Genetik Algoritma Kullanılarak Noktadan Noktaya Yol ve Rota Planlama

Bülent SİYAH
Mustafa Kemal Üniversitesi Mühendislik Fakültesi
Bilgisayar Mühendisliği Bölümü,
Telefon: (537) 3453878 , e-posta:bulentsyh@gmail.com

1.GİRİŞ

Bu çalışmada bir kroki üzerinde bulunan noktalar arası seyahati genetik algoritma kullanarak optimum yol ve optimum yola alternatif yollar bulunması amaçlanmıştır.

Genetik algoritma rastgele arama metodu olduğu için tek bir çözüm aramak yerine bir çözüm kümesi üzerinde çalışır. Optimum çözüme olası çözümlerin bir bölümü üzerinde gidilir.Böylece çalışmadaki sonuçlar her zaman en iyi olmaz. Çalışmada genetik algoritmanın kullanılmasının nedeni,genetik algoritmanın problemin doğasıyla ilgili herhangi bir bilgiye ihtiyaç duymamasıdır.

Temelinde gezgin satıcı problemine benzeyen çalışmanın , bu probleme benzer problemler içinde çözüm olması amaçlanmıştır.

2. MATERYAL VE YÖNTEM

2.1.Genetik Algoritma

Genetik Algoritmalar,daha iyi çözümlere yavaş yavaş bir yaklaşım sağlayan geniş bir problem uzayı boyunca yönlendirilmiş rastgele bir araştırmaya imkan sağlar.Temel avantajı optimize edilmeye çalışılan problemin doğasıyla ilgili herhangi bir bilgiye ihtiyaç göstermemeleridir.

2.1.1.Genetik Algoritmaların Temel Kavramları

Evrimsel hesaplama topluluğunda yerleşik, tek bir Genetik Algoritma tanımı bulunmamakla beraber, Genetik Algoritma sınıfına girdiği kabul edilen yöntemlerde en azından şu ortak unsurlara rastlanır: kromozom topluluğu, uygunluğa göre seçilim, yeni bireyler üretmek için çaprazlama, yeni bireylerin rastgele mutasyonu. Holland'ın GA'nın unsurları içinde saydığı ters çevirme operatörü günümüzde nadiren kullanılmaktadır.

2.1.1.1.Kromozom ve Topluluk

GA'da kromozomlar, problem için olası çözümleri temsil ederler. Bu çözümlerden her birine birey adı verilir. Topluluk (popülasyon) kromozomlardan (bireylerden) oluşan kümedir. GA'nın kromozom toplulukları üzerinde işlemler yapması ile, bir önceki topluluklar her seferinde yeni üretilen topluluklarla yer değiştirir.

2.1.1.2.Uygunluk

Uygunluk değeri, çözümün kalitesini belirler ve uygunluk fonksiyonu kullanılarak hesaplanır. Uygunluk fonksiyonu mevcut topluluktaki her kromozoma bir puan (uygunluk değeri) atar. Kromozomun uygunluğu, o kromozomun eldeki problemi ne kadar iyi çözdüğüyle ilişkilidir.

2.1.1.3.Genetik Operatörler

GA'nın en basit formu üç tip genetik operatör içerir: seçilim, çaprazlama (tek noktalı) ve mutasyon.
Seçilim: Bu operatör topluluktaki kromozomları üreme için seçer. Kromozomun yüksek uygunluk değerine sahip olması, üremek için seçilmesi ihtimalini arttırır.
Çaprazlama: Bu operatör rastgele bir locus belirleyip, iki kromozomun bu locustan önceki ve sonraki kısımlarını değiş tokuş ederek iki yeni birey yaratılmasını sağlar.
Mutasyon: Bu operatör kromozomdaki bazı bitleri rastgele olarak ters çevirir.

2.1.2.Genetik Algoritma Süreci

Çözülmek üzere tanımlı bir problem verilmesi ve aday çözümlerin birer bit dizisi ile temsil edilmesi halinde, GA şu şekilde çalışır:

1. N adet kromozom (problem için aday çözümler) içeren rastgele oluşturulmuş bir topluluk ile başla.
2. Topluluktaki her x kromozomu için f(x) uygunluk değerini hesapla.
3. Aşağıdaki adımları birey n (topluluk büyüklüğü) oluşturuluncaya kadar tekrarla.

a. Güncel topluluktan, yüksek uygunluk değerinin seçilme ihtimalini arttırdığını göz önünde bulundurarak, iki ebeveyn kromozom seç. Seçilim, aynı kromozomun birden çok defa ebeveyn olarak seçilmesine olanak verecek şekilde yapılır.

b. Pc olasılığı ("çaprazlama olasılığı" ya da "çaprazlama oranı") ile seçilen çifti iki yeni birey oluşturmak üzere rastgele belirlenen bir noktadan çaprazla. Eğer çaprazlama gerçekleşmezse ebeveynlerinin birebir kopyası olan iki çocuk oluştur.

c. Pm olasılığı ("mutasyon olasılığı" ya da "mutasyon oranı") ile, oluşan iki çocuğu tüm veya bazı locuslarında mutasyona uğrat.

d. Sonuçta elde edilen kromozomları yeni topluluğa ekle.

4. Önceki topluluğu yeni topluluk ile değiştir.
5. Sonlandırma koşulu sağlandıysa mevcut topluluktaki en iyi çözümü döndür, sağlanmadıysa 2. adıma dön.

