Gazi Üniv. Müh. Mim. Fak. Der.
Cilt 22, No 2, 431-439, 2007

J. Fac. Eng. Arch. Gazi Univ.
Vol 22, No 2, 431-439, 2007

GENETİK ALGORİTMA İLE WEB SAYFASI DÜZENİNİN GERÇEK ZAMANLI OPTİMİZASYONU

Yaşar GÖZÜDELİ ve M.Ali AKCAYOL*
Bilgisayar Mühendisliği Bölümü, Mühendislik Mimarlık Fakültesi, Gazi Üniversitesi, 06570, Maltepe, Ankara
ygozudeli@verivizyon.com, *akcayol@gazi.edu.tr

(Geliş/Received: 08.05.2006; Kabul/Accepted: 04.06.2006)

ÖZET

Günümüzde, haber ve makale sunan web sitelerinden ürün satışı yapan e-ticaret sitelerine kadar bir çok web sitesi çok büyük boyutlarda içerik sunmaktadır. Çok fazla bilgi sunan siteler için en büyük problem, kullanıcının rahat okuyabileceği biçimde ve en az yer kaplayacak şekilde dinamik olarak web sayfasının oluşturulmasıdır. Ancak kullanıcının isteği ve tercihleri doğrultusunda gerçek zamanlı olarak şekillenmesi gereken bu sayfaların statik olarak ve elle yapılabilmesi mümkün değildir. Gerçek zamanlı olarak oluşturulan bu tür sayfalarda istenmeyen boşluklar ve sayfa düzenleri ortaya çıkabilmektedir. Bu çalışmada, genetik algoritma kullanılarak bir sayfada bulunan bilgilerin iki sütun halinde en az yer kaplayacak şekilde gerçek zamanlı olarak oluşturulması gerçekleştirilmiştir. Böylece makalelerin sayfaya yerleştirilmeleri güncellikleride gözönüne alınarak düzenlenmiş ve sayfadaki toplam satır sayısında belirgin bir azalma sağlanmıştır. Yapılan deneysel çalışmalar genetik algoritmanın gerçek zamanlı sayfa optimizasyonunda başarılı olduğunu göstermiştir.

Anahtar Kelimeler: Genetik algoritma, dinamik web içerik yerleşimi, gerçek zamanlı optimizasyon.

REAL-TIME OPTIMIZATION OF WEB PAGE LAYOUT USING GENETIC ALGORITHM

ABSTRACT

Today, many web sites from which presents newspapers and papers to e-commerce web sites which sells products presents very large contents. It is a big problem for web sites with large information that web site should be prepared dynamically and users should read it conveniently and also it should require minimum page size. It is not possible to prepare these pages being created in real-time depending on users' choices and prefers by means of manual. In this kind of web sites there may be bad page layout and spaces. In this study, it has been realized that web pages are created in real-time with two columns and least spaces. So that, papers in the web page are arranged depending on expire time and total lines are significantly decreased. The experimental results have showed that genetic algorithm is successful in optimization of real-time web page layout.

Keywords: Genetic algorithm, dynamic web context layout, real-time optimization.

1. GİRİŞ (INTRODUCTION)

Optimizasyon, hemen her alanda kullanılan bir kavram olup, kazancı maksimum ve maliyeti minimum yapmayı amaçlamaktadır. Optimizasyon için bir çok yöntem kullanılabilir. Eğer kullanılan yöntem, belirli bir probleme her uygulanışında aynı sonucu veriyorsa, bu tür yöntemlere deterministik yöntemler denir. Deterministik yöntemler genellikle en iyi bir tek çözümü bulmak için kullanılırlar. Deterministik yöntemler bazı problemlerin çözümü için mümkün olmayabilmekte veya çok zaman ve maliyet gerektirebilmektedir. Bu tür problemlerde deterministik olmayan yöntemlerin kullanımı başarılı olabilir. Deterministik olmayan yaklaşımlar, aynı başlangıç durumu için farklı sonuçlar verebilmekte, ancak en iyi olmasada kabul edilebilir bir sonuca belirli bir süre sonunda ulaşabilmektedirler. Meta-sezgisel yöntemler deterministik olmayan yöntemlerdir ve en iyi çözümü garanti etmemekle birlikte, denenmesi gereken olasılıkların çok fazla

olduğu durumlarda daha az deneme ile kabul edilebilir seviyede iyi bir çözüm bulmaktadırlar.

