T.C.
HALİÇ ÜNİVERSİTESİ
FEN BİLİMLER ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ PROGRAMI

GENETİK ALGORİTMALARIN OPTİMAL GÜZERGAH BELİRLENMESİNE UYGULANMASI

YÜKSEK LİSANS TEZİ

Hazırlayan
İsmail KAYA

Danışman
Yrd. Doç. Dr. Ulviye HACIYEVA

İstanbul - 2012

T.C.

HALİÇ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRLÜĞÜNE

Bilgisayar Mühendisliği Anabilim Dalı Bilgisayar Mühendisliği Programı Tezli Yüksek Lisans öğrencisi **İsmail KAYA** tarafından hazırlanan **"Genetik Algoritmaların Optimal Güzergah Belirlenmesine Uygulanması"** adlı bu çalışma jürimizce Yüksek Lisans Tezi olarak kabul edilmiştir.

Sınav Tarihi : 18.01.2012

(Jüri Üyesinin Ünvanı , Adı , Soyadı ve Kurumu) :	İmzası :
Jüri Üyesi: Yrd.Doç.Dr.Ulviye HACIYEVA **Dan.-HAL.Üniv.Bilgisayar Müh. ABD Öğr.Üyesi**	.....
Jüri Üyesi : Yrd.Doç.Dr.Oğuz KARAN **HAL.Üniv. Bilgisayar Müh. ABD Öğr.Üyesi**	.....
Jüri Üyesi : Yrd.Doç.Dr.Murat BEKEN **HAL.Üniv. Matematik ABD Öğr.Üyesi**	.....
Jüri Üyesi :Prof.Dr.Mübariz EMİNLİ **HAL.Üniv. Bilgisayar Müh.ABD Öğr.Üyesi (Yedek)**	.....
Jüri Üyesi : Yrd.Doç.Dr.M.Burcu I.YAZICIOĞLU **HAL.Üniv. Mol.Biy.ve Gen. ABD Öğr.Üyesi (Yedek)**	.....

ÖNSÖZ

"Genetik Algoritmaların Optimal Güzergah Belirlenmesine Uygulanması" isimli araştırma Haliç Üniversitesi Fen Bilimler Enstitüsü Bilgisayar Mühendisliği Ana Bilim Dalı Yüksek Lisans Programı'nda tez olarak hazırlanmıştır.

Araştırmanın her aşamasında desteğini esirgemeyen danışmanım Haliç Üniversitesi öğretim üyesi Yrd. Doç. Dr. Ulviye HACIYEVA'ya şükranlarımı sunarım.

Beni hiçbir zaman yalnız bırakmayan, maddi ve manevi desteklerinden dolayı eşim Sevcan KAYA'ya teşekkür ederim.

İstanbul, 2012 İsmail KAYA

İÇİNDEKİLER

KISALTMALAR

GA : Genetik Algoritmalar
VB : Ve Benzeri
DNA : Deoxyribo Nucleic Acid
EA : Evrim Algoritması
GP : Genetik Programlama
EP : Evrimsel Programlama
EV : Evrimsel Strateji
PC : Personal Computer
GSP : Gezgin Satıcı Problemi
TSP : Traveling Salesman Problem

TABLOLAR

Sayfa No.

ŞEKİLLER

Sayfa No.

GENEL BİLGİLER

Adı ve Soyadı : İsmail KAYA
Anabilim Dalı : Bilgisayar Mühendisliği
Programı : Bilgisayar Mühendisliği
Tez Danışmanı : Yrd. Doç. Dr. Ulviye HACIYEVA
Tez Türü ve Tarihi : Yüksek Lisans – Ocak 2012

ÖZET
GENETİK ALGORİTMALARIN OPTİMAL GÜZERGAH BELİRLENMESİNE UYGULANMASI

Bu çalışmanın amacı İETT'nin uygulama alanında çalışan denetim ekibinin durak, hat, araç ve işletmeleri denetlemesi için gereken optimal güzergahın belirlenmesi hedeflenmektedir. Bu kapsamda en çok bilinen ilişkisel optimizasyon problemi olan Gezgin Satıcı Problemi belirlenmiş olup çözüm yöntemi olarak genetik algoritmalar kullanılmıştır. Gezgin Satıcı Problemi (Traveling Salesman Problem) veya kısaca GSP; aralarındaki mesafeleri bilinen belirli sayıdaki şehirlerin (nokta, düğüm veya parça gibi) her birinden yalnızca bir defa geçerek başlangıç şehrine dönen en kısa turun bulunmasına denir. Gezgin Satıcı Problemlerinin pek çok çözüm yöntemi vardır. GSP'de kullanılan kesin çözüm algoritmaları, orta büyüklükteki problemleri çözmek için çok uzun sürelere ihtiyaç duyduğu için pek tatmin edici değildir. Bunun yerine Sezgisel yöntemler ise kısa sürede optimuma yakın sonuçlar vermektedir. Bu çalışmada GSP çözüm yöntemi olarak sezgisel algoritmalardan genetik algoritmalar yöntemi ile çözülmüştür. Genetik Algoritmalar (GA) doğal evrimin teorisinden esinlenilerek biyolojik süreci modelleyerek fonksiyonları optimize eden evrim algoritmalarıdır. Genetik algoritmalar özellikle; kaynak tahsisi, iş atölyesi çizelgelemesi, makine parça gruplaması ve bilgisayar ağ tasarımı, lojistik alanda, okul otobüsü rotalaması, posta kutusuna dağıtım problemleri, çöp toplama, baskı devre kartlarının montajı veya muayenesi, ders programı hazırlama, GSM operatörlerinin baz istasyonlarının yerleşim yerlerinin belirlenmesi problemi, malzeme akış sistem tasarımı, araç rotalama problemleri, depolardaki vinç güzergâhlarının programlaması, stok alanındaki malzeme toplama problemleri, uçaklar için havaalanı rotalaması, elektronik devre tasarımı gibi çeşitli alanlarda uygulanmaktadır.

Bu çalışmada: optimizasyonun ne olduğu ve bunların bir çeşidi olan genetik algoritmaların tarihçesi, tanımı, süreci, kodlama yöntemleri, operatörleri, parametreleri, avantaj ve dezavantajları, uygulama alanları hakkında bilgi verilmiştir. Gezgin satıcı problemin tarihçesi, çözüm yöntemleri, literatürdeki çalışmalar hakkında detaylı bilgi verilmiştir. Uygulama kısmında ise optimal güzergah belirlenmesi için genetik algoritma çözüm yöntemi ile Micirosoft Visual Studio 2010 ortamında Asp.Net, C# programlama dili, Java Script ve Google Maps Apisi ve Microsoft SQL Server 2008 veri tabanı kullanılarak bir yazılım geliştirilmiştir.

Anahtar Kelimeler: Genetik Algoritmalar, Gezgin Satıcı Problemi, Optimal Güzergah Belirlenmesi, Optimizasyon,

GENERAL INFORMATION

Name and Surname : İsmail KAYA
Field : Computer Engineering
Program : Computer Engineering
Supervisor : Assist. Prof. Dr. Ulviye HACIYEVA
Degree Awarded and Date : Master of Science – January 2012

SUMMARY

OPTIMAL GENETIC ALGORITHMS DETERMINING THE IMPLEMENTATION OF ROUTE

The purpose of this study İETT application stops working in the field audit team, line, equipment, and businesses need to oversee the optimal route will be determined. In this context, the most well-known traveling salesman problem with relational optimization problem is defined as a method of solution used in genetic algorithms.Traveling Salesman Problem (Traveling Salesman Problem), or GSP for short, known distances between them a certain number of cities (points, nodes, or as part) only once from each of the short tour through the initial finding is returning to the city. Travelling Salesman Problem has many solutions method. GSP algorithms used in the final solution, to solve problems of medium size needed for very long periods is not very satisfactory. Instead, heuristic methods results in the shortest period of time is near optimum. GSP in this study as a method of solution is solved with heuristic algorithms such as genetic algorithms method. Genetic Algorithms (GA), inspired by the biological process of natural evolution, the theory of evolution by modeling functions that optimize the algorithm. Genetic algorithms, in particular, resource allocation, job shop scheduling, machine parts grouping and computer network design, logistics area, school bus rotalaması, mail box, distribution problems, garbage collection, assembly, or inspection of printed circuit boards, curriculum development, placement of base stations of GSM operatorsLocating the problem, the material flow system design, vehicle routing problems, scheduling routes crane warehouses, inventory problems in the collection of material, rotalaması airport for aircraft, applied in various fields such as electronic circuit design.

In this study, which is a variant of genetic algorithms and their optimization is what the history, definition, process, coding methods, operators, parameters, advantages and disadvantages, are given information on the application areas. The history of traveling salesman problem, the solution methods, are given detailed information about the studies in the literature. In the second part of the solution of genetic algorithm to determine the optimal route in Visual Studio 2010 environment Micirosoft Asp.Net, C # programming language, java Script and apis google maps and the software has been developed using Microsoft SQL Server 2008 database.

Keywords: Genetic Algorithms, Traveling Salesman Problem , Optimal Route Determination, Optimization.

1. GİRİŞ

Optimal güzergahın belirlenmesi günümüzde gittikçe gelişmekte ve önem kazanmaktadır. Düzgün bir güzergahın belirlenmesi, zamanında varış noktasına ulaşılması, kısa mesafede yakıt tasarrufunun yapılması büyük önem taşımaktadır.

Güzergah belirleme yıllardan beri insanların üzerinde çalıştığı bir konudur. Bu konuda birçok araştırmalar yapılmış ve yapılmaya devam etmektedir. Bu araştırmalar daha çok Gezgin Satıcı Problemi olarak karşımıza çıkmaktadır. Bu tür problemler için kullanılan modellerin çoğu elle çözümü olanaksız denecek kadar zor, diğer taraftan ise bilgisayar yardımı ile çözümü uzun süreler alan problemlerdir.

Gezgin Satıcı Problemi matematiksel modellerle çözümü oldukça uzun zaman almaktadır. Matematiksel çözümlerde problem alt gruplara ayrılmakta ve her gruba farklı yöntemler uygulanmaktadır.

Günümüzde karmaşık matematiksel çözümler yerini bilgisayar kullanımının artmasıyla sezgisel yöntemler almıştır. Sezgisel yöntemlerden biri genetik algoritmalardır. Bu optimizasyon yöntemi evrim teorisinden esinlenerek ortaya atılmıştır. Gezgin Satıcı Problemin çözümünde GA yaygın olarak kullanılmaktadır. Genetik algoritmanın kullanım nedeni kolay ve hızlı bir yöntem olmasındandır.

GA optimizasyon problemi olarak tasarlanan fonksiyonlar, amaç fonksiyonu formüllerini ve kısıtlayıcı fonksiyonları içermektedir. Yapılan optimizasyonun amacı, minimum amaç değerlerini kısıtları sağlayacak şekilde bulmaktır.

İETT ulaşım sektöründe hizmet veren bir kamu kurumudur. Bu kurum İstanbul'un hemen hemen her noktasında hizmet vermektedir. Bu hizmetin aksamaması için sürekli bir denetim ekibi tarafından sahada denetlenmektedir.
Bu tezde ana amaç belirli noktaları denetleyen ekibin optimal güzergah kullanarak noktaları (Durakların) hızlı ve kısa sürede kontrol etmeleri için etkin bir yöntemin geliştirilmesidir. Daha sonra optimal güzergahının bulunmasını sağlayan genetik algoritma temelli yöntemini uygulayan etkin bir yazılımın geliştirilmesidir.

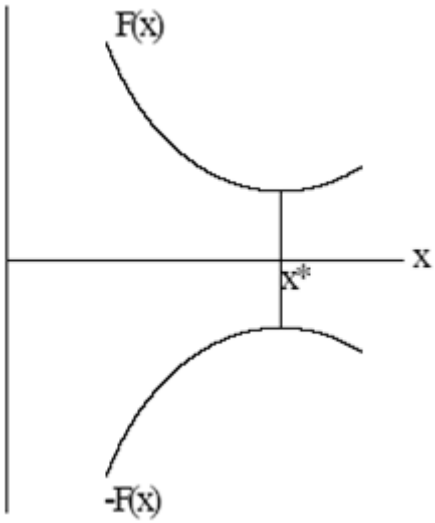
Tezin ikinci bölümünde optimizasyon hakkında bilgi verilmiştir. Üçüncü bölümde genetik algoritmalar hakkında detaylı bilgi verilmiştir. Dördüncü bölümde gezgin satıcı problemi, gezgin satıcı probleminin tarihçesi, gezgin satıcı problemi için çözüm yöntemleri, literatürde gezgin satıcı problemi çalışmaları konuları açıklanmıştır. Beşinci bölümde yapılan uygulama ayrıntılı olarak açıklanmış, bu probleme genetik algoritmanın uygulama nedenleri, problemin tanıtımı ve uygulamanın tanıtımı yapılmıştır. GA-temelli yönteminin GSP'ye uygulanması örnek ile açıklanmıştır. Son bölümde ise yapılan çalışmalarının sonuçları verilmiştir.

2. OPTİMİZASYON

Optimum kelimesi Latince bir kelime olup nihai ideal manasına gelmektedir. Optimizasyon ise bir problemin mümkün çözümleri arasından en iyisinin belirlenmesi işlemine denir. Optimizasyon problemlerinin çözümü belirli sınırlamaları sağlayacak şekilde matematiksel ifadelere veya kurallara dayanan algoritmalarla mümkün olmaktadır. Tüm optimizasyon problemlerinin çözümünde etkin bir metot yoktur ve kullanılan metotlar gerçek çözümü bulmayı garanti etmezler. Ancak makul bir çözüm bulmayı hedefler.

Matematiksel bir ifade ile optimizasyon sayısal bir fonksiyonu maksimize veya minimize etme probleminin çözümü için bir küme dahilindeki izin verilen değerlerini sistematik bir şekilde kullanarak arama işlemi olarak da tanımlanabilir. Bir denklemin kök belirleme ve optimizasyonu birbirine benzetebileceğimiz kavramlardır. Kök belirleme bir fonksiyonun veya fonksiyonların sıfır olduğu noktaların aranmasını içerir. Optimizasyonda ise minimum veya maksimum noktaların aranması söz konusudur. Optimum nokta denilince akla bu noktadaki f(x)'in türevinin sıfır olduğu x değerine karşı gelir. Ayrıca f''(x) de yani ikinci türev de optimum un maksimum veya minimum olduğunu belirtir. Eğer f''(x)<0 ise nokta maksimumdur, f''(x)>0 ise nokta minimumdur. f'(x)=0 kök problemini çözerek optimumu bulmaya yarar.

Çoğu optimizasyon metotları, kök veya sıfır araştırma işlemini kullanır. Optimizasyon için türevin sıfir olduğu yerleri araştırmak gerekmektedir. Teknik problemlerin birçoğu köklerini bulmak üzere formülize edilebilir. Fakat bir kısım optimizasyon yöntemleri bu kökleri bulmada yetersiz kalmaktadır (Pierre, 1992). Optimizasyonda diğer bir zorluk; elde edilen bir sonucun, global veya lokal bir çözüm olup olmadığının belirlenmesidir, özellikle de lineer olmayan bir fonksiyon için. Bulunan kökün optimal bir çözüm olduğunu anlamak zordur. Çünkü bütün kökler, fonksiyonu sıfır yapmaktadır. Bu tip problemler ya lineer bir yaklaşımla veya optimizasyon bölgesini küçük bir bölge ile sınırlamakla çözülür.



Şekil 2.1. Optimizasyon eğrisi

Şekil 2.1'de görülen x* noktası F(x) fonksiyonun minimum noktası ise -F(x) fonksiyonunun maksimum noktasıdır. Bir problemin birden fazla çözümü varsa en iyi çözümü bulmak gerekir. Bazı problemlerin tam cevabı bulunurken, bazıları optimal noktalar olarak bilinen değişik minimum ve maksimum noktalarına sahiptir. Optimizasyon ile bu noktalar bulunabilir. Optimizasyon temel aşamaları Şekil 2.2'de gösterilmiştir.

Şekil 2.2. Optimizasyonun temel aşamaları (Haupt, 1998)

Bilimin gelişmesiyle birlikte geçmişte çözülemeyen lineer olmayan optimizasyon problemler için yeni çözümler üretilmiştir. Gelişmenin bu sürecinde optimizasyon önemli bir rol oynamıştır.

Farklı optimizasyon problemleri için farklı metotlar ve farklı çözüm yöntemleri geliştirilmiştir. Bunlardan bazılar; Stokastik, İstatistiksel, Deterministik, Matematiksel, Sezgisel gibi optimizasyon çözüm yöntemleridir (Haataja, 1994). Sezgisel yöntemlerden karmaşık ve zaman alan bir problemin çözümünde en çok tercih edilen ve son zamanlarda yaygın olarak genetik algoritmalar kullanılmaktadır (Palko, 1996). Günümüzde genetik algoritmalar hakkında çok çeşitli araştırmalar yapılmaktadır (Broyden, 1965). Son yıllarda bilgisayarların hızlarındaki artış bu algoritmaların uygulama sahasında sıkça görülmesine neden olmuştur (Wurtz ve diğ., 1997). Özellikle kombinasyonel optimizasyon problemlere yaklaşık iyi sonuçlar bulmayı hedefleyen arama yöntemlerinden biri de genetik algoritmalardır.

3. GENETİK ALGORİTMALAR

Genetik Algoritmalar doğal evrim süreçlerini modelleyerek olası çözüm uzayında optimum çözüm arayan ve etkin çözümler sunan bir araştırma tekniğidir. Çalışmanın bu bölümünde genetik algoritmanın tarihçesini, tanım ve temel kavramlarına, evrimsel genetik işlemlere, bir problemin Genetik Algoritma (GA) ile çözüm aşamalarına, GA'nın avantajları ve dezavantajlarına yönelik bilgiler verilmektedir.

3.1. Genetik Algoritmanın Tarihçesi

Evrimsel hesaplama, kökeni 1950'lere kadar dayanmasına rağmen son on yıldır daha çok araştırılmaya başlanmıştır. Bilgisayarın kullanılmaya başlanması ve bilimin gelişmesi ile evrimsel hesaplama önemli derecede hız kazanmıştır (Parlak, 2007).

Evrimsel hesaplama ilk olarak 1960'larda I. Rechenberg tarafından "Evrim Stratejileri" isimli eserinde tanıtılmıştır. I. Rechenberg'in tezinden sonra başka araştırmacıların da ilgisini çekmiş ve geliştirilmiştir. Gelişimsel süreci taklit eden GA hakkında ilk çalışmalar 1975 yıllında Michigan Üniversitesinde psikoloji ve bilgisayar bilimi uzmanı olan John Holland tarafından yapılmıştır. Makina öğrenme (machine learning) konusunda çalışan Holland, Darwin'in evrim teorisinden etkilenerek canlılarda yaşanan genetik süreci bilgisayar ortamında gerçekleştirmeyi düşündü.