2.1.2.1.Kromozomların Kodlanması

Bir problemin çözümünde Genetik Algoritmalar kullanılacaksa çözümün ilk adımı kromozomların nasıl kodlanacağına karar vermektir. Kodlama yaklaşımı çözümün başarısına doğrudan etki eder ve problemin türüne ve özelliklerine göre farklılık gösterir.

2.1.2.2.Seçilim

GA'da bir sonraki nesle geçiş sırasında, yeni topluluğu oluşturmak için mevcut topluluktan çaprazlama ve mutasyon işlemlerine tabi tutulacak bireylerin seçilmesi gerekir. Teoriye göre iyi olan, bir başka deyişle uygunluk değeri yüksek olan bireyler yaşamını sürdürmeli ve bu bireylerden yeni bireyler oluşturulmalıdır. Bu nedenle tüm seçilim yöntemlerinde uygunluk değeri fazla olan bireylerin seçilme olasılığı daha yüksek tutulur. Seçilim sırasında dengenin gözetilmesi gerekir. Çok güçlü bir seçilim, o seviyede yüksek uygunluğa sahip bireylerin topluluğu kaplamasını ve çözüme ulaşmak için gereken çeşitliliğin azalmasını beraberinde getirir. Çok zayıf bir seçilim ise aşırı yavaş bir evrime neden olur.

2.1.2.3.Çaprazlama

GA'da çaprazlama işlemi, iyi çözümlerin farklı bölümlerini birleştirip daha iyi çözümler oluşturabilmek amacıyla kullanılır. Çaprazlamanın en kolay yolu rastgele bir çaprazlama noktası belirleyip, bu noktadan önceki bölümü ilk ebeveynden, sonraki bölümü ise diğer ebeveynden alarak yeni bir birey oluşturmaktır.

Tek Noktalı Çaprazlama: En temel çaprazlama yöntemidir. Rastgele bir çaprazlama noktası seçilir ve iki ebeveynin o noktadan sonra gelen kısımları değiş tokuş edilir.

2.1.2.4.Mutasyon

Bireyin bir sonraki nesle geçirilmesi sırasında kromozomu olu turan karakter dizisinde yapılan rastgele değişikliğe mutasyon denir. Mutasyon, oluşan yeni çözümlerin önceki çözümü kopyalamasını önleyerek çeşitliliği sağlamak ve sonuca daha hızlı ulaşmak amacıyla gerçekleştirilir. Mutasyon olasılığı çok düşük (%0.01 gibi) tutulmalıdır. Yüksek mutasyon olasılığı uygun çözümleri de bozacak ve GA'nın çalışma sırasında problemlerle karşılaşılmasına yol açacaktır.

2.1.2.5.Seçkincilik (elitizim)

Seçilim, çaprazlama ve mutasyon işlemleri sonrasında mevcut topluluğun en iyi uygunluk değerine sahip bireyi bir sonraki nesle aktarılamayabilir. Bunu önlemek için bu işlemlerden sonra, bir önceki topluluğun en iyi (elit) bir veya daha çok bireyi, yeni oluşturulan topluluğa doğrudan aktarılır.

2.1.3.Genetik Algoritma Parametreleri

2.1.3.1.Topluluk Büyüklüğü

Topluluk büyüklüğü için seçilen değer, algoritmanın performansını iki şekilde etkilemektedir. Birincisi, topluluk büyüklüğünün aşırı küçülmesi araştırma uzayının yetersiz örneklenmesine sebep olacağından kontrollü ıraksamayı sağlamak zorlaşacak ve araştırma belirli bir alt optimal noktaya doğru sürüklenecektir. ikincisi, topluluk için aşırı yüksek değer seçildiğinde bir nesillik gelişim oldukça uzun süreye ihtiyaç duymaktadır.

2.1.3.2.Çaprazlama Oranı

Bireylerin çoğalması sırasında kromozomlara uygulanacak çaprazlama operatörünün frekansını belirlemek amacıyla kullanılan parametredir. Düşük çaprazlama oranı yeni nesle çok az sayıda yeni yapının (ebeveyninden farklı yeni bireyler) girmesine sebep olmaktadır. Dolayısıyla tekrar üreme operatörü algoritmada aşırı etkili bir operatör haline gelmekte ve araştırmanın yakınsama hızı düşmektedir. Yüksek çaprazlama oranı araştırma uzayının çok hızlı bir şekilde araştırılmasına sebep olmaktadır. Ama oran aşırı yüksek ise, çaprazlama operatörü benzer veya daha iyi yapıları üretmeden kuvvetli olan yapılar çok hızlı olarak bozulduğundan, algoritmanın performansı düşmektedir.

2.1.3.3.Mutasyon Oranı

Mutasyon operasyonun frekansı, etkili bir genetik algoritma tasarlamak için çok iyi kontrol edilmelidir. Mutasyon operasyonu, araştırma sahasına yeni bölgelerin girmesini sağlar. Yüksek mutasyon oranı,araştırmaya aşırı bir rastgelelik kazandıracak ve araştırmayı çok hızlı olarak ıraksatacaktır. Başka bir deyişle, topluluğun gelişmesine değil tahribatına sebep olacaktır. Bu durumun tersine, çok düşük mutasyon oranının kullanılması ıraksamayı aşırı düşürecek ve araştırma uzayının tamamen araştırılmasını engelleyecektir. Dolayısıyla, algoritmanın alt optimal çözüm bulmasına sebep olacaktır.