Bu makalede ele alınan problem, gerçek zamanlı dinamik olarak oluşturulan bir web sayfasında, boş alanları en aza indirerek daha fazla makalenin daha az alanı kaplayacak şekilde ve daha güncel olanları olabildiğince başa alarak yerleşimini sağlamaktır. Bu nedenle her ne kadar ele alınan problem yerleşim tasarımı gibi görünse de her bir içerik dikdörtgen şeklinde bir alan kaplayacağından, problem temelde dikdörtgensel şerit paketleme'nin (strip packing) uygulama alanıdır. Şerit paketleme, genişliği sabit iki boyutlu sele paketleme (bin packing) 'nin bir özel türüdür. Burada ele alınan problemde, yerleştirme sırasında şerit 90° döndürülerek yerleşim iyileştirmesi yapılamamaktadır. Bu yüzden, problemin çözümü NP (Non-Polynomial) zor bir problem kadar karmaşık ve zaman alıcıdır. Sele paketlemesi ile ilgili ilk çalışmalar Hopper tarafından yapılmıştır[1,2]. Sele paketlemenin genetik algoritma kullanılarak yapıldığı çalışmalarda bulunmaktadır[3,4]. Şerit paketleme alanında ise, Lesh ve arkadaşları tarafından elde edilen BLD(Bottom-Left-Decreasing) adlı çözüm [5,6], benzetilmiş tavlama algoritmasına göre daha iyi sonuç vermiştir. Iori ve arkadaşları tarafından yapılan araştırmada şerit paketlemede meta-sezgisel çalışmalar değerlendirilmiştir[7]. Bortfeldt tarafından ikili şerit paketlemede genetik algoritma kullanımı ele alınmıştır[8]. Hadjiconstantinou ve Iori ise iki boyutlu knapsack probleminde genetik algoritmayı başarıyla kullanmıştır[9]. Elektronik içerik düzeni hakkında Krista ve arkadaşları yapmış oldukları araştırmada benzetilmiş tavlama temelli üç farklı yöntem önermişlerdir[9]. Gonzales ve arkadaşları web sayfalarında içerik yerleşim optimizasyonu için benzetilmiş tavlama algoritmasını kullanmışlardır[10]. Canbek ve Akcayol ise yaptıkları çalışmada, istemci tarafında çalışan benzetilmiş tavlama algoritmasıyla birden fazla sütunu bir makaleye ayırabilen, esnek ve daha görsel sonuçlar elde etmişlerdir[11].

Bu çalışmada, genetik algoritma kullanılarak bir sayfada bulunan bilgilerin iki sütun halinde en az yer kaplayacak şekilde ve daha güncel olanların daha üst kısımda gösterimi gerçekleştirilmiştir. Makalede, web sayfası düzeni optimizasyonu hakkında bilgi Bölüm 2'de verilmiştir. Bölüm 3'te genetik algoritmayla web sayfası düzeni optimizasyonunun gerçekleştirilmesi detaylı bir şekilde anlatılmıştır. Bölüm 4'te gerçekleştirilen uygulamanın sonuçları sunulmuş ve Bölüm 6'da ise yapılan çalışmanın genel sonuçları verilmiştir.