John Holland 1975 yılında yaptığı çalışmaları "Adaptation in Natural and Artificial Systems" adlı kitabında bir araya getirmiştir. Ancak 1985 yılında Holland'ın öğrencisi olarak doktorasını yapan David Edward Goldberg adlı inşaat mühendisi 1989'da bir klasik sayılan kitabını yayınlayana dek genetik algoritmaların pek pratik yararı olmayan bir araştırma konusu olduğu düşünülüyordu. Genetik Algoritma ilk olarak Hollanda'da makine öğrenme sistemlerine yardımcı olarak

kullanılmış, daha sonra David Edward Goldberg ve diğerleri tarafından analiz edilmiştir. David Edward Goldberg tezinde gaz boru hattının kontrolünü içeren bir problemin çözümünü genetik algoritma ile gerçekleştirdi (Goldberg,1989). Goldberg, GA'nın çok sayıda kollara ayrılmış gaz borularında gaz akışını düzenlemek ve kontrol etmek için uygulamasını tanımlamıştır. Ayrıca David Edward Goldberg makine öğrenmesi, nesne tanıma, görüntü işleme ve işlemsel arama gibi alanlarda kullanılabildiğini göstermiştir.

Goldberg'in gaz boru hatlarının denetimi üzerine yaptığı doktora tezi ona sadece 1985 "National Science Foundation" genç araştırmacı ödülünü kazandırmakla kalmadı, genetik algoritmaların pratik kullanımının da olabilirliğini kanıtladı. David Edward Goldberg 1983'te yayınladığı "Computer-aided gas pipeline operation using genetic algorithms and rule learning" adlı kitabında genetik algoritmalara dayalı tam 83 uygulamaya yer vererek GA'nın dünyanın her yerinde çeşitli konularda kullanılmakta olduğunu gösterdi.

3.2. Genetik Algoritma Tanımı

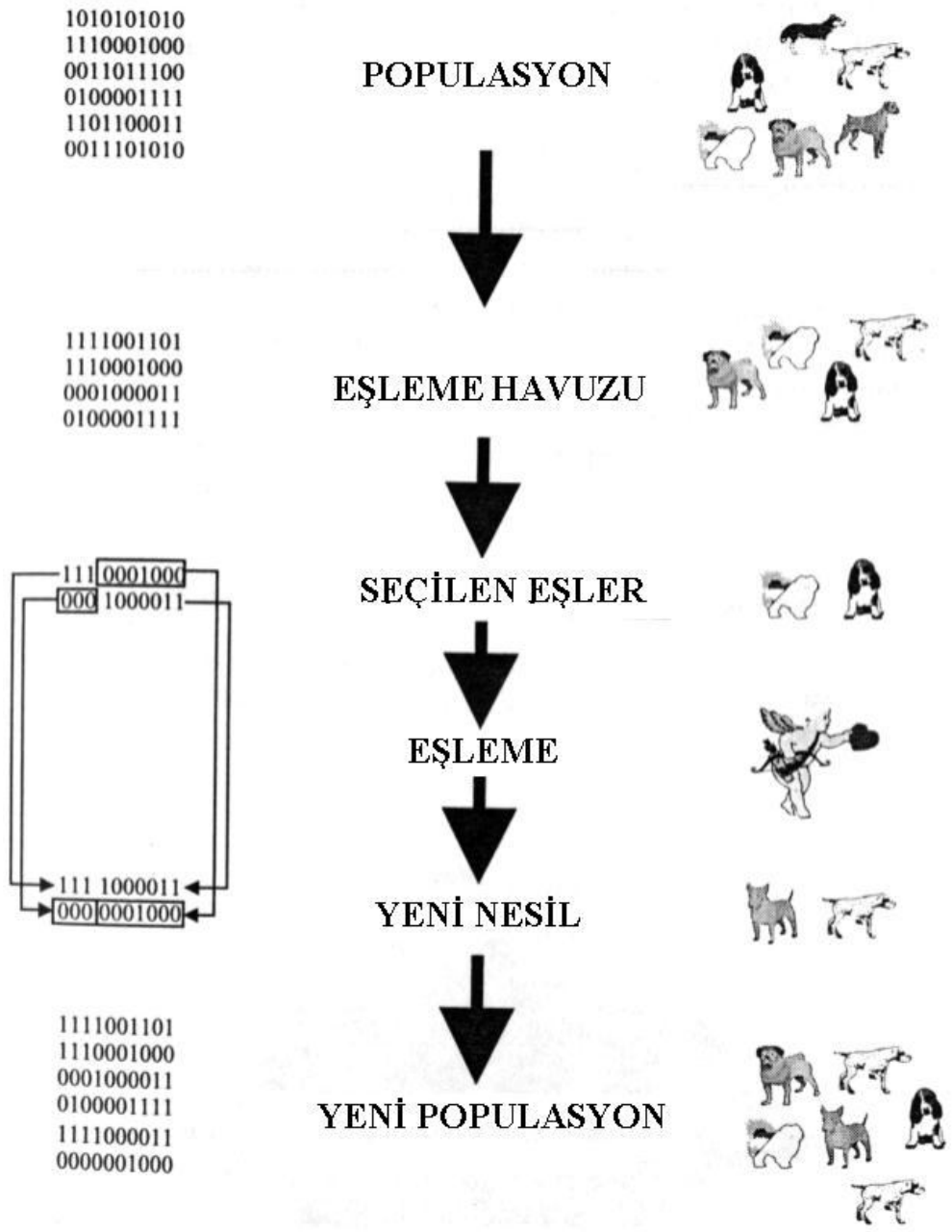
Evrimsel Algoritmalar (EA) canlı popülasyonlarının biyolojiden ve özellikle çevrelerine adapte olmalarına izin veren biyolojik işlemlerden (genetik kalıtım, en iyinin yaşaması) esinlenmiş sezgisel optimizasyon algoritmalarını geniş bir sınıfıdır. Evrimsel algoritmalar (EA); genetik algoritmalar (GA), genetik programlama (GP), evrimsel programlama (EP), evrimsel strateji (EV) ve benzerleri gibi alt bölümlere ayrılabilir. Genetik algoritmalar (GA) doğal evrimin teorisinden esinlenilerek biyolojik süreçleri modelleyen bir optimizasyon tekniğidir.

Genetik algoritmanın anlaşılması için "doğal seçim" in (seleksiyonun) anlaşılması gerekir (Grant, 1985). Adaptasyon ve çevreye uygunluk düzeyi dünyada uzun süre yaşayabilmenin göstergesi haline gelmiştir. Genetik algoritmalar problemlerin çözümü için evrimsel süreci bilgisayar ortamında taklit ederler. Diğer optimizasyon yöntemlerinde olduğu gibi çözüm için tek bir yapının geliştirilmesi yerine böyle yapılardan meydana gelen bir küme oluştururlar.

Genetik algoritmalar doğada geçerli olan en iyinin yaşaması kuralına dayanarak sürekli iyileşen çözümler üretir. Bunun için "iyi"nin ne olduğunu belirleyen bir uygunluk (fitness) fonksiyonu ve yeni çözümler üretmek için yeniden

kopyalama (recombination), değiştirme (mutation) gibi operatörleri kullanır. Genetik algoritmaların bir diğer önemli özelliği de bir grup çözümle uğraşmasıdır. Bu sayede çok sayıda çözümün içinden iyileri seçilip kötüleri elenebilir.

Biyolojik evrim ile ikili kodlarla çalışan GA arasındaki benzerlik Şekil 3.1'de gösterilmiştir. Her ikisinde de popülasyonun üyeleri rastgele seçilmiştir. Köpeklerin her birinin karakteristik özellikleri sol taraftaki satırlarda verilmiştir.

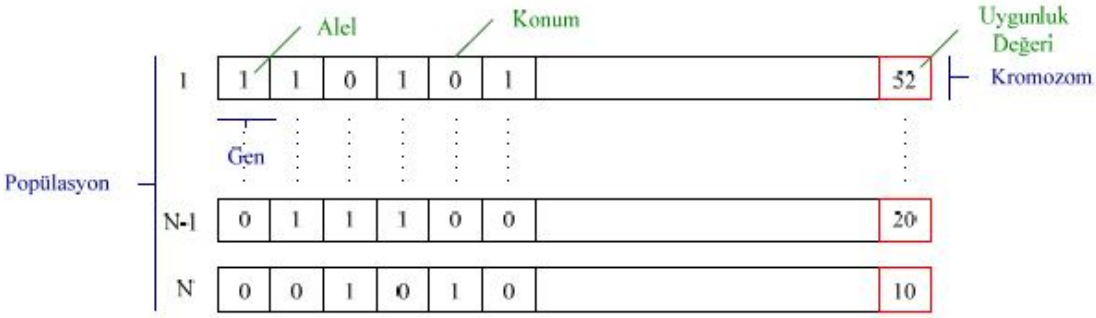


Şekil 3.1. İkili kodlu GA ile biyolojik evrim arasındaki benzetim (Haupt, 1998)

En iyi havlayan köpekler üretilmek istenirse sadece birkaç köpeğin elde tutulması gerekir. Her bir köpek ikili sayı sistemiyle kodlanır. Yeni yavruların üretilmesi için bu popülasyonda rastgele iki adet köpek seçilir ve eşleştirilir. Eşleştirmenin sonucunda iyi havlayan köpeğin çıkma olasılığı yüksektir. Oluşan yeni köpek eşleştirme havuzuna tekrar atılır. Başa dönülerek yeniden eşleştirme prosedürü tekrarlanır (Goldberg, 1993). Bu işleme en iyi havlayan köpek elde edilene kadar devam edilir.

Genetik bilimi ile yakın bir benzerlik gösterdiğinden dolayı benzer terminolojileri kullanılmaktadır. Genetik algoritmada bireylerin oluşturduğu bir topluluk bulunmaktadır. Bu topluluğa popülasyon adı verilmektedir. Popülasyon içerisinde her bir bireyin kalıtsal özelliklerinin taşındığı kromozomlar bulunmaktadır (Goldberg, 1989). Her bir kromozom aslında bir bireydir (Michalewicz, 2004).

Kromozomlar genlerin birleşiminden meydana gelmiştirler. Bu genler yapay sistemlerde kromozom üzerinde bilgilerin tutulduğu karakterler veya özellikler olarak gösterilmektedir (Curtis, 1975). Genlerin taşıdıkları değere alel, her alelin bilgisinin tutulduğu yere konum adı verilmektedir (Man, Tang ve Kwong, 1999). Popülasyon yapısına örnek Şekil 3.2'de gösterilmiştir.



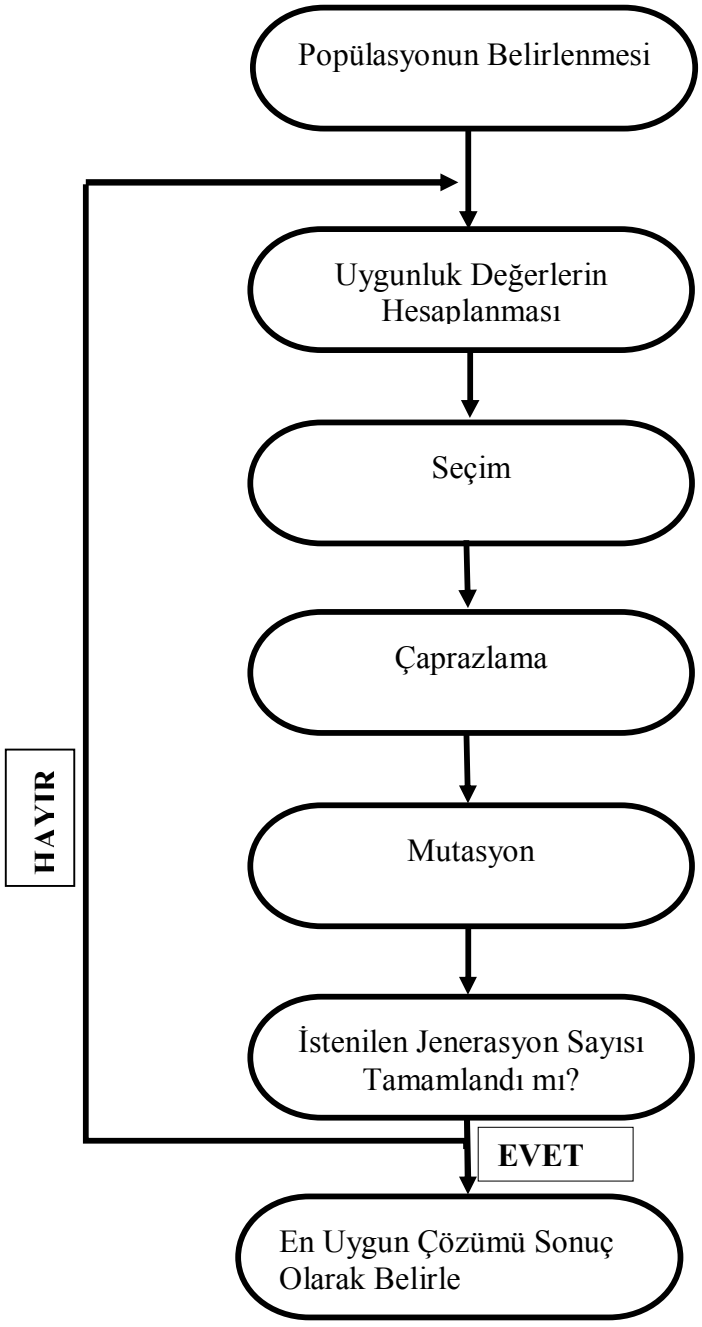
Şekil 3.2. Genetik algoritmada popülasyon yapısı (Tabak, 2008)

3.3. Genetik Algoritma Süreci

Basit bir genetik algoritma (Simple Genetic Algorithm) optimizasyon için son derece uygun ve güçlü bir yöntemdir (Deb, 2001). Genetik algoritma en uygun (optimal) çözümü bulmak için birden çok noktadan aramaya başladığından bu arama noktalarının başlangıç değerlerini oluşturmak önemlidir. Başlangıç popülasyonu genellikle rastgele olarak oluşturulur (Reeves ve Rowe, 2002).

Genetik algoritma sürecinin başlaması için öncelikle başlangıç popülasyonundaki bireylerin her birinin uygunluk değerleri hesaplanması

gerekmektedir. Daha sonra seçim yöntemleri kullanılarak bu bireyler içinde yeni popülasyona aktarılacak olanlar seçilecektir. Seçilen popülasyon arasında evrimsel işlemler uygulanmaktadır. Önce çaprazlamaya (crossing-over) maruz kalan kromozomlar daha sonra mutasyon (mutation) geçirmektedirler. Oluşan yeni kromozomların uygunluk fonksiyonları yeniden hesaplanmaktadır. Uygunluk değerine göre bir sonraki popülasyona aktarılacak kromozomların belirlenmesi için tekrar seçim işlemi yapılmaktadır (Reeves ve Rowe, 2002). Bu işlem süreci problemin niteliğine ve beklentilerine göre en uygun sonuç elde edilinceye kadar sürer. Genetik algoritmanın akış diyagramı Şekil 3.3'te gösterilmiştir.



Şekil 3.3. Genetik algoritmanın akış diyagramı

Genel bir GA aşamaları aşağıdaki gibi özetlenebilir.

1. t=0 G(t) jenerasyonunda N adet kromozom içeren popülasyon rasgele oluşturulur (Her bir kromozom problemin olası çözümüdür).

2. Popülasyonda rastgele büyüklükte kromozom seç (Başlangıç Popülasyon)

3. Her x kromozomu için f(x) uygunluk değerini hesaplanır.

4. t=t+1. G(t) jenerasyonu oluşturulur Bunun için aşağıdaki adımlar gerçekleştirilir.

a. Kromozomlar bir önceki jenerasyonda uygunluk değerlerine göre G(t) jenerasyonuna seçilme şansını kazanırlar. Yüksek uygunluk değerlerine sahip olan kromozomların G(t) jenerasyonuna seçilme şansı (seçilme ihtimali) daha yüksektir. Düşük uygunluk değerine sahip kromozomlar G(t) jenerasyonuna geçmeyebilirler. Yüksek uygunluk değerine sahip kromozomlar birkaç defa G(t) jenerasyonuna geçerler.

b. Seçim sonucunda oluşan popülasyonundaki kromozomlar belli bir olasılıkla çaprazlamaya tabi tutulur. Çaprazlama sonucunda meydana gelen yeni kromozomlar eski kromozomların yerine geçerler.

c. Çaprazlama sonucunda oluşan popülasyonda genetik çeşitliliği sağlamak amacıyla belli bir olasılık değeri ile mutasyon uygulanır.

5. Mutasyon sonucunda oluşan popülasyondaki yeni kromozomlar eski kromozomların yerini alır.

6. Popülasyondaki tüm kromozomların uygunluk değerleri hesaplanır.

7. Baştan belirlenmiş olan jenerasyon sayısı tamamlanmamışsa 4. Adıma geri dönülür.

8. Yukarıdaki en iyi uygunluk değerine sahip olan kromozom problemin optimal çözümünü göstermektedir.

Algoritma en uygun sonucun bulunması ile süreci durduracaktır. Ya da beli bir iterasyonda durdurulacaktır (Tabak, 2008) .

3.4. Genetik Algoritmalarda Kodlama Yöntemleri

Genetik algoritmalarda önemli aşamalardan biri kromozomun nasıl kodlanacağıdır. Genetik algoritmanın kısa sürede ve optimal çalışmasını sağlamak için probleme uyan en uygun kodlama türü seçilmesi gerekir. Kodlama diğer operatörlerin işleyişini etkilediği için ayrı bir önem taşımaktadır. Genellikle en çok ikili kodlama, tamsayı kodlama ve permutasyon kodlama kullanılmaktadır.

Genetik algoritmada en yaygın kullanılan kodlama ikili kodlamadır. Diğer optimizasyon metotlarında olduğu gibi ikili kodlu GA'da da amaç fonksiyonu,

parametreler ve sınırlar tanımlanır. Popülasyon içindeki kromozomların her biri 0 ve 1'lerden oluşmuş ikili diziler şeklinde ifade edilir.

İkili kodlama

Kromozom A: 1 1 0 1 0 1 0 0 0

Kromozom B: 0 1 1 0 1 0 1 1 1

Tamsayı kodlamada ise birey, tamsayıların ard arda yazılmasıyla oluşturulmaktadır.

Tamsayı (Gerçel) kodlama

Kromozom A: 6.378 0.917 0.598 1.376 3.077 6.748

Kromozom B: 0.966 4.280 3.410 2.308 2.213 0.814

Permutasyon kodlama, Gezgin Satıcı Problemi (GSP) ve iş sıralama problemleri gibi permutasyon problemlerinde kullanılır. Burada her kromozom sayıları bir sırada temsil eden sayılar dizisidir.

Permutasyon Kodlama

Kromozom A: 3 4 5 1 2 7 6 0

Kromozom B: 7 0 5 6 2 3 4 1

Gezgin Satıcı Problemi (GSP) ve iş sıralama problemlerinde olan ikili kodlama da kullanılabilir. Örneğin, her onluk sayı için ikili tabandaki sayılar yazılır. Bu durumda 8 genden (örneğin 8 şehirden) oluşan permutasyon kodlamalı kromozomlarımız ikili kodlama da 24 genden oluşacaktır ve her üç gen bir şehri ifade edecektir.