2.2.YÖNTEMİN PROBLEME UYGULANIŞI

Problemin çözümü için populasyon büyüklüğü karar verilmeli, populasyon büyüklüğü seçilirken aşırı yüksek seçilirse gelişim yavaşlar, aşırı küçük seçilmesi durumunda da araştırma uzayı yetersiz olucağından problemimize en uygun populasyon büyüklüğü 30 birey olarak seçilmiştir. Programızın arayüzünde birey sayısı değiştirilebilir. Her kromozom 20 genle temsil edilip, genler kodlanırken değer kodlama yöntemi kullanılmıştır. Her gen yönleri temsil eden 0;Batı, 1;Kuzey, 2;Doğu ve 3;Güney ile belirtilmiştir. Örnek olarak 5 nolu kromozomun genleri:01102033021011023221 gibi 20 gene sahiptir.

Populasyon büyüklüğü tamamlanıp, kromozomlar kodlandıktan sonra uygunluklarına göre seçilim yapıldı. Uygunluk değerleri hesaplanırken noktalar arası seyahat olduğu için kromozomun en son noktası ile ulaşmak istenen arasındaki farka bakılır. Bu yüzden uygunluk fonksiyonumuz 2 fonksiyonun toplamına eşittir. Birinci fonksiyonumuz en son nokta uzaklığı f(y) , ikinci fonksiyonumuz ise kromozomun aldığı yolların toplamı olan toplam mesafe f(z) fonksiyonudur. Uygunluk fonksiyonumuz f(x)=5*f(y)+ f(z) olarak belirlenmiştir. Burada son nokta uzaklığını fonksiyonu problem için daha önemli olduğundan katsayısı artılmıştır. Örneğin 6 nolu kromozom son noktası 21 , ulaşılmak istenen nokta 22 ise f(y)=20, ve aldığı yolların mesafesini ölçen f(z)=550 olduğunu düşünelim ,6 nolu kromozomun uygunluk değeri f(x)=5*20+550=650 olur. Kromozom ulaşılmak istenilen noktaya varmış olsaydı bu değer 550 olacaktı. Burada görüldüğü gibi son noktaya ulaşmak uygunluk değerini hesabının doğruluğunu kanıtlıyor.

Seçilimde ikili turnuva seleksiyonu kullanıldı. İkili turnuva seçiminde populasyon içinden rastgele iki birey seçilir ve uygunlarına göre iyi olan alınır, daha sonra tekrar rastgele iki birey seçilir ve yine uygunluklarına göre en iyi olan alınır. Böylece elde olan iki tane birey çaprazlanarak yeni topluma katılır. Çaprazlama yapılırken rastgele bir lopus seçilir ve iki kromozom o lopustan değiştirilir. Örnek rastgele 2 ve 16 nolu bireyler seçildi bunlardan uygunluğu en iyi olan 2, tekrar rastgele iki birey seçildi 11 ve 7, bunlara arasındada uygun olan 7 , bu kazanan 2 ve 7 bireyi rastgele bir noktadan değişirme hazırlar. Rastgele noktamızın 9 olduğunu düşünürsek her kromozom 20 gen olduğu için ilk 9 gen 2 nolu kromozomdan geriye kalan 11 gen 7 nolu kromozomdan alınır(2 ile 7 ikili turnuva sonucunda eşleşen kromozomlardı). Aynı işlem 7nin ilk 9 geni alınır geriye kalan 11 gende 2den alınır. Turnuva metodun da seçilen birey tekrar sisteme dahil edilir yani çaprazlamaya uğrayan birey tekrar çaprazlamaya katılabilir.

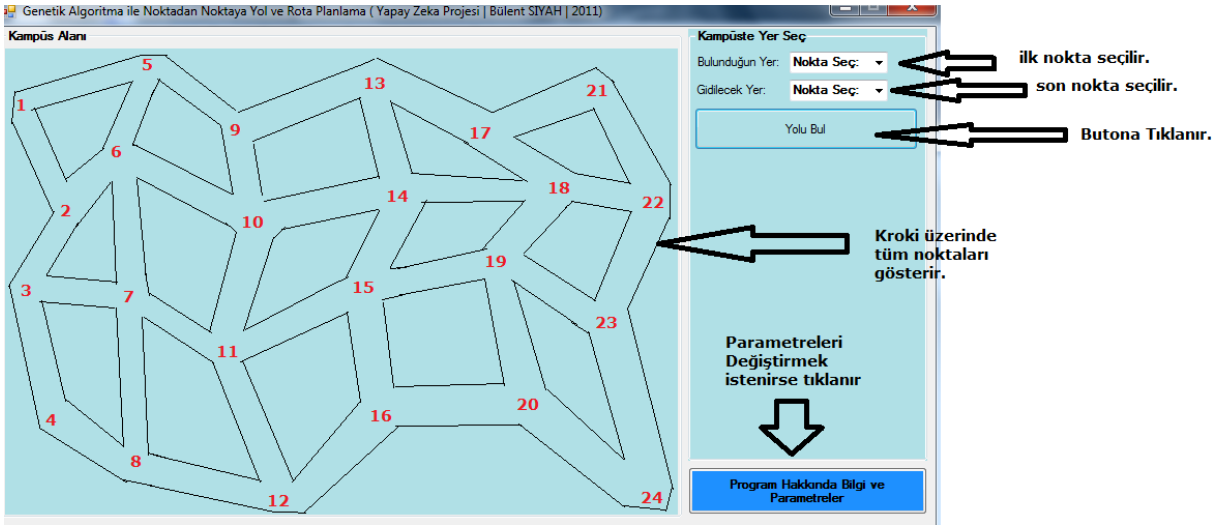
Populasyondaki en iyi birkaç birey doğrudan yeni topluma aktarıldı. Geriye kalan bireyler çaprazlama ile yaratıldı. Çaprazlamaya ugrayan bireylerden biri mutasyona uğratıldı. Mutasyon yapılmasının nedeni önceki çözümlerin kopyalanmasını önlemek. Mutasyon için rastgele bir gen seçilir ve değiştirilir.