2. WEB SAYFASI DÜZENİ OPTİMİZASYONU (WEB PAGE LAYOUT OPTIMIZATION)

İçerik sağlayan siteler veya e-ticaret siteleri sık sık üyelerine son ürün veya yeni eklenen içerikleri hakkında bilgilendirici özet bültenler oluşturmak zorundadırlar. Bu şekilde içerik özetlerinin yer aldığı bülten veya ürün tanımı katalogların, güncellik, kişiye özgü olarak hazırlanma, bir veritabanına bağlı olarak tek merkezden yönetilebilme gibi gereksinimlerden ötürü dinamik olarak gerçek zamanlı oluşturulmaları gerekmektedir. Canbek ve Akcayol'un araştırmalarına [11] göre en çok ziyaret edilen haber içerikli bazı İnternet sitelerine ait toplam sayfa sayıları Çizelge 1.'de görülmektedir.

Çizelge 1. Bazı haber sitelerinde sunulan toplam çevrimiçi sayfa sayısı (Total online pages presented by some newspaper websites)

Internet Haber Sitesi	Ülke	Sayfa Sayısı
bbc.co.uk	İngiltere	1.270.000
cnn.com	ABD	1.220.000
hurriyetim.com.tr	Türkiye	834.000
usatuday.com	ABD	813.000
timesonline.co.uk	İngiltere	740.000
guradian.co.uk	İngiltere	639.000
washingtonpost.com	ABD	631.000
nytimes.com	ABD	627.000
cbsnews.com	ABD	491.000
independent.co.uk	İngiltere	346.000
sabah.com.tr	Türkiye	298.000
msnbc.com	ABD	298.000
ntvmsnbc.com	Türkiye	86.700
milliyet.com	Türkiye	25.400

İnternet'te içerik yerleşimi sorunu için en basit çözüm, Şekil 1'de görüldüğü gibi her bir içeriği dikey olarak alt alta yerleştirmektir. Ancak bu şekildeki bir yaklaşım, gerek boş yerlerin fazlalığı gerekse okuyucunun daha aşağıda kalan içerikler için sayfayı aşağı doğru hareket ettirme zorunluluğu açısından etkin bir çözüm değildir.

Özet içerik yerleşimi konusunda diğer bir yaklaşım tablo kullanımıdır. Bu tür bir yaklaşımında yanyana gelen makalelelerin yüksekliğinin yaklaşık aynı olması gerekliliği gibi kendine özgü sakıncaları vardır. Tablo kullanılarak yapılan yerleştirmede ise Şekil 2'de görüldüğü gibi boş alanlar oluşmaktadır.

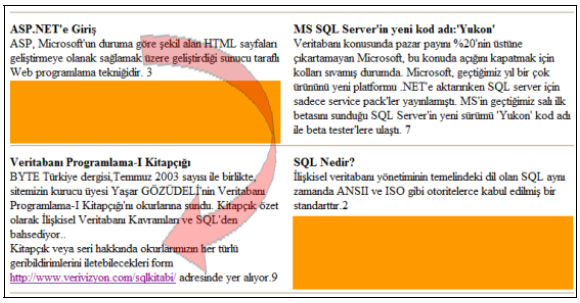
Bu çalışmada, makale özetleri sunan bir web sitesinin daha az satır kullanarak daha fazla makaleyi, rahat okunabilir bir şekilde sunması gerçekleştirilmiştir. Bunun için, birbirine daha yakın yükseklikteki özetleri yan yana koymanın yanı sıra, daha güncel makaleyi daha yukarıda göstermek de amaçlanmıştır. Eşit uzunlukta olmayan makalelerin Şekil 3'teki gibi yan yana gelmesi, makale özet yerleşiminde boşluklara neden olmaktadır. Yakın uzunlukta iki

Şekil 1. Dikey sayfa düzeni (Vertical page layout)

Şekil 2. Tablo ile sayfa düzeni (Page layout with table)

özet seçildiğinde ise, boş olarak geçilmek durumunda kalınan satırların sayısında belirgin bir azalma olduğu Şekil 4'te görülmektedir.

Toplam makale sayısının 20 olduğu bir durumda, 20! farklı olasılıkta yerleşim elde edilmesi mümkündür. Yapılan çalışmada, rastsal bir başlangıçtan hareket ederek en iyi dizilişe yakın kabul edilebilir bir diziliş genetik algoritma kullanılarak elde edilmiştir.