3.5. Uygunluk Fonksiyonu

Bir jenerasyonun oluşturulmasından sonraki adım, popülasyondaki her bireyin uygunluk değerinin hesaplanmasıdır. Uygunluk değeri, mevcut popülasyondan hangi bireylerin bir sonraki popülasyonun oluşturulmasında kullanılacağının belirlenmesinde kullanılır. Popülasyondaki en uygun birey uygunluk değeri en yüksek olan birey, en zayıf birey ise uygunluk değeri en düşük olan birey

anlamına gelir. Örneğin, maksimizasyon problemi için i. bireyin uygunluk değeri $f(x_i)$ genellikle o noktadaki amaç fonksiyonunun değeridir. Fakat minimumum probleminde i. bireyin uygunluk değeri $g(x_i)$'tir formül 3.1'de bağıntı gösterilmiştir. Burada;

$$f = -g \text{ yani } \min f(x_i) = \max g(x_i) = \max[-f(x_i)] \qquad (3.1)$$

Çözümü aranan her problem için bir uygunluk fonksiyonu mevcuttur. Bir çözümün uygunluk değeri ne kadar yüksekse yaşama ve çoğalma olasılığı o kadar fazladır ve bir sonraki jenerasyonda temsil edilme oranı da o kadar yüksektir. Genetik algoritmanı ile iyi çözümün bulunması için uygunluk fonksiyonu iyi seçilmelidir. Kromozomun seçim sırasında seçilme olasılığı, uygunluk fonksiyonuyla hesaplanan kromozomun uygunluk değerine bağlıdır.

3.6. Başlangıç Popülasyonu

Genetik algoritmayı diğer yöntemlerden ayıran bir özelliği de arama işlemine belirli bir noktadan değil de çözüm havuzunu oluşturan bir grup başlangıç çözümü ile başlamasıdır. Bundan dolayı ilk olarak popülasyon büyüklüğü kadar rastgele bir çözüm grubu oluşturulur. Başlangıç popülasyonu; genetik algoritmanın çözüme arayışına başlayabilmesi için ilk olarak rastgele oluşturulan ve içerisinde problemin değişkenlerinin kodlarını bulunduran bir gen havuzudur. Bu adıma popülasyonda bulunacak birey sayısı (popülasyon büyüklüğü) belirlenerek başlanmaktadır. Popülasyon bu işlemden sonra rastgele oluşturulur.

3.7. Genetik Algoritma Operatörleri

Popülasyondaki bireyler uygunluk değerleri hesaplandıktan sonra bir takım genetik operatörleri tabi tutulurlar. En çok kullanılan genetik algoritma operatörleri; seçim, çaprazlama ve mutasyon olarak açıklanabilir.

3.7.1. Seçim

Genetik algoritmalarda bir sonraki popülasyona geçiş sırasında yeni popülasyonu oluşturmak için mevcut popülasyonu içerisinden çaprazlama ve

mutasyon işlemlerine tabi tutulacak bireylerin seçilmesi gerekir. Uygunluk değeri yüksek olan bireyler yaşamlarını sürdürmeli ve bu bireylerden yeni bireyler oluşturulmalıdır (Özkan vd., 2008). Aynı şekilde düşük uygunluk değerine sahip bireyler de zamanla yok olmalıdırlar. Bu nedenle tüm seçilim yöntemlerinde uygunluk değeri fazla olan bireylerin seçilme olasılığı daha yüksek tutulur. Yaygın olarak bilinen seçilim yöntemleri Rulet Tekerleği Seçimi (Roulette Wheel Selection), Turnuva Seçilimi (Tournament Selection) ve Sıralı Seçilimdir. Burada en çok kullanan ve tercih edilen Rulet Tekerleği Seçimi anlatılacaktır.

Rulet Tekerleği Seçim (Roulette Wheel Selection): Rulet tekerleğinde seçme ilkesi, bir çarkın döndürülmesi ve rastgele olarak herhangi bir dilimde durmasının beklenmesi esasına dayanır (Şen, 2004). Rulet çarkı seçimi bireylerin uygunluk fonksiyonlarının çark üzerinde kapladığı alanla ilişkilidir. Alanların büyüklüğü ise bireyin uygunluğu ile orantılıdır. Rulet tekerleği seçim metodunun uygulanması için, ilk önce kromozomların toplam uygunluk değeri hesaplanır. Daha sonra her bir bireyin uygunluk değeri, kendisi de dahil olmak üzere tüm bireylerin uygunluk değerleri toplamına (formül 3.2'de gösterilmiştir.) bölünerek 0 ile 1 aralığında seçilme olasılıkları (formül 3.3'de gösterilmiştir.) belirlenir. Hesaplanan seçilme olasılıkları rulet tekerleği üzerinde kaplayacağı alanı da temsil etmektedir. Kromozomların uygunluk değerleri ne kadar iyiyse seçilme şansı o kadar fazla olmaktadır. Fakat uygunluğu düşük olan bireylere de seçilme şansı doğmaktadır. Bireylerin sahip olduğu seçilme olasılıkları birikimli olarak toplanmaktadır (formül 3.4'de gösterilmiştir). Seçim işlemi gerçekleştirilirken seçilecek kromozomu belirlemek için 0 ile 1 arasında rastgele bir sayı üretilmektedir. Üretilen sayı hangi alana denk geliyorsa o alanla temsil edilen birey seçilir.

$$F=\sum_{i}^{n} f(x_i) \qquad (3.2)$$

$$p(x_i)=\frac{f(x_i)}{F} \qquad (3.3)$$

$$q_i=\sum_{j=1}^{i} p_j \qquad (3.4)$$

Burada

F: Popülasyondaki bireylerin uygunluk değerlerinin toplamı,

p(x_i): Popülasyonunun x_i 'nci bireyin seçilme olasılığı,

q_i : Popülasyonun q_i 'nci bireyin birikimli (ardışık) olasılığı,

n: Popülasyonda birey sayısı,

$f(x_i)$: Uygunluk fonksiyonudur.

Örneğin 4 tane bireyin uygunluk fonksiyonları sıra ile 45, 21, 9, 75 dir. Bu bireylerin uygunluk değerlerinin toplamı aşağıdaki formül ile hesaplanmaktadır.

$$\mathrm{F} = \sum_{i=1}^{4} f(x_i) = f(x_1) + f(x_2) + f(x_3) + f(x_4) = 45+21+9+75 = 150$$

Rulet tekerleği seçim metodunun uygulanması için bu bireylerin seçilme ve birikimli (ardışık) olasılıkları aşağıdaki formüllerle hesaplanmıştır.

$$\mathrm{p}(x_1) = \frac{f(x_1)}{\mathrm{F}} = \frac{45}{150} = 0.30 \qquad q_1 = 0.30$$

$$\mathrm{p}(x_2) = \frac{f(x_2)}{\mathrm{F}} = \frac{21}{150} = 0.14 \qquad q_2 = 0.44$$

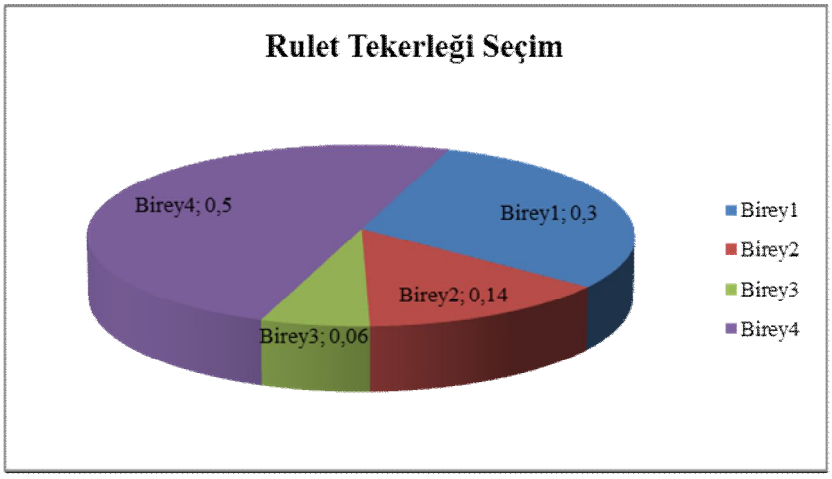
$$\mathrm{p}(x_3) = \frac{f(x_3)}{\mathrm{F}} = \frac{9}{150} = 0.06 \qquad q_3 = 0.50$$

$$\mathrm{p}(x_4) = \frac{f(x_4)}{\mathrm{F}} = \frac{75}{150} = 0.50 \qquad q_4 = 0.10$$

Tablo 3.1'de bireyler, seçilme olasılıkları verilmiştir. 0 ile 1 arasında rastgele 4 tane sayı üretilsin. Bunlar 0.18, 0.37, 0.68, 0.91 dir. Bu durumda yeni bir popülasyon oluşturmak için seçilen popülasyon bireyleri Birey1, Birey2, Birey4, Birey4 şeklinde olacaktır. Bireylerin rulet tekerleğindeki dağılımı Şekil 3.5'te gösterilmiştir.

Tablo 3.1. Bireylerin seçilme olasılıkları

Bireyler	Uygunluk Fonksiyonları(F)	Seçilme Olasılığı(p)	Ardışık Toplam Seçilme Olasılığı(q)	Aralık
Birey1	45	0.3	0.3	0.00-0.30
Birey2	21	0.14	0.44	0.3-0.44
Birey3	9	0.06	0.5	0.44-0.50
Birey4	75	0.5	1.0	0.50-1.00



Şekil 3.4. Rulet tekerleği seçimi

3.7.2. Çaprazlama

İki adet yeni kromozom (çocuklar) elde etmek için kromozomların bulunduğu eşleme havuzundan iki adet kromozom seçilir. Eşleme sürecinde seçilen kromozomlardan bir ve birden fazla yeni kromozomlar (çocuklar) oluşturma olayına "çaprazlama" denir (Şen, 2004). Çaprazlamanın kullanım amacı bir önceki popülasyondan gelen kromozomlardan daha iyi uygunluk değerine sahip kromozomlar oluşturmaktır. Çaprazlamayı popülasyondaki bütün bireylere uygulama zorunluluğu yoktur. Popülasyondaki bazı bireylerin çaprazlama işlemine uygulanmadan bir sonraki popülasyona geçmesi istenirse, çaprazlama oranı belirlenir. Daha sonra popülasyondaki her birey için 0 ile 1 aralığında rastgele olarak bir reel sayı üretilir. Bu sayı çaprazlama oranından küçük ise bireye çaprazlama

işlemi uygulanır. Sayı büyük ise çaprazlama işlemi uygulanmaz. Örneğin çaprazlama oranı $p_c = 0.9$ olduğunu kabul edersek çaprazlamaya tabi tutulacak kromozomların sayısı pop_size x $p_c = 5 \times 0.9 = 4.5 \approx 5$ formülü ile hesaplanmaktadır. Burada pop_size: kromozomların popülasyondaki sayısıdır. Eğer formülün sonucu tek sayı ise bu sayı çift sayıya dönüştürülecektir. Örnekte formül sonucu olan 5 kromozom çaprazlamaya tabi tutulamaz (çaprazlamanın genel olarak çiftlere uygulandığından dolayı). Bunun için 5 - 1 = 4 işlemi yapılarak kromozom sayısını çift sayıya dönüştürülür. Çaprazlamaya uğrayacak bu kromozomların belirlenebilmesi için her bir birey için 0 ile 1 arasında rastgele bir r reel sayı üretilir. Sayılar Tablo 3.2' de gösterilmektedir. Sonra bireye kromozomun kaçıncı geninden itibaren çaprazlama yapılacağının belirlenmesi için rastgele bir tam sayı üretilir.

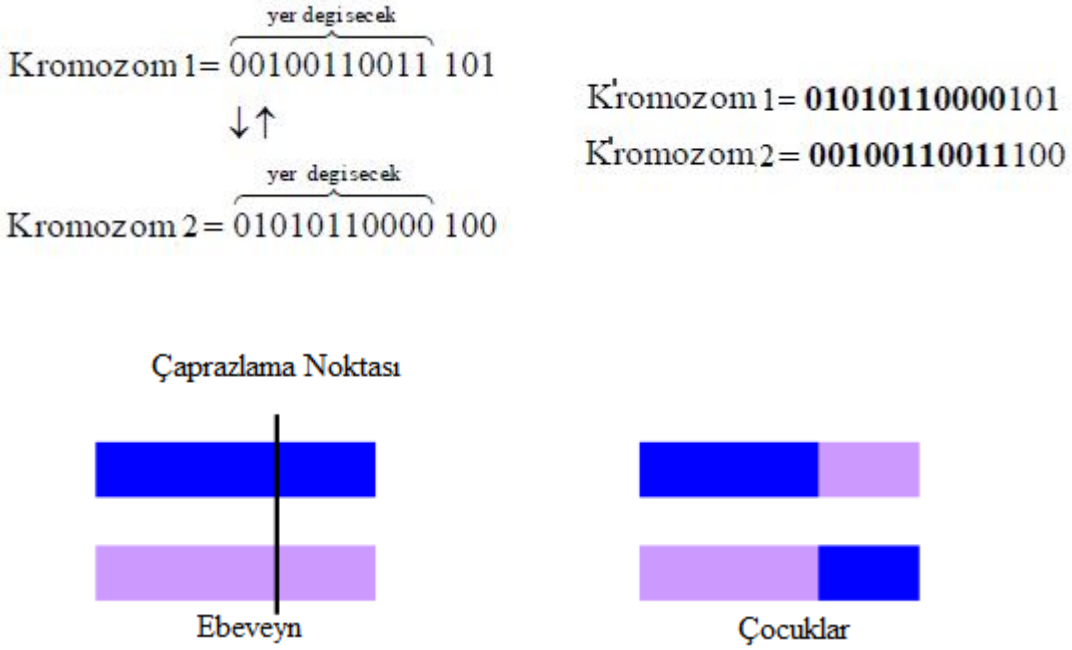
Tablo 3.2. Çaprazlama için üretilen rastgele sayılar

r_1	0.75
r_2	0.97
r_3	0.24
r_4	0.95
r_5	0.81

- $r_1 > p_c$ $0.75 < 0.9$ olduğundan 1. birey çaprazlama işlemi için seçilir.
- $r_2 > p_c$ $0.97 > 0.9$ olduğundan 2. birey çaprazlama işlemi için seçilmez.
- $r_3 < p_c$ $0.24 < 0.9$ olduğundan 3. birey çaprazlama işlemi için seçilir.
- $r_4 > p_c$ $0.9\ 5 > 0.9$ olduğundan 4. birey çaprazlama işlemi için seçilmez
- $r_5 < p_c$ $0.81 < 0.9$ olduğundan 5. birey çaprazlama işlemi için seçilir.

Çaprazlama için çok farklı yaklaşımlar vardır. Adewuya (1996) ve Michalewitcz (1994) tarafından ilginç metotlar ortaya konulmuştur. Bunlar tek noktalı, çift noktalı, çok noktalı ve düzgün (uniform) çaprazlama metotlarıdır. Burada en çok kullanılan yöntem olan tek noktalı çaprazlama metot açıklanacaktır. **Tek Noktalı Çaprazlama Metodu**: Bu tip çaprazlama işleminde ikişerli olarak

eşleştirilen bireylerin çaprazlanması için rastgele olarak bireyin genleri üzerinde bir nokta seçilir. Bu nokta sıfır ile bireyin gen sayıları arasında bir sayının üretilmesi ile belirlenir. Kromozomlar belirlenen bu noktadan ikiye ayrılır. Başlangıç kısımları veya ikinci kısımları yer değiştirir. Örneğin Şekil 3.6' da gösterildiği gibi 14 genden oluşan bir birey için 1 ile 14 arasında bir sayı üretilir ve üretilen sayının solundaki rakamlar çaprazlamaya tabi tutulur.



Şekil 3.5. Tek Noktalı Çaprazlama

3.7.3. Mutasyon

Mutasyon, kromozomların çeşitliğini artırmak ve arama uzayını genişletmek amacıyla rastgele bir şekilde bireylerin genlerinde küçük değişiklikler yapmak için kullanılır. Mutasyon işlemi çaprazlama işleminden sonra gerçekleşir. Mutasyon işlemi yeni üretilmiş bir kromozomun genleri üzerinde rastgele olarak seçilmiş olan genin değerini 0 'dan 1 ' e (genin değeri 0 ise) veya 1 'den 0 'a (genin değeri 1 ise) değiştirerek yapılır. Mutasyon işlemi için genlerin ne oranda seçileceği mutasyon oranı ile belirlenir. Bu oran $p_m = 0.02$ ise bireylerin genlerinin %2'inin değişmesi öngörülüyor demektir. Hangi genlerin mutasyona uğrayacağını belirlemek için bireylerin genlerinin sayısı kadar rastgele r sayısı üretilir. Örneğin 20 genden oluşan 20 kromozom için N = 20 x 20 = 400 adet rastgele r sayıları üretilir, $r_i \in R$ (R-reel

sayılar kümesidir), $0 \leq r_i \leq 1$, $i \in [1,400]$. Oluşturulan bu değerler belirlenen $p_m = 0.03$ mutasyon ihtimali ile karşılaştırılır ve $r_i < p_m$ durumunu sağlayan genler mutasyon işlemine tabi tutulur. Tablo 3.3' te görülen r_i değerlerine göre

$r_5 < p_m$ (0.02 < 0.03) 1. Bireyin 5 numaralı geni mutasyona tabi tutulur.

$r_{397} < p_m$ (0.01 < 0.03) 20. Bireyin 17 numaralı geni mutasyona tabi tutulur.

Tablo 3.3. Her gen için üretilen mutasyon oranı

r_i	p_m
r_1	0.27
r_2	0.36
r_3	0.79
r_4	0.67
r_5	0.02
r_6	0.95
…	..
r_{397}	0.01
r_{398}	0.19
r_{399}	0.87
r_{400}	0.43

Mutasyon operatörüne tabi tutulacak genler bu şekilde belirlenir. Diğer genler ise oldukları gibi popülasyonda kalır. İkili kodlama (Binary) sisteminde mutasyon işlemi daha önce belirtildiği gibi değeri '0' olan genlerin '1', '1' olan genlerin '0' olarak değiştirilmesi şeklinde olur.

m'_1 kromozomun 5. geni mutasyona uğradığından sonra Şekil 3.7' de görüldüğü gibi m''_1 şeklinde yeni bir kromozom oluşturulur.

$m'_1 = 1\ 0\ 1\ 0\ \mathbf{1}\ 1\ 1\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 0$ ⇦ Önce

$m''_1 = 1\ 0\ 1\ 0\ \mathbf{0}\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 0$ ⇦ Sonra

Şekil 3.6. Mutasyon operatörünün uygulanışı

Değiştirme (mutasyon) olasılığı 0.1-0.01 aralığında olması tavsiye edilir. Aşağıdaki eşitlik (denklem 3.5 de) Bäck tarafından önerilmiştir (Bäck, 1993).

$$\frac{1}{\text{Populasyon Buyuklugu}} \leq P_{\text{Mutasyon Orani}} \leq \frac{1}{\text{Kromozom Uzunlugu}} \quad (3.5)$$

3.8. Genetik Algoritmaların Performansını Etkileyen Parametreler

Parametreler, genetik algoritmaların performansı üzerinde önemli derecede etki yaparlar. En uygun parametreler bulunması için birçok çalışmalar yapılmıştır. Fakat her problem için genel olarak kullanılabilecek parametreler bulunamamıştır. Bu parametreler kontrol parametreleri olarak isimlendirilmektedir. Bunların en önemlileri: Popülasyon Büyüklüğü (N), Çaprazlama Oranı (p_c) ve Mutasyon Oranı (p_m) olarak sayılabilir.

