,
- Önce grupla sonra tur belirle algoritması örnek verilebilir.
- Lin-Kernighan Algoritması

3.5.2.1.Clarke ve Wright Algoritması

Bu konudaki ilk çalışma 1964'te Clarke ve Wright tarafından gerçekleştirilmiştir. Clarke ve Wright algoritmasında temel mantık her adımda eldeki turlar kümesini daha iyiye değiştirerek iyi bir çözüme erişmektedir. Bu algoritmanın en önemli noktası kazanım değerlerinin nasıl hesaplanacağıdır. Bu nedenle kazanım değerlerini hesaplamak için farklı pek çok formülasyon önerilmiş, en fazla incelenen ve türevleri geliştirilen algoritma olmuştur. Algoritmanın 100 düğüme kadar iyi sonuç verdiği gözlenmiştir (Sateesh ve Ray,1992).

3.5.2.2. İyileştirme değiştirme yöntemi

Bu algoritma dal(ayrıt) değiştirme veya p-opt. Değişimi adıyla anılmaktadır. Yöntem problem için bir başlangıç çözüm bulunması ve bunun her adımda çözümdeki ayrıtlardan bazılarının çözüm dışı ayrıtlarla değiştirilerek iyileştirilmesi mantığına dayanır. İyileştirme, her aracın kendi turu içindeki i-j bağlantılarını değiştirerek gerçekleştirilir. İşlemler toplam maliyeti düşürecek bir değişiklik bulunduğu sürece devam eder. Yöntem her zaman uygunluğu korur ve en iyi çözümü arar.

3.5.2.3. Önce tur belirle sonra grupla algoritması

Önce tur belirle sonra grupla algoritması (Route First-Cluster Second), önce kapasite koşullarını dikkate almadan bütün düğümleri kapsayan en kısa turun G.S.P. ile bulunması, sonra da bu turun kapasite koşullarını sağlayacak şekilde makul alt turlara ayrılması mantığına dayanır. Yöntem, 1969'da Newton ve Thomas tarafından geliştirilmiştir. (Sipahioğlu A., 1996). Daha sonra 1979'da Bodin ve Berman yöntemi, bir okul için otobüs turunun belirlenmesinde, Stern ve Dror ise elektirk sayacı okuyucularının turlarının belirlenmesine başarıyla uygulamışlardır (Sateesh ve Ray,1992).

3.5.2.4. Önce grupla sonra tur belirle algoritması

Önce grupla sonra tur belirle algoritması (Cluster First- Route Second), Sweep algoritması adıyla da anılmaktadır. Yaklaşımın mantığı önce tur belirle sonra grupla algoritmasının tam tersidir. İlk olarak serimdeki düğümler araç kapasitesini aşmayacak şekilde gruplandırılır. Sonra da her grup için uygun (ekonomik) bir araç turu belirlenir. Bu konudaki ilk çalışma 1974'te Gillet ve Miller tarafından yapılmıştır (Lawler ve diğ.,1985). Yöntemin 250 düğüme kadar iyi sonuç verdiği gösterilmiştir (Sateesh ve Ray, 1992).

3.5.2.5. Lin-Kernighan algoritması

Lin-kernighan algoritması simetrik gezgin satıcı problemi için optimum yada optimumu sonuçları bulmada en etkili algoritmadır. *k*-en yakın komşu algoritmasına benzer. Algoritma, problemin çözümünde tur kalitesi ve hesaplama maliyetini göz önünde bulundurur. Algoritma *k* göz önünde bulundurularak her iterasyonda eğer tur uzunluğunda olası bir iyileşme varsa iyileştirmeyi gerçekleştirir ve daha iyi bir tur oluşmasını sağlar. Algoritma ara adımlarda kötü turları eleyerek, iyi turları döndürür.

3.5.3. Meta Sezgisel Yöntemler

Optimizasyon, hayatın hemen her alanında gerekliliği kaçınılmaz bir kavram olup, kazancı maksimize veya kaybı minimize etmeyi hedefler. Bu amaç için birçok yöntem kullanılabilir. Şayet kullanılan yöntem(veya algoritma) parametreleri belli bir probleme her uygulanışında aynı sonucu veriyorsa, bu tür yöntemlere deterministik yöntemler denir. Deterministik yöntemler, genellikle en iyi bir tek çözüm için kodlanırlar. Deterministik olmayan yaklaşımlar, aynı durum için farklı çalışmalarında aynı sonucu garanti etmeyen yöntem veya algoritmalardır. Yani bir satranç oyununda aynı pozisyon için program ilk çalışmasında A7 karesine oynamayı çözüm olarak verdiği halde, sonraki denemede A3'ü uygun çözüm olarak verebilir.

Meta Sezgisel(meta-heuristic) yöntemler deterministik olmayan yöntemlerin bir alt grubudur ve en iyi çözümü garanti etmemekle birlikte, denenmesi gereken ihtimallerin çok fazla olduğu durumlarda, daha az deneme ile "iyi" bir çözüm önermek amacıyla kullanılırlar. "meta-heuristic" optimizasyon yöntemidir ve her zaman en iyiyi bulmayı vaat etmediği halde, hep en iyiye yakın çözümlerden birini elde etmemizi sağlar.

3.5.3.1.Tavlama benzetimi

Tavlama benzetimi, teorik olarak bir yerel en iyi araştırma algoritmasıdır. Ama yöntemin en önemli özelliği, algoritmanın yerel en iyi noktaları bulduktan sonra da işleme devam ederek yeni çözümler aramasıdır. Bu nedenle yeteri kadar beklenirse

en iyi çözümün bulunması olasılığı vardır. Bu konudaki ilk gelişme, 1953 yılında yapılan bir çalışmada katıların yavaş yavaş soğutulurken oluşan ısıl denge denkleminin belirlenmesiyle başlar. Bundan 30 yıl sonra 1983'te Kirkpatrick, Gelatt ve Vecchi, kombinatoryal en iyileme problemlerinde karşılaşılan maliyetin en küçüklenmesi biçimindeki amaç fonksiyonuyla, daha önce bulunan ısıl denge denkleminin benzerliğini göstermişlerdir. Bu benzerlikten yararlanılarak bir yöntem geliştirilmiş ve adına da tavlama benzetimi denmiştir. (Koulamas,1994)

Tavlama benzetimi, metalürjik tavlama işlemini taklit eden sezgisel arama yöntemidir. Monte Carlo modeline dayanan bu yaklaşım, kareli atama probleminin çözümünde ikili yer değiştirmelere karşı gelmektedir. Aynı değiştirmeyi yapan geleneksel iyileştirme yöntemleri, sırası gelen ikili değiştirmeyi değerlendirmekte, eğer amaç fonksiyonunda iyileşme olacaksa, bu değişikliği yapmaktadır. Tavlama benzetiminin, yapay zekâ dışındaki sezgisel algoritmalardan temel farkı da, burada ortaya çıkmaktadır. Tavlama benzetimi, değerlendirmeyi yapar, iyileşme varsa ikili değiştirmeyi kabul eder. Ancak iyileşme yoksa bile, bu değişimi hemen reddetmez, belli bir olasılıkla kabul eder. Bu strateji, amaç fonksiyonunun değerinde zaman zaman gerilemelere neden olsa da, çözüm uzayının daha geniş bir bölümünü tarama olasılığını yükselttiğinden, alt en iyilere takılma tehlikesini azaltmış, daha kaliteli çözümler bulma şansını da arttırmış olur.

3.5.3.2. Tabu arama

Tabu arama algoritması; öğrenme, geliştirme gibi stratejiler ile bölgesel optimumdan sakınmak için iyi bir komşuluk üretme mekanizması kullanmaktadır. Burada, bir yasaklama stratejisi ile araştırmayı sınırlama ve yasaklama, sadece tabu listesini güncellemek ve kontrol etmekle gerçekleştirilmektedir. Yasaklama stratejisinde hedef, araştırmanın daha önce görülen çözümlere tekrar dönmesini önlemek ve araştırmayı yeni bölgelere yönlendirmektir.

Tabu Arama Algoritmasının genel işleyişinde öncelikle başlangıç çözümü oluşturulur. İkinci adımda seçilen kabul edilebilir en iyi hareket, hedef değer ve tabu şartlarına bağlı olarak devam eden, komşuluğunda en yüksek gelişme değerini

sağlayan harekettir. En yüksek değerlendirme fonksiyonu, en fazla iyileşen veya önceki iterasyonun en iyi hareketini seçer. Kabul edilebilir hareketlerin karakteristikleri tabu listesinde depolanarak daha sonraki iterasyonda tabu olarak sınırlandırılır ve sonraki hareketlerin belirlenmesinde kullanılır. Belirli şartların sağlanmasına bağlı olarak her hangi bir adımda iyileşmeyen hareketlerinde kabul edilebilmesi daha önceki adımlarda görülen çözüm noktalarına geri dönülmesine engel olmaktadır. Tabu arama algoritması, çeşitli optimizasyon problemlerinin çözümünde etkin olarak kullanılmaktadır.

3.5.3.3 Genetik algoritmalar

Genetik Algoritmalar (G.A), evrim ve genetiğin doğal sürecine dayalı stokastik bir araştırma tekniğidir. Bir problemin iyi çözümleri, olası çözümlerin jenerasyonlarının seçkin üremeleri ile geliştirilir. G.A' lar, daha iyi çözümlere doğru yavaş yavaş bir yaklaşım sağlayan geniş bir problem uzayı boyunca yönlendirilmiş rasgele bir araştırmaya olanak sağlarlar. Bu algoritma canlılarda bulunan genetik gelişimini simüle etmektedir. G.A' ların genel avantajlarından biri optimize etmeye çalıştıkları problemin doğası ile ilgili herhangi bir bilgiye ihtiyaç duymamalarıdır. Özel bilgi gerekmeksizin bir kısım zor problemlerin çözümü mümkündür. Problemin parametre değerlerinin ve bunların uygunluğunun değerlendirilmesinde sadece bir kodlamaya ihtiyaç duyarlar. Bu, araştırma prosedürünün karmaşıklığını azaltır ve diğer araştırma metotlarına göre daha etkin bir yol sağlar. G.A' lar birçok problem için optimale yakın çözümlerin bulunmasında başarılı olmuşlardır (Knosala ve Wal, 2001).

Veri madenciliği teknikleri ile Gezgin Satıcı Problemi çözüm yöntemlerinde kullanılan bazı teknikler aynıdır. Bunlardan birkaçı; genetik algoritmalar, yapay sinir ağları, en yakın komşu algoritması gibi.

4. GEZGİN SATICI PROBLEMİ İÇİN VERİ MADENCİLİĞİ TABANLI SEZGİSEL BİR YAKLAŞIM

Bu çalışmada G.S.P. için veri madenciliği tabanlı sezgisel bir yaklaşım geliştirilmiştir. Bu yaklaşımda 15 ayrı gezgin satıcı problemi (gr17, ulysses22, bays29, dantzig42, eil51, brasil58, st70, eil76, gr96, rd100, ch130, si175, gr202, pr299, lin318) ve optimum çözümleri TSPLIB internet sitesinden elde edilmiştir. Bu problemler için ilk önce eşik değer ($\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S ve $\bar{x}$-3S) hesabı için kaç örnek sayısı kullanılması gerektiğinin deneyi yapılmıştır. Farklı örnek sayılarında rassal turlar üretilmiş ve tur uzunluklarının ortalaması ($\bar{x}$), standart sapması, $\bar{x}$-S, $\bar{x}$-2S, ve $\bar{x}$-3S değerlerinin hangi örnek değeri ile bulunmasının doğru olacağı saptanmıştır. Belirlenen örnek sayısı kadar rassal tur oluşturulup, eşik değerler ($\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S ve $\bar{x}$-3S) bulunmuştur. Daha sonra tur deneme sayısını belirlemek için, tur uzunluğu belirlenen eşik değerden (yani $\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, ve $\bar{x}$-3S tur uzunluklarından biri) küçük rassal turlar farklı deneme sayıları için üretilmiştir. Bu deneyler sonucunda da hangi deneme sayısının minimum tur uzunluğunu belirleyebileceği saptanmıştır. Belirlediğimiz deneme sayısı kadar turları tüm eşik değerler için($\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, ve $\bar{x}$-3S) 10'ar kez üretip, tur uzunluk değerleri bulunmuştur. Turların uzunluk değerleri eşik değere göre ağırlıklandırılarak, bağlı olan şehirlerin tur uzunluğuna etkisi dikkate alınmış ve yeni turlar oluşturulmuştur. Böylelikle daha kaliteli deneme sayısı kadar turlara sahip olunmuştur. Bulunan yeni turların, tur uzunlukları ve matlab programında ne kadar sürede hesaplandığı bulunmuştur. Bu turların içinde minimum tur uzunluğuna sahip tura yerel arama yapılarak sonucun etkinliği artırılmaya çalışılmış ve optimum sonuç ile karşılaştırılmıştır. Geliştirilen sezgisel yaklaşımın adımları aşağıda gösterilmiştir.

1. İlk önce eşik değer ($\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S ve $\bar{x}$-3S) hesabı için kaç tane rassal örnek tur kullanılması gerektiğinin deneyi yapılmıştır.

2. Belirlenen rassal örnek tur sayısı kadar rassal tur oluşturup, bu turların uzunluklarının örnek ortalaması ($\bar{x}$), standart sapması(S) ve bu bilgilerle eşik değer olarak kullanacağımız ($\bar{x}$-S, $\bar{x}$-2S , $\bar{x}$-3S) değerler bulunmuştur.

3. Tur deneme sayısını belirlemek için, yani 2. adımda belirlenen eşik değerden küçük kaç tur üretmemiz gerektiğini bulmak için, farklı deneme sayıları kadar rassal turlar üretilmiştir. Bu deneyler sonucunda da hangi deneme sayısının minimum tur uzunluğunu belirleyebileceği saptanmıştır.