Böylece yeni populasyon oluşturuldu. Problemin çözümü için belirelenen iterasyon sayısı kadar döngü devam eder. Döngü sonlanınca problemin en ugun çözümü elde edilmiş olur. Böylece optimum değer elde edilmiş oldu fakat çalışmanın diğer amacı olan alternatif yollar üretmek için son nesilden önceki nesillerin tümünden en iyi olanlarda seçilip , seçilen yollar arasından birbirinden farklı olan diger yollarda projede tutularak arayüzde yansıtıldı.

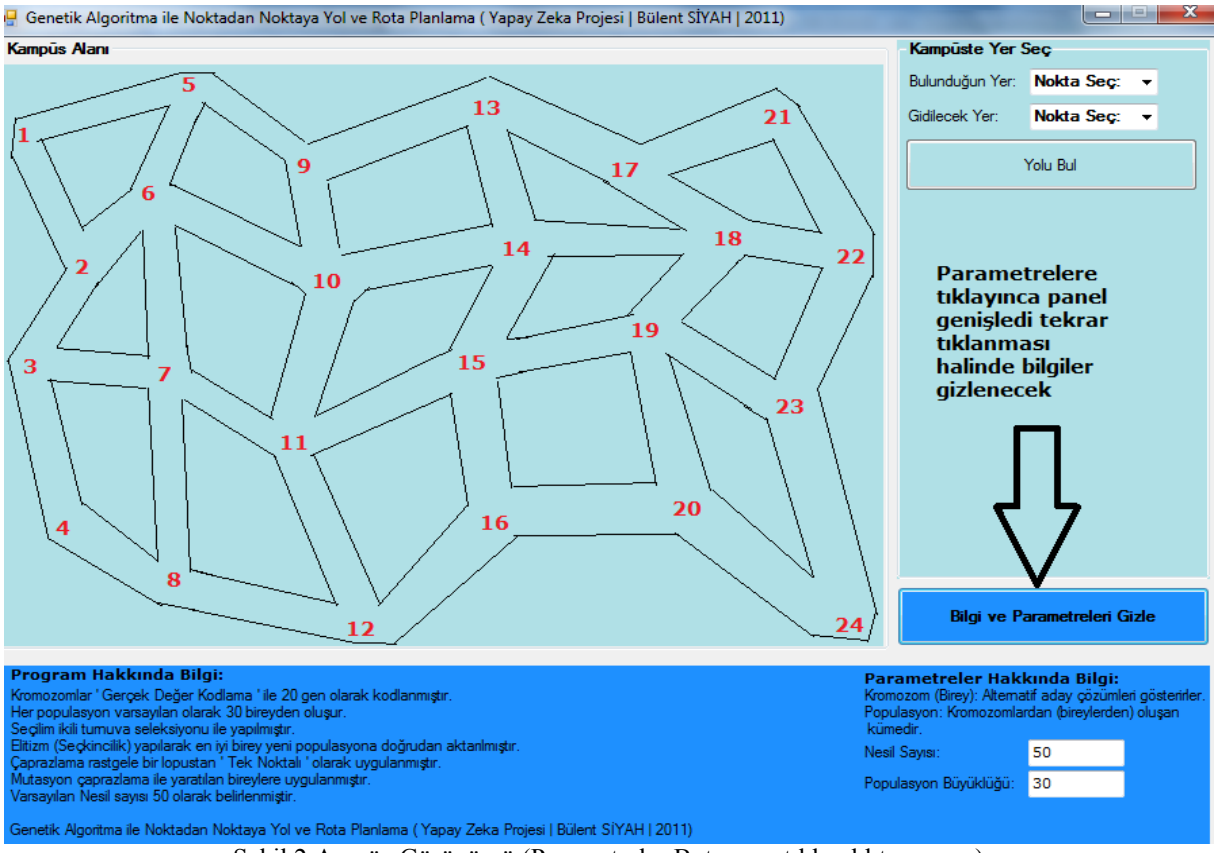
3. PERFORMANS ANALİZİ

Problemin çözümünde elde edilen bazı veriler;

Başlangıç noktasından bitiş noktasına gitmeye çalışınca elde edilen veriler grafikte gösterilmiştir. Y ekseni Uzaklık piksel , x ekseni nesil sayısnı gösterir.