Popülasyon Büyüklüğü: Genetik algoritmalarda popülasyondaki kromozom (birey) sayısına popülasyon büyüklüğü denir. GA'nın başarısını ya da optimum sonuca erişim süresini etkileyen önemli unsurlardan biri de popülasyon büyüklüğüdür. Popülasyon büyüklüğü problemin türü ve kapsamına bakılarak belirlenmelidir. Popülasyonda çok fazla birey varsa GA bayağı yavaşlayacaktır (Özkan, 2008). Araştırmalar belli bir noktadan sonra popülasyon sayısının artırımının bir yararı olmadığını göstermiştir (Cevre, 2008). Bu problemin daha hızlı bir şekilde çözülmesine yardımcı olmamaktadır. Özellikle dinamik gezgin satıcı problemi gibi tepki süresinin düşük olmasını gerektiren gerçek zamanlı problemlerde uygulamanın çalışma hızı daha büyük önem kazandığından topluluk büyüklüğü çok daha dikkatli bir şekilde belirlenmeli ve gerektiğinden daha büyük olmamalıdır (Kalaycı, 2006). Bu yüzden topluluk büyüklüğü için uygun bir değer belirlenmelidir (Goldberg, 1989).

Çaprazlama Oranı: Çaprazlama işlemi, popülasyonda bulunan kromozomların belirli bir oranına uygulanmaktadır. Çaprazlama oranı olarak adlandırılan bu oran, algoritmanın başında ya da her yeni popülasyon oluşturmadan önce belirlenmektedir. Böylece seçilen ebeveyn kromozomlar, çaprazlama oranı ölçüsünde yeni bireyler oluşturmak üzere çaprazlanırlar (Kalaycı, 2006). Çaprazlama oranının yüksek olması popülasyonda değişime uğrayan kromozom sayısının fazla olacağını, diğer bir ifadeyle, yeni kromozom sayısının artacağını göstermektedir (Kahvecioğlu, 2004).

Gereğinden yüksek olarak belirlendiği durumda, çözüm uzayını hızlı bir şekilde aramayı sağlarken, diğer yandan da iyi sonuçlar verecek kromozomların atlanmasını ya da GA'nın performansının düşmesine neden olabilir (Nearchou, 1998). Çaprazlama oranın düşük olarak belirlenmesi halinde ise değişime uğrayacak kromozom sayısı azalacak ve dolayısıyla algoritma yavaşlayarak sonuca geç ulaşılacaktır. Bu da çözüme daha uzun sürede ulaşılmasına yol açar. Literatürde ideal çaprazlama oranı 0.7 ya da 0.75 olarak belirtilir (Özkan, 2008).

Mutasyon Oranı: Kromozomdaki genlerin ne kadar sıklıkla mutasyon geçireceğini gösterir. Mutasyonun amacı popülasyondaki genetik çeşitliği korumaktır. Yüksek mutasyon oranı çözüm adaylarında aşırı rastgeleliğe yol açacağından dolayı optimum çözümden uzaklaşmamıza neden olur. Çok düşük mutasyon oranı ise topluluk içerisinde yer alan çözüm adaylarının çeşitliliğini azaltacağından çözüme ulaşmayı güçleştirir. Literatürde ideal mutasyon oranı 0.001 olarak belirtilmekle birlikte mutasyon oranı belirlenirken problemin türünün dikkate alınmasında yarar vardır. Etkili bir genetik algoritma tasarlamak için çok iyi kontrol edilmelidir.

3.9. Genetik Algoritmaların Avantajları Ve Dezavantajları

Genetik algoritmalar diğer optimizasyon yöntemlerinden dört temel farkı vardır (Goldberg, 1989):

1. Genetik algoritmalar, problemlerin çözümünü parametrelerin değerleriyle değil kodlarıyla arar. Parametreler kodlanabildiği sürece çözüm üretilebilir.

2. Genetik algoritmalar çözüm uzayından rastgele seçilen noktalar üzerinde operatörler yardımıyla daha iyi noktalara ulaşmayı amaçlar.

3. Genetik algoritmalar türev yerine uygunluk fonksiyonunun değerini kullanır.

4. Genetik algoritmalar olması gereken kuralları değil olasılıksal kuralları kullanır (Söke, 2003).

Genetik algoritmaların çeşitli avantajları olduğu kadarı ile dezavantajları da vardır.

Avantajları;

- Genetik Algoritmalar karmaşık ve büyük bir çözüm uzayına sahip problemlerde çok etkilidir.
- Geliştirilmesi ve gerçekleştirimi düşük maliyetlidir.
- Çözüm uzayında aynı anda geniş bir alandan çok sayıda noktadan araştırmaya başlar ve pek çok alternatif çözüm sunar.
- Diğer yöntemlerle birlikte melez (hibrid) olarak kullanılabilir.
- Sürekli ve ayrık parametreleri optimize eder.
- Türevsel bilgiler gerektirmez.
- Amaç fonksiyonunu geniş bir spektrumda araştırır.
- Çok sayıda parametrelerle çalışma imkânı sağlar.
- Paralel PC'ler kullanılarak çalıştırılabilir.
- Karmaşık amaç fonksiyonu parametrelerini, lokal minimum veya maksimumlara takılmadan optimize eder.

Dezavantajları:

- Genetik algoritmalar problem uzayının özelliklerinden yeterince faydalanmazlar ve teorik altyapıları oldukça basittir.
- Genetik algoritmalarda rastgeleliğe dayalı bir arama yapıldığı için problemin türüne bağlı olarak çözüme ulaşmak uzun sürebilir.
- Genetik algoritmalar kesin sonuç isteyen problemler için uygun değildir. Çünkü sınırlı bir sürede mükemmel çözüme ulaşmayı garanti etmez.
- Yerel optimizasyonlarda etkili değildir.
- Çözüm uzayının özelliklerinden fazla yararlanmaz.

Genetik algoritmalar arama ve optimizasyon için sezgisel yöntemlerdir. Geniş arama algoritmalarının aksine, genetik algoritmalar en iyiyi seçmek için tüm farklı durumları üretmez. Bundan dolayı mükemmel çözüme ulaşamayabilir.

Her problemin çözümü için GA kullanmak iyi bir yol değildir. Birkaç parametreli analitik fonksiyonun çözümünde klasik metotlar daha hızlıdır. Böyle durumlarda, nümerik metotlar tercih edilmelidir. Paralel bilgisayarlar kullanılırsa GA daha hızlı sonuç verebilir.

3.10. Genetik Algoritmaların Uygulama Alanları

Genetik algoritmaların en uygun olduğu problemler, geleneksel yöntemler ile çözümü mümkün olamayan ya da çözüm süresi problemin büyüklüğü ile üstel orantılı olarak artan yerlerde yoğun bir şekilde kullanmaktadır. Genetik algoritmalar özellikle kaynak tahsisi, iş atölyesi çizelgelemesi, makine parça gruplaması ve bilgisayar ağ tasarımı, lojistik, okul otobüsü rotalaması, posta kutusuna dağıtım problemleri, çöp toplama, baskı devre kartlarının montajı veya muayenesi, ders programı hazırlama, GSM operatörlerinin baz istasyonlarının yerleşim yerlerinin belirlenmesi problemi, malzeme akış sistem tasarımı, araç rotalama problemleri, depolardaki vinç güzergâhlarının programlaması, stok alanındaki malzeme toplama problemleri, uçaklar için havaalanı rotalaması, elektronik devre tasarımı gibi çeşitli alanlarda uygulanmaktadır.

Genetik algoritmaların kullanıldığı bazı spesifik problemler şu şekilde sıralanabilir:

- Montaj Hattı Dengeleme Problemi
- Çizelgeleme Problemi
- Tesis Yerleşim Problemi
- Baskı Devre Kartlarında İşlem Sırası Belirleme Problemi
- Atama Problemi
- Hücresel Üretim Problemi
- Sistem Güvenilirliği Problemi
- Taşıma Problemi
- Gezgin Satıcı Problemi
- Araç Rotalama Problemi
- Minimum Kapsayan Ağaç Problemi

Bu tez çalışmasında Gezgin Satıcı Probleminin Genetik Algoritma ile çözüm yöntemini geliştirerek İETT personelin uygulama alanındaki (sahadaki) denetim ekibinin durak, hat, araç ve işletmelerini denetleyerek optimal güzergahı belirlemesi amaçlanmaktadır.

4. GEZGİN SATICI PROBLEMİ

Bu bölüm Gezgin Satıcı Problemi (Traveling Salesman Problem) (TSP) (GSP) hakkında, tarihçesi, çözüm yöntemleri ve genel bir literatür bilgisi içermektedir.

Gezgin satıcı problemi (Traveling Salesman Problem) veya kısaca GSP (TSP) aralarındaki mesafeleri bilinen belirli sayıdaki şehirlerin (nokta, düğüm veya paraca gibi) her birinden yalnızca bir defa geçerek başlangıç şehrine dönen en kısa turun bulunmasına denir.

GSP veya onun bir türevi şeklinde modellenen problemler, matematik, yapay zeka ve fizik gibi farklı alanlardaki çok sayıda araştırmacının ilgisini çekmektedir. Gezgin satıcı problemi muhtemelen en çok bilinen optimizasyon problemidir.

4.1. Gezgin Satıcı Probleminin Tarihçesi

Gezgin Satıcı Problemi ilk kez 1800 li yıllarında ortaya atılmış olup günümüzde bu konuda birçok çalışma yapılmış ve yapılmaya devam edilmektedir.

GSP birçok akademisyen tarafından farklı isimlerde kullanılmıştır. Karl Menger (1932), GSP'nin bir çeşidini ulak problemi olarak adlandırmıştır. Ulak problemi ile gezgin satıcı problemi birkaç küçük farklılık içermesine rağmen aslında aynı problemdir. Menger ulak problemini şöyle tanımlamıştır. "Ulak problemi karşılıklı eşit uzaklıkta olan belirli sayıdaki noktalar bilinirken, bu noktalar birleştirilerek en kısa yolun bulunması problemidir. Bu problem belirli sayıdaki noktalar için her zaman kesin olarak çözüm sağlamaktadır. Başlangıç noktasından sonra en yakın komşuya gitme kuralı, en kısa yol problem için geçerli değildir." Karl Menger'in (1932) GSP alanında ilk yayın yapan kişi olduğuna inanılmaktadır. Menger bütün olurlu yollar test edilerek, optimum sonuca ulaşma stratejisini

önermiştir ve en yakın komşu algoritmasının optimum sonucu garanti etmediğini belirtmiştir. Hatta bazı durumlarda en yakın komşuyu seçme stratejisi en kötü sonucu vermektedir (Gutin ve Punnen, 2002).

Bu alandaki ikinci çalışma Mahalanobis'e aittir. Mahalanobis 1939'da Hindistan Bilim Kongresi'nde, Bengal'deki hintkeneviri tarım alanı tahmin problemi ile ilgili bir konuşma yapmıştır. Amaç sonuçların güvenilirliğini maksimize ederek, alandaki faaliyetlerin maliyetlerini belirli bir seviyenin altında tutarak kontrol etmektir. 1949 yılında Gezgin Satıcı Problemi adını kullanarak ilk makale J.B. Robinson tarafından yayınlanmıştır. Fakat İlişkisel optimizasyona giren GSP ile ilgili sistematik bir çalışma Danztig, Fulkerson ve Johnson (1954) tarafından yapılmıştır.

4.2. Gezgin Satıcı Problemi İçin Çözüm Yöntemleri

Gezgin satıcı problemini çözmek için araştırmacılar tarafından pek çok yöntem geliştirilmiştir. Bu çözüm yöntemleri optimal çözüme ulaşıp ulaşmamasına göre kesin çözüm yöntemleri ve sezgisel yöntemler olarak ikiye ayrılır. Gezgin Satıcı Problemi için kullanılan başlıca kesin çözüm yöntemleri: Sayma yöntemi, Dinamik programlama, Dal ve sınır yöntemi, Doğrusal çözüm yaklaşımları, Dantzig-Fulkerson-Johnson (DFJ) modeli, Miller-Tucker-Zemlin (MTZ) modeli. Sezgisel çözüm yöntemleri: En yakın komşu yöntemi, En yakın ekleme yöntemi, Geometrik yöntem, Greedy yöntemi, Bovurka yöntemi, Christofides algoritması, Grasp yöntemi, R-opt denemeleri, Lin-kerninghan yöntemi, Hyperopt denemeleri, S&C (stem and cycle) yöntemi, Genetik algoritmalar, Karınca algoritması, Tabu arama, Tavlama benzetimi, Yapay sinir ağlarıdır. Bu tez çalışmasında sezgisel yöntemlerden genetik algoritmalar üçüncü bölümde açıklanmıştır.

4.3. Literatürde Gezgin Satıcı Problemi Çalışmaları

Literatürde, yerel arama algoritmaları, genetik algoritmalar (Goldberg, 1989), benzetimli tavlama (simulated annealing) (Laarhoven, 1987), yasak arama (tabu search) (Fiechter, 1994), karınca kolonisi optimizasyonu (Angus ve Hendtlass, 2005) ve yapay sinir ağları gibi eniyileme tabanlı birçok yaklaşım yöntemi Gezgin Satıcı Problemini çözümü için önerilmiştir (Johnson, McGeoch, 1997). Dallandır ve Kes

(Branch and Cut) yöntemi tabanlı büyük GSP örneklerinde bile kesin sonuç verebilen algoritmalar da tanımlanmıştır. Concorde bilgisayar kodu, simetrik GSP çözücülere en meşhur örnektir ve 2006'da bir devre kartı içerisindeki 85900 noktanın tümünün en etkin hangi sırada dolaşılması gerektiğini belirlemiştir. Sengoku ve Yoshihara'nın 1998'te yaptıkları çalışma GSP'nin GA ile çözümü alanındaki en önemli işlerden biridir. Çünkü geliştirdikleri algoritma, mutasyon evresinde çok etkili olan 2-opt yöntemini kullanmaktadır. Pullan 2003 yılında yayınladığı "Genetik Algoritmaları Gezgin Satıcı Problemi' ne Uyarlamak (Adapting Genetic Algorithms to Traveling Salesman Problem)" (Pullan, 2003) bildirisinde yerel arama sezgileri ile genetik algoritmaların birlikte kullanımının GSP'nin çözümündeki etkinliğine değinmiştir. Bu melez yaklaşımı sadece gezgin satıcı probleminde değil buna benzer biçimde ağ modellemesin uygulanacağını savunmuştur. Ray, Bandyopadhyay ve Pal 2004 yılında yayınladıkları "Gezgin Satıcı Problemi için Genetik Algoritmaların Yeni İşlemcileri (New Operators of Genetic Algorithms for Traveling Salesman Problem)" bildiride yeni bir bilgi temelli çoklu ters çevirme (multiple inversion) operatörü ile bir bilgi temelli komşuluk yer değiştirme (neighborhood swapping) operatörü önermişlerdir. Bilgi temelli çoklu ters çevirme operatörü doğal seçilimden hemen önce, bilgi temelli komşuluk yer değiştirme operatörü ise çaprazlama ve mutasyon evrelerinin arasında kullanılmaktadır. Takahashi ise 2005 yılında gerçekleştirdiği "Gezgin Satıcı Problemini Değişken Çaprazlama İşlemcileri ile Genetik Algoritmalar Aracılığıyla Çözmek (Solving the Traveling Salesman Problem through Genetic Algorithms with Changing Crossover Operators)" (Takahashi, 2005) adlı çalışmada GSP'nin çözümü için geliştirdiği genetik algoritmada oldukça farklı bir çaprazlama yöntemi önermiştir. "Değişken Çaprazlama İşlemcileri (Changing Crossover Operators - CXO)" adını verdiği bu yaklaşım genetik algoritmanın çalışması sırasında mevcut çaprazlama operatörünün herhangi bir anda uygun bir başka çaprazlama operatörü ile esnek olarak değiştirilebilmesini sağlamaktadır. Takahashi çalışması sırasında elde ettiği deneysel sonuçlar doğrultusunda geliştirdiği CXO fikri yalnızca bir çaprazlama yöntemi kullanan genetik algoritmalardan daha etkin olduğunu ifade etmiştir.

Cevre U, Özkan B ve Uğur A (2007), Gezgin satıcı probleminin genetik algoritmalarla eniyilemesi ve etkileşimli olarak Internet üzerinde görselleştirilmesi ile, Özkan B, Cevre U ve Uğur A (2008), Melez Bir Eniyileme Yöntemi ile Rota Planlama, Özkan B (2008), Dinamik Gezgin Satıcı Probleminin Çözümü için Bir

Eniyileme Kütüphanesinin Tasarımı ve Görsel Yazılım Geliştirme Ortamı ile Birlikte Gerçekleştirim adlı tezinde gezgin satıcının yazılım sayesinde pratik hayat uygulanmasını göstermiştir. Cevre U (2008), Çoklu Gezgin Satıcı Probleminin Çözümü için Bir Eniyileme Kütüphanesinin Tasarımı Ve Görsel Yazılım Geliştirme Ortamı ile Birlikte Gerçekleştirimi, Çetin M (2007), Gezgin Satıcı Örnek Problemlerinin Optimum Sonuçlarının Grid Aracılığı ile Hesaplanması vb. birçok akademik çalışmalar gezgin satıcı hakkında yapılmaya devam etmektedir. Keskintürk T (2006), Gezgin Satıcı Probleminin Diferansiyel Gelişim Algoritması ile Çözümü, VI. Ulusal Üretim Araştırmaları Sempozyumu yayını, Gazi Üniversitesi Mühendislik Dergisi için hazırlanan "Gezgin Satıcı Problemini Melez Bir Eniyileme Yöntemi ile Çözen Web Tabanlı Etkileşimli Bir Simülasyon Aracı Geliştirilmesi" başlıklı yayın çalışması ise hakemler tarafından önerilen düzeltmelerin gerçekleştirilmesi aşamasındadır.

5. YAPILAN UYGULAMA

5.1. Genetik Algoritmanın Kullanım Amacı

Gezgin Satıcı Problem yıllardır birçok kişi tarafından farklı yöntemler ile çözülmüştür. Bu çözüm yöntemleri dördüncü bölümde anlatılmıştır. Ancak problemin çözümü için en garanti yöntem bütün olası güzergahların denenmesidir. Az sayıda şehir için bu mantıklı olabilir. Fakat şehir sayısı büyüdükçe n! (Faktöriyel) şeklinde şehirlerarası olası güzergah sayısı da artıyor. Bunların içinden de en uygun olanın seçilmesi bilgisayarla bile yıllarca hesaplamalar yapılmasını gerektirebilir. Örneğin 20 şehir bulunduğu bir bölgede (n-1)!/2 = 19!/2 = 60.822.550.204.416.000 adet farklı tur mevcuttur. Sadece her bir turun uzunluğunun belirlenmesi ve en iyi turun bulunması işlemi için 3 Ghz işlemci hızına sahip bir bilgisayarda denemeler yaklaşık olarak 234 gün sürer (Terzi, 2009). Yani şehir sayısının artması ile orantılı olarak arama süresi artmaktadır. Çözüm süresine bakıldığında bu yöntemin uygun olmayacağı anlaşılmaktadır. GSP'nin çözümünün genetik algoritma ile bulunması daha kısa sürmektedir. Yapılan uygulama da bunu göstermektedir.