Şekil 3. Tablo ile düzenlemede oluşan boşluklar (Spaces at layout with table)

3. GENETİK ALGORİTMA İLE WEB SAYFASI DÜZENİ OPTİMİZASYONU (WEB PAGE LAYOUT OPTIMIZATION WITH GENETIC ALGORITHM)

Genetik algoritma'da, çözümün her bir parçasına gen denilmektedir. Genetik algoritma, aynı genlerin çeşitli operatörler yardımıyla farklı dizilişlerini sağlayarak farklı bireyler elde etmekte ve bu bireylerden en iyi olanı seçmektedir.

3.1. Gen ve Birey (Gene and Choromosome)

Genetik algoritmayı probleme uygularken, her bir makale özeti bir gen olarak alınmıştır. Yine genetik algoritmada, genlerden her birinden eksiksiz-fazlasız sadece bir tane bulunduran her bir yapıya birey (kromozom) denilmektedir. Buna göre her bir makale özetinin bir defa yer aldığı dizilişlerden her biri bir birey olarak modellenmiştir. 4 genden oluşan örnek duruma göre, Şekil 3'te ve Şekil 4'te iki farklı birey görülmektedir.

3.2. Uygunluk Fonksiyonu (Fitness Function)

Bireyler arasındaki farklılık genlerin farklı sıranmasıyla ortaya çıkmaktadır. Her bir bireyin gen dizilişindeki bu farklılıklarının çözüm üzerindeki başarı oranını gösterecek bir fonksiyona gerek vardır. Bu fonksiyon probleme özgüdür ve modeldeki her bir bireyin çözüme uygunluk değerini hesaplamaktadır. Bu fonksiyona, uygunluk fonksiyonu (fitness function) denmektedir. Bu fonksiyondan elde edilen sonuca da bireyin uygunluk değeri denmektedir. Bu çalışmada, sayfadaki toplam kullanılan satır sayısı ceza puanı olarak, her bir özetin güncelliği de ödül puanı olarak değerlendirilmiştir. Bu çalışmada kullanılan uygunluk fonksiyonu aşağıdaki gibidir;

$$UD = GD - (BSS / BS) \quad (1)$$

Şekil 4. Tablo ile düzenlemede boşlukların giderilmesi (Remove spaces at layout with table)

Burada, *UD* uygunluk değerini, *BSS* makale yerleştirilmeden boş geçilen satır sayısını ve *BS* ise toplam boşluk sayısını göstermektedir. Makalelerin yayında kalma süreleri ile makalenin yayında kalması için kalan süre arasındaki fark hesaplanmış ve bu orana güncellik eğimi adı verilmiştir. Güncellik değeri ise güncellik eğimi kullanılarak aşağıdaki gibi hesaplanmıştır;

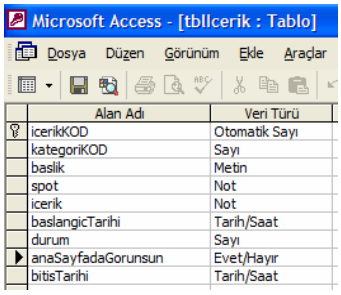
$$GD = (YKT - BT) / (YKT - YGT) \quad (2)$$