Şekil 1.Arayüz Görünümü

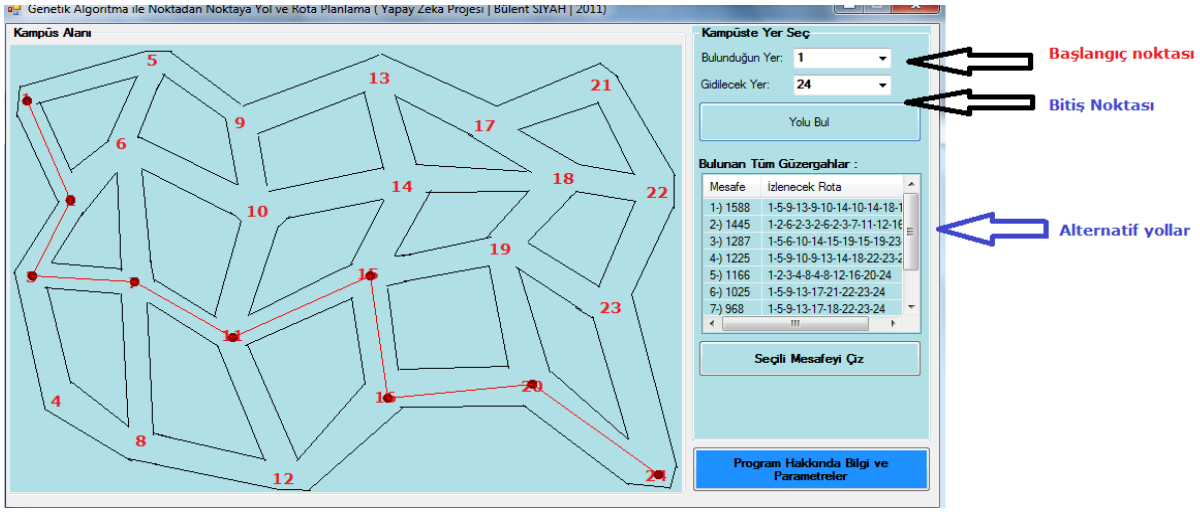


Şekil 2.Arayüz Görünümü (Parametreler Butonuna tıklandıktan sonra)

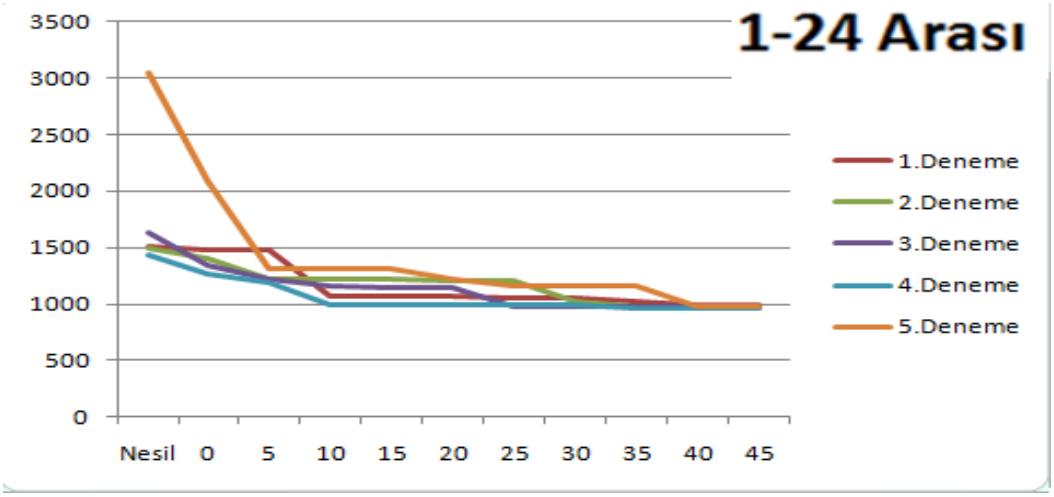
Noktaların x ve y bileşenleri: dizinin ilk elemanı noktanın kendisini,ikinci elemanı x veya y eksenindeki değerini tutuyor.

NoktaKoordinatlari[1, 0] = 16; // 0 ‘lar x eksenindeki yerini temsil ediyor.
NoktaKoordinatlari[1, 1] = 56; // 1 ‘lar y eksenindeki yerini temsil ediyor.
NoktaKoordinatlari[2, 0] = 58; NoktaKoordinatlari[2, 1] = 156; NoktaKoordinatlari[3, 0] = 21;
NoktaKoordinatlari[3, 1] = 232; NoktaKoordinatlari[4, 0] = 44; NoktaKoordinatlari[4, 1] = 356;
NoktaKoordinatlari[5, 0] = 137; NoktaKoordinatlari[5, 1] = 13; NoktaKoordinatlari[6, 0] = 108;
NoktaKoordinatlari[6, 1] = 100; NoktaKoordinatlari[7, 0] = 120; NoktaKoordinatlari[7, 1] = 238;
NoktaKoordinatlari[8, 0] = 127; NoktaKoordinatlari[8, 1] = 400; NoktaKoordinatlari[9, 0] = 222;
NoktaKoordinatlari[9, 1] = 76; NoktaKoordinatlari[10, 0] = 241; NoktaKoordinatlari[10, 1] = 171;
NoktaKoordinatlari[11, 0] = 215; NoktaKoordinatlari[11, 1] = 294; NoktaKoordinatlari[12, 0] = 265;
NoktaKoordinatlari[12, 1] = 437; NoktaKoordinatlari[13, 0] = 358; NoktaKoordinatlari[13, 1] = 34;
NoktaKoordinatlari[14, 0] = 380; NoktaKoordinatlari[14, 1] = 144; NoktaKoordinatlari[15, 0] = 348;
NoktaKoordinatlari[15, 1] = 232; NoktaKoordinatlari[16, 0] = 365; NoktaKoordinatlari[16, 1] = 355;
NoktaKoordinatlari[17, 0] = 460; NoktaKoordinatlari[17, 1] = 81; NoktaKoordinatlari[18, 0] = 536;
NoktaKoordinatlari[18, 1] = 135; NoktaKoordinatlari[19, 0] = 473; NoktaKoordinatlari[19, 1] = 206;
NoktaKoordinatlari[20, 0] = 505; NoktaKoordinatlari[20, 1] = 341; NoktaKoordinatlari[21, 0] = 573;
NoktaKoordinatlari[21, 1] = 41; NoktaKoordinatlari[22, 0] = 626; NoktaKoordinatlari[22, 1] = 150;
NoktaKoordinatlari[23, 0] = 581; NoktaKoordinatlari[23, 1] = 265; NoktaKoordinatlari[24, 0] = 627;
NoktaKoordinatlari[24, 1] = 432;