Son yıllarda bilgisayar gücündeki gelişmeler ve etkin algoritmaların ortaya çıkması ile GSP çözümünde önemli gelişmeler kaydedilmiştir. Applegate, Bixby, Chavatal ve Cook gezgin satıcı test problemlerinden bazı çözülmemiş problemleri çözmüşlerdir. Bunlardan, 7397 şehirli problem 1994 yılında, 13509 şehirli problem 1998 yılında, 15112 şehirli problem 2001 yılında ve 24978 şehirli problem 2004 yılında çözülmüştür. Bu başarılara rağmen GSP hala başarılı bir şekilde çözümlemeden uzaktır (TSP History, 2011). Süper bilgisayarlar, paralel bilgisayarlar gibi bilgisayar teknolojilerindeki gelişmelere karşın pek çok problemin çözümü hala çok zordur. Bir grup araştırmacının (Applegate, 1998) 13509 şehirlik bir problemi çözmesi 48 bilgisayar ile 3 ayı almıştır. Bu durumda normal bir bilgisayar ile bu problemi çözmek için 12 yıl gerekmektedir (TSP History, 2011).

5.2. Problemin Tanıtılması

İETT (İstanbul Elektrik Tramvay ve Tünel) dünyada kent içi ulaşımda önde gelen firmalardan biridir. İstanbul kent içi ulaşımı 1869 yılında Dersaadet Tramvay Şirketi`nin kurulması ve Tünel Tesisleri`nin inşasıyla başladı.

Şekil 5.1. 1869 yılında Dersaadet Tramvay (İETT, 2011)

1871 yılında ilk atlı tramvay hizmete girdi. 1913 yılında Silahtarağa'da Türkiye'nin ilk elektrik fabrikası kurulur. Ardından Şubat 1914'te elektrikli tramvay işletmeciliğine geçilir. 1926 yılında ilk otobüsler alınır. Bir süre muhtelif yabancı şirketler tarafından işletilen elektrik, tramvay ve tünel işletmeleri 1939 yılında millileştirilerek 3645 sayılı yasa ile İstanbul Elektrik Tramvay ve Tünel (İETT) İşletmeleri Umum Müdürlüğü adı altında bugünkü hüviyetine kavuşur. Bugün yalnızca kent içi toplu ulaşım hizmeti veren İETT: otobüs, tramvay ve tünel işletmeciliğinin yanında Özel Halk Otobüsleri'nin yönetim, yürütüm ve denetiminden sorumludur. İETT ayrıca İstanbul`daki raylı sistemlerin (Metro, Hafif Metro) bir bölümünün yapımını (Eminönü-Kabataş, Sultançiftliği-Edirnekapı, Edirnekapı-Topkapı, Otogar-Başakşehir) üstlenmiştir. Tarih 2007 yılının Eylül ayını gösterdiğinde şehir için tamamen yeni bir sistem olan ve İstanbul'a özgün şekilde tasarlanan metrobüs devreye alınır. İlk etapta Avcılar-Topkapı arasında hizmete giren metrobüs bir yıl sonra Söğütlüçeşme'ye uzatılarak şehrin iki yakasını en kısa yoldan birbirine bağlar. Bu haliyle dünyada iki kıtayı birbirine bağlayan tek sistem olarak yerini alan metrobüs projesi İETT'ye ulusal ve uluslararası alanda pek çok ödül kazandırır.

Şekil 5.2. İETT metrobüs (İETT, 2011)

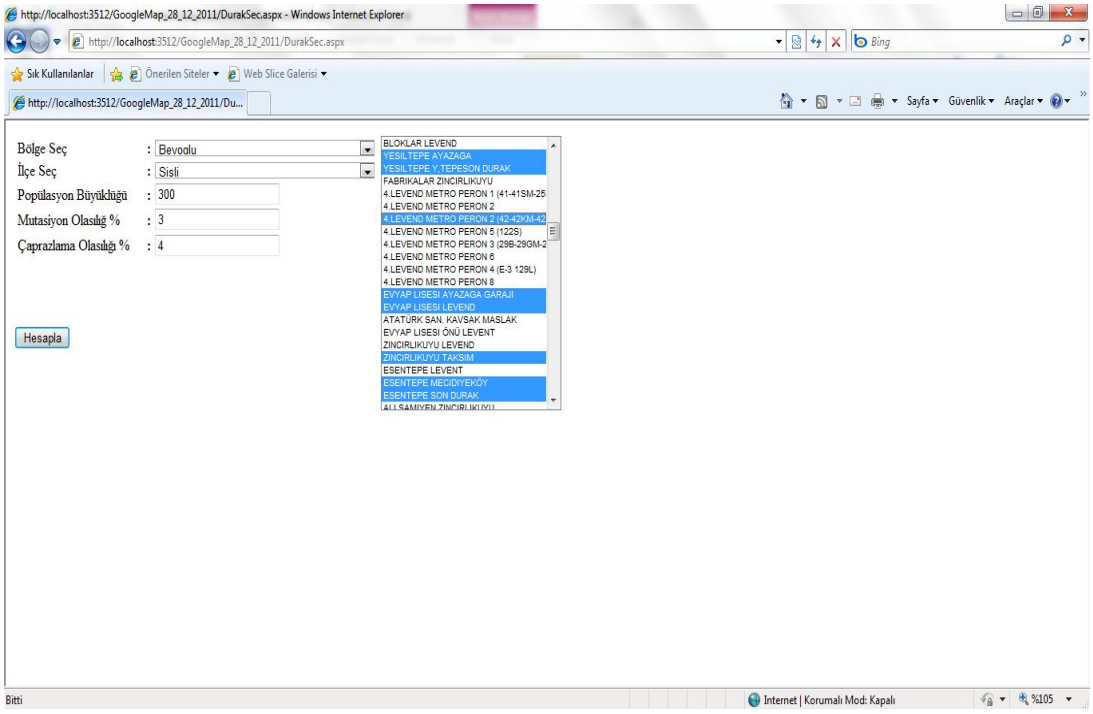
İstanbul'da sürekli artan yolculuk talebini karşılamak ve İETT'nin mevcut filosunu desteklemek amacıyla İstanbul Büyükşehir Belediyesi iştirakiyle Otobüs A.Ş. firması kurularak 2011 yılının mayıs ayında hizmet vermeye başlar (İETT, 2011). 2661 araç filosu ile 11 lokasyon (garaj) ile 10.000'i aşkın çalışanı, 10854 durak noktası ile aylık ortalama 46.004.998 yolcu taşımakta İstanbul halkına hizmet vermektedir. Böyle büyük bir kuruluşun iyi bir hizmet vermesi ancak iyi bir yönetim ve denetim ile mümkündür. Özellikle durak, hat ve araç denetiminin çok iyi yapılması gerekir. Bunun için İETT bünyesinde denetim amir unvanı ile 80'e yakın personel bu görevi sahada yürütmektedir. Personelin bölgeleri öncelikle Anadolu bölgesi, Beyoğlu bölgesi, İstanbul bölgesi olarak üçe ayrılmaktadır. Bu bölgelerin her biri 100'e yakın işletme şefliği olarak ayrılmıştır. 2011 Ekim ayı 514 denetim yapılmıştır. Bu denetimin 10854 noktadan ve İstanbul gibi büyük bir şehirde bütün alanın dağılımı düşünüldüğü zaman çok uzun bir süre alan ve çok maliyetli bir iş olduğu ortaya çıkmaktadır. Bu denetimler sırasında her bir denetimci kendi bölgesinin bütün noktalarını dolaşmak zorundadır. Problemin bu yönü ile gezgin satıcı problemine çok uygun olduğu aşikardır. Şuan bu noktalar personelin kendi tahminine göre bu denetlemektedir. Bu denetim düzgün bir şekilde yapılmadığında kurum zarar etmektedir.

5.3 Uygulamanın Tanıtılması

Gezgin Satıcı Problemin çözümü için geliştirilen GA temeli yöntem İETT denetim ekibi tarafından optimal güzergahın bulunması için uygulanmıştır. Yöntemin probleme uygulanması için problemde bazı verilerin eksik olmasından dolayı çeşitli varsayımlar yapılarak bu veriler elde edilmiştir. Özellikle bütün durakların bir birine uzaklığı verisi olmamasına karşın durakların koordinatları (enlem, boylam) mevcuttur. Koordinatlardan yola çıkılarak duraklar arasındaki mesafe bulunmuştur. Bunun için Google maps apisinden faydalanmıştır.

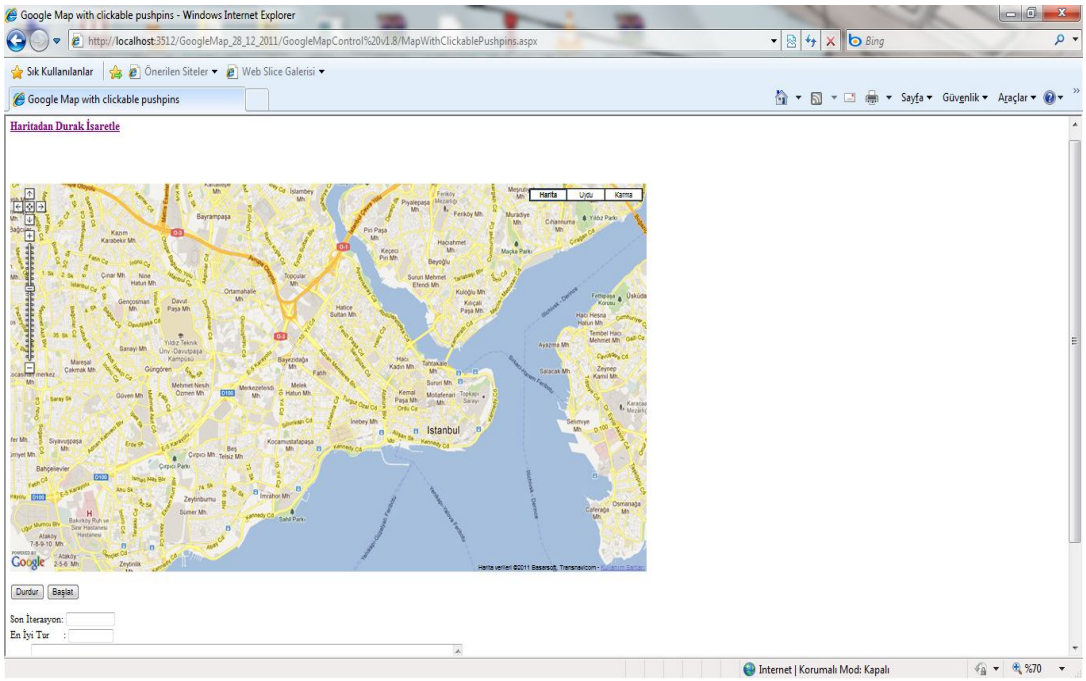
Bu tezde genetik algoritmaların aşamaları gezgin satıcı problemine uygulandı. Yapılan uygulama Microsoft Visual Studio 2010 ortamında Asp.Net ve C# programlama dili, Java Script, Google Maps apisi kullanılarak geliştirilmiştir.

Uygulama dört sayfadan oluşmaktadır. Uygulamanın "Durak Seç" sayfasında önce bölge seçilir. Sonra bölgeye ait ilçe seçilir. İlçede bulunan duraklar çoklu olarak seçilir. İlgili değerler girildiğinde "Hesapla" butonuna basılır. "Durak Seç" sayfası Şekil 5.3'de gösterilmiştir.



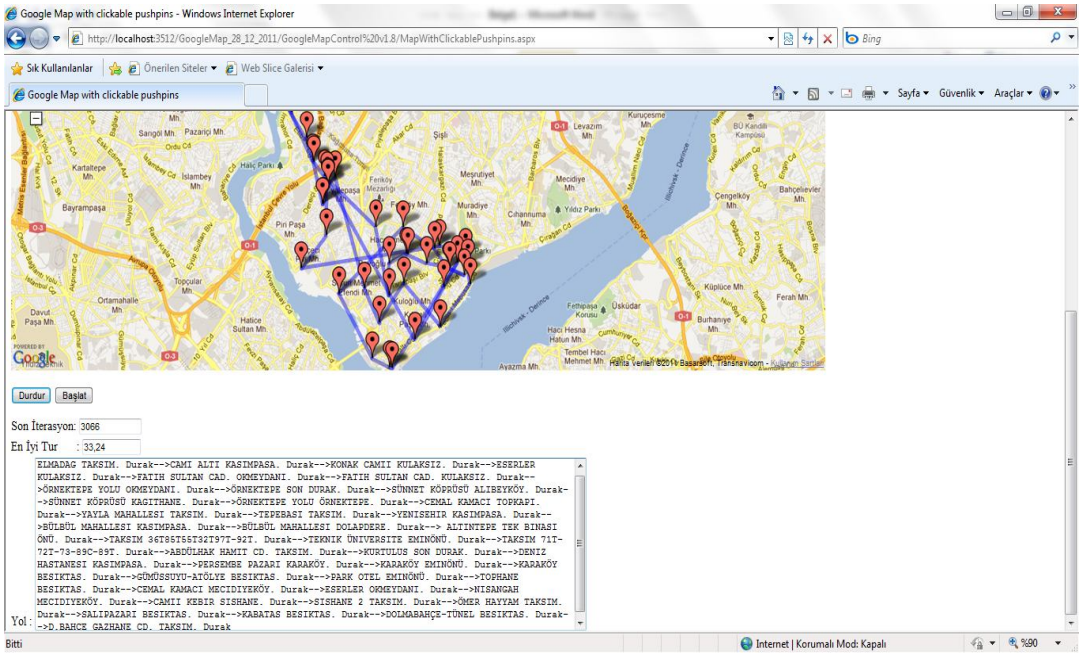
Şekil 5.3. Programın durak seç sayfası

Yeni gelen hesaplama ekranda "Durdur" butonuna basılarak hesaplama o anki iterasyonda durdurulur. "Hesaplama" ekranı Şekil 5.4'de gösterilmiştir.



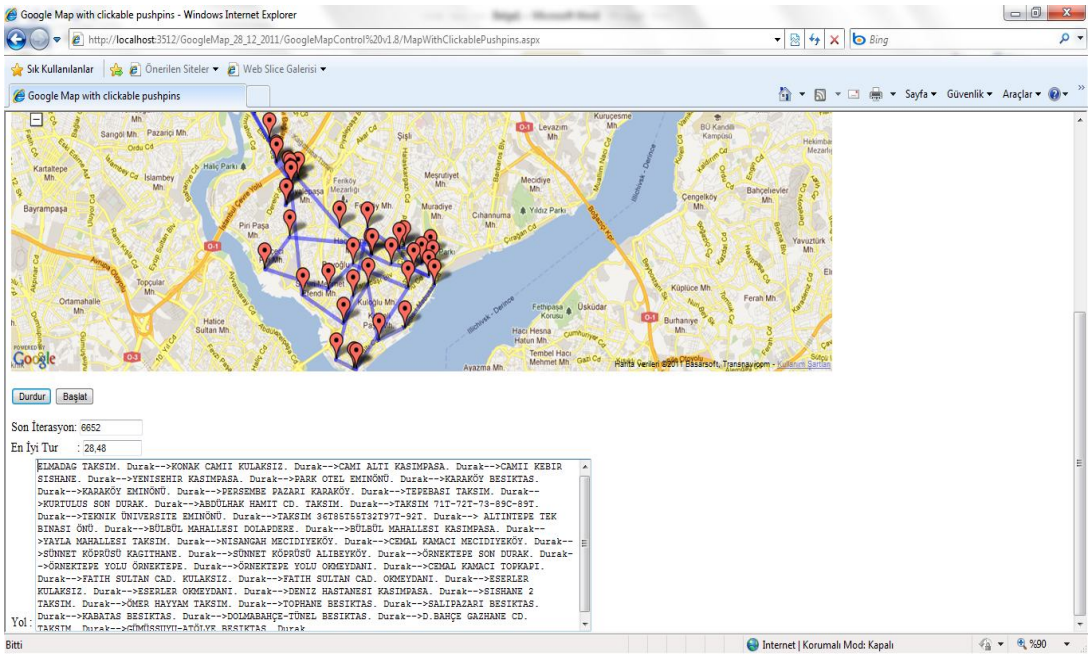
Şekil 5.4. Programın hesaplama sayfası

İterasyon sayısı 3066 olduğunda bulunan optimal güzergah ve duraklar arası toplam mesafesi Şekil 5.5'te gösterilmiştir.



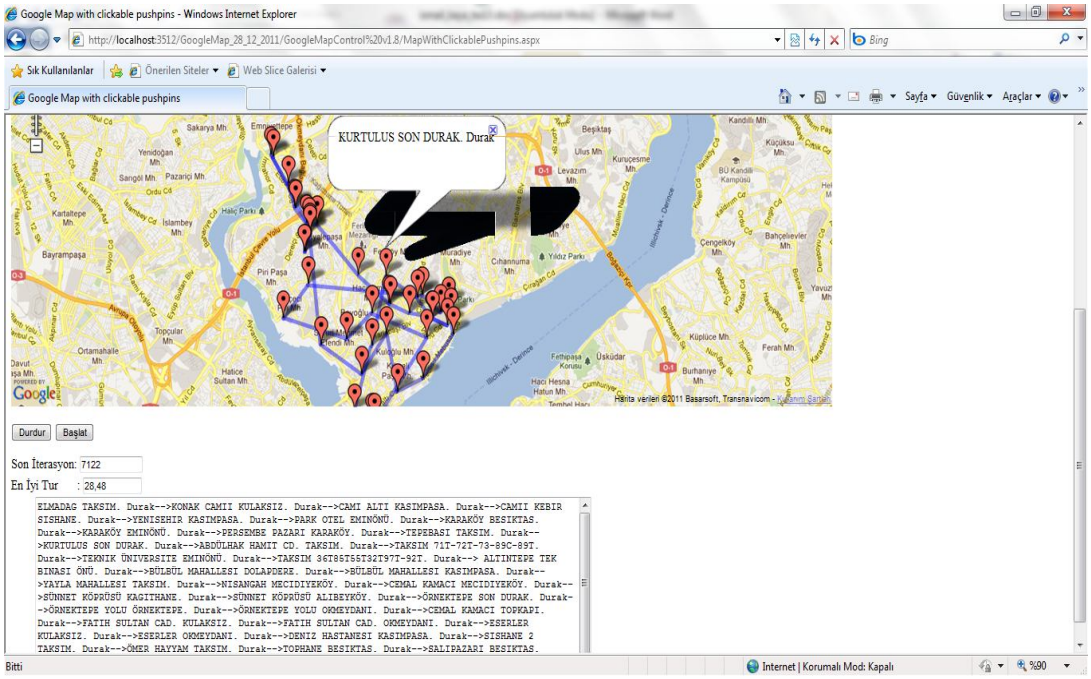
Şekil 5.5. Hesaplama sayfasında 3066 iterasyondaki tur

İterasyon sayısı 6652 olduğunda bulunan optimal güzergah ve duraklar arası toplam mesafesi Şekil 5.6'te gösterilmiştir.