4. Deneme sayısı kadar rassal turları, tüm eşik değerler için($\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, ve $\bar{x}$-3S) 10'ar kez üretip, uzunluk değerleri bulunur.

5. Tüm turların uzunluk değerleri teker teker eşik değere göre ağırlıklandırılarak bir oran bulunur ve ilgili turun tüm ikili bağlarına bu oran atanır. Diğer turların oranları da bu matriste bağlı olan şehirlere eklenir. Böylelikle tüm bağlar üzerinde, tur uzunluğunun etkisi dikkate alınmış olur. Elde edilen yeni matriste en büyük orana sahip bağdan başlanarak tüm şehirler kullanılarak yeni tur oluşturulur.

6. Bulunan yeni turların, tur uzunlukları ve matlab programında ne kadar sürede bulunduğu hesaplanır.

7. Bu turlar içinde minimum tur uzunluğuna sahip tura yerel arama yapılarak sonucun etkinliği artırılmaya çalışılmış ve optimum sonuç ile karşılaştırılmıştır.

Algoritma kodlaması MATLAB 7.6.0 programıyla ve 2 GB RAM'e sahip bilgisayarda gerçekleştirilmiştir.

MATLAB; (MATrix LABoratory); ilk defa 1985'de C.B Moler tarafından matematik ve özellikle de matris esaslı matematik ortamında kullanılmak üzere geliştirilmiş etkileşimli bir paket programlama dilidir. Matlab tüm mühendislik alanında, sayısal hesaplamalar, veri çözümlenmesi ve grafik işlemlerinde kolaylıkla kullanılabilen bir programlama dilidir. Fortran ve C dili gibi yüksek seviyeden

programlama dilleri ile yapılabilen hesaplamaların pek çoğunu Matlab ile yapmak mümkündür. Matlab'ın kullanım yerleri;

- Denklem takımlarının çözümü, doğrusal ve doğrusal olmayan diferansiyel denklemlerinin çözümü, integral hesabı gibi sayısal hesaplamalar,
- Veri çözümleme işlemleri,
- İstatistiksel hesaplamalar ve çözümlemeler,
- Grafik çizimi ve çözümlemeler.

4.1. Eşik Hesaplama

Yarattığımız algoritmada n şehirli her problemde, minimum tur uzunluğunu (optimumu) bulabilmek için, önceden bu optimuma yakın bir tur uzunluğu belirlememiz gerekmektedir. Böylelikle yazdığımız algoritma, istediğimiz deneme sayısı kadar, belirlediğimiz eşik değerden küçük tur uzunluğuna sahip rassal turlar üretecektir. Bunun için de belirlediğimiz örnek sayısı kadar rassal ürettiğimiz turların, tur uzunluklarının ortalamasını ve standart sapmasını hesapladık. Eşik değer olarak tur uzunluğunun örnek ortalaması ($\bar{x}$), $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S değerlerini kullandık. Örneğin; şehir sayısı 4 olan ve örnek sayısını 3 olarak belirlediğimiz gezgin satıcı problemini ele alalım. 4. adımda yer alan rassal olarak deneme sayısı kadar örnek üret ile 3 adet rassal 1-2-3-4, 4-1-2-3, 3-1-2-4 turları üretilmiştir. Tablo 4.1' de şehirlerarası uzaklık matrisi verilmiştir.

Tablo 4.1: Şehirlerarası uzaklık matrisi

Şehirlerarası uzaklık matrisi	1	2	3	4
1	-	12	15	10
2	12	-	18	20
3	15	18	-	16
4	10	20	16	-

Bu turların uzunlukları 5. adımda yer alan, her örneğin tur uzunluklarını hesapla ile şehirlerarası uzaklık matrisindeki uzaklıklara göre sırasıyla 46, 40, 47 hesaplanmıştır. 3 rassal turun uzunluklarının ortalaması $\bar{x}$=44.33 ve standart sapması S=3.78 bulunmuştur. Bulunulan bu sonuçlara göre, eşik değerleri $\bar{x}$=44.33, $\bar{x}$-S=40.55, $\bar{x}$-2S= 36.77, $\bar{x}$-3S=32.99 olarak hesaplanmıştır.

Algoritma adımları aşağıda gösterilmiştir.

Örnek ortalaması: $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$

Standart Sapma: $S = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}$

1.Adım: Başla,
2.Adım: Deneme sayısı, şehirlerarası uzaklık matrisi, şehir sayısı oku,
3.Adım: Toplam=0, i=1,
4.Adım: Rassal olarak deneme sayısı kadar örnek üret,
5.Adım: Her örneğin tur uzunluklarını hesapla,
6.Adım: Tur uzunluğu(i)= örnek tur uzunluğu, toplam=toplam+örnek tur uzunluğu,
7.Adım: i=i+1,
8.Adım: Eğer i<=deneme sayısı ise git 4'e,
9.Adım: Ortalama tur uzunluğu= toplam/ deneme sayısı,
10.Adım: Kare toplam=0,
11.Adım: i=1,
12.Adım: Kare toplam=kare toplam+ (tur uzunluğu(i)- tur uzunluğu)^2
13.Adım: i=i+1,
14.Adım: Eğer i<1=deneme sayısı ise git 12'ye,
15.Adım: Standart sapma=(kare toplam/(deneme sayısı-1))$^{1/2}$
16.Adım: Standart sapma ve ortalama tur uzunluğu yaz,
17.Adım: Bitir.

İlk önce 17 şehirli ve 130 şehirli gezgin satıcı problemi için farklı örnek sayıları 2'şer kez tekrarlanıp örnek sayısının eşik değer üzerindeki etkisine bakılmıştır. Tablo 4.2' de gr17 problemi için 500'den başlayarak 700000'e kadar artarak 2'şer kez tekrarlanan farklı örnek sayıları için tur uzunluklarının örnek ortalaması ($\bar{x}$), standart sapması, $\bar{x}$-S, $\bar{x}$-2S ve $\bar{x}$-3S değerleri gösterilmiştir.

Tablo 4.2: Gr17 probleminin farklı örnek sayıları için eşik değerleri

Örnek sayısı	**Tur uzunluklarının örnek ortalaması ($\bar{x}$)**	**Tur uzunluklarının standart sapması(S)**	**$\bar{x}$-S**	**$\bar{x}$-2S**	**$\bar{x}$-3S**
500	4650	439,7491	4210,2509	3770,5018	3330,7527
500	4660	431,471	4228,529	3797,058	3365,587
1000	4669	438,1749	4230,8251	3792,6502	3354,4753
1000	4671	446,5739	4224,4261	3777,8522	3331,2783
5000	4665	431,39	4233,61	3802,22	3370,83
5000	4666	444,3711	4221,6289	3777,2578	3332,8867
10000	4669	433,4712	4235,5288	3802,0576	3368,5864
10000	4667	434,2718	4232,7282	3798,4564	3364,1846
50000	4670	435,4729	4234,5271	3799,0542	3363,5813
50000	4670	429,4721	4240,5279	3811,0558	3381,5837
100000	4668	431,3721	4236,6279	3805,2558	3373,8837
100000	4668	432,4729	4235,5271	3803,0542	3370,5813
300000	4668	432,5732	4235,4268	3802,8536	3370,2804
300000	4667	432,4728	4234,5272	3802,0544	3369,5816
400000	4667	433,351	4233,649	3800,298	3366,947
400000	4668	433,4779	4234,5221	3801,0442	3367,5663
500000	4668	432,472	4235,5279	3803,0558	3370,5837
500000	4668	432,4873	4234,5127	3802,0254	3369,5381
600000	4668	432,4729	4235,5271	3803,0542	3370,5813
600000	4668	432,4677	4235,5323	3803,0646	3370,5969
700000	4668	432,4721	4235,5279	3803,0558	3370,5837
700000	4668	432,4728	4235,5272	3803,0544	3370,5816

Tablo 4.2'de de görüldüğü gibi örnek sayısı 500'den 100000'e kadar değişkenlik gösteriyor. 500000'den itibaren ise tur uzunluklarının örnek ortalaması ve standart sapması değişkenlik göstermemektedir. 500000 ve üstü örnek sayıları için tur uzunluklarının örnek ortalaması ($\bar{x}$),standart sapması, $\bar{x}$-S, $\bar{x}$-2S ve $\bar{x}$-3S sonuçları birbirine çok yakın çıkmıştır. Gr17 problemi için tur uzunluklarının örnek ortalaması ($\bar{x}$), standart sapması**,** $\bar{x}$-S, $\bar{x}$-2S, ve $\bar{x}$-3S sonuçları 500000 örnek sayısından itibaren anlamlı bir değişkenlik göstermemektedir.Gr17 problemi için bu sonuçlara göre tüm problemlerde eşik değer hesabı için örnek sayısının 500000 alınması uygun görülmüştür.

Tablo 4.3:Ch130 probleminin farklı örnek sayıları için eşik değerleri

Örnek sayısı	**Tur uzunluklarının örnek ortalaması($\bar{x}$)**	**Tur uzunluklarının standart sapma(S)**	**$\bar{x}$-S**	**$\bar{x}$-2S**	**$\bar{x}$-3S**
500	46413	1689,2718	44723,7282	43034,4564	41345,1846
500	46311	1637,355	44673,645	43036,29	41398,935
1000	46415	1679,3771	44735,6229	43056,2458	41376,8687
1000	46214	1673,361	44540,639	42867,278	41193,917
5000	46321	1647,5472	44673,4528	43025,9056	41378,3584
5000	46321	1651,4619	44669,5381	43018,0762	41366,6143
10000	46315	1655,361	44659,639	43004,278	41348,917
10000	46300	1668,3619	44631,6381	42963,2762	41294,9143
50000	46302	1663,4619	44638,5381	42975,0762	41311,6143
50000	46307	1670,7193	44636,2807	42965,5614	41294,8421
100000	46300	1665,3791	44634,6209	42969,2418	41303,8627
100000	46305	1666,729	44638,271	42971,542	41304,813
300000	46304	1659,2711	44644,7289	42985,4578	41326,1867
300000	46310	1663,4699	44646,5301	42983,0602	41319,5903
400000	46307	1663,4281	44643,5719	42980,1438	41316,7157
400000	46308	1663,371	44644,629	42981,258	41317,887
500000	46308	1662,361	44645,639	42983,278	41320,917
500000	46308	1662,3619	44645,6381	42983,2762	41320,9143
600000	46308	1665,4619	44642,5381	42977,0762	41311,6143
600000	46308	1665,3618	44642,6382	42977,2764	41311,9146
700000	46308	1665,4181	44642,5819	42977,1638	41311,7457
700000	46308	1665,1739	44642,8261	42977,6522	41312,4783

Tablo 4.3'de görüldüğü gibi ch130 probleminde, 500000 ve üstü örnek sayıları için tur uzunluklarının örnek ortalaması ($\bar{x}$), standart sapma, $\bar{x}$-S, $\bar{x}$-2S, ve $\bar{x}$-3S sonuçları birbirine çok yakın çıkmıştır. Gr17 ve ch130 problemleri için tur uzunluklarının örnek ortalaması ($\bar{x}$), standart sapması**,** $\bar{x}$-S, $\bar{x}$-2S, ve $\bar{x}$-3S sonuçları 500000 örnek sayısından itibaren anlamlı bir değişkenlik göstermemektedir. Bu sonuçlara göre tüm problemlerde eşik değer hesabı için örnek sayısının 500000 alınması uygun görülmüştür.

4.2. Rassal Tur Deneme Sayısının Belirlenmesi

Gr17 ve gr96 problemleri belirlediğimiz eşik değer ile minimum tur uzunluğunu bulabilmek için kaç deneme yapmamız gerektiğini bulabilmek adına kullanılmıştır.

Gr17 problemi için eşik değeri $\bar{x}$=4668 verilip, farklı örnek büyüklükleri ile 2'şer kez tekrarlanmış sonuçlar tablo 4.4' te gösterilmiştir.

Tablo 4.4: gr17 probleminin farklı deneme sayıları sonuçları

Deneme sayısı	**Tur uzunluğu ve tekrar sayısı**
500	2232/2
1000	2232/2
5000	2232/1, 2168/1
10000	2168/1, 2232/1
50000	2152/2
100000	2152/2
200000	2152/1, 2168/1
300000	2104/1, 2168/1
400000	2152/1, 2104/1
500000	2104/2
600000	2104/2
700000	2104/2

Gr17 problemi için 500000 deneme sayısından sonra tur uzunluğunda anlamlı bir iyileşme görülmemiştir.

Gr96 problemi için eşik değeri $\bar{x}$=366520 verilip, farklı örnek büyüklükleri ile 2'şer kez tekrarlanmış sonuçlar tablo 4.5' te gösterilmiştir.

Tablo 4.5: gr96probleminin farklı deneme sayıları sonuçları

Deneme sayısı	**Tur Uzunluğu ve tekrar sayısı**
500	223431/1, 238823/1
1000	199369/1, 180909/1
5000	149364/1, 138441/1
10000	111109/1, 104545/1
50000	81101/1, 76057/1
100000	66680/1, 66730/1
200000	70423/1, 64052/1
300000	64271/1, 62664/1
400000	63471/1, 61441/1
500000	60653/1, 60456/1
600000	61400/1, 60976/1
700000	60457/1, 61212/1

Gr96 problemi içinde gr17'de olduğu gibi 500000 deneme sayısının üstündeki deneme sayılarında tur uzunluğunda anlamlı bir iyileşme görülmemiştir. Bu

sonuçlara göre tüm problemler için deneme sayısı 500000 alınması uygun görülmüştür.