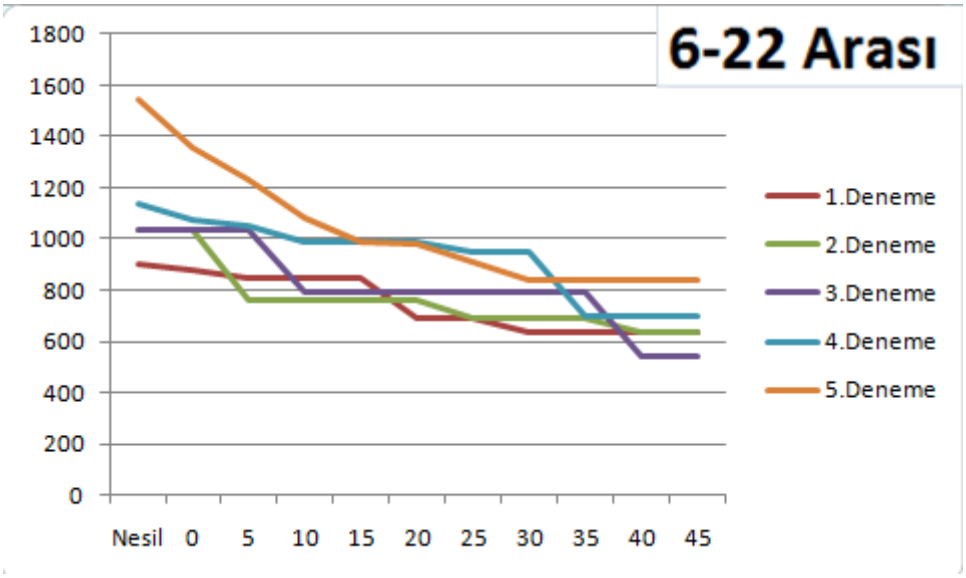
1 ile 24 arasındaki elde edilen veriler aşağıda listelenmiştir. 50 nesil sayısı ile sonuçlar gözlenmiştir.



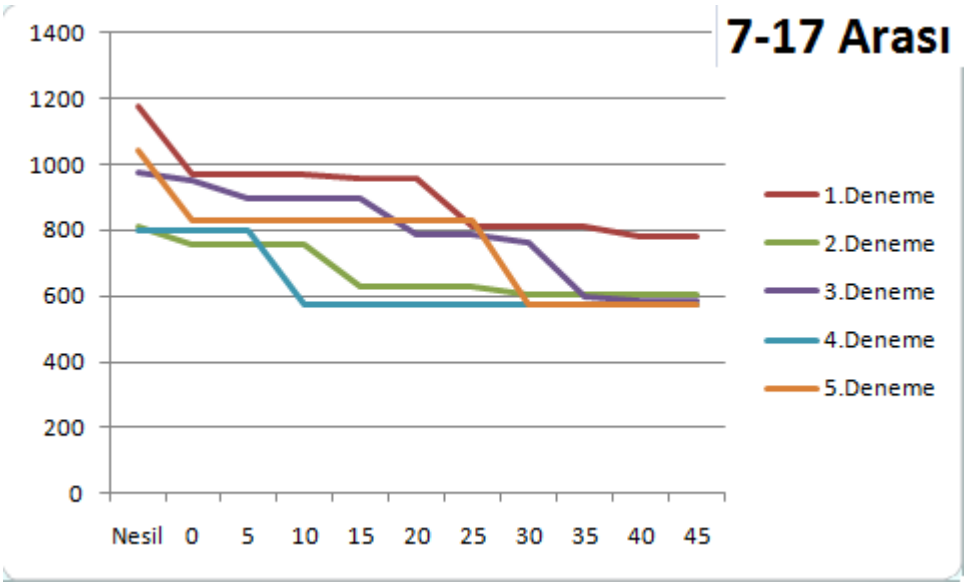
Şekil 2. 1-24 arası nesil sayısı=50 Programın Arayüzü



Şekil 3.1-24 arası elde edilen veriler.