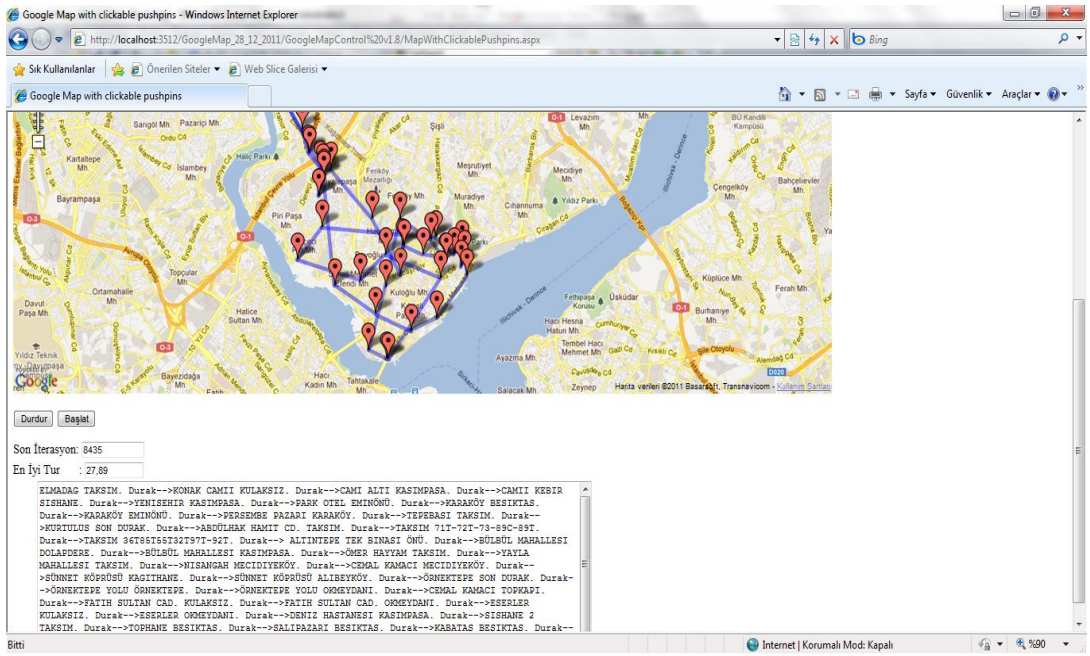
Şekil 5.6. Hesaplama sayfasında 6652 iterasyondaki tur

İterasyon sayısı 7122 olduğunda bulunan optimal güzergah ve duraklar arası toplam mesafesi Şekil 5.7'te gösterilmiştir.



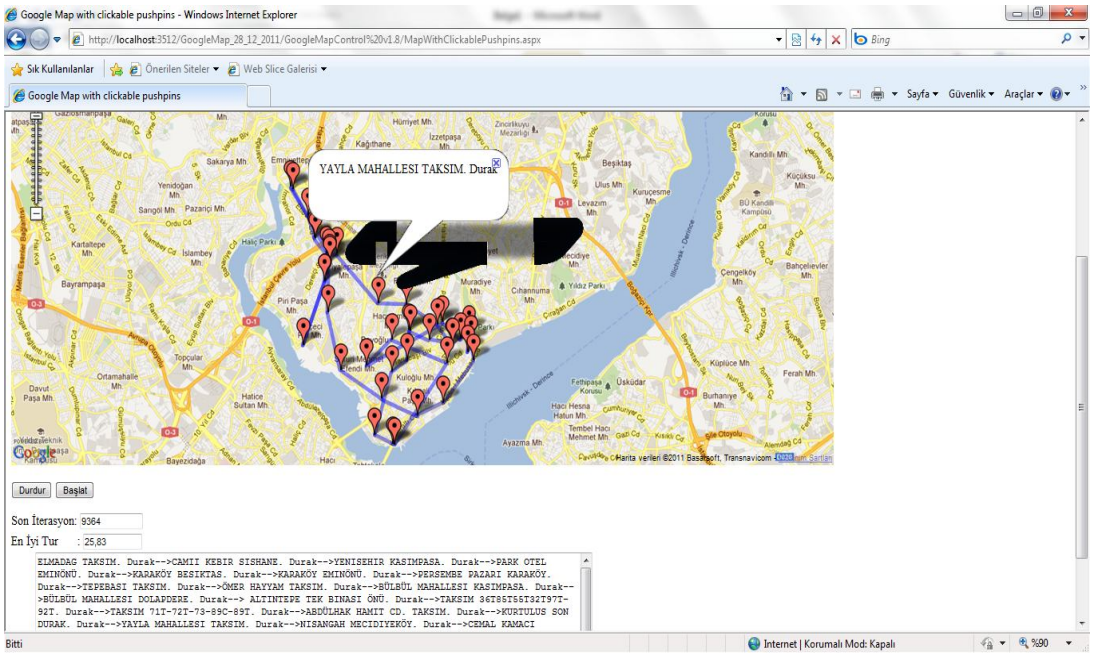
Şekil 5.7. Hesaplama sayfasında 7122 iterasyondaki tur

İterasyon sayısı 8435 olduğunda bulunan optimal güzergah ve duraklar arası toplam mesafesi Şekil 5.8'da gösterilmiştir.



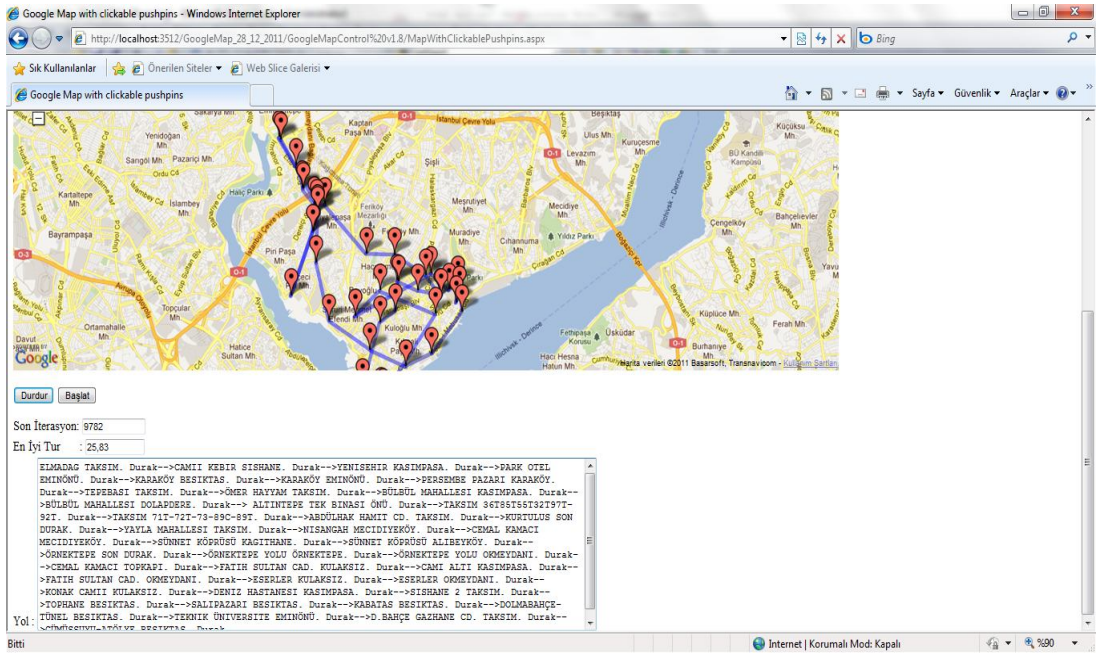
Şekil 5.8. Hesaplama sayfasında 8435 iterasyondaki tur

İterasyon sayısı 9364 olduğunda bulunan optimal güzergah ve duraklar arası toplam mesafesi ve herhangi bir durağa tıklandığında durak ismi Şekil 5.9'de gösterilmiştir.



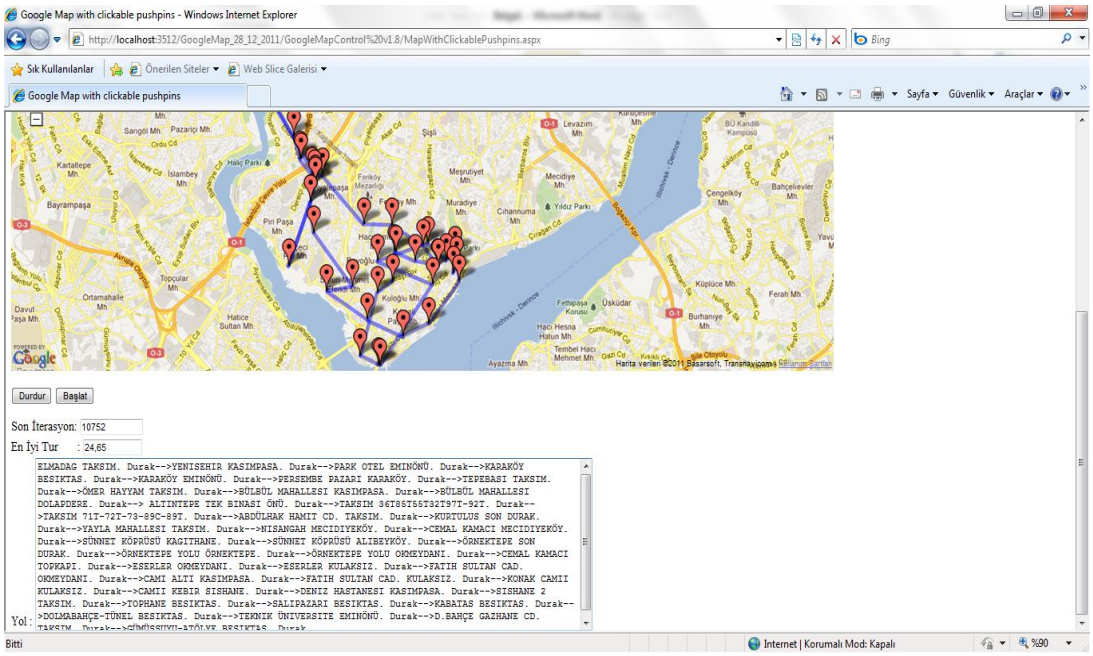
Şekil 5.9. Hesaplama sayfasında 9364 iterasyondaki tur

İterasyon sayısı 9782 olduğunda 40 durak seçildiğinde yaklaşık bir dakika içerisinde bulunan optimal güzergah ve duraklar arası toplam mesafesi Şekil 5.10'de gösterilmiştir.



Şekil 5.10. Hesaplama sayfasında 9782 iterasyondaki tur

İterasyon sayısı 10752 olduğunda 40 durak seçildiğinde yaklaşık bir dakika içerisinde bulunan optimal güzergah ve duraklar arası toplam mesafesi Şekil 5.11'da gösterilmiştir.



Şekil 5.11. Hesaplama sayfasında 10752 iterasyondaki tur

Tablo 5.1. Farklı iterasyondaki en iyi tur mesafesi

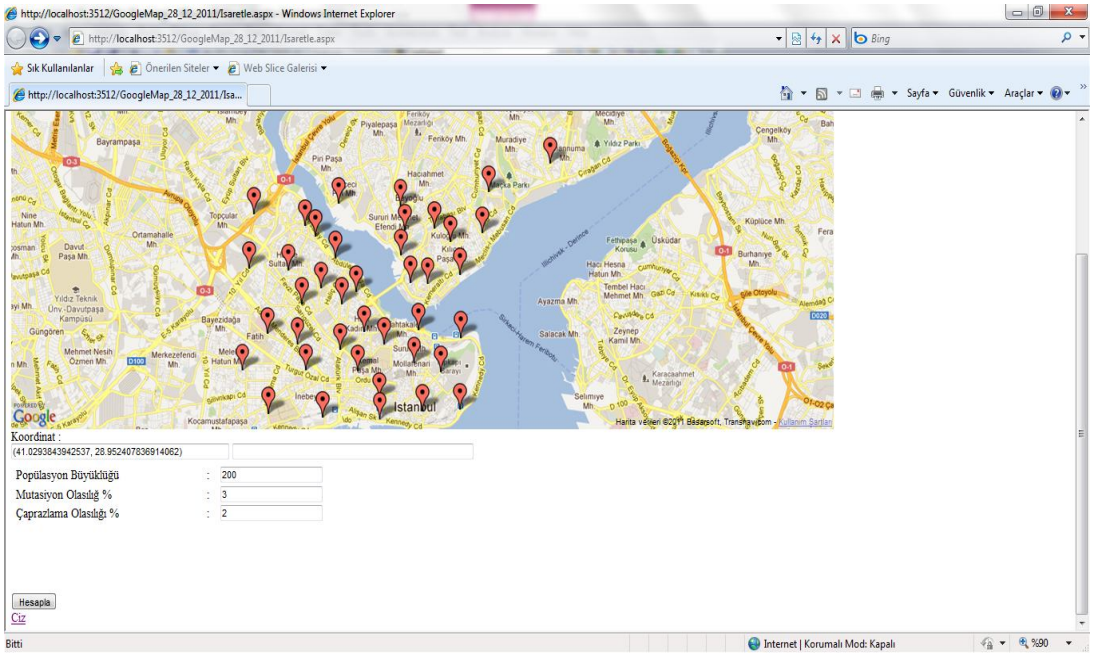
İterasyon Sayısı	En İyi Tur Mesafesi
3066	33.24
6652	28.48
7122	28.48
8435	27.89
9364	25.83
9782	25.83
10752	24.65

Tablo 5.1 de görüldüğü gibi farklı iterasyonda optimal güzergah ve tur mesafelere değiştiği görülmüştür. Bu da iterasyon sayısı artıkça genelde optimal güzergah ve tur mesafesi azalmaktadır. Bazen de belli iterasyonda tur mesafesi aynı olduğu görülmektedir. Örneğin 6652 iterasyondaki tur ve 7122 iterasyondaki tur mesafeleri aynı olduğu görülmüştür. Ancak 9782 ve 10752 iterasyonlara bakıldığında tur mesafesinin azaldığı ve daha optimal güzergahlar bulunduğu görülmektedir.

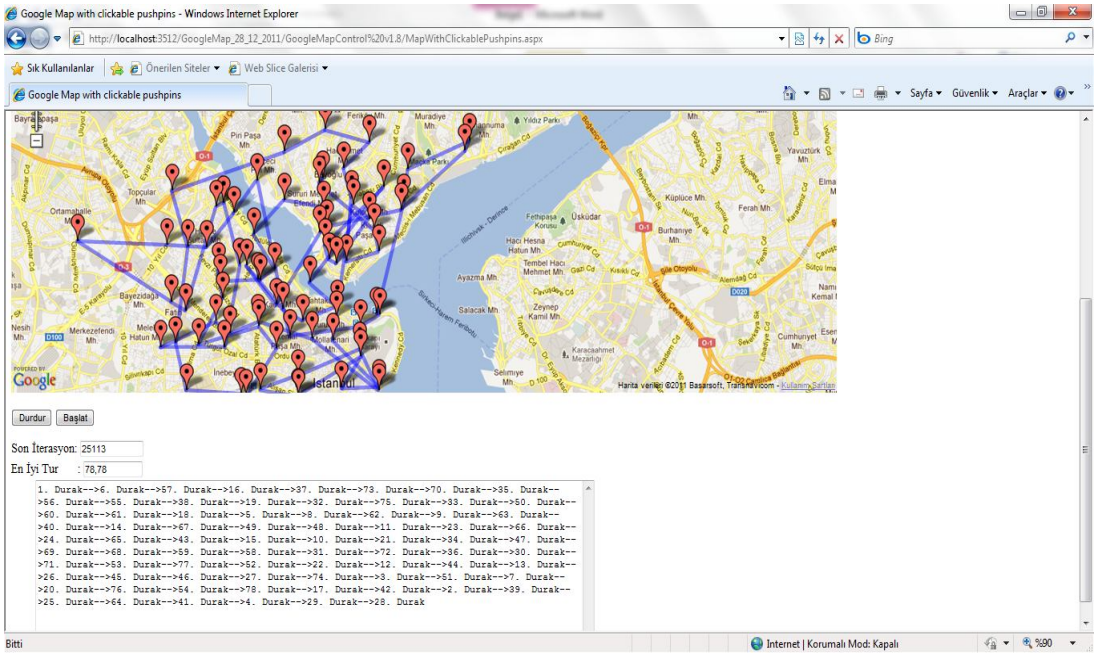
Kesin çözüm yöntemi ile 20 şehir bulunduğu bir bölgede (n-1)!/2 = 19!/2 = 60.822.550.204.416.000 adet farklı tur mevcuttur. Sadece her bir turun uzunluğunun belirlenmesi ve en iyi turun bulunması işlemi için 3 Ghz işlemci hızına sahip bir bilgisayarda denemeler yaklaşık olarak 234 gün sürer (Terzi, 2009). Bu uygulamada

40 durağın bulunduğu turun uzunluğu belirlenmesi ve en iyi turun bulunması işlemi için 2.53 Ghz işlemci hızına sahip bilgisayarda yaklaşık olarak bir dakika sürmüştür.

Haritadan "Durak İşaretle" sayfasında kullanıcı rastgele haritada istediği noktaları işaretleyerek ve ilgili değerler girilerek "Hesapla" butonuna basıldığında hesaplama işlemi yapılmaktadır. Şekil 5.12'de ve Şekil 5.13'de gösterilmiştir.

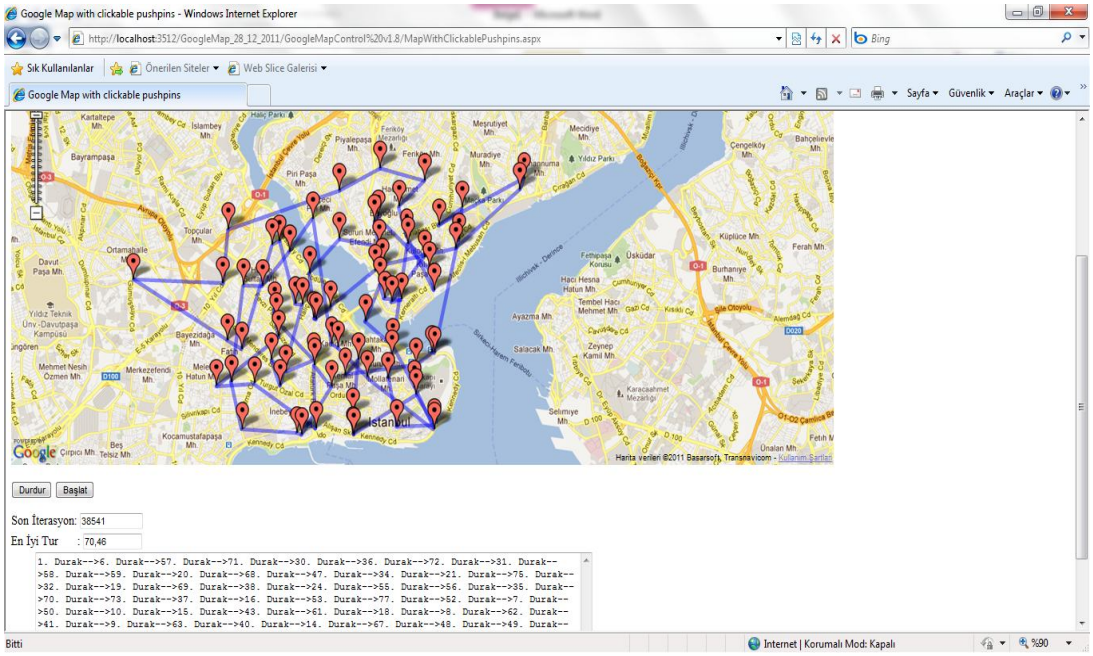


Şekil 5.12. Haritadan durak işaretleme



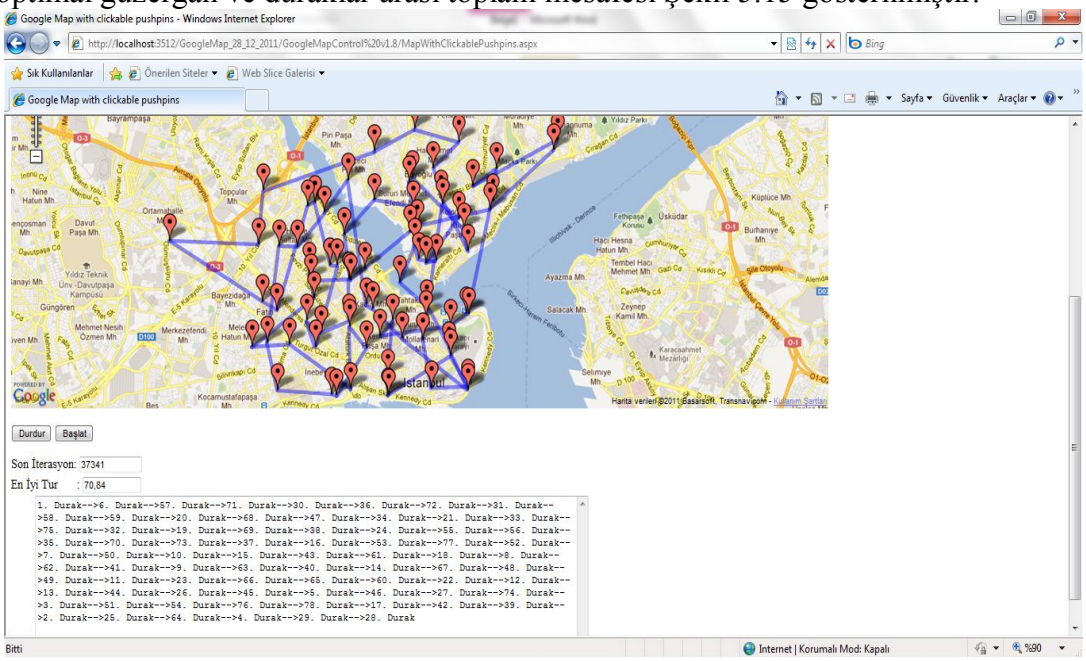
Şekil 5.13. 25113 iterasyondaki tur

Haritadan durak işaretle sayfasında iterasyon sayısı 38541 olduğunda bulunan optimal güzergah ve duraklar arası toplam mesafesi Şekil 5.14 de gösterilmiştir.