4.3. Rassal Tur Oluşturma

Matlab programla dilinde, n şehirli gezgin satıcı probleminde, rassal turlar oluşturmak için aşağıdaki gibi bir kod yazılmıştır. Oluşturulacak tur sayısı, rassal tur deneme sayısının belirlenmesinde belirlediğimiz deneme sayısını aşmayacak kadar olmalıdır. Deneme sayısı kadar rassal tur oluşturma 10. adımda kontrol edilmiştir. Ayrıca rassal turlar, belirlediğimiz eşik değerden küçük tur uzunluklarına sahip olmalıdır. Bunun kontrolü de 6. adımda yapılmıştır. Rassal tur oluşturmayı daha iyi anlayabilmek için 4 şehirli bir gezgin satıcı için, rassal tur deneme sayısının belirlenmesi sonucu 4 deneme sayısının uygun görüldüğünü farz edelim. Eşik değer olarak eşik hesaplamada bulunan $\bar{x}$=44.33 ele alalım. 2. adımda deneme sayısı 4, şehirlerarası uzaklık matrisi ve eşik değer $\bar{x}$=44.33 okunur. Deneme sayısı kadar yani 4 tane ve tur uzunlukları 44.33'den küçük rassal turlar oluşturulur. Turlar 4-1-2-3, 4-1-3-2, 1-4-3-2, 2-1-4-3 olmak üzere, deneme sayısı kadar yani 4 rassal tur, tur uzunlukları eşik değerden yani 44.33'den küçük olmak koşuluyla türetilmiştir.

Algoritma adımları aşağıda gösterilmiştir.

1.Adım: Başla,
2.Adım: Deneme sayısı, şehirlerarası uzaklık matrisi, eşik değeri oku,
3.Adım: i=1,
4.Adım: Rassal örnek oluştur(randperm kodu ile sıra halinde),
5.Adım: Rassal örnek tur uzunluğunu, tur uzunluğu fonksiyonu ile hesapla,
6.Adım: Eğer rassal örnek tur uzunluğu>= eşik değeri ise git 4'e,
7.Adım: Rassal örnekler(i)=rassal örnek,
8.Adım: Örnek tur uzunlukları(i)= rassal örnek tur uzunluğu,
9.Adım: i=i+1,
10.Adım: Eğer i<deneme sayısı ise git 4'e.
11.Adım: Rassal örnekler ve örnek tur uzunlukları yaz,
12.Adım: Bitir.

4.4. Tur Uzunluğu Hesaplama

n şehirli rassal oluşturduğumuz örneklerin tur uzunluklarını hesaplamak için aşağıdaki kod yazılmıştır. Her turda, tüm şehirlerarasındaki uzunluğu hesaplayabilmek için, şehirlerarası uzaklık matrisinden her bir şehrin diğer şehir arasındaki uzunluk değeri okunup, turdaki tüm şehirlerarasındaki uzunluklar toplanarak hesaplanır. Rassal tur oluşturmada ki örnekte oluşturulan turların uzunluğunu tur uzunluğu hesaplama kodu ile hesaplayalım. Tablo 4.1'de şehirlerarası uzaklık matrisi verilmiştir. Turlar 4-1-2-3, 4-1-3-2, 1-4-3-2, 2-1-4-3 olmak üzere, deneme sayısı kadar yani 4 rassal tur, tur uzunlukları eşik değerden yani 44.33'den küçük olmak koşuluyla türetilmişti. Şehirlerarası uzaklık matrisinden, tüm bağlı şehirlerin birbirine olan uzaklık değerleri toplanarak sırasıyla tur uzunlukları şu şekilde bulunmuştur; 40, 43, 44 ve 38.

Algoritma adımları aşağıda gösterilmiştir.

1.Adım: Başlat,
2.Adım: Rassal örnekler ve şehirlerarası uzaklık matrisi oku,
3.Adım: Şehirlerarası uzaklık matrisinden şehir sayısı, rassal örnekler matrisinden tur sayısı hesapla,
4.Adım: j=1;
5.Adım: Örnek tur uzunlukları(j)=şehirlerarası uzaklık matrisi (rassal örnekler (j,1), rassal örnekler (j,şehir sayısı),
6.Adım: i=1,
7.Adım: Örnek tur uzunlukları(j)= örnek tur uzunlukları(j)+ şehirlerarası uzaklık matrisi(rassal örnekler (j,i), rassal örnekler (j,i+1))
8.Adım: i=i+1
9.Adım: Eğer i<şehir sayısı ise git 7'ye.
10.Adım: j=j+1,
11.Adım: Eğer j<=tur sayısı ise git 5'e.
12.Adım: Örnek tur uzunlukları yaz,
13.Adım: Bitir.

4.5. Kural İlişki Matrisi

Kural ilişki matrisi, problem büyüklüğü kadar satır ve sütundan oluşmaktadır. Bu matriste her turda, tüm bağlı olan şehirler için bir oran hesaplanmıştır. Bu oran, ilgili turun ((eşik değeri-tur uzunluğu)/tur uzunluğu)'dur. Her tura ait bu oran hesaplanıp tüm bağlı olan şehirlere bu oran atanır. Diğer tüm turların oranları bu matriste bağlı olan şehirlere eklenerek, toplanır.

Örneğimizdeki; 4-1-2-3, 4-1-3-2, 1-4-3-2, 2-1-4-3 turlarını ele alalım. Kural ilişki matrisi aşağıdaki matriste gösterildiği gibi 4*4'lük bir matris olacaktır. İlk önce 4-1-2-3 turundan başlayalım, bu turun uzunluğu 40 ve eşik değeri de 44.33. Bu turun oranı ((44.33-40)/40)= 0,10825 'dur. Aşağıdaki tablo 4.6' da kural ilişki matrisinin 1. adımın da gösterildiği gibi bu tur için 4-1, 1-4, 1-2, 2-1, 2-3, 3-2 bağlarına 0,10825 değeri atanır.

Tablo 4.6: Kural ilişki matrisinin 1. adımı

	1	2	3	4
1	0	0,10825	-	0,10825
2	0,10825	0	0,10825	-
3	-	0,10825	0	-
4	0,10825	-	-	0

Sonraki tur olan 4-1-3-2 turunun uzunluğu 43 olup, tur oranı ((44.33-43)/43)=0,03 hesaplanmıştır. Bu oran tablo 4.7' de kural ilişki matrisinin 2. adımında 4-1, 1-4, 1-3, 3-1, 3-2, 2-3 bağlarına eklenir.

Tablo 4.7: Kural ilişki matrisinin 2. adımı

	1	2	3	4
1	0	0,10825	0,03	0,13825
2	0,10825	0	0,13825	-
3	0,03	0,13825	0	-
4	0,13825	-	-	0

Bir sonraki tur olan 1-4-3-2 turunun uzunluğu 44 olup, tur oranı ((44.33-44)/44)=0,0075 hesaplanmıştır. Bu oran tablo 4.8' de kural ilişki matrisinin 3. adımında 1-4, 4-1, 4-3, 3-4, 2-3, 3-2 bağlarına eklenir.

Tablo 4.8: Kural ilişki matrisinin 3. adımı

	1	2	3	4
1	0	0,10825	0,03	0,14575
2	0,10825	0	0,14575	-
3	0,03	0,14575	0	0,0075
4	0,14575	-	0,0075	0

Son tur olan 2-1-4-3 turunun uzunluğu 38 olup, tur oranı ((44.33-38)/38)=0,1665 hesaplanmıştır. Bu oran tablo 4.9' da kural ilişki matrisinin 4. adımında 2-1, 1-2, 1-4, 4-1, 4-3, 3-4 bağlarına eklenir.

Tablo 4.9: Kural ilişki matrisinin 4. adımı

	1	2	3	4
1	0	0,27475	0,03	0,31225
2	0,27475	0	0,14575	-
3	0,03	0,14575	0	0,174
4	0,31225	-	0,174	0

Kural ilişki matrisinin en son bilgisayar çıktısı, tablo 4.10' da gösterildiği gibi 4. adımda oluşturulan matris gibidir.

Tablo 4.10: Kural ilişki matrisi

	1	2	3	4
1	0	0,27475	0,03	0,31225
2	0,27475	0	0,14575	-
3	0,03	0,14575	0	0,174
4	0,31225	-	0,174	0

Algoritma adımları aşağıda gösterilmiştir.

1.Adım: Başla,

2.Adım: Rassal örnekler, örnek tur uzunlukları, eşik değeri oku,

3.Adım: Rassal örnek sayısı hesapla(rassal örnekler içerisinden oku), problem büyüklüğü hesapla,

4.Adım: i=1, j=1,

5.Adım: Kural ilişki matrisi(çözüm(i,j),çözüm(i,j+1)) = kural ilişki matrisi(çözüm(i, j), çözüm(i, j+1)) + ((eşik değeri - örnek tur uzunlukları(i)) / - örnek tur uzunlukları(i)),

6.Adım: Kural ilişki matrisi(çözüm(i,j+1),çözüm(i,j)) = kural ilişki matrisi(çözüm(i, j+1), çözüm(i, j)) + ((eşik değeri- örnek tur uzunlukları(i)) / örnek tur uzunlukları(i)),

7.Adım: Eğer j<problem büyüklüğü ise git 6'ya,

8.Adım: Kural ilişki matrisi(çözüm(i, 1), çözüm(i, problem büyüklüğü)) = kural ilişki matrisi(çözüm(i, 1), çözüm(i, problem büyüklüğü)) + ((eşik değeri - örnek tur uzunlukları(i)) / örnek tur uzunlukları(i)),

9.Adım: Kural ilişki matrisi(çözüm(i, problem büyüklüğü), çözüm(i, 1)) = kural ilişki matrisi(çözüm(i, problem büyüklüğü), çözüm(i, 1)) + ((eşik değeri - örnek tur uzunlukları(i)) / örnek tur uzunlukları(i)),

10.Adım: i=i+1,

11.Adım: Eğer i<=rassal örnek sayısı ise git 5'e.

12.Adım: Kural ilişki matrisi yaz.

13.Adım: Bitir.

4.6. Kural İlişki Matrisine Göre Tur Oluşturma

Tur oluşturmada, kural ilişki matrisinde yer alan şehirlerarasında ki en büyük oranlı iki şehir birbirine bağlanır. Daha sonra 2. en büyük orana sahip şehirler birbirine bağlanarak, tüm şehirler bitene kadar bu işleme devam edilir ve tur oluşturulur. Yazılmış olan kodda, şehirler birbirine bağlanırken, tüm şehirler bitmeden tur oluşmasını engellemek için kontroller gerçekleştiriyoruz. Kodda yazılı olan diğer uç vektörü, birbirine bağlanan 2 şehirden, diğer şehri göstermektedir. Ayrıca n şehirli bir turda en fazla (şehir sayısı*2)-2) bağ olacağı için, kodun en sonunda, "bağ sayısı vektörü toplamı [(satır sayısı*2)-2]" ile turun bitiminin kontrolünü gerçekleştiriyoruz.

Ele aldığımız örnekte bulduğumuz tablo 4.10'da ki kural ilişki matrisine göre tur oluşturalım. Kural ilişki matrisinde ki şehirlerarasında ki en büyük oran 0,31225 ile 1-4 veya 4-1 bağlarıdır. Kodlama 1. satır ve 1. sütundan başlayarak satır bazında en büyük değeri aradığı için ilk 1-4 bağını görecektir. 1 ile 4 numaralı şehirler(1-4) birbirine bağlanır ve diğer uç vektöründe tablo 4.11' de 1 numaralı şehirde 4, 4 numaralı şehirde 1 şehri yer alır.

Tablo 4.11: Diğer uç vektörünün 1. adımı

Şehir numaraları	1	2	3	4
Diğer uç vektör değerleri	4	-	-	1

Bağ sayısı vektörü de hangi şehirlerin bağlandığının kontrolünü yapar. Örneğin; 1 ile 4 numaralı şehirler birbirine bağlanmış ise, bağ sayısı vektörü tablo 4.12' de 1 ve 4 numaralı şehirde 1 rakamı yer alır.

Tablo 4.12: Bağ sayısı vektörünün 1. adımı

Şehir numaraları	1	2	3	4
Bağ sayısı vektör değerleri	1	-	-	1

Kural ilişki matrisinde diğer en büyük oranlı bağ 0,27475 ile 1-2 bağıdır. Bu durumda 2 numaralı şehir, 1 şehri ile bağlanır ve 4-1-2 turu elde edilir. Diğer uç vektörü tablo 4.13' de, 2 numaralı şehre 4 değeri, 4 numaralı şehre 2 numaralı şehir atanır. 1 numaralı şehre ise 0 değeri atanır, çünkü 1 ve 4 numaralı şehirler iki şehir arasında kalmıştır, artık uçları yoktur.

Tablo 4.13:Diğer uç vektörünün 2. adımı

Şehir numaraları	1	2	3	4
Diğer uç vektör değerleri	0	4	-	2

Bağ sayısı vektörü tablo 4.14' de, 1 numaralı şehre 2, 2 ve 4 numaralı şehre 1 atanır. Eğer bağ sayısı vektörü 2 ise, o şehir 2 bağa sahip olmuş olur ve o şehir tur oluşturma işlemine dâhil edilmez.

Tablo 4.14: Bağ sayısı vektörünün 2. adımı

Şehir numaraları	1	2	3	4
Bağ sayısı vektör değerleri	2	1	-	1

Kural ilişki matrisinde 3. en büyük değer 0,174 ile 3-4 bağıdır. Elimizde 4-1-2 turu vardı, bu durumda 3-4-1-2 turu oluşur. Diğer uç vektörü tablo 4.15' de, 3 numaralı şehre 2 değeri, 2 numaralı şehre 3 numaralı şehir atanır. 1 ve 4 numaralı şehirlerin ucu olmadığı için 0 atanır.