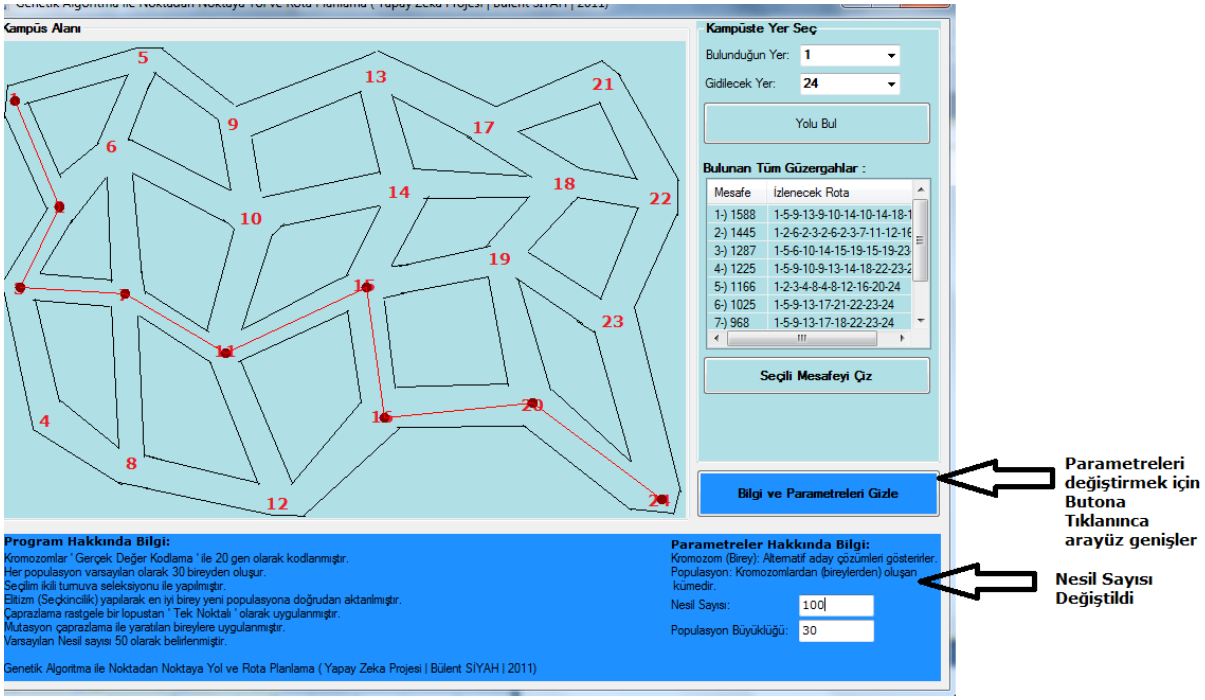


Şekil 4.6-22 arası elde edilen veriler.



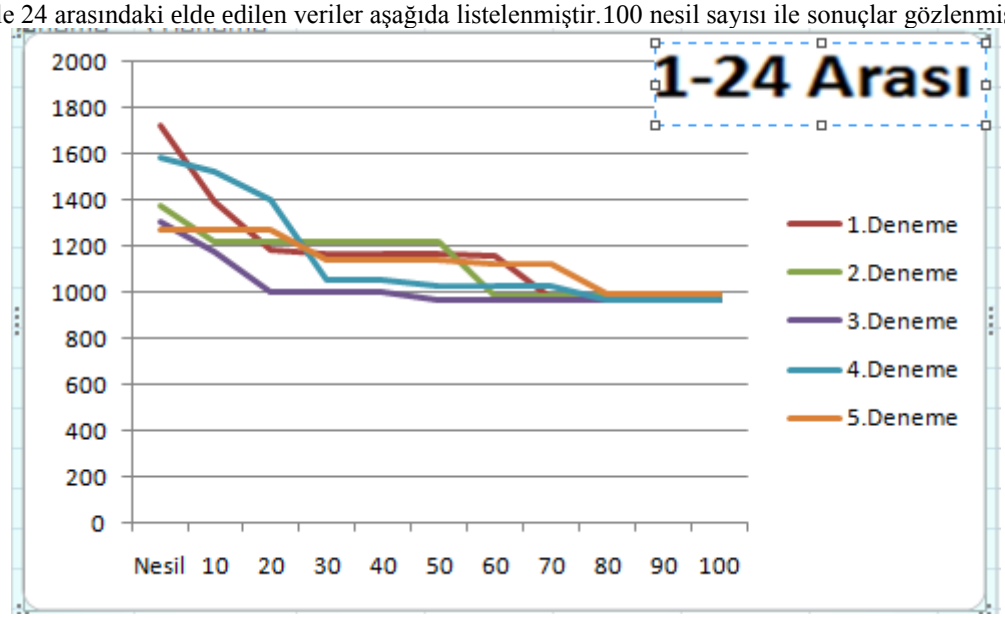
Şekil 5.7-17 arası elde edilen veriler.

Nesil sayısı değiştirmek için arayüzde şöyle değiştirilir.