Burada, *GD* güncellik değerini, *YKT* yayından kalkma tarihini, *BT* bugünün tarihini ve *YGT* ise yayına giriş tarihini göstermektedir. Bir makalenin güncellik değeri yayınlandığı tarihte 1 değerine sahiptir ve her geçen gün 0'a doğru yaklaşarak, yayından kalkma tarihinde 0 değerine ulaşır. Bu nedenle güncellik değeri her zaman 1 ile 0 aralığındadır. Bir makalenin daha yukarıda görünmesini sağlamak amacıyla, makalelerin güncellik değerleri bulundukları sıra numarası ile çarpılarak ceza puanı hesaplanmıştır. Makalelelerin sıra numaraları bulundukları sütuna göre değerlendirmeye alınmıştır. Bir makalenin her bir kromozom içinde farklı sütun ve sıralarda yer alma olasılığı bulunmaktadır. Makalenin bulunduğu sıra numarasına göre ait olduğu kromozomun uygunluk değerine etkisi olmakta ve bir sonraki popülasyona aktarılma olasılığını değiştirmektedir. Bütün makaleler için güncelliklerine göre genel olarak daha iyi sıra numarasına sahip olan kromozomların seçilme olasılığı daha yüksek olmaktadır. Algoritma sonlandırıldıktan sonra alınan en iyi kromozom tüm makaleler için genelde en iyi sıralamaya sahip olmaktadır. *k* makalelerin kaç sütunda gösterileceğini, *n* makalelerin kaç adet olduğunu göstermek üzere, makalelerin bulundukları sıra numaraları 1 ile *n / k* arasında bir sayıdır. Bu çalışmada *n / k* oranı da her zaman 1'den büyük alınmıştır.

3.3. Popülasyon Oluşturulması (Generate Population)

Birden fazla bireyden oluşan topluluğa populasyon denilmektedir. Literatürde genetik algoritmanın genellikle 100-300 arasında bireyden oluşan popülasyonlarla uygulamaları görülmektedir[12,13, 14,15]. Bu çalışmada problemin çözümünde en iyi birey sayısını bulmak için farklı sayılardaki bireyler için deneysel sonuçlar elde edilmiştir.

Herhangi bir bireyde, genlerin dizilişlerinde bir kurala bağlı olmaksızın, nedensiz ve düşük ihtimallerle meydana gelen değişikliğe mutasyon denmektedir. Mutasyon genetik çeşitliliği artırmaktadır. Bu çalışmada, mutasyon oranı için %0,001 - %0,05 arasında farklı değerler alınarak en iyi değer araştırılmıştır.



Şekil 5. Veritabanı alanları (Database fields)

Çaprazlama işlemiyle, popülasyonda yer alan iki bireyin rastgele bir noktadan bölünüp parçaların karşılıklı değiştirilmesiyle eski iki bireyden farklı iki yeni birey üretilmiştir. Bu işlemin sonucunda, her iki bireyde de tekrarlayan ve dolayısıyla da eksik kalan genler olabileceği için çaprazlamadan sonra bireyler bir onarım işlemine tabi tutulmuşlardır. Böylece kromozomların anlamsal bütünlüğü sağlanmıştır. Yeni bir nesil, eldeki popülasyondan çaprazlama, mutasyon ve elitizm gibi kriter ve yöntemlerle elde edilmiştir. Her bir popülasyonda önceden belirlenen sayıdaki en iyi birey, yeni popülasyona hiç bir işleme tabi tutulmadan, elitizm kriteri çerçevesinde aktarılmıştır. Bu çalışmada elitizm kriteri %2 olarak alınmıştır. Böylece genetik değişimlerle elde edilen iyi bireylerin kaybının önlenmesi sağlanmıştır.

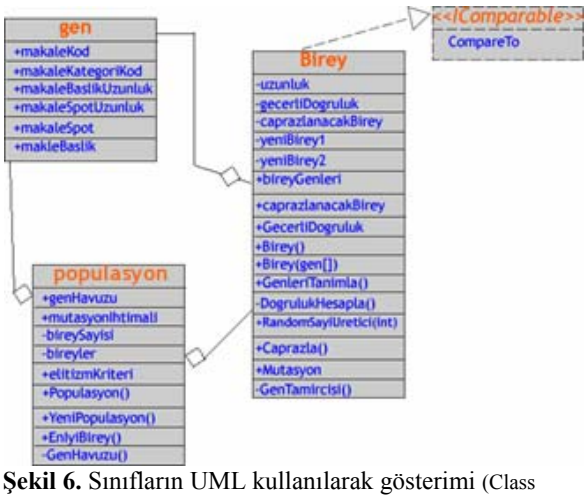
4. GELİŞTİRİLEN YAZILIM (DEVELOPED SOFTWARE)

Bu çalışmada genetik algoritmayla problemin çözümünün gerçekleştirilmesi için Visual C#.NET programlama dili, MS Access ilişkisel veritabanı ve ortam olarak ASP.NET kullanılmıştır.