Tablo 4.15: Diğer uç vektörünün 3. adımı

Şehir numaraları	1	2	3	4
Diğer uç vektör değerleri	0	3	2	0

Bağ sayısı vektörü tablo 4.16' da, 1 ve 4 numaralı şehre 2, 2 ve 3 numaralı şehre 1 atanır.

Tablo 4.16: Bağ sayısı vektörünün 3. adımı

Şehir numaraları	1	2	3	4
Bağ sayısı vektör değerleri	2	1	1	2

4 şehirli bir turda en fazla 6 bağ olacağı için, kodun en son satırında "bağ sayısı vektör toplamı=[(satır sayısı*2)-2]" ile bağ sayısı vektör toplamını 6 hesaplayıp, 3-4-1-2 turunu bitirmiştir.

Kural ilişki matrisine göre tur oluşturma için yazılan kodun algoritma adımları aşağıda verilmiştir.

1.Adım: Başlat,
2.Adım: Kural ilişki matrisinden satır sayısı ve sütun sayısı oku,
3.Adım: Tur matrisini sıfır matris olarak ve diğer uç vektörü ile bağ sayısı vektörünü sıfır vektör olarak kur,
4.Adım: k=o,
5.Adım: Kural ilişki matrisinde ki en büyük değeri bul,
6.Adım: i=1'den başlasın satır sayısına kadar devam etsin,

7.Adım: j=1'den başlasın sütun sayısına kadar devam etsin,

8.Adım: Eğer en büyük değer kural ilişki matrisi(i,j)'de ise,

9.Adım: Eğer bağ sayısı vektörü(i)=0 ise,

10.Adım: Eğer bağ sayısı vektörü(j)=0 ise,

11.Adım: Diğer uç vektörü(i)=j,

12.Adım: Diğer uç vektörü(j)=i,

13.Adım: Bağ sayısı vektörü(i)=1,

14.Adım: Bağ sayısı vektörü(j)=1,

15.Adım: Tur matris(i,j)=1,

16.Adım: Tur matrisi(j,i)=1,

17.Adım: Bağ sayısı vektörü(i)=1,

18.Adım: Bağ sayısı vektörü(j)=0,

19.Adım: Diğer uç vektörü(j)= diğer uç vektörü(i),

20.Adım: Diğer uç vektörü(diğer uç vektörü(j))=j,

21.Adım: Bağ sayısı vektörü(i)=2,

22.Adım: Bağ sayısı vektörü(j)=1,

23.Adım: Tur matris(i,j)=1,

24.Adım: Tur matrisi(j,i)=1,

25.Adım: Eğer bağ sayısı vektörü(j)=1 ise,

26.Adım: Eğer diğer uç vektörü(i)≠j ise,

27.Adım: Diğer uç vektörü(diğer uç vektörü(i))= diğer uç vektörü(j),

28.Adım: Diğer uç vektörü(diğer uç vektörü(j))= diğer uç vektörü(i),

29.Adım: Diğer uç vektörü(i)=-1,

30.Adım: Diğer uç vektörü(j)=-1,

31.Adım: Bağ sayısı vektörü(i)=2,

32.Adım: Bağ sayısı vektörü(j)=2,

33.Adım: Tur matris(i,j)=1, tur matrisi(j,i)=1,

34. Adım: Bağ sayısı vektör toplamı=[(satır sayısı*2)-2] ise 5'e git.

4.7. Yerel Arama

Kural ilişki matrisine göre bulunan en küçük tur uzunluğuna sahip tura, yerel arama yapılarak, optimum tur uzunluğuna yaklaşmak yada optimum tur uzunluğu bulmak

istenmiştir. Yerel arama algoritmasında ilgili problem için bulduğumuz en küçük tur uzunluğuna sahip tur ele alınır. Bu turda ilk şehirden başlamak üzere bir şehir diğer tüm şehirler ile yer değiştirerek, tur uzunluğundaki iyileşme hesaplanır. Tur uzunluğunda azalış var ise yeni tur ve yeni tur uzunluğu bulunur. Örneğin; kural ilişki matrisinde oluşturulan 3-4-1-2 turuna yerel arama uygulayalım. 3 numaralı ilk şehirden başlanarak tüm şehirler diğer şehirler ile yer değiştirir. Elde edilen yeni turlar 4-3-1-2, 1-4-3-2, 2-4-1-3, 3-1-4-2, 3-2-1-4, 3-4-2-1 turlarıdır. Bu turların uzunluk değeri hesaplanır, tur uzunluk değerleri sırasıyla 43, 44, 45, 45, 40, 48 bulunmuştur. 3-4-1-2 turunun uzunluk değeri 38' di, yerel arama ile bulunan yeni turların uzunluğunda, veri madenciliği tabanlı yaklaşım ile bulunan tur uzunluğuna göre azalma olmamıştır. Eğer tur uzunluğunda azalma olsaydı yeni tur ve yeni tur uzunluğu yazılacaktı.

Algoritma adımları aşağıda verilmiştir.

1.Adım: Başla,
2.Adım: Rassal örnekler ve şehirlerarası uzaklık matrisi oku ve k=1,
3.Adım: Şehirlerarası uzaklık matrisinden şehir sayısını hesapla ve i=1,
4.Adım: j=i,
5.Adım: Aday rassal örnekler= rassal örnekler,
6.Adım: g= aday rassal örnekler(i),
7.Adım: Aday rassal örnekler(i)= aday rassal örnekler(j),
8.Adım: Aday rassal örnekler(j)=g,
9.Adım: Aday rassal örnekler tur uzunluklarını, tur uzunluğu hesapla ile hesapla,
10.Adım: Eğer aday rassal örnekler tur uzunlukları > örnek tur uzunlukları ise git 15'e.
11. Adım: Aday konum(k,1)=i,
12.Adım: Aday konum(k,2)=j,
13.Adım: Aday tur uzunluğu(k)= rassal örnekler tur uzunlukları - aday rassal örnekler tur uzunlukları
14.Adım: k=k+1, j=j+1,
15.Adım: Eğer j<=şehir sayısı ise git 5'e,
16.Adım: i=i+1,

17.Adım: Eğer i<şehir sayısı ise git 4'e.

18.Adım: Eğer k=1 ise git 25'e,

19.Adım: m=aday tur uzunlukları içindeki en büyük değeri bul,

20.Adım: g= rassal örnekler(aday konum(m,1)),

21.Adım: Rassal örnekler(aday konum(m,1))=rassal örnekler(aday konum(m,2)),

22.Adım: Rassal örnekler(aday konum(m,2))=g,

23.Adım: Git 2'ye,

24.Adım: Yerel arama tur uzunluğu yaz,

25.Adım: Bitir.

4.8. Problem Çözümleri

Tüm problemler için 500000 deneme sayısı ile $\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S eşik değerleri için 10'ar kez bulunan sonuçlar tablolaştırılmıştır.

4.8.1. Gr17 problemi

17 şehirli gr17 probleminde, 500000 tur deneme sayısı ile her bir eşik değeri ($\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S, $\bar{x}$-4S) için 10'ar kez bulunan sonuçlar aşağıdadır.

Tablo 4.17: Gr17 probleminin sonuçları

Eşik değeri	**Tur uzunluğu ve tekrar sayısı**	**Süre(saniye)**	**Min. tur uzunluğu ve tekrar sayısı**
$\bar{x}$**=4668**	(2232/3),(2152/3),(2168/3), (2096/1)	3.749841, 2.472621, 2.478218, 3.56382, 3.563722, 2.467332, 3.58433, 3.573334, 3.574214, 3.572691,	2096/1
$\bar{x}$**-S=4236**	(2168/4),(2152/3),(2096/2), (2104/1)	16.604280,14.371942, 9.583184, 10.483911,11.584721,10.462782, 11.578421,9.483214, 9.582713, 9.57381	2096/2
$\bar{x}$**-2S=3803**	(2096/2),(2104/4),(2152/2), (2168/2)	80.301915, 89.472912, 73.582711, 52.589433, 69.583721 63.567211, 62.573819, 51.583711, 50.584721, 62.563781	2096/2
$\bar{x}$**-3S=3370**	(2096/7),(2104/2),(2152/1)	1365.793963, 1747.100599, 1483.291554, 1216.877304, 1463.291667, 1222.112504, 1321.567313, 1248.573172, 1574.578291, 1064.462683	2096/7
$\bar{x}$**-4S=2940**	(2096/1)	189473.849822	2096

Tablo 4.17' de gr17 problemi için $\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S, $\bar{x}$-4S eşik değerleri ile bulunan tur uzunlukları, bu uzunlukların tekrar sayıları, süre, bulunan minimum tur uzunluğu ve minimum tur uzunluğunun tekrar sayısı gösterilmiştir.

Gr17 probleminin $\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S, $\bar{x}$-4S eşik değerleri ile bulunan minimum tur uzunlukları aynı olup 2096 bulunmuştur. 4668 eşik değerinde 10 tekrarda, bulunan minimum tur uzunluğu olan 2096 bir kere bulunmuşken, 3372 eşik değeri ile 10 tekrarda 7 kere bulunmuştur. Eşik değer küçüldükçe minimum tur uzunluğu olan 2096 daha fazla görülmektedir. $\bar{x}$-4S eşik değeri ile bulunan sonucun süresi 189473,849822 sn. (yaklaşık 52 saat) olduğu için, süre açısında $\bar{x}$-4S eşik değeri kullanılması uygun görülmemiştir.

4.8.2. Ulysses22 problemi

22 şehirli ulysses22 probleminde 500000 tur deneme sayısı ile her bir eşik değeri($\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S, $\bar{x}$-4S) için 10'ar kez bulunan sonuçlar aşağıdadır. Tablo 4.18' de ulysses probleminin $\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S eşik değerleri ile bulunan tur uzunlukları, tur uzunluklarının tekrar sayıları, süre, bulunan minimum tur uzunluğu ve minimum tur uzunluğunun tekrar sayısı gösterilmiştir.

Tablo 4.18: Ulysses22 probleminin sonuçları

Eşik değeri	**Tur uzunluğu ve tekrar sayısı**	**Süre(saniye)**	**Min. tur uzunluğu ve tekrar sayısı**
$\bar{x}$ **=16615**	(7089/3),(7131/1),(7087/2), (7106/3),(7112/1),	5.110164,4.872401,4.676850, 4.161646, 5.734427,4.220117, 4.112238, 3.366440, 4.238689, 4.914510	(7087/2)
$\bar{x}$ **-S=15219**	(7089/4),(7106/2),(7087/3), (7314/1)	18.529112,11.591397,10.574832, 15.584733,15.584732,14.584722, 10.675832,15.684573,14.587321, 12.583721	(7087/2)
$\bar{x}$ **-2S=13823**	(7298/2),(7314/2),(7087/3), (7089/3)	94.718894,91.715158,85.573621, 79.573611,98.573821,86.587321, 79.583713,84.564728,88.574837, 81.463718	(7087/3)
$\bar{x}$ **-3S=12427**	(7089/4),(7106/2),(7087/2), (7314/2)	1854.206384,1894.372819,1945.564728, 1957.573827,1375.549321,1854.463829, 1785.473829,1894.564738,1945.473811, 1877.566313	(7087/2)

Ulysses probleminin $\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S eşik değerleri ile bulunan minimum tur uzunlukları aynı olup 7087 bulunmuştur.

4.8.3. Bays29 problemi

29 şehirli bays29 probleminde 500000 tur deneme sayısı ile her bir eşik değeri ($\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S, $\bar{x}$-4S) için 10'ar kez bulunan sonuçlar Tablo 4.19 'da verilmiştir.

Tablo 4.19: Bays29 probleminin sonuçları

Eşik değeri	Tur uzunluğu ve tekrar sayısı	Süre(saniye)	Min. tur uzunluğu ve tekrar sayısı
$\bar{x}$**=5974**	(2234/3), (2127/1),(2139/2), (2075/1),(2210/2),(2075/1)	7.875206,5.283011,4.595056, 5.074038,6.594306,4.892126, 5.564728,4.573912,4.127495, 5.957211	(2075/1)
$\bar{x}$**-S=5545**	(2093/1),(2210/1),(2124/3), (2108/1),(2139/2),(2135/1), (2194/1)	12.094684,21.564738,13.578223, 18.453666,19.564729,18.473822, 19.564121,19.573817,16.573613, 17.539372	(2093/1)
$\bar{x}$**-2S=5116**	(2281/2),(2194/2),(2124/3), (2075/1),(2108/2)	108.350901,96.878051,115.584721, 123.564721,92.564722,104.574621, 85.573611,81.758321,101.563821, 104.583721	(2075/1)
$\bar{x}$**-3S=4687**	(2075/1),(2135/2),(2143/1), (2124/2),(2093/3),(2075/1)	1935.050628,1875.564782,1213.473611, 1422.471019,1192.462387,1118.568213, 1184.563821,1254.573819,1125.472618, 1183.482719	(2075/1)

4.8.4. Dantzig42 problemi

42 şehirli dantzig42 probleminde 500000 tur deneme sayısı ile her bir eşik değeri ($\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S, $\bar{x}$-4S) için 10'ar kez bulunan sonuçlar Tablo 4.20 'de verilmiştir.