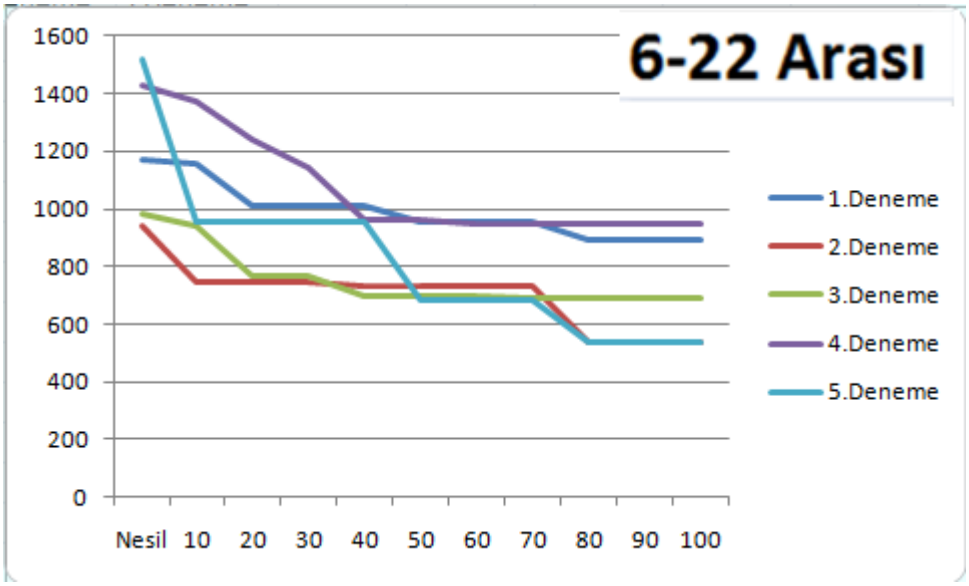


Şekil 6.Nesil sayısı değiştirilişi.

1 ile 24 arasındaki elde edilen veriler aşağıda listelenmiştir.100 nesil sayısı ile sonuçlar gözlenmiştir.



Şekil 7. 1-24 arası elde edilen veriler.Nesil sayısı 100.



Şekil 8. 6-22 arası elde edilen veriler.Nesil sayısı 100.

4.SONUÇLAR VE ÖNERİLER

Elde edilen tüm sonuçlara göre bir problemin genetik algoritma uygulanmasıda uygun parametreler seçilmediği taktirde en uygun çözümden uzaklaşır. Çalışmada populasyon büyüklüğü ayarlanırken varsayılan değer 30 ile 100 değeri arasında gözlemlerim populasyon büyüdükçe çözüm olması muhtemel bireylerin çoğalması ve çözüm uzayının genişlemesiyle en iyiye daha da yaklaşılmıştır.

Uygunluk fonksiyonu oluşturulurken çalışma için önemli noktalar belirlenmemesi halinde uygun olmayan bireyler uygun sanılıp problemin çözümünden uzaklaşılır. Bu çalışmada ugunluk fonksiyonu 2 tane fonksiyonun toplamına eşit. Çünkü en kısa yol bulunmasının dışında önemli olan parametre kromozomun istenilen noktaya ulaşıp ulaşmadığıdır. Yönetimin probleme uygulanışında bir örnek ile bu durumun anlatılmaya çalışıldı.

Seçilim için ikili turnuva seçilimi haricinde rulet tekeli seçilimi çalışmada kullanılmamasının sebebi rulet tekeri yöntemi ile iyi olan birey nesiller sonra kendisiyle aynı bireyler üreterek çözüm uzayını o nokta etrafında toplamasıdır. Bu durum çalışmada daha iyi olması muhtemel çözümlerin araştırılmasını engellediğinden kullanılmamıştır.

Çaprazlamada ise çaprazlama oranının probleme uygun ayarlanmaması halinde çalışmada hatalar gözlendi. Örneğin en iyi bireylerin yok olması istenilen bir durum değildi ve tüm bireyler çaprazlama ile oluşturulması durumunda bu hata açığa çıktı. Bu durumu engellemek için elitizm yapılarak en iyi bireyler doğrudan yeni topluma aktarılarak elenmeleri engellenmiş oldu. Çaprazlama sayesinde çözüm uzayı genişledi. En iyi bireyler doğrudan yeni topluma aktarılmasına rağmen ikili turnuva yönteminde kullanılan kümeden çıkarılmadılar. Çaprazla için seçilen bireyler de ikili turnava yöntemi için bulunan kümesinden çıkarılmadı.

Mutasyon çözüm yoğunluğunu dağıttı gözlendi fakat mutasyon oranı fazla tutulması halinde en uygun değerden uzaklaşıldığı gözlendi. Bu yüzden çalışmamızda mutasyon oranı 0.01 olarak tutulmasına karar verildi.

Projenin geliştirlmesi halinde bir bina içerisinde özellikle çok sayıda geçiş bulunduğu karmaşık binalarda istenilen yerler arasında en uygun şekilde yol ve rota bulunabilir. Çalışmanın ileriki aşamalarında program arayüzüne kullanıcı tarafından kroki,harita veya plan yüklenmesi sağlanabilir. Gezgin satıcı problemine benzer problemler için de kullanılabilir. Örneğin bilgisayar ağlarında olası bir kopmada diğer en uygun yollar bulunabilir.

5.KAYNAKLAR

1. Utku Cevre, Çoklu gezgin satıcı probleminin çözümü için bir eniyileme kütüphanesinin tasarımı ve görsel yazılım geliştirme ortamı ile birlikte gerçekleştirimi, 2008
2. Aytekin Bağış,Genetik Algoritma kullanılarak Ders Programının Optimum Şekilde Hazırlanması,1996
3. Mustafa Kaya, Genetik Algoritma ve Gezgin Satıcı Probleminin Çözümü,1999
4. Barış Özkan, Dinamik Gezgin Satıcı Probleminin Çözümü için bir eniyileme kütüphanesinin tasarımı ve görsel yazılım geliştirme ortamı ile birlikte gerçekleştirimi 2008