4.1. Veritabanı Tasarımı (Database Design)

Bu çalışmada MS Access veritabanı kullanılmıştır. Şekil 5'te oluşturulan veritabanına ait alan bilgileri görülmektedir.

icerikKOD alanında her bir makaleye ait tekil (unique) kod saklanmaktadır. *baslik*, her bir makalenin başlık bilgisini tutan alandır. *spot*, her bir makaleye ait özet bilgilerin yer aldığı alandır. *baslangicTarihi*, makalelerin yayına giriş tarihini, *durum* makalenin yayında olup olmadığını, son olarak *bitisTarihi* ise makalenin yayından kalkacağı tarihi göstermektedir.



Şekil 6. Sınıfların UML kullanılarak gösterimi (Class representation using UML)

Diğer alanlar bu çalışmadaki yerleştirme probleminin parçası olmamasına karşın, bir içerik yönetim sisteminde bulunması gerektiğinden dolayı oluşturulmştur. *icerik*, makalelerin içeriğini göstermektedir. *kategoriKOD*, her bir makalenin hangi kategori altında çıkacağını belirlemek için kullanılmıştır. Şekil 6'da bu çalışmada oluşturulan sınıfların UML (Unified Modeling Language) kullanılarak gösterimi sunulmuştur.

4.2. Gen Yapısı (Gene Structure)

Gen bilgileri stack türü bir değişken olan *struct* şeklinde tanımlanmıştır. Şekil 7'de oluşturulan gen yapısına ilişkin kod görülmektedir.

Genler, her bir makaleye ait bilgileri tutmaktadır. Her bir bireyde, sayfaya yerleşecek makale sayısı kadar gen bulunmaktadır.

```
public struct gen
{
    public int makaleKod;
    public int makaleKategoriKod;
    public int makaleBaslikUzunluk;
    public int makaleSpotUzunluk;
    public string makaleSpot;
    public string makaleBaslik;
}
```

Şekil 7. Gen yapısı (Gene structure)

4.3. Birey Yapısı (Choromosome Structure)

Genetik algoritmada kullanılan birey tanımı *IComparable* sınıfından türetilmiştir. Böylece bireylerin uygunluk fonksiyonlarından elde edilen değerlere göre hızlı bir şekilde sıralanabilmeleri sağlanmıştır. Şekil 8'de oluşturulan birey yapısı görülmektedir.

uzunluk özelliği, bireydeki gen sayısını, *gecerliDogruluk* özelliği dışarıdan erişilemeyen yapıda olup çözüme uygunluk derecesini saklamaktadır. Böylelikle uygunluk fonksiyonunun her çalışmasından doğacak kaynak tüketimi en aza indirilmiştir ve birey türetilmesinden hemen sonra doğruluk değeri hesaplanarak saklanmıştır. *bireyGenleri* dizisinde herhangi bir bireye ait genlerin dizilişleri tutulmaktadır, *caprazlanacakBirey* bir birey

```
public class Birey:IComparable
{
    int uzunluk;
    private int gecerliDogruluk = 100;
    private Birey caprazlanacakBirey;
    private Birey yeniBirey1;
    private Birey yeniBirey2;
    public gen[] bireyGenleri = new gen[Populasyon.genHavuzu.Length] ;
//Metodlar
    public Birey CaprazlanacakBirey...
    public int GecerliDogruluk...
    public Birey YeniBirey1...
    public Birey YeniBirey2
    {
        get...
    }
    public Birey()...
    public Birey(gen[] genHavuzu)...
    private void GenleriTanimla(gen[] genHavuzu)...
    public void DogrulukHesapla()...
    public static Random RandomSayiUreteci(int seed) ...
    public Birey Caprazla()...
    public void Mutasyon()...
    private void GenTamircisi()...
    IComparable Members
```