Tablo 4.20 Dantzig42 probleminin sonuçları

Eşik değeri	Tur uzunluğu ve tekrar sayısı	Süre(saniye)	Min. tur uzunluğu ve tekrar sayısı
$\bar{x}$**=3110**	(1032/1),(1147/2),(1121/1), (1125/1),(1280/1),(1037/1), (1240/1),(1136/1),(1100/1)	6.812896,5.810944,4.730465, 4.894516, 6.292226,5.171487, 4.719247,6.239808,5.093015, 4.803412	(1032/1)

Tablo 4.20: (Devam) Dantzig42 probleminin sonuçları

$\bar{x}$ **-S=2898**	(743/2),(764/1),(731/1), (744/3),(737/3)	23.656638,22.747865,23.992156, 23.453402,13.068942,21.538291, 22.573612,11.473621,19.573642, 18.573628	(731/1)
$\bar{x}$ **-2S=2686**	(727/2),(724/1),(741/1), (725/1),(735/2),(731/1), (745/2)	126.049054,139.705830,168.574822, 124.587301,191.573191,220.573822, 120.478281,149.573819,129.572910, 162.583929	(725/1)
$\bar{x}$ **- 3S=2474**	(724/2),(740/2),(735/2), (725/3),(735/1)	2390.281611,2482.538291,2183.573990, 2184.462819,2137.573912,2194.472911, 2183.482373,2102.381101,2001.472910, 2381.716134	(724/1)

4.8.5. Eil51 problemi

51 şehirli eil51 probleminde 500000 tur deneme sayısı ile her bir eşik değeri ($\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S, $\bar{x}$-4S) için 10'ar kez bulunan sonuçlar Tablo 4.21 'de verilmiştir.

Tablo 4.21: Eil51 probleminin sonuçları

Eşik değeri	**Tur uzunluğu ve tekrar sayısı**	**Süre(saniye)**	**Min. tur uzunluğu ve tekrar sayısı**
$\bar{x}$ **=1652**	(584/1),(641/1),(780/1), (727/1),(668/1),(635/1), (778/1),(719/1),(677/1), (717/1)	8.694818,6.778106,7.765396,7.711532, 7.776608,7.635983,7.154743,7.419305, 7.471454, 8.609664	(584/1)
$\bar{x}$ **-S=1563**	(445/3),(452/2),(447/3), (430/2)	25.063917,24.680485,24.903057, 23.675832, 25.674983,14.684921, 24.674831,15.673913,23.573621, 16.473292	(430/2)
$\bar{x}$ **-2S=1474**	(428/3),(430/2),(432/3), (429/2)	340.700332,313.573821,308.572915, 236.472914,237.57392,239.562914, 239.573792,240.573921,278.572915, 280.572914	(428/3)
$\bar{x}$ **-3S=1385**	(428/4),(429/3),430/3)	3042.472945, 28392.482937, 2194.582911, 3012.582391, 3321.458291, 2143.582101, 3141.482910, 3147.573929, 2135.582017, 2015.382618	(428/4)

4.8.6. Brasil58 problemi

58 şehirli eil58 probleminde 500000 tur deneme sayısı ile her bir eşik değeri ($\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S, $\bar{x}$-4S) için 10'ar kez bulunan sonuçlar Tablo 4.22 'de verilmiştir.

Tablo 4.22: Brasil58 probleminin sonuçları

Eşik değeri	Tur uzunluğu ve tekrar sayısı	Süre(saniye)	Min. tur uzunluğu ve tekrar sayısı
$\bar{x}$ **=123660**	28685, 28509, 29741, 28904,27551, 27183, 27036, 27029,27179, 27427	9.986057,9.669700,10.05224, 10.002831,9.595454,9.572915, 9.583201,9.184927,8.472915, 8.463929	27036
$\bar{x}$ **-S=116629**	27046, 27858, 26765, 26922,26928, 26490, 26359, 26837,26318, 26113	25.761379,25.295170,26.573924, 28.684021,26.573920,25.673921, 27.572391,24.673911,23.583433, 13.674394	26113
$\bar{x}$ **-2S=109598**	26770, 27007, 27005, 26682,26619, 26686, 27010, 26769,25943, 26936	248.576590,299.674921,259.381013, 263.482921, 239.482390, 238.475329, 285.572925, 334.574238, 333.483092, 278.684930	25943
$\bar{x}$ **-3S=102567**	26409, 26833, 26113, 26654,26318, 26036, 26770, 27112,27083, 26837	3888.854391, 3672.914826,2395.195391, 3185.381675, 3271.582372,2758.482572, 3282.579163,2237.537754, 3381.722184, 3163.583291	26036

4.8.7. St70 problemi

70 şehirli st70 probleminde 500000 tur deneme sayısı ile her bir eşik değeri ($\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S, $\bar{x}$-4S) için 10'ar kez bulunan sonuçlar Tablo 4.23 'de verilmiştir.

Tablo 4.23: St70 probleminin sonuçları

Eşik değeri	Tur uzunluğu ve tekrar sayısı	Süre(saniye)	Min. tur uzunluğu ve tekrar sayısı
$\bar{x}$ **=3657**	765, 832, 783, 773, 751, 824, 745, 787, 735, 757	10.548566,11.022121,10.853538, 10.841141, 10.854875,10.696851, 10.591441,10.741308, 9.895321, 10.572945	735
$\bar{x}$ **-S=3477**	765,719, 751, 763, 724/2, 727, 745, 749, 739	27.013705, 26.992593, 25.684930, 25.482014, 26.483262, 25.547320, 25.673923, 24.673921,27.573912, 26.472657	719
$\bar{x}$ **-2S=3297**	705,742, 730, 718, 756, 758, 733/2, 710,729	367.822372, 381.002376, 283.763935, 381.763935, 482. 973772, 380.805769, 368.558112, 341.362910, 381.699768, 329.467105	705
$\bar{x}$ **-3S=3117**	739, 725, 721, 735, 750, 727, 739, 746, 717, 705	3738.630113, 3946.214654, 3426.357432, 3382.185285, 3684.210582, 3803.274272, 3375.582194, 3948.583185, 3483.284687, 3722.560571	705

4.8.8. Eil76 problemi

76 şehirli eil76 probleminde 500000 tur deneme sayısı ile her bir eşik değeri($\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S, $\bar{x}$-4S) için 10'ar kez bulunan sonuçlar Tablo 4.24 'de verilmiştir.

Tablo 4.24: Eil76 probleminin sonuçları

Eşik değeri	Tur uzunluğu ve tekrar sayısı	Süre(saniye)	Min. tur uzunluğu ve tekrar sayısı
$\bar{x}$**=2523**	603, 600, 590, 599, 591, 599, 594, 601, 624, 616	11.111194, 11.318636, 11.360362, 11.128697, 11.037332, 11.386302, 11.047909, 11.214405, 10.137995, 11.268355	590
$\bar{x}$**-S=2410**	592, 584, 597, 590, 585, 575, 570, 583, 581, 585	28.730362, 30.290992, 27.806795, 28.857946, 38.583920, 39.739202, 30.573924, 40.573920, 30.573910, 31.573291	570
$\bar{x}$**-2S=2297**	561, 564, 572, 570, 565, 583, 573, 567, 574, 567	379.618559, 381.588702, 382.322561, 363.247264,352.680779, 384.007299, 362.581328, 376.780430, 363.184001, 381.191937	561
$\bar{x}$**-3S=2184**	559, 562, 567, 568, 565, 569, 567, 563, 573, 581	3712.428648, 3938.53154, 3938.531541, 3753.321687, 3975.425765, 3974.436526, 3895.4263754, 3746.755578, 3537.545754, 3878.543656	559

4.8.9. Gr96 problemi

Tablo 4.25: Gr96 probleminin sonuçları

Eşik değeri	Tur uzunluğu ve tekrar sayısı	Süre(saniye)	Min. tur uzunluğu ve tekrar sayısı
$\bar{x}$**=366520**	60318, 60489, 63613, 61149, 61331, 61804, 60056, 59086, 60530, 61196	12.573247, 13.684314, 12.573930, 12.684933, 13.539235, 12.573921, 12.573925, 12.539935, 12.583920, 13.568403	59086
$\bar{x}$**-S=349961**	59756, 59980, 59795, 59938, 59935, 58888, 58867, 61889,59890, 60790	89.937528, 91.916075, 95. 475871, 91.059170, 92.586391, 93.275681, 79.361038, 89.593183, 91.584625, 95.583601	58260
$\bar{x}$**-2S=333402**	60968, 60170,59183, 60423, 61746, 58314, 61559, 58227, 57386, 60266	350.283401, 344.971389, 347. 481279, 376.584145, 358.882312, 419.552210, 357.973374, 358.515183, 357.963559, 456.138213	57386
$\bar{x}$**-3S=316843**	59938, 58589, 59423, 59935, 59890, 57925, 58867, 58837, 63452, 60666	8.184.196.350, 7.584.215.325, 7.634.346.421, 8.156.325.235, 7.535.215.646,8.146.654.646, 8.134.654.534, 7.346.456.276, 8.253.253.554, 7.953.634.166	57925

96 şehirli gr96 probleminde 500000 tur deneme sayısı ile her bir eşik değeri($\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S, $\bar{x}$-4S) için 10'ar kez bulunan sonuçlar yukarıda Tablo 4.25 'de verilmiştir.

4.8.10. Rd100 problemi

100 şehirli rd100 probleminde 500000 tur deneme sayısı ile her bir eşik değeri($\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S, $\bar{x}$-4S) için 10'ar kez bulunan sonuçlar Tablo 4.26 'da verilmiştir.

Tablo 4.26: Rd100 probleminin sonuçları

Eşik değeri	Tur uzunluğu ve tekrar sayısı	Süre(saniye)	Min. tur uzunluğu ve tekrar sayısı
$\bar{x}$=55562	9241, 9131, 9152, 9108, 9051, 9299, 8632, 9233, 8874, 8893	14.573946, 13.674924, 13.684921, 13.583024, 14.105736, 13.584028, 13.574302, 14.044547, 14.649340, 13.574906	8632
$\bar{x}$-S=53215	8532, 8628, 8723, 9133, 8578, 8908, 8861, 8760, 8871, 8783	81.088462, 91.539382, 98.537592, 112.458362, 100.775932, 99.578292, 111.573920, 105.482659,89.583920, 96.573920	8532
$\bar{x}$-2S=50868	8574, 8684, 8652, 8780, 9016, 8649, 9096, 8991, 9018, 8606	414.294743,447.532749, 397.385734, 384.384702, 387. 407017, 381.573910, 469.329055, 399.049345, 481.919375, 392.160174	8574
$\bar{x}$-3S=48521	8467, 9299, 8467, 8628, 8795, 8536, 8489, 8893, 8841, 8278	8.473.482.534, 8.975.322.556, 7.984.535.667, 8.146. 577.566, 8.136.757.565,8.135.656.675, 8.145.657.767,8.987.436.324,8.535.355.313, 8.454.641.113	8278

4.8.11. Ch130 problemi

130 şehirli ch130 probleminde 500000 tur deneme sayısı ile her bir eşik değeri($\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S, $\bar{x}$-4S) için 10'ar kez bulunan sonuçlar Tablo 4.27 'de verilmiştir.

Tablo 4.27: Ch130 probleminin sonuçları

Eşik değeri	Tur uzunlukları	Süre(saniye)	Min. tur uzunluğu ve tekrar sayısı
$\bar{x}$=46308	6980, 7041, 7316, 7750,7942, 7875, 7456, 8123,7926, 8120	55.495194,53.478923,57.488922, 54.398951,41.125784,51.473848, 51.476562,44.562611,55.472311, 43.58332	6980/1

Tablo 4.27: (Devam) Ch130 probleminin sonuçları

$\bar{x}$**-S=44643**	7053, 7095, 7097, 7051,7196, 6999, 7033, 7318,7085, 6893	152.632672,134.567211,141.473821, 132.471814,138.572619,139.573819, 137.573812,143.478881,139.483721, 146.57477	6893/1
$\bar{x}$**-2S=42978**	6711, 6632, 6999, 7033, 6818, 6860, 6702, 6800,6910, 6797	477.418276,432.473385,422.473822, 442.928337,590.979360,457.482716, 462.467732,472.481723,442.471198, 421.48773	6632/1
$\bar{x}$**-3S=41313**	6622, 6555, 6648, 6498, 6753, 6734, 6761, 6702, 6903, 6580	9668,236554,8735.851890,7593.281945, 8478.487821,8822.578833,7473.471823, 9874.584211,9845.398812,8873.581921, 9743.557122	6498/1

4.8.12. Si175 problemi

175 şehirli si175 problemi için bilgisayar hafızası yetersiz geldiği için 500000 tur deneme sayısı yerine 400000 deneme sayısı yapılmıştır. 400000 tur deneme sayısı ile $\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S eşik değerleri için 10'ar kez bulunan sonuçlar Tablo 4.28 'de verilmiştir.

Tablo 4.28: Si175 probleminin sonuçları

Eşik değeri	**Tur uzunlukları**	**Süre(saniye)**	**Min. tur uzunluğu ve tekrar sayısı**
$\bar{x}$**=48120**	22308, 22170, 22363, 22325, 22181, 22197, 22346, 22303, 22277, 22271	69.533030,72.240576,72.501441, 68.204934,78.697424,74.346969, 74.667136,75.301215, 76.608719, 77.781966,	22170
$\bar{x}$**-S=47364**	21922, 22020,21939, 21990, 21933, 21996, 22185, 21960, 21938, 22060	197.600503,186.033018,185.755521, 184.215198,184.638391,188.072044, 186.784164, 171.268037,168.330465, 193.396772	21922
$\bar{x}$**-2S=46608**	21858, 22041, 21999, 21882, 21968, 21997, 21990, 21967, 21847, 22060	483.399712, 479.360472, 475.676889, 538.110258, 532.272178, 536.261484, 536.341353, 532.382471, 521.844070, 531.846985	21847
$\bar{x}$**-3S=45852**	21854, 21717, 21978, 21814, 21831, 21815, 21898, 21721, 21791, 21762	9553.873912, 9764.792749, 9648.382103, 10474.381295, 12336.583025, 9649.342394, 1007.372944, 10599.739573, 10504.483025, 9622.573920	21717

4.8.13. Gr202 problemi

202 şehirli gr202 probleminde bilgisayar hafızası yetersiz geldiği için 350000 tur deneme sayısı ile $\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S eşik değerleri için 10'ar kez bulunan sonuçlar Tablo 4.29 'da verilmiştir.