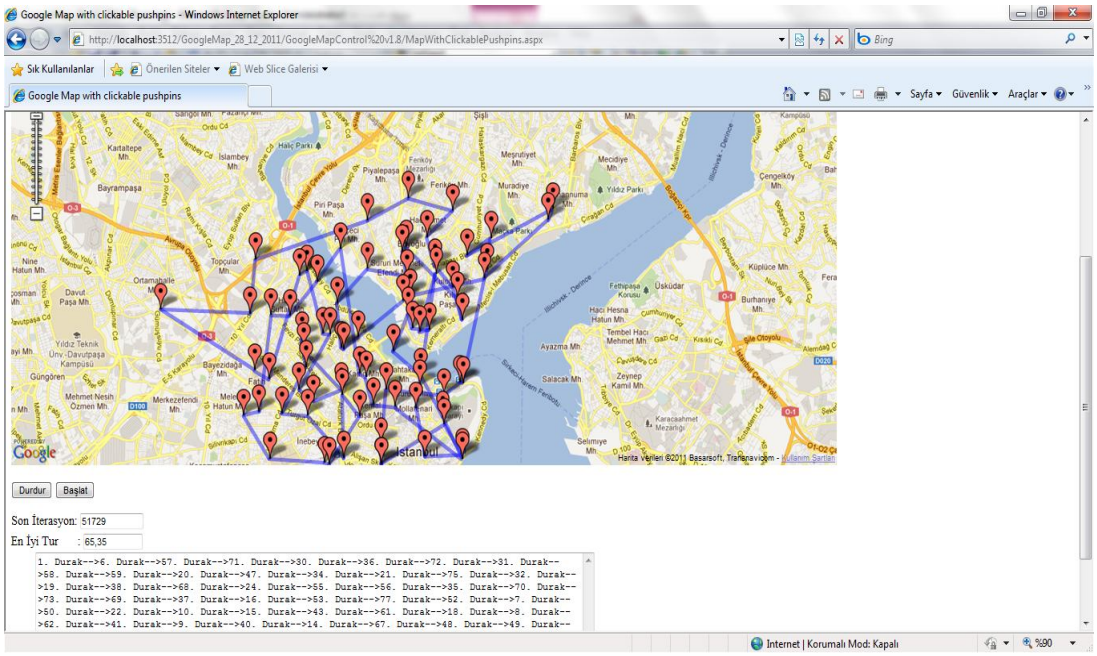
Şekil 5.14. 38541 iterasyondaki tur

Haritadan durak işaretle sayfasında iterasyon sayısı 37341 olduğunda bulunan optimal güzergah ve duraklar arası toplam mesafesi Şekil 5.15 gösterilmiştir.



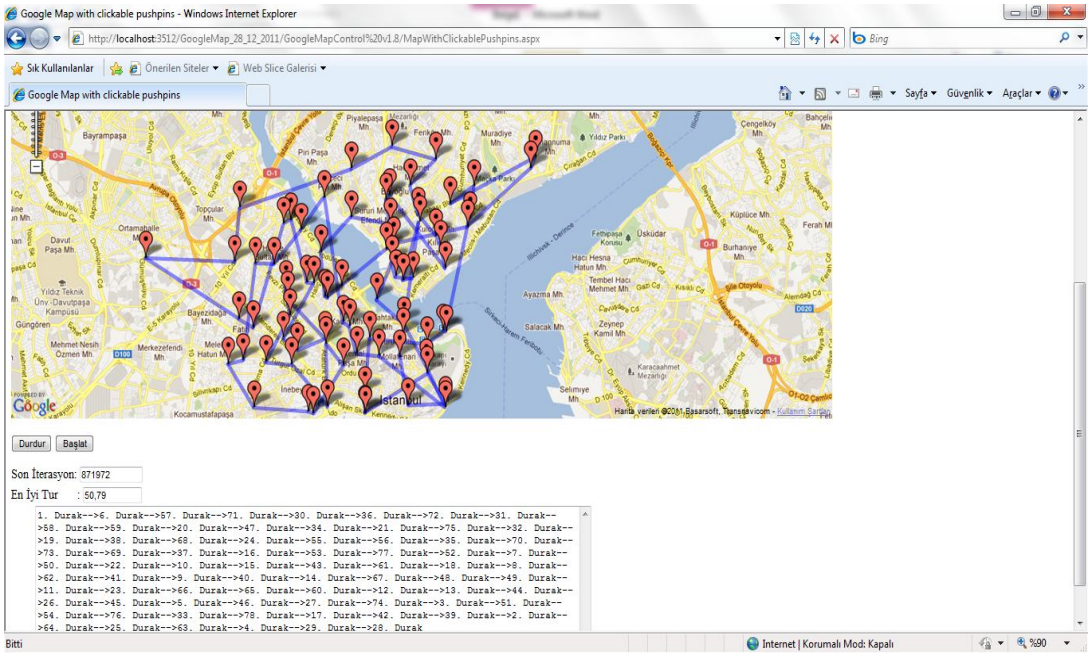
Şekil 5.15. 37341 iterasyondaki tur

Haritadan durak işaretle sayfasında iterasyon sayısı 51729 olduğunda bulunan optimal güzergah ve duraklar arası toplam mesafesi Şekil 5.16 de gösterilmiştir.



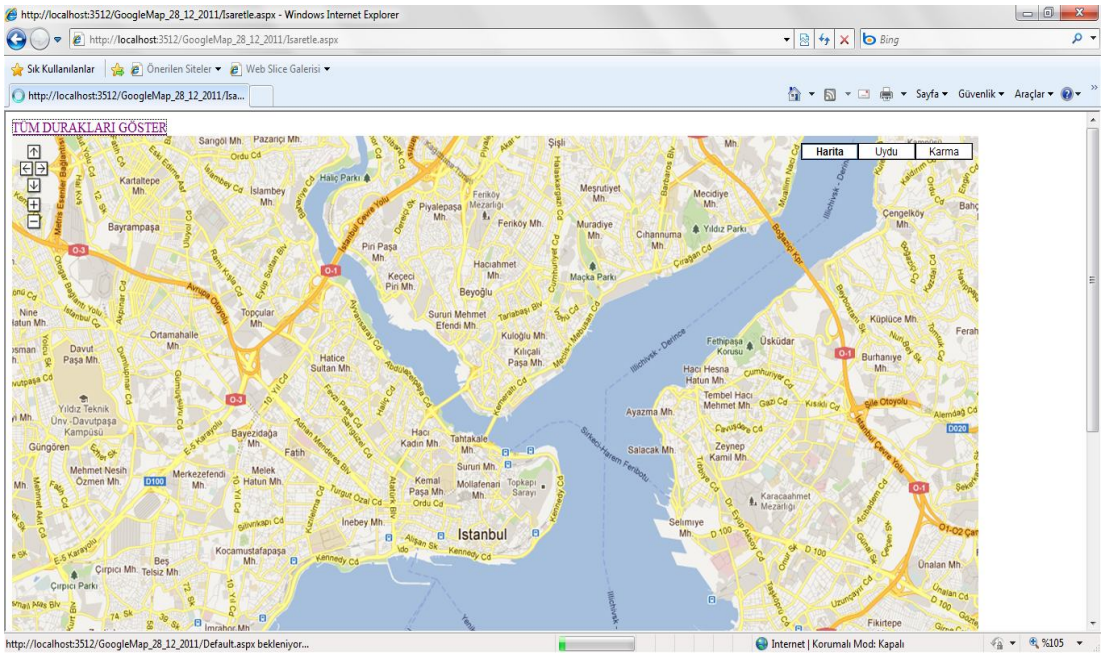
Şekil 5.16. 51729 iterasyondaki tur

Haritadan durak işaretle sayfasında iterasyon sayısı 871972 olduğunda bulunan optimal güzergah ve duraklar arası toplam mesafesi Şekil 5.17 gösterilmiştir.



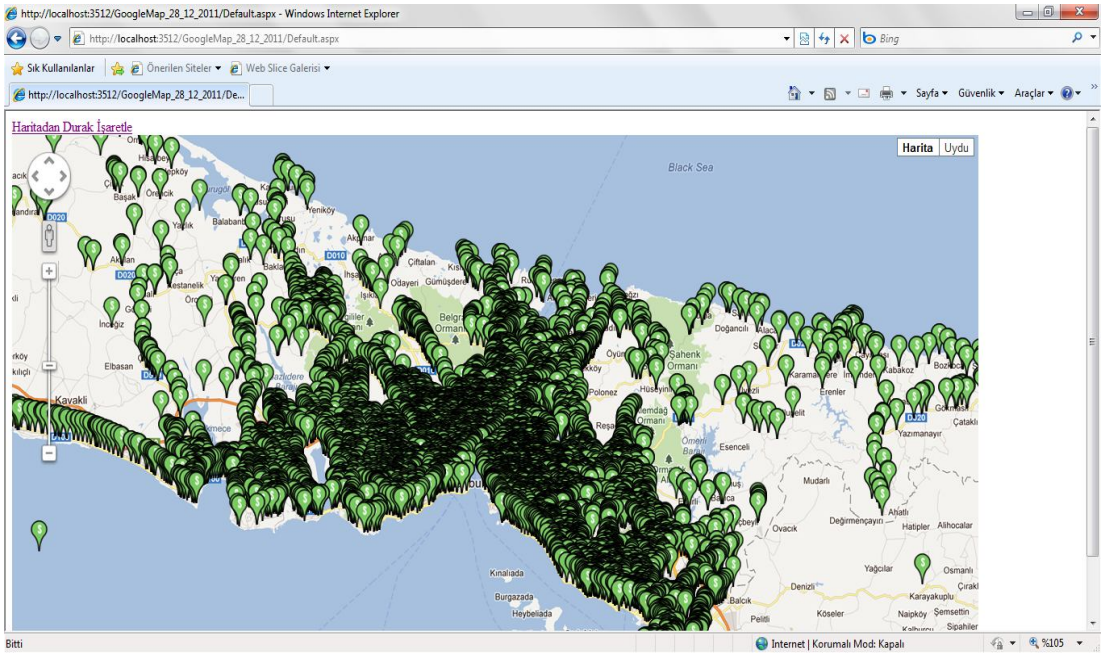
Şekil 5.17. 871972 iterasyondaki tur

"Tüm Durakları Göster" sayfasında İETT ye ait bütün duraklar haritadan gösterilmektedir. Ekran görüntüleri Şekil 5.18'da gösterilmiştir.



Şekil 5.18. Tüm durakları gösteren sayfa

Programda haritada gösterilen duraklara tıklandığında durak adlarını göstermektedir. “Haritadan Durak İşaretle” ekranında tıklanan yerler sıra numaralar olarak durak adları verilmektedir. “Haritadan Durak İşaretle” sayfası ve bu harita üzerinde seçilen tüm duraklar Şekil 5.19’da gösterilmiştir.



Şekil 5.19. Haritada tüm durakların gösterimi

Örnek Gezgin Satıcı Problemi

Genetik Algoritmaların uygulanmasına ilişkin incelenecek olan Gezgin Satıcı Problemine ait örnekte 4 durak alınmıştır. Amaç deneticinin her durağa yalnızca bir kere uğrayarak turunu en kısa yoldan tamamlamasıdır. Deneticinin başladığı durağa geri dönme zorunluluğunun olmadığı varsayılacaktır. Duraklar birbirine bağlı olmakta ve uzaklıklar verilmektedir. Bu şartlar içerisinde toplam alınan yolun minimum değerini bulmaya çalışılacaktır. Tablo 5.1'de duraklar ve duraklar arası mesafeler verilmiştir. Duraklar arası mesafeler bir n x n matrisi şeklinde verilmektedir (n durak sayısı).

Tablo 5.2. Duraklar arası mesafe

Duraklar	0	1	2	3
0	0	3	4	7
1	3	0	5	4
2	4	5	0	6
3	7	4	6	0

1.Adım

t=0. G(t) jenerasyonunda

N adet kromozom içeren ilkel popülasyon rastgele oluşturulur. Her bir kromozom problemin olası çözümüdür. Bu örnekte N=6 olsun.

6 adet kromozom içeren popülasyon rastgele oluşturulur.

Rastgele oluşturulan 6 tane kromozom aşağıda gösterilmektedir:

v_1=[1 2 3 0]

v_2=[1 3 0 2]

v_3=[2 1 3 0]

v_4=[3 2 0 1]

v_5=[0 1 3 2]

v_6=[2 0 1 3] .

Her v_i, i $\in$ [1,6], kromozomu için f(v_i) uygunluk değeri aşağıdaki formül ile

hesaplanır:

$$f(v_i)=\frac{1}{g(v_i)}.$$

Burada $g(v_i)$ deneticinin geçtiği toplam yolun mesafesidir. Açıktır ki, $g(v_i)$ yolu ne kadar kısa olursa v_i kromozomun uygunluk değeri o kadar yüksek olacaktır.

v_1 = [1 2 3 0] için $g(v_1)$ =5 km +6 km +7 km = 18 km

v_2 = [1 3 0 2] için $g(v_2)$ =4 km +7 km +4 km = 15 km

v_3 = [2 1 3 0] için $g(v_3)$ =5 km +4 km +7 km = 16 km

v_4 = [3 2 0 1] için $g(v_4)$ = 6 km +4 km +3 km = 13 km

v_5 = [0 1 3 2] için $g(v_5)$ = 3 km +4 km +6 km = 13 km

v_6 = [2 0 1 3] için $g(v_6)$ = 4 km +3 km +4 km = 14 km

$$f(v_i)=\frac{1}{g(v_i)} \rightarrow f(v_1)=\frac{1}{18}=0.06$$

$$f(v_2)=\frac{1}{15}=0.07$$

$$f(v_3)=\frac{1}{16}=0.06$$

$$f(v_4)=\frac{1}{13}=0.08$$

$$f(v_5)=\frac{1}{13}=0.08$$

$$f(v_6)=\frac{1}{14}=0.07$$

$f_{max}=\frac{1}{13}$ olduğundan 1. Jenerasyonda en iyi güzergahları içeren kromozomlar v_4 ve v_5 kromozomlarıdır.

2.Adım

Uygunluk değerlerinin toplamı aşağıdaki formül ile hesaplanır:

$$F=\sum_{i=1}^{6} f(v_i) = 0.06+0.07+0.06+0.08+0.08+0.07 = 0.42$$

Daha sonra kromozomların bir sonraki G(t) jenerasyonuna (t=t+1) seçilme olasılıkları hesaplanır ve rulet çarkı tasarlanır.

$p_1 = \frac{f_1}{F} = \frac{0.06}{0.42} = 0.14$ $q_1 = 0.14$

$p_2 = \frac{f_2}{F} = \frac{0.07}{0.42} = 0.17$ $q_2 = 0.31$

$p_3 = \frac{f_3}{F} = \frac{0.06}{0.42} = 0.14$ $q_3 = 0.45$

$p_4 = \frac{f_4}{F} = \frac{0.08}{0.42} = 0.19$ $q_4 = 0.64$

$p_5 = \frac{f_5}{F} = \frac{0.08}{0.42} = 0.19$ $q_5 = 0.83$

$p_6 = \frac{f_6}{F} = \frac{0.07}{0.42} = 0.17$ $q_6 = 1.00.$

Önceki bölümlerde belirttiğimiz gibi, doğal seçilimde sadece uygun türler hayatta kalabilir, üreyebilir ve sonraki jenerasyona onların genleri geçilebilir. Genetik Algoritmalar benzer bir yaklaşım kullanır. Fakat doğadan farklı bir şekilde, kromozom popülasyonunun boyutu bir sonraki jenerasyonda değiştirilmemiş kalır. Örneğin v_1 ve v_2 kromozomların çok düşük bir seçilme olasılığı varken v_4 ve v_5 kromozomlarının yeterli şansı vardır.

Örnekte, altı kromozomlu bir başlangıç popülasyonu oluşturuldu. Böylece sonraki jenerasyonda aynı popülasyon boyutunu korumak için altı tane rastgele sayı üretilecektir (bu, rulet tekerleğinin altı kez dönmesi gibidir.).

[0-1] aralığında rastgele 6 sayı üretilir:

$r_1 = 0.12$ $r_2 = 0.30$ $r_3 = 0.82$ $r_4 = 0.60$ $r_5 = 0.03$ $r_6 = 0.09.$

Rulet tekerleği 6 kere döndürülür. Her i'inci adımda aşağıdaki karşılaştırma yapılır. Eğer $r_i < q_1$ ise i. Kromozomun yerine bir önceki jenerasyondan 1. Kromozom geçer. Aksi taktirde i. kromozomun yerine $q_{j-1} < r_i \leq q_j$ koşulunu sağlayan j. kromozom geçer. Seçim sonucunda oluşan 2. jenerasyonun 1. popülasyonu aşağıdaki gibidir:

$v_1 = v_{1eski} = [\ 1\ 2\ 3\ 0\ ]$

$v_2 = v_{2eski} = [\ 1\ 3\ 0\ 2\ ]$

$v_3 = v_{5eski} = [\ 0\ 1\ 3\ 2\ ]$

$v_4 = v_{4eski} = [\ 3\ 2\ 0\ 1\ ]$

$v_5 = v_{1eski} = [\ 1\ 3\ 0\ 2\ ]$

$v_6 = v_{1eski} = [\ 1\ 3\ 0\ 2\ ].$

Seçim sonucunda oluşan popülasyonundaki kromozomlar belli bir olasılıkla çaprazlamaya tabi tutulur. Çaprazlama sonucunda meydana gelen yeni kromozomlar eski kromozomların yerine geçerler.