Şekil 8. Birey yapısı (Choromosome structure)

sınıfını göstermektedir ve çaprazlama aşamasında dışarıdan gelen birey nesnesini işaret etmek için kullanılmıştır. *yeniBirey1*, çaprazlamanın sonucunda elde edilen ilk bireyi, *yeniBirey2* ise aynı çaprazlamadan elde edilen ikinci bireyi göstermektedir. *CaprazlanacakBirey*, sadece yazılabilir bir özellik olup, dışarıdan gelen bir *Birey* nesnesini private değişkene aktarmaktadır. *GecerliDogruluk* okunabilir bir özellik olup bireye ait geçerli uygunluk değerini vermektedir. *yeniBirey1* ve *yeniBirey2* özellikleri, sadece okunabilir özellikler olup çaprazlamadan sonra oluşan bireylerin okunmasını sağlamak amacıyla kullanılmıştır. Bunların dışında, birey sınıfına ait iki adet yapılandırıcı (constructor) kullanılmıştır. Bunlardan biri boş bir birey üretirken, diğeri başlangıç aşamasında kullanılmak üzere bireye ait girilen gen bilgilerini alarak yapılandırma işlemini hızlandırmaktadır.

GenleriTanimla metodu boş bir bireyin genlerini gen havuzundan rastgele seçip yeni bir birey oluşturmak için kullanılmaktadır. Bu nedenle, *Birey* metodu sınıfın kendisine özeldir. *BireyCaprazla* metodu public tanımlanmıştır ve nesne dışından da çağrılabilmektedir. Ancak bu metod çağrılmadan önce, *CaprazlanacakBirey* özelliği ile birey nesnesine çaprazlanacağı bireyin atanması gerekmektedir.

Mutasyon metodu, belirtilen mutasyon değerine bağlı olarak ilgili bireyin seçilen geninde mutasyon olup olmayacağını belirler ve gerektiğinde mutasyon işlemini gerçekleştirir. Bu metod, rastgele bir sayı üretip elde edilen sayının aralığına göre rastgele iki genin yerlerini değiştirip değiştirmeyeceğine karar vermektedir. *GenTamircisi* metodu ise genlerin mutasyon sonucunda denetlenmesi ve kromozomda aynı genden birden fazla olması veya bazı genlerin çaprazlama sonucunda kaybolması durumunda tamir edilmesini gerçekleştirmektedir.

RandomSayiUretici metodu random sayı üretecini yeniden başlatmak için kullanılmıştır. Web uygulamaları, her çağrılışında aynı çekirdek değer ile başladıklarından her yeni istek geldiğinde aynı rastsal sayıyı üretirler. Bunu önlemek için, *Birey* sınıfının bu metodunda bir rastgele sayı üreticisi temel kütüphane'den türetilmiş ve http response buffer'a kaydedilerek durumunu her bir istek için koruması sağlanmıştır. Böylece, her seferinde aynı çekirdek değer ile başlayan rastsal sayıların üretimi engellenmiştir.

4.4. Popülasyon Tanımı (Population Description)

Popülasyon sınıfı public olarak tanımlanan *genHavuzu* adında bir gen dizisinden oluşmaktadır. Bu dizi *GenHavuzu()* adlı metod tarafından veritabanına bağlanılarak oluşturulmaktadır. Bu diziyi kullanmaktaki amaç, her genden sadece bir tane bulundurarak, eksik genin belirlenmesini