Tablo 4.29: Gr202 probleminin sonuçları

Eşik değeri	Tur uzunlukları	Süre(saniye)	Min. tur uzunluğu ve tekrar sayısı
$\bar{x}$ **=264620**	52176, 52081,54810, 52658, 52951,52793, 54486, 52235, 52353, 50385	62.768964, 67.707978, 70.276080, 85.040257, 75.172514, 73.997648, 76.772421, 83.478780,72.446645, 89.569834	50385
$\bar{x}$ **-S=257858**	46485, 47728, 48164, 47575, 47328, 47831, 47869, 49172, 47636, 47444	150.646912, 158.448951,169.598316, 171.754221, 170.853569, 163.222282, 173.324717, 187.622975, 186.472352, 158.952055	46485
$\bar{x}$ **-2S=251096**	45650, 46796, 46872, 46261, 47564, 47310, 47945, 46152, 46237, 48073,	742.401708, 556.610372, 526.472955, 525.573920,581.483045, 581.473922, 575.749302, 558.472920, 527.463829, 527.473292	45650
$\bar{x}$ **-3S=244334**	46527, 46170, 44238, 44854, 43725, 44797, 45386, 45486, 43593, 43548	12302.473929, 10361.464836, 11256.257495, 10319.473902, 10354.573922, 10848.573921, 13453.46044, 10575.473299, 11575.267440, 11481.008549	43548

4.8.14. Pr299 problemi

299 şehirli pr299 problemi için bilgisayar hafızası yetersiz geldiği için 250000 tur deneme sayısı ile $\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S eşik değerleri için 10'ar kez bulunan sonuçlar Tablo 4.30 'da verilmiştir.

Tablo 4.30: Pr299 probleminin sonuçları

Eşik değeri	Tur uzunlukları	Süre(saniye)	Min. tur uzunluğu ve tekrar sayısı
$\bar{x}$ **=759710**	142025, 144437, 14262, 145580, 144150, 141042, 138532, 139050, 143503, 143447	111.009145, 129.856518, 106.900751, 96.862616, 106.258395, 115.184193, 97.677032, 89.726319, 93.027396, 93.135415,	138532
$\bar{x}$ **-S=737224**	116229, 109658, 112598, 109826, 121164, 111226, 116604, 118174, 118597, 123950	198.125484, 195.012126, 175.665996, 165.234493, 138.397301, 138.080049, 148.826016, 155.261594, 151.830366, 205.626083	109826
$\bar{x}$ **-2S=714738**	93254, 97580, 95733, 91701, 94172, 95066, 91407, 100191,94219, 93325	674.721382, 610.277480, 575.835963, 575.905831, 580.533410, 574.590869, 574.721382, 579.811782, 582.566500, 583.614803	91407
$\bar{x}$ **-3S=692252**	75405, 81638, 81933, 83027, 82279, 85841, 85607, 84313, 79009 80351	13018.860810, 12600.52553, 11623.52504, 11528.75478, 11592.52638, 11786.25193, 10635.684166, 10878.642576, 10823.990808, 10825.500686	75405

4.8.15. Lin318 problemi

318 şehirli lin318 problemi için bilgisayar hafızası yetersiz geldiği için 150000 tur deneme sayısı ile $\bar{x}$, $\bar{x}$-S, $\bar{x}$-2S, $\bar{x}$-3S eşik değerleri için 10'ar kez bulunan sonuçlar tablo 4.31'de verilmiştir.

Tablo 4.31: Lin318 probleminin sonuçları

Eşik değeri	**Tur uzunlukları**	**Süre(saniye)**	**Min. tur uzunluğu ve tekrar sayısı**
$\bar{x}$ **=587970**	133054, 141671, 134604, 136082, 138822, 133738, 138783, 132985,139598, 133820	152.531547, 156.951607, 127.369227, 135.884313, 128.548393,111.563829, 118.573920, 120.473920, 138.545382, 148.171671	132985
$\bar{x}$ **-S=573833**	85455, 103514, 101204, 106070, 106666, 105592, 104273, 106462,100673, 106463	225.035451,216.616537,190.815371, 192.466520,207.379585,179.539375, 186.473922,194.574829,185.743921, 184.674935	85455
$\bar{x}$ **-2S=559696**	71528, 63379,72449, 71221, 72574, 69494, 70926, 72812, 72869 82671	848.837718,861.025068,594.894164, 619.894164,618.712232,602.215427, 613.745307,619.408910,601.700343, 650.536454	63379
$\bar{x}$ **-3S=545559**	62134, 57493,58505, 57053, 57169,60699, 58001, 58592,60380, 61120	13061.540863,12979.800686,12609.148380, 12678.002117,13658.472914,13462.567493, 14457.547383,14574.329101,13574.320113, 14274.472911	57053

4.9. Yerel arama

Her şehir için veri madenciliği tabanlı yaklaşım (V.M.T.Y.) ile bulunan minimum tur uzunluğuna yerel arama yapılmıştır. Tablo 4.32'de problem adları, V.M.T.Y. ile bulunan minimum tur uzunluğu, yerel arama(Y.A.) ile bulunan minimum uzunluğu, problemlerin bugüne kadar bilinen optimum tur uzunlukları, V.M.T.Y. ile bulunan minimum tur uzunluğunun hatası(%), Y.A. ile bulunan tur uzunluğunun hatası(%) gösterilmiştir. V.M.T.Y. ile bulunan min. tur uzunluğunun Hatası(%) ve Y.A. ile bulunan min. tur uzunluğunun Hatası(%)'nın formülasyonu aşağıda gösterilmiştir.

V.M.T.Y. ile bulunan min. tur uzunluğunun Hatası(%)=[(V.M.T.Y. bulunan min. tur uzunluğu- Optimum tur uzunluğu)/ Optimum tur uzunluğu]*100

Y.A. ile bulunan min. tur uzunluğunun Hatası(%)=[(Y.A. ile bulunan tur uzunluğu - Optimum tur uzunluğu)/ Optimum tur uzunluğu]*100

Tablo 4.32: Yerel arama sonuçları

Problem adı	**V.M.T.Y. ile bulunan min. tur uzunluğu**	**Y.A. ile bulunan min. tur uzunluğu**	**Optimum tur uzunluğu**	**V.M.T.Y. ile bulunan min. tur uzunluğunun hatası(%)**	**Y.A. ile bulunan tur uzunluğunun hatası(%)**
Gr17	2096	2095	2085	0,5275	0,4796
Ulysses22	7089	7083	7013	0,1083	0,09981
Bays29	2075	2046	2020	2,7722	1,2871
Dantzig42	724	724	699	3,5765379	3,5765379
Eil51	428	428	426	0,4694836	0,4694836
Brasil58	25943	25813	25395	2,1579	1,16460
St70	705	693	675	4,4444	2,6666
Eil76	559	551	538	3,9	2,41
Gr96	57386	56020	54645	5,016	2,5
Rd100	8278	8165	7910	4,6523	3,2238
Ch130	6499	6349	6110	6,3666	5,9410
Si175	21717	21641	21407	1,4481	1,0931
Gr202	43548	42224	40160	8,4362	5,1394
Pr299	75405	58705	48191	56,4711	21,8174
Lin318	57169	49281	42029	36,0227	17,2548

Dantzig42 ve eil51 hariç diğer tüm problemlerde Y.A. ile bulunan tur uzunluğunun hatasının, V.M.T.Y. ile bulunan minimum tur uzunluğunun hatasından daha küçük olduğu ve Y.A. ile bulunan tur uzunluğunun ilgili problemin optimumuna daha yakın olduğu görülmüştür.

V.M.T.Y. ve Y.A. ile bulunan tur uzunluğu hataları 299 şehirli gezgin satıcı problemine kadar en fazla %8 bulunmuş ve başarılı sonuçlar elde edilmiştir. Fakat 299 ve 318 şehirli gezgin satıcı problemi için elde edilen sonuçların optimumdan sapması yüksek bulunmuştur.

5. SONUÇLAR VE ÖNERİLER

Gerçek hayatta G.S.P. olarak düşünülebilecek çok sayıda problem mevcuttur. Bu problemlerin ifadesi G.S.P.' nin klasik tanımı ile benzeşmese de G.S.P. formülasyon ve algoritmaları bu tür problemlere daha doğru sonuçlar vermektedir. Böylelikle problemin türüne bağlı olarak, gerek süre gerekse maliyet açısından daha avantajlı olunabilmektedir. Günümüzde G.S.P. için birçok çözüm yöntemi bulunmaktadır, gelişmenin durmayacağı ve gerçek hayata çok daha uygun yeni problem türleriyle, bilgisayar teknolojisindeki gelişmelere paralel olarak, farklı çözüm yöntemlerinin türetileceği açıktır.

Bu çalışma iki temel kavram olan Gezgin Satıcı Problemi ve Veri Madenciliği üzerinde durmaktadır. Burada ki her iki kavram merak uyandıran isimlerinden dolayı popüler kazanan ve gelişen kavramlardır. Bu yüzden çalışmada G.S.P. için veri madenciliği tabanlı bir yaklaşım denenmiştir. Bu yaklaşımda 15 ayrı G.S.P. için, eşik değer yani tur uzunluğunun belirli bir değerin altında kalması şartıyla 500000 tane rassal tur Matlab programlama dili ile üretilmiştir. Turların uzunluk değeri bulunup eşik değere göre ağırlıklandırılarak, yeni 500000 tur oluşturulmuş böylelikle daha kaliteli turlara sahip olunmuştur. Bu turların tur uzunlukları, matlab programında ne kadar sürede bulunduğu hesaplanmıştır. Bu turların içinde minimum tur uzunluğuna sahip tura yerel arama yapılarak sonucun etkinliği artırılmaya çalışılmış ve optimum sonuç ile karşılaştırılmıştır. Uygulanan yöntemin küçük boyutlu problemler için başarılı sonuçlar verdiği, fakat büyük boyutlu problemlerde optimumdan önemli ölçüde saptığı görülmektedir.

Ayrıca ekte kodu verilen değiştirilmiş kural ilişki matrisi ile gr202 problemi 264620 eşik değeri için denenmiştir. Bu kural ilişki matrisinde tur uzunluk sayıları düşük olan az sayıda tura daha yüksek oranlar atanarak, daha az sayıda tur ile daha başarılı sonuçlar hedeflenmiştir.

Tablo 4.33: Gr202 problemi için farklı kural ilişki matrisi sonuçları

Deneme sayısı	Kural ilişki matrisi	Tur Uzunluğu ve tekrar sayısı	Süre(saniye)
100000	2. kural ilişki matrisi	62349, 61327, 64228, 62706, 63711	183.573930, 174.295736, 195.383264, 159.674932, 174.5662814
350000	2. kural ilişki matrisi	51381, 53121, 52989, 51772, 54290	455.033387, 519.564883, 467.371547, 486.748290, 461.573932
350000	Tezde kullanılan kural ilişki matrisi	52176, 52081, 54810, 52658, 52951,52793, 54486, 52235, 52353, 50385	62.768964, 67.707978,70.276080, 85.040257, 75.172514,73.997648, 76.772421, 83.478780,72.446645, 89.569834

Ek tablo 1'de de görüldüğü üzere 100000 deneme sayısı ile tur uzunluğu sonuçları 350000 deneme sayısı ile bulunan sonuçlara göre daha kötüdür. 350000 deneme sayısında her iki kural ilişki matrisi için yakın değerler bulunmuştur. Fakat tezde kullanılan kural ilişki matrisi daha kısa sürede sonuç bulmaktadır.

Ayrıca eil51 ve ch130 problemleri üzerinde deneme sayısının etkisinin irdelendiği 2 tablo aşağıda verilmiştir.

Tablo 4.34: Eil51 probleminin farklı deneme sayıları sonuçları

Deneme sayısı	Tur Uzunluğu ve tekrar sayısı
100000	461, 448, 445/2, 450, 444, 453, 441, 470, 440
200000	443/2, 439, 428, 439, 445, 449, 436, 449/2
300000	428/2, 443, 446, 445/2, 438/2, 433, 439
400000	452/3, 445, 446, 452, 440, 430, 445, 453

Tablo 4.35: Ch130 probleminin farklı deneme sayıları sonuçları

Deneme sayısı	Tur Uzunluğu ve tekrar sayısı
100000	8962, 8457, 8591, 8173, 8682, 9122, 8426, 8467, 7781, 8572
200000	7738, 8114, 8136, 7693, 7877, 8253, 7449, 7842, 7946, 8252
300000	7250, 7503, 7420, 7215, 7126, 7560, 7503, 7445, 7409, 7475
400000	6952/2, 7232, 7203/2, 7316, 7374, 7211, 7036, 7351

Eil51 problemini ele aldığımızda ek tablo 1'de 100000 deneme sayısı ile 200000 deneme sayısı sonuçlarının korelasyonu 0,38, 100000 ile 300000 deneme sayısı sonuçlarının korelasyonu -0,53, 100000 ile 400000 deneme sayısı sonuçlarının korelasyonu 0,05, 200000 ile 300000 deneme sayısı sonuçlarının korelasyonu -0,42, 200000 ile 400000 deneme sayısı sonuçlarının korelasyonu 0,27, 300000 ile 400000 deneme sayısı sonuçlarının korelasyonu -0,10'dur.