3.Adım

Çaprazlama kullanım amacı bir önceki popülasyondan gelen kromozomlardan daha iyi uygunluk değerine sahip kromozomlar oluşturmaktır. Çaprazlamayı popülasyondaki bütün bireylere uygulama zorunluluğu yoktur. Popülasyondaki bazı bireylerin çaprazlama işlemine uygulanmadan bir sonraki popülasyona geçmesi için çaprazlama oranı belirlenir. Popülasyondaki her birey için 0 ile 1 aralığında rastgele bir reel sayı üretilir. Bu sayı çaprazlama oranından küçük ise birey çaprazlama işlemine maruz kalır. Sayı büyük ise çaprazlama işlemi uygulanmaz.

Seçilen her bir çift ebeveyn kromozom için çaprazlama uygulanır. Rastgele olarak seçilen bir çaprazlama noktasından sonraki kromozomlardaki parçalar yer değiştirir. Sonuç olarak iki yeni yavru oluşturulur. Örneğin, çaprazlama ihtimali p_c=0.5 olduğunda p_c x pop_size = 0.5 x 6 = 3, 3 – 1 = 2 kromozomun çaprazlamaya tabi tutulacağı beklenilmektedir. Kromozomların seçimi aşağıdaki adımlar ile gerçekleşmektedir. Önce 6 tane rastgele sayılar üretilir: r_1 = 0.12 r_2 = 0.72 r_3 = 0.82 r_4 = 0.60 r_5 = 0.53 r_6 = 0.09. Daha sonra sayıların her biri p_c ile karşılaştırılır. Sayı p_c den küçük olduğunda bu sayı ile temsil edilen kromozom çaprazlama için seçilir. Örneğin, $r_1 < p_c$ ve $r_6 < p_c$ olduğu için v_1 ve v_6 kromozomlar çaprazlamaya tabi tutulur. Rastgele seçilen 2. Pozisyonundan sonra kromozomlarda gen değişimi yapılır:

v_1 = [1 | 2 3 0] → [1 3 0 2]

v_6 = [1 | 3 0 2] → [1 2 3 0].

Fakat v_1 ve v_4 çaprazlama sonucu aşağıdaki gibidir:

v_1 = [1 | 2 3 0] → [1 2 0 1]

v_4 = [3 | 2 0 1] → [3 2 3 0].

Görüldüğü gibi yeni oluşan kromozomlar, Gezgin Satıcı Problemi için olumlu bir sonuç vermez. Çünkü denetimci tüm durakları gezmeli ve her durağa yalnızca bir kez uğramalıdır. Fakat çaprazlama sonrası kromozomlarda görüldüğü gibi, denetimci tarafında denetimci tarafında bazı duraklara hiç gidilmemekte ya da aynı duraklara

iki kere gidilmektedir. v_1 kromozomundan görüldüğü gibi 1. Durağa iki kere gidilmekte, v_4 'ten görüldüğü gibi gidildiği 3. Durağa ise hiç gidilmemektedir. Böyle bir olumsuz durumda ikinci bir işleme gerek duyulur. Yani kısıtlamaları sağlamayan uygunsuz kromozomların standartlaştırılması gerekmektedir. Standartlaştırma için şöyle basit bir kural uygulanabilir: "Kromozom içinde tekrar eden ilk durak ziyaret edilmeyen en küçük numaralı durak ile değiştirilsin". Bu standartlaştırma kuralı akla ilk gelen kurallardan biridir. Herhangi başka bir kural da standartlaştırma amacı için uygulanabilirdi. Genetik Algoritma böyledir: hemen hemen her şey plansız programsızdır. Tıpkı doğadaki gibidir. Örnekte sorunlu kromozomların standartlaştırılması aşağıdaki şekilde yapılmaktadır:

v_1 = [1 | 2 3 0] → [1 2 0 1] →[1 2 0 3]

v_4 = [3 | 2 0 1]→ [3 2 3 0] → [3 2 1 0].

Çaprazlama sonucunda oluşan yeni popülasyon aşağıdaki gibidir:

v_1 = [1 3 0 2]

v_2 = [1 3 0 2]

v_3 = [0 1 3 2]

v_4 = [3 2 0 1]

v_5 = [1 3 0 2]

v_6 = [1 2 3 0] .

Çaprazlama sonucunda oluşan popülasyonda genetik çeşitliliği sağlamak amacıyla belli bir olasılık değeri ile mutasyon uygulanır.

4.Adım

Mutasyon işlemi çaprazlama işleminden sonra gerçekleşir. Mutasyon işlemi yeni üretilmiş bir kromozomun genlerinde rastgele 0 'dan 1 ' e veya 1 'den 0 'a değişiklik yapmak için kullanılır. İkili kodlamada mutasyon işlemi için genlerin ne oranda seçileceği mutasyon oranı ile belirlenir. Hangi genlerin mutasyona uğrayacağını belirlemek için bireylerin genlerinin sayısı kadar rastgele r sayısı üretilir.

Örnekte permütasyon kodlama türü kullanıldığından dolayı, mutasyon rastgele seçilen kromozomlar üzerindeki iki genin yerlerini değiştirmekle gerçekleştirilir. Örneğin, mutasyon ihtimali p_m =0.3 olduğunda p_m x pop_size = 0.3 x 6 = $1.8 \approx 2$ gen

mutasyona tabi tutulacaktır. 6x4 = 24 rastgele sayı üretilir. $r_i < p_m$ şartını sağlayan ve ondan bir sonraki gen mutasyon için seçilir. [0-1] aralığında rastgele 24 sayı üretilir:

$r_1 = 0.42 \quad r_2 = 0.82 \ldots r_{22} = 0.23, \quad r_{23} = 0.32 \quad r_{24} = 0.69.$

$r_{22} < p_m$ olduğundan 6. kromozomun 2. ve 3. genleri yer değiştirir:

$v_6 = [\ 1\ \mathbf{2}\ \mathbf{3}\ 0\ ] \rightarrow [\ 1\ 3\ 2\ 0\ ].$

Mutasyon sonucunda oluşan popülasyondaki yeni kromozomlar eski kromozomların yerini alır. Mutasyon soncunda oluşan yeni popülasyon aşağıdaki gibidir:

$v_1 = [\ 1\ 3\ 0\ 2\ ]$

$v_2 = [\ 1\ 3\ 0\ 2\ ]$

$v_3 = [\ 0\ 1\ 3\ 2\ ]$

$v_4 = [\ 3\ 2\ 0\ 1\ ]$

$v_5 = [\ 1\ 3\ 0\ 2\ ]$

$v_6 = [\ 1\ 3\ 2\ 0\ ].$

5.Adım

Popülasyondaki tüm kromozomların uygunluk değerleri hesaplanır:

$$g(v_1) = 15 \text{ km} f(v_1) = \frac{1}{g(v_1)} = \frac{1}{15} = 0.06$$

$$g(v_2) = 15 \text{ km} f(v_2) = \frac{1}{g(v_2)} = \frac{1}{15} = 0.06$$

$$g(v_3) = 16 \text{ km} f(v_3) = \frac{1}{g(v_3)} = \frac{1}{16} = 0.06$$

$$g(v_4) = 13 \text{ km} f(v_4) = \frac{1}{g(v_4)} = \frac{1}{13} = 0.07$$

$$g(v_5) = 15 \text{ km} f(v_5) = \frac{1}{g(v_5)} = \frac{1}{15} = 0.06$$

$$g(v_6) = 14 \text{ km} f(v_6) = \frac{1}{g(v_6)} = \frac{1}{14} = 0.07.$$

$f_{2.\max} = \frac{1}{13}$ olduğundan 2. Jenerasyonda en iyi güzergahları içeren kromozom v_4 kromozomdur.

6.Adım

Baştan belirlenmiş olan jenerasyon sayısı tamamlanmamışsa 2. Adıma geri dönülür

7.Adım

Her bir t jenerasyon sonucunda bulunan $f_{t.\max}$ değerleri birileri ile karşılaştırılır. Karşılaştırma sonucu olarak bulunan en yüksek uygunluk fonksiyonu değerini sağlayan kromozom en iyi çözümü (en optimal güzergahı) göstermektedir. Örnekte 2. jenerasyon sonunda 4. Kromozom en iyi güzergahı: 3-2-0-1 temsil eder.

6. SONUÇ

Gezgin Satıcı Problemi hayatın birçok alanında karşımıza çıkan bir problemdir. Bunların bir kısmı: ulaşım, lojistik, haberleşme şebekeleri tasarımı, elektronik devre dizaynı, gaz boruları şebekeleri optimizasyonu, görüntü ve ses tanıma, veri tabanı sorgulama optimizasyonu, uçak tasarımı, GSM operatörlerinin baz istasyonlarının yerleşim yerlerinin belirlenmesi, fiziksel sistemlerin kontrolü gibi birçok sektörde kullanılmaktadır. Bu problem kompleks olmasından ve gerçek dünya'da çok geniş uygulama alanı olmasından dolayı araştırmacıların ilgisini çeken bir konudur. Ekonomi alanında önemli bir yere sahip olan bu problem için araştırmacılar yıllardır çalışmalarını sürdürmektedirler.

Yapılan bu çalışmada Gezgin Satıcı Probleminin çözümü için geliştirilen GA-temelli yöntem İETT denetim ekibi tarafından optimal güzergahın bulunması için istenilen optimizasyon probleminin iyi tanımlanması, problem için uygun kodlama yönteminin seçilmesi, Genetik Algoritma operatörlerinden seçim operatörü için probleme uygun yöntemin seçilmesi, Genetik Algoritmanın performansını etkileyen parametrelere uygun değerler verilmesi ve uygunluk fonksiyonunun iyi belirlenmesi ile optimal çözümler alınabilmektedir.

Çalışmanın uygulama kısmında Gezgin Satıcı Problemine benzer bir yapıya sahip İETT de uygulama yapılarak denetim personelinin optimal güzergah belirlemesi için çalışılmıştır. Optimal güzergah belirlenmesi probleminin tanımlanması kolay, fakat çözümü matematiksel olarak çok zor olan bir problemdir. Son yıllarda optimal güzergah belirleme problemini çözmek için çeşitli kesin ve sezgisel çözüm yöntemleri ortaya konulmuştur. Bilgi teknolojilerin son yıllarda hızla gelişmesine ve iyi çözüm yöntemleri ortaya çıkmasına karşın, pek çok büyük boyuttaki probleme çözüm bulunamamıştır. Bu çalışmada optimal güzergah belirlenmesine sezgisel bir yaklaşım olan genetik algoritma yöntemi ile çözüm aranmıştır. Bu yöntem ile uygulamada çok kısa süreler içinde başarılı sonuçlar elde edilmiştir.

7. KAYNAKLAR

Adewuya A A(1996). *New methods in Genetic Search with Real Valued Chromosomes*, Master‟s Thesis, Cambridge:Massachusetts Institute of Technology.

Altay A (2007). *Genetik Algoritma Ve Bir Uygulama*. Yüksek Lisans Tezi. İTÜ. Fen Bilimleri Enstitüsü, İstanbul

Angeline P J (1995). *Evolution revolution: An introduction to the special track on genetic and evolutionary programming*. IEEE Expert Intelligent Systems and their Applications 10

Bäck, T (1993) *Optimal Rates Mutataion Rates in Genetic Search*, Proceedings of the 5th International Conference on Genetic Algorithms, Morgan Kaufmann, Los Angeles

Beasley D , Bull D R ve Martin R R (1993). *An Overview of Genetic Algorithms*: Part 1, Fundamentals,
Broyden G C (1965). *A Class Of Methods for Solving Nonlinear Simultaneous Equations,* Math. Comput.

Cevre U (2008). *Çoklu Gezgin Satıcı Probleminin Çözümü İçin Bir Eniyileme Kütüphanesinin Tasarımı Ve Görsel Yazılım Geliştirme Ortamı İle Birlikte Gerçekleştirimi*, Yüksek Lisans Tezi. Ege Üniversitesi. Fen Bilimleri Enstitüsü, Bornova- İzmir.

Cevre U, Özkan B ve Uğur A (2007). *Gezgin Satıcı Probleminin Genetik Algoritmalarla Eniyilemesi Ve Etkileşimli Olarak Internet Üzerinde Görselleştirilmesi*. Ankara.

Curti H(1975). *Biology*, 2nd Ed., New York: Worth publisher.

Çunkaş M, Akkaya R (2004). *İkili Ve Gerçek Kodlu Genetik Algoritmaların Karşılaştırılması*. S.Ü. Müh. Mim. Fak. Dergisi

Davis L(1991). *Hybridization and numerical representation*.in L.Davis(Ed.), The Handbook of Genetic Algorithms. New York: Van Nostrand Reinhold

Kalaycı T E (2006). *Yapay Zeka Teknikleri Kullanan Üç Boyutlu Grafik Yazılımları İçin“Extensıble 3d” (X3d) İle Bir Altyapı Oluşturulması Ve Gerçekleştirimi*, Yüksek Lisans Tezi. Ege Üniversitesi. Fen Bilimleri Enstitüsü. Bornova- İzmir.

Eshelman L J, Shafter D J (1993). *Real-Coded Genetic Algorithms And İnterval-*

Schemata. in D.L. Whitley (Ed.), Foundations of Genetic Algorithms 2, San Mateo, CA: Morgan Kaufman

Deb K (2001).*Multi-objective Optimization Using Evolutionary Algorithms*. John Wiley and Sons.

Goldberg D E (1993). Making Genetic Algorithms Fly: A Lesson From The Wright Brothers. Adv. Technol. Dev. 2

Goldberg D E (1989). *Genetic Algorithms in Search, Optimization, and machine Learning*, Addisson-Wesley, Reading, MA.

Grant V (1985). *The Evolutionary process*. Columbia University Press,New York

Gutin G, Punnen A P (2002). *The Travelling Salesman Problem*: Applications, Formulations and Variations, Kluwer Academic Publishers

Haataja (1994). *Solving Optimization Problems. CSC- Center for Scientific Computing Ltd, Yliopistopaino*. ISBN 952-9821-02-6. Ed. 1. 232p.(In Finnish)

Haupt Randly L, Haupt ve Sue E (2004*). Practical Genetic Algorithms*. A Willey Interscience Publication, USA

Holland J H (1992). *Genetic Algorithms*. Sci. Am.

Kahvecioğlu A (2004). *Onarılabilir Elemanlara Önleyici Bakımın Etkisi Ve Optimizasyonu*, Mühendis ve Makine, Cilt.45, Sayı: 531, ss.43-51.

Man K F, Tang K S ve Kwong S (1999). *Genetic Algorithms*. Springer. Publishing.
Michalewicz, Z.,*Genetic Algorithms + Data Structures = Evaluation Programs*, Springer-Verlag, New York, 1994

Nabiyev V V (2003). *Yapay Zeka*. Seçkin Yayınları, Ankara

Nearchou, Andreas C (1998), *Path planning of mobile robot using genetic heuristics*, Robotica , Cambridge University Pres, Vol.16, ss.575-588.

Özkan B (2008). *Dinamik Gezgin Satıcı Probleminin Çözümü İçin Bir Eniyileme Kütüphanesinin Tasarımı ve Görsel Yazılım Geliştirme Ortamı ile Birlikte Gerçekleştirimi*. Yüksek Lisans Tezi. Ege Üniversitesi. Fen Bilimleri Enstitüsü, Bornova- İzmir.

Özkan B, Cevre U ve Uğur A (2008). *Melez bir eniyileme yöntemi ile rota planlama*. Akademik Bilişim 2008, Çanakkale.

Özkan R(2003). *Tek Modelli Deterministik Montaj Hattı Dengelem Problemlerine Genetik Algoritma ile Çözüm Yaklaşımı*. Yüksek Lisans Tezi. İ.T.Ü. Fen Bilimleri Enstitüsü, İstanbul.

Palko S (1996). *Structural Optimization of Induction Motor using a Genetic Algorithm and a Finite Element Method*. Acta Polytechnica Scandinavvica, Electrical Engineering Series No. 4, Helsinki

Parlak M (2007). *Genetik Algoritmaların Hesapsal Ve Yapısal Olarak İncelenmesi.* Yüksek Lisans Tezi. On dokuz Mayıs Üniversitesi. Fen Bilimleri Enstitüsü, Samsun.

Pierre D A (1992). *Optimization*.in McGraw-Hill Encyclopedia of Science and Technology 12, New York: McGraw-Hill

Radcliff N J (1991). *Forma analysis and random respectful recombination*. In Proc. of Fourth International Conference on Genetic Algorithms, San Mateo, CA: Morgan Kauffman.

Reeves C R ve Rowe J E(2002). *Genetic Algorithms: Principles and Perspectives: A Guide to GA Theory*, Kluwer Academic Publishers, Norwell, MA, 1-63.

Terzi Ü (2009). *Gezgin Satıcı Problemi İçin Diferansiyel Gelişim Algoritması Tabanlı Bir Metasezgisel Önerisi.* Doktora Tezi. Kocaeli Üniversitesi. Fen Bilimleri Enstitüsü, Kocaeli.

Söke A (2003). *Genetik Algoritma ve Benzetilmiş Tavlama ile İki Boyutlu Giyotinsiz Kesme Problemlerine Olasılıksal Yaklaşım*. Yüksek Lisans Tezi. Kocaeli Üniversitesi. Fen Bilimleri, Enstitüsü Kocaeli.

Şen Z (2004). *Genetik Algoritmalar ve En İyileme Yöntemleri*. Su Vakfı Yayınları, İstanbul.

Tabak Ö (2008). *Genetik Algoritma ile Kapasiteli Servis Güzergahı Belirlenmesi ve Bir Uygulama.* Yüksek Lisans Tezi. Anadolu Üniversitesi. Fen Bilimleri Enstitüsü, Eskişehir.

Wright A (1991). *Genetic Algorithms For Real Parameter Optimization* in G.J.E. Rawlins (Ed.), Foundations of Genetic Algorithms 2, San Mateo, CA: Morgan Kaufman

Wurtz F, Richomme M, Bigeon J ve Sabonnadiere J C(1997). *A Few Results For Using Genetic Algorithms İn Design Of Electrical Machines*. IEEE Transactions on Magnetics, Vol.33

İNTERNET

TSP History.(2011). Erişim Tarihi:28 Aralık 2011,

http://www.tsp.gatech.edu/history/milestone.html

http://www.tsp.gatech.edu/history/milestone.html

İstanbul Elektrik Tramvay ve Tünel [İETT].(2011). Erişim Tarihi:25.Aralık 2011,

http://www.iett.gov.tr/metin.php?no=190

8. ÖZGEÇMİŞ

İsmail KAYA, 1982 yılında Siverek'te doğdu. 2000 yılında Diyarbakır Cumhuriyet Fen Lisesi'nden, 2007 yılında da Sakarya Üniversitesi Bilgisayar Mühendisliği Bölümünden mezun olmuştur. 2009 yılından beri Haliç Üniversitesinde yüksek lisans eğitimini sürdürmektedir. Çalışma hayatına 2007 yılında 4T bilişim firmasında yazılımcı olarak başladı. 2008 yılında İETT'de yazılımcı olarak çalışmaya başladı ve halen burada çalışmaya devam etmektedir.