Ch130 problemini ele aldığımızda ek tablo 2'de 100000 deneme sayısı ile 200000 deneme sayısı sonuçlarının korelasyonu 0,27, 100000 ile 300000 deneme sayısı sonuçlarının korelasyonu 0,06, 100000 ile 400000 deneme sayısı sonuçlarının korelasyonu 0,17, 200000 ile 300000 deneme sayısı sonuçlarının korelasyonu 0,39, 200000 ile 400000 deneme sayısı sonuçlarının korelasyonu 0,01, 300000 ile 400000 deneme sayısı sonuçlarının korelasyonu 0,28'dir. Bu sonuçlara göre deneme sayısı artıkça korelasyonda bir artış görülmemiştir, korelasyon bizim için belirleyici bir katsayı oluşturamamıştır, sebebi de her seferinde rassal veri üretmemizdir.

Fakat eil51 problemi için deneme sayısının artışı tur uzunluğunda anlamlı bir iyileştirme yapmamışken, ch130 probleminde deneme sayısının artışı tur uzunluğunda iyileşme sağlamıştır. Bu sonuçlardan kısmen küçük ölçekli problemlerde deneme sayısının artışı tur uzunluğu iyileşmesinde anlamlı katkısı olmadığını fakat büyük ölçekli problemler için tur deneme sayısının artışı tur uzunluğu iyileşmesinde anlamlı katkı sağladığını söyleyebiliriz.

Veri madenciliği tabanlı yaklaşımın kısa zamanda optimum sonuca ulaşması ve çözüm uzayının büyüklüğü ne olursa olsun çalışma süresinin makul sınırlar içerisinde kalması istenir. Kritik bir işlem için uzun zaman beklenemeyeceği gibi, sonucunda istenilen verimlilikte olması istenmektedir. Yapılan tüm deney sonuçlarının daha da arttırılması ve daha küçük eşik değerleri için daha az sürede tekrarlanması sağlanarak yeni modellerin etkinliği konusunda daha güvenilir değerlerin elde edileceği açıktır. Gelecekle ilgili bir çalışma konusu da veri madenciliğinin büyük ölçekli problemlerde iyi sonuçlar verecek şekilde kullanılması olabilir.

Ayrıca G.S.P.'nin çözümünde veri madenciliği yaklaşımlarının performanslarının değerlendirildiği çalışma bulunmamaktadır. Gelecekte yapılacak çalışmalarda daha iyi yerel aramalar veya olasılıksal yapıların kullanılması ile üretilen sonuçların kalitesinde yapılacak değerlendirmeler sayesinde, veri madenciliğinin optimizasyon problemlerine çözüm üretmedeki yetenekleri ve diğer yaklaşımlar üzerine avantajları açığa çıkarılmış olacaktır.

KAYNAKLAR

Hui S., Jha G., "Application data mining for customer service support", ***Information and Management***, 38: 1-13 (2000).

Anand A., "A study and comparison of data clustering techniques", **Master Thesis,** ***Faculty of The Graduate School of The University of Texas El Paso*****,** Texas, 21-22 (2003).

San O., Huynh V., Nakamori Y., "An alternative extention of the k means algorithm for clustering categorical data", ***Applied Math. and Computer Science,*** 14 (2): 241-247 (2004).

Bland C., "The discovery of multiple level profile association rules", Doctoral Thesis, ***Graduate School of Computer and Information Science of University of Missisipi***, Missisipi, 43-50 (2002).

Hudairy H., "Data mining and decision making support in the governmental sector", Master Thesis, ***Faculty of Graduate School of The University of Louisville***, Kentucky, 1-5 (2004).

Huberty C., "Applied Discriminant Analysis", ***John Wiley & Sons Inc***., Canada, 25-35 (1994).

Wei C., Chiu T., "Turning telecommunications call details to churn prediction: a data mining approach,", ***Expert Systems with Applications***, 23: 103-102 (2002).

Aryeetey K., "Data analysis and predictive modelling using the variable precision rough set approach", **Master Thesis,** ***Faculty of Graduate Study and Research of University of Regina***, Canada, 28-33 (2003).

Pawlak Z., "Rough sets, decision algorithms and bayes theorem", ***Europen Journal of Operation Research***, 136: 181-189 (2002).

Shah S., Kursak A., "Data mining and genetic algorithms based gene / SNP selection", ***Artificial Intelligence in Medicine,*** 31: 183 -196 (2004).

Hui S., Jha G., "Application data mining for customer service support", ***Information and Management***, 38: 1-13 (2000).

Yalçıntaş G.,"Veri Madenciliği", Yüksek Lisans Tezi, ***Gazi Üniversitesi Fen Bilimleri Enstitüsü***, Ankara, 6-7 (2003).

Tantuğ A.C. ," Veri Madenciliği ve Demetleme" , Yüksek Lisans Tezi, ***İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü***, İstanbul,3 (2002).

Goebel, M., Gruenwald, L., "A Survey of Data Mining and Knowledge Discovery Software Tools", ***ACM SIGKDD Explorations Newsletter***, 1, 1, pp. 20-33, (1999).

Bayram E., "Customer segmentation and churn modeling in wireless communication", Yüksek Lisans Tezi, ***Boğaziçi Üniveristesi Fen Bilimleri Enstitüsü***, İstanbul, 1-5 (2001).

Restivo K., "The drill on data mining", ***Computer Dealer News***, 15 (14): 29-30(1999).

Bilen Ö., "ÖSS Sınav Sonuçlarının Okul Bazında Veri Madenciliği ile İncelenmesi", Yüksek Lisans Tezi, ***Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü***, İstanbul,10-13 (2004).

Rushing, J., 1997, CS 687 Technology Assessment Paper, www.cs.uah.edu/~thinke/CS 687/Fall 97/Tech/Rushing.html **(Ziyaret tarihi:18.Mart 2008).**

Han, j. And Kamber, M. , "Data Mining- Concept, Techniques, academic" ***PRESS, USA***, 550p (2001).

E. Alpaydın, "Zeki Veri Madenciliği- Ham veriden altın Bilgiye Ulaşma Yöntemleri" , Boğaziçi Üniversitesi, ***Bilişim 2000 Eğitim Semineri***, (2000).

H. Akpınar, "Veri Tabnalarında Bilgi Keşfi ve Veri Madenciliği", İstanbul Üniversitesi, ***İşletme Fakültesi Dergisi***, C29, S:1-22, (2000).

Berson,Alex- Smith, Stephen- Thearling, Kurt, ***Building Data Mining applications for CRM***, McGraw- Hill, USA, (2000).

Freitag alex A., Data Mining and Knowledge Discovery with Evolutionary algorithms***.Springer- Verlag Berlin Heidelberg***,Germany, (2002).

Giudici, Paolo, Applied Data Mining, Statistical Methods for Business and Industry.***John Wiley – Sons***, england, (2004).

Javovic, N.- Milutinovic, V. –Obradovic, Z. "Foundations of Predictive Data Mining, Member", ***6th seminar on neural network applications in electrical engineering Neural,*** 53, -58,IEEE, (2002).

Akbulut S.," Veri Madenciliği Teknikleri ile Bir Kozmetik Markanın Ayrılan Müşteri Analizi ve Müşteri " , Yüksek Lisans tezi, ***Gazi Üniversitesi Fen Bilimleri Enstitüsü***, (2006)

Özmen, Şule (2001), İş Hayatı Veri Madenciliği ile İstatistik Uygulamalarını Yeniden Keşfediyor. suleozmen.com.tr/teblig sunumlar/3is hayati veri madenciligi.ppt **(Ziyaret tarihi :5 Mart 2008).**

Davis B., "Data mining transformed*",* ***Information Week***, 751: 86 (1999). Ronald Swift; Accelerating Customer Relationship; *Prentice Hall PTR,*(2001).

Fayyad,U.M.;Piatesky-Shapiro,G.;Smyth,P. "From Data Mining to Knowledge Discovery in Databases", ***USA,AAAI Pres***,(1996).

Han,J;Kamber,W. "Data Mining Concepts and Techniques", ***Morgan Kaufmann Publishers Inc.***,s 63,(2001).

İnternet: Current data mining applications / percentage in different industries, http://www.kdnuggets.com/polls/ 2003/data mining applications industries.html **(Ziyaret tarihi:13 Mart 2005).**

Aydoğan F., "E-ticarette veri madenciliği yaklaşımlarıyla müşteriye hizmet sunan akıllı modüllerin tasarımı ve gerçekleştirimi", Yüksek Lisans Tezi, ***Hacettepe Üniversitesi Fen Bilimleri Enstitüsü,*** Ankara, 12-16 (2003).

ALATAŞ, Bilal – Akın, Erhan; Veri Madenciliğinde Yeni Yaklaşımlar, ***Ya/Em-2004- Yöneylem Araştırması/Endüstri Mühendisliği Xxıv Ulusal Kongresi***, 15-18 Haziran, Gaziantep-Adana (2004)

Piramuthu S., "Evaluating feature selection for learning in data mining applicatyions",***Thirty-Fisrt annual Hawai International Conference on System Sciences,*** 5:294(1998).

Aydoğan F., "E-ticarette veri madenciliği yaklaşımlarıyla müşteriye hizmet sunan akıllı modüllerin tasarımı ve gerçekleştirimi", Yüksek Lisans Tezi**,** ***Hacettepe Üniversitesi Fen Bilimleri Enstitüsü,*** Ankara, 12-16 (2003).

Shearer C., "The Crisp-DM model: The new blueprint for data mining", ***Journal of Data Warehousing,*** 5 (4): 13-23 (2000).

Adriaans, P., Zantinge, D.,Data Mining, ***Addison Wesley Longman Limited,*** **Harlow**, 159, (1996)

Gondran M., Minoux M., Graphs and Algorithms, ***John Wiley and Sons Inc***., 650 Sayfa, New York, (1984).

Sipahioğlu A, "Gezgin Satıcı ve Araç Turu Belirleme Problemleri İçin Yeni Alt Tur Engelleme Kısıtları*",* ***Osmangazi Üniversitesi, Fen Bilimleri Enstitüsü,*** Doktora Tezi, 80 (1996).

BAYZAN Şahin.,"Araç Rotalarının En kısa Yol algoritmaları Kullanılarak Belirlenmesi ve .Net Ortamında Simülasyonu" , Yüksek Lisans Tezi**,** ***Pamukkale Üniversitesi Fen Bilimleri Enstitüsü***, Denizli,17-32(2005) .

Yeniay M.Ö."Gezgin Satıcı Problemi" Yüksek Lisans Tezi, ***Hacettepe Üniversitesi Fen Bilimleri Enstitüsü***, Ankara,12-16,(1994).

Minieka E., "Optimization algorithms for network and graphs", ***Marcel Dekker Inc.,*** New York, 355 sayfa (1978).

Gondran M., Minoux M.,Vajda S., , Graphs and Algorithms, ***John Wiley& Sons*****, Newyork**, 650 sayfa (1984).

Canen A.G., Scott L.G., "Bridging theory and practise in vehicle routing problem", ***JORS***, 46,1-8 (1995).

Lawler E.L., Lenstra J.K., Rinnooy Kan A.H.G., Shmoys D.B.," The traveling salesman problem", **John Wiley&Sons,** Newyork,450 sayfa (1985).

Weisstein, Eric W., 1999, Complexity Theory. From MathWorld--A Wolfram Web Resource. http://mathworld.wolfram.com/ComplexityTheory.html **(Ziyaret tarihi: 10 Ocak 2008).**

Eiselt, H.A. and Sandblom, C.L., "Integer programming and network models", ***Springer-Verlag***, Berlin,315-341 (2000).

Lenstra, J.K., Rinnoy Kan, A.H.G., "some simple applications of the travelling salesman problem", ***Opl Res. Q.***,26, 717-733, (1975).

Arthur, J.L., Frendewey, J.O., "Generating travelling salesman problems with known optimal tours", ***J.Opl.Res. Soc***., 39, 153-159, (1988).

Belmore,M.Nemhauser, G.L., "The traveling salesman problem", ***Operations research***, 16, 538-558, (1968).

Christofides, N., "Bounds for the traveling salesman problem", ***Operations Research***, 20, 1044-1056, (1972).

Dantzig, G., Fulkerson, R.,Johnson, S., "Solution of large scale traveling salesman problem", ***Operations Research***, 393-410, (1954).

Golden, B., Bodin, L., Doyle,T.,Stewart, W., "Approximate traveling salesman algorithms", ***Operations Rese***arch, 694-712, (1980).

Ong, H.L.,Huang, H.C., "Asymptotic expected performance of some T.S.P heuristics", ***European Journal of Operational Research***,43, 231-238,(1989).

Lin S, Kernighan B., "An effective heuristic algorithm for the traveling salesman problem." **Operations Research**, 498–516 (1973).

Or I. “Traveling salesman-type combinatorial problems and their relation to the logistics of regional blood banking.” ***Ph.D. thesis, Evanston, IL: Northwestern University,*** (1976).

Knox J. “Tabu search performance on the symmetric traveling salesman problem.” ,***Computers and Operations Research,*** 21, (1994).

Tsubakitani S, Evans J. “Optimizing tabu list size for the traveling salesman problem.” ***Computers and Operations Research,*** 25, (1998).

Bui T, Moon B. “A new genetic approach for the traveling salesman problem.” ***Proceedings of the first IEEE conference on evolution computation*** , 7–13, (1994).

Chatterjee S, Carrera C, Lynch L. “Genetic algorithms and traveling salesman problems.” ***European Journal of Operational Research,*** 490–510, (1996).

Poon P, Carter J. “Genetic algorithm crossover operations for ordering applications.” ***Computers and Operations Research,*** 135–47, (1995).

Potvin J. “Genetic algorithm for the traveling salesman problem.” ***Annals of Operations Research,*** 63-70, (1996).

Schmitt L, Amini M. “Performance characteristics of alternative genetic algorithmic approaches to the traveling salesman problem using path representation: an mpirical study**.”** ***European Journal of Operational Research,*** 108, (1998).

Alaykıran K., Engin O. “Karınca Kolonileri Meta-Sezgiseli ve Gezgin Satıcı Problemleri Üzerinde Bir Uygulaması.” ***Gazi. Üniv. Müh. Mim. Fak. Dergisi*** 21:1 69-76, (2005).

Stützle T., Hoos TH., “The MAX-MIN Ant System and Local Search for the Traveling Salesman Problem***.” In Proceedings of the Fourth International Conference on Evolutionary Computation (ICEC'97),*** 308-313, (1997)

Dorigo M., and Gambardella L. M., “Ant colonies for the traveling salesman problem.” ***Biosystems*** ,73-81, (1997).

Bauk, Sanja; Nataša Kova; “Modelıng Shıp’s Route By The Adaptatıon of Hopfıeld - Tank T.S.P Neural Algorıthm*”, **Journal of Maritime Research***, Vol. I. No. 3, 51-66, (2004).

Koulamas C.,Antony S.R., Jean R.,”A survey of simulated annealing applications to operations research problems”, ***Omega,*** 22, 41-56, (1994).

Chee-kit L., Neural Networks methods in combinatorial optimization, ***Comp. & Ops. Res.***, 3-4, 191-208, (1994).

Burke L.I., Ignizio J.P., Neural Networks and operations research: An overview, ***Comp.&Ops. Res***.,3-4, 179-189, (1992).

Öztemel, Ercan; Yapay Sinir Agları***, Papatya Yayıncılık***, 1. baskı, İstanbul, (2003).

Sağıroğlu, Seref; Erkan Besdok; Mehmet Erler; Mühendislikte Yapay Zeka Uygulamaları-I: Yapay Sinir Agları***, Ufuk Kitap Kırtasiye Yayıncılık***, 1. Baskı, Kayseri, (2003).

SEZEN, Kemal; Dinamik Programla, ***Ekin Yayınevi***, Bursa, 1998

Golden B.L., Wong R.T., 1981, Capaciatated are routing problems, ***Network***, 11, 305-315.

İnternet: "Seyyar Satıcı Problemi-vikipedi, Vikipedi özgür ansiklopedi", http://tr.wikipedia.org/wiki/Seyyar sat%C4%B1c%C4%B1 problemi.**(Ziyaret tarihi:30 Kasım 2008).**

EKLER

Aşağıda Matlab dili ile kodlanmış tüm program kodları verilmiştir.

Eşik Hesaplama

```
function [standart_sapma] = esik_hesapla(uzmat, deneme_sayisi)
   [sehir_sayisi, n] = size(uzmat);
   t = 0;
   for i=1:deneme_sayisi
      l = randperm(n) ;
      uygunluk(i)  = uygunluk_degeri_hesapla_gsp(l, uzmat);
      t = t + uygunluk(i) ;
   end
   esik_orta = t / deneme_sayisi
   kare_toplam = 0;
   for i=1:deneme_sayisi
      kare_toplam = kare_toplam + ((uygunluk(i)- esik_orta) * (uygunluk(i)-
esik_orta));
   end
   standart_sapma = sqrt(kare_toplam / (deneme_sayisi - 1))
end
```

Rassal Tur Oluşturma

```
function [cozum uygunluk degerleri] = rassal olustur(uygunluk matrisi, tekrar sayisi,
esik)
   [boyut1, boyut2] = size(uygunluk matrisi);

   cozum = zeros(tekrar sayisi, boyut1);
   uygunluk degerleri = zeros(tekrar sayisi, 1);

   i = 1;
   while i <= tekrar sayisi
      rassal vektor = randperm(boyut1);
      uygunluk rassal vektor = uygunluk degeri hesapla gsp(rassal vektor, uygunluk
matrisi);
      if(uygunluk rassal vektor <= esik)
         cozum(i, :) = rassal vektor;
         uygunluk degerleri(i) = uygunluk rassal vektor;
         i = i + 1;
      end
   end
end
```

Kural İlişki Matrisi

```
function [kural iliski matrisi] = kural iliski matrisi kur(cozum, uygunluk degerleri, esik degeri)

   [cozum veri sayisi, problem buyuklugu] = size(cozum);
   kural iliski matrisi = zeros(problem buyuklugu, problem buyuklugu);

   for i=1:cozum veri sayisi
      for j=1:problem buyuklugu-1
         kural iliski matrisi(cozum(i, j), cozum(i, j+1)) = kural iliski matrisi(cozum(i, j), cozum(i, j+1)) + ((esik degeri - uygunluk degerleri(i)) / uygunluk degerleri(i));
         kural iliski matrisi(cozum(i, j+1), cozum(i, j)) = kural iliski matrisi(cozum(i, j+1), cozum(i, j)) + ((esik degeri - uygunluk degerleri(i)) / uygunluk degerleri(i));
      end
      kural iliski matrisi(cozum(i, 1), cozum(i, problem buyuklugu)) = kural iliski matrisi(cozum(i, 1), cozum(i, problem buyuklugu)) + ((esik degeri - uygunluk degerleri(i)) / uygunluk degerleri(i));
      kural iliski matrisi(cozum(i, problem buyuklugu), cozum(i, 1)) = kural iliski matrisi(cozum(i, problem buyuklugu), cozum(i, 1)) + ((esik degeri - uygunluk degerleri(i)) / uygunluk degerleri(i));
   end
end
```

Kural İlişki Matrisine Göre Tur Oluşturma

```
function tur = tur olustur en buyuk(kural iliski matrisi)
   [satir sayisi, sutun sayisi] = size(kural iliski matrisi);
   tur = zeros(satir sayisi, sutun sayisi);

   diger uc vektoru = zeros(satir sayisi, 1);
   bag sayisi vektoru = zeros(satir sayisi, 1);

   while (sum(tur(:)) ~= (satir sayisi * 2) - 2)
      en buyuk deger = max(max(kural iliski matrisi));
      for i=1:satir sayisi
         for j=1:sutun sayisi
            if (en buyuk deger == kural iliski matrisi(i, j))
               if (bag sayisi vektoru(i) == 0)
                  if (bag sayisi vektoru(j) == 0)
% İki nokta da bağlı değilse
                     diger uc vektoru(i) = j;
                     diger uc vektoru(j) = i;
                     bag sayisi vektoru(i) = 1;
                     bag sayisi vektoru(j) = 1;
                     tur(i, j) = 1;
                     tur(j, i) = 1;
```

```
                elseif (bag sayisi vektoru(j) == 1)
% Bir nokta bağlı ise
                    diger uc vektoru(i) = diger uc vektoru(j);
                    diger uc vektoru(diger uc vektoru(j)) = i;
                    diger uc vektoru(j) = -1;
                    bag sayisi vektoru(i) = 1;
                    bag sayisi vektoru(j) = 2;
                    tur(i, j) = 1;
                    tur(j, i) = 1;
                end
            elseif (bag sayisi vektoru(i) == 1)
                if (bag sayisi vektoru(j) == 0)
                    diger uc vektoru(j) = diger uc vektoru(i);
                    diger uc vektoru(diger uc vektoru(j)) = j;
                    diger uc vektoru(i) = -1;

                    bag sayisi vektoru(i) = 2;
                    bag sayisi vektoru(j) = 1;
                    tur(i, j) = 1;
                    tur(j, i) = 1;
                elseif (bag sayisi vektoru(j) == 1)
% İki taraftada bir bağ var ise
% Diğer uç aynı olmamalı
                    if(diger uc vektoru(i) ~= j)
% Diğer uç aynı değilse
                        diger uc vektoru(diger uc vektoru(i)) = diger uc vektoru(j);
                        diger uc vektoru(diger uc vektoru(j)) = diger uc vektoru(i);
                        diger uc vektoru(i) = -1;
                        diger uc vektoru(j) = -1;
                        bag sayisi vektoru(i) = 2;
                        bag sayisi vektoru(j) = 2;
                        tur(i, j) = 1;
                        tur(j, i) = 1;
                    else
% matris simetrikse ve diğer uç aynı ise
                        kural iliski matrisi(i, j) = 0;
                        kural iliski matrisi(j, i) = 0;
                    end
                end
            end
% matris simetrikse
            kural iliski matrisi(i, j) = 0;
            kural iliski matrisi(j, i) = 0;
        end
    end
  end
%    diger uc vektoru'
end
```

Tur Uzunluğu Hesaplama

```
function [ud] = uygunluk degeri hesapla gsp(cozum, uzaklik matrisi)
   [sehir sayisi, sehir sayisi] = size(uzaklik matrisi);
   tursay=size(cozum,1);
   for j=1:tursay
      ud(j) = uzaklik matrisi(cozum(j,1), cozum(j,sehir sayisi));

      for i=1:sehir sayisi-1
         ud(j) = ud(j) + uzaklik matrisi(cozum(j,i), cozum(j,i+1));
      end
   end

end
```

Yerel Arama

```
function cozum = yerel arama 2li(cozum, uzaklik matrisi, m)
   if(m >= 490)
      return;
   else
      m = m + 1;
   end

   [sehir sayisi, sehir sayisi] = size(uzaklik matrisi);
   cozum uygunluk degeri = uygunluk degeri hesapla gsp(cozum, uzaklik matrisi);
   k = 1;
   for i=1:sehir sayisi-1
      clear aday;
      clear aday uygunluk degeri;
      for j=i:sehir sayisi
         aday cozum = cozum;
         g = aday cozum(i);
         aday cozum(i) = aday cozum(j);
         aday cozum(j) = g;
         aday cozum uygunluk degeri = uygunluk degeri hesapla gsp(aday cozum,
uzaklik matrisi);
         if(aday cozum uygunluk degeri < cozum uygunluk degeri)
            aday konum(k, 1) = i;
            aday konum(k, 2) = j;
            aday uygunluk(k) = cozum uygunluk degeri - aday cozum uygunluk
degeri;
            k = k + 1;
         end
      end
   end
   if(k > 1)
      l = find(aday uygunluk == max(aday uygunluk));
```

```
        g = cozum(aday konum(l, 1));
        cozum(aday konum(l, 1)) = cozum(aday konum(l, 2));
        cozum(aday konum(l, 2)) = g;
        cozum = yerel arama 2li(cozum, uzaklik matrisi, m);
    end
end
```

Değiştirilmiş Kural İlişki Matrisi

```
function [kural_iliski_matrisi] = kural_iliski_matrisi_kur_duzeltilmis(cozum, uygunluk_degerleri, esik_degeri)
[cozum_veri_sayisi, problem_buyuklugu] = size(cozum);
kural_iliski_matrisi = zeros(problem_buyuklugu, problem_buyuklugu);

tek_uygunluk_degerleri = unique(uygunluk_degerleri);
[tek_deger_sayisi g] = size(tek_uygunluk_degerleri);
tek_uygunluk_degerleri_sayisi_carpimi = 0;

for i=1:tek_deger_sayisi
tek_uygunluk_degerleri_sayisi(i)=sum((find(uygunluk_degerleri==tek_uygunluk_degerleri(i))) > 0);
tek_uygunluk_degerleri_sayisi_carpimi = tek_uygunluk_degerleri_sayisi_carpimi + log(tek_uygunluk_degerleri_sayisi(i));
end

 for i=1:cozum_veri_sayisi

k=log(tek_uygunluk_degerleri_sayisi(find(tek_uygunluk_degerleri==uygunluk_degerleri(i))));
kural_iliski_matrisi(cozum(i, j), cozum(i, j+1)) = kural_iliski_matrisi(cozum(i, j), cozum(i, j+1)) + (((esik_degeri - uygunluk_degerleri(i)) / uygunluk_degerleri(i)) * (tek_uygunluk_degerleri_sayisi_carpimi - k));
 kural_iliski_matrisi(cozum(i, j+1), cozum(i, j)) = kural_iliski_matrisi(cozum(i, j+1), cozum(i, j)) + (((esik_degeri - uygunluk_degerleri(i)) / uygunluk_degerleri(i)) * (tek_uygunluk_degerleri_sayisi_carpimi - k));
 end
kural_iliski_matrisi(cozum(i, 1), cozum(i, problem_buyuklugu)) = kural_iliski_matrisi(cozum(i, 1), cozum(i, problem_buyuklugu)) + (((esik_degeri - uygunluk_degerleri(i))/uygunluk_degerleri(i))*(tek_uygunluk_degerleri_sayisi_carpimi - k));
 kural_iliski_matrisi (cozum (i, problem_buyuklugu), cozum(i, 1)) = kural_iliski_matrisi(cozum(i, problem_buyuklugu), cozum(i, 1)) + (((esik_degeri - uygunluk_degerleri(i))/uygunluk_degerleri(i))*(tek_uygunluk_degerleri_sayisi_carpimi - k));
end
end
```

ÖZGEÇMİŞ

1984 yılında Mardin'de doğdu. İlk ve orta öğrenimini Sakarya'da tamamladı. 2002 yılında girdiği Gazi Üniversitesi İstatistik Bölümünden, 2006 yılında bölüm 2.'si olarak mezun oldu. Lisans eğitimi sırasında Ankara'da özel bir araştırma şirketinde part time SPSS analisti olarak görev yaptı. 2006 yılında Kocaeli Üniversitesi Endüstri Mühendisliğinde yüksek lisansa başladı. 2007 yılında Sakarya'da gıda sektöründe bir fabrikada üretim planlama uzman yardımcısı olarak çalıştı. 2008 yılında Tübitak yüksek lisans bursu ile yüksek lisans öğrenimine devam etti.