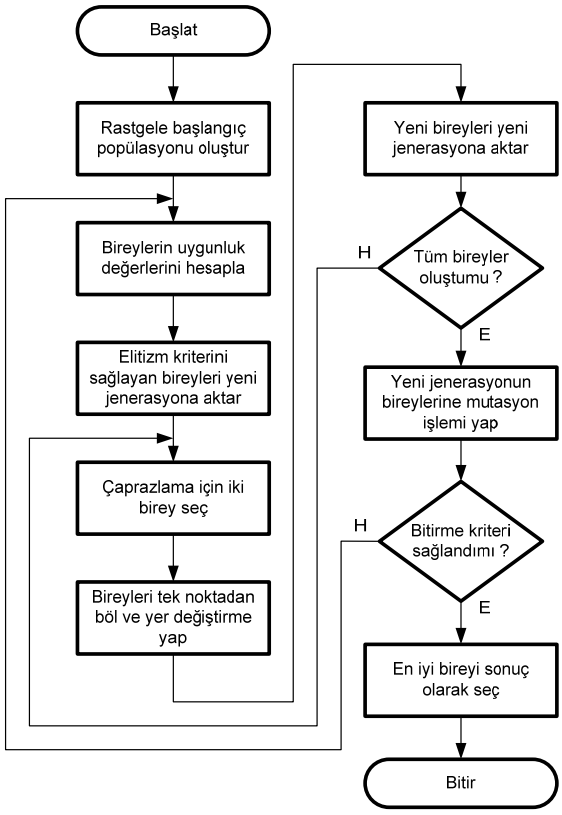
```
public class Populasyon
{
    int populasyondakiBireySayisi= 100;
    public static gen[] genHavuzu;
    public static float mutasyonIhtimali = 0.01f;
    private int bireySayisi;
    public int IterasyonSayisi=10;
    private ArrayList Bireyler=new ArrayList();
    public int elitizmKriteri = 2;
    public Populasyon(int bireySayisi)...
    private void YeniPopulasyon()...
    public Birey EnIyiBirey()...
    private void GenHavuzu()...
}
```

Şekil 9. Populasyon sınıfı tanımlamaları (Population class definition)

sağlamaktadır. Şekil 9'da popülasyon sınıfına ait tanımlamalar görülmektedir.

mutasyonIhtimali değeri bir float değer olup, %0.01 başlangıç değeri ile tanımlı ve bir genin mutasyona uğrama ihtimalini belirlemek için kullanılmaktadır. *bireySayisi* değişkeninde, popülasyonun bulunduracağı birey sayısı, *IterasyonSayisi* değişkeninde ise, genetik algoritmanın kaç tane yeni nesil üreteceği saklanmakta olup başlangıç değeri olarak 10 alınmıştır.

YeniPopulasyon metodu, asıl genlerin değişimini ve bireylerin gelişimini sağlamaktadır. Bu metod ile sırayla bütün bireyler için uygunluk değeri hesaplanmakta, ardından en iyi bireyler elitizm kriteri dahilinde yeni nesile aktarılmaktadır. Daha sonra boş bireyler için rastgele seçilen iki bireyin çaprazlanması ile elde edilen iki yeni birey aktarılarak yeni nesil oluşturulmakta ve son olarak da yeni nesil mutasyona



Şekil 10. Uygulama yazılımının akış şeması (Flow chart of application software)

tabi tutulmaktadır. Bütün bu işlemler, iterasyon sayısı kadar tekrarlanmaktadır. *EnIyiBirey* metodu, mevcut popülasyonu ceza puanına göre sıralayarak en az ceza alan bireyi elde etmektedir. Gerçekleştirilen uygulama yazılımına ait akış şeması Şekil 10'da görülmektedir.

5. DENEYSEL SONUÇLAR (EXPERIMENTAL RESULTS)

Genetik algoritmayla gerçekleştirilen web sayfası optimizasyonunun deneysel çalışmaları Intel Centrino 1.3GHz işlemciye ve 512MB hafızaya sahip bir bilgisayar kullanılarak elde edilmiştir. Programlama dili olarak Visual C#.NET ve veritabanı olarak MS Access ilişkisel veritabanı kullanılmıştır. Windows XP işletim sisteminde çalışılmış ve ASP.NET sayfaları oluşturulmuştur. Sayfalar Internet Explorer 6.0 kullanılarak görüntülenmiştir.

Servis sunucu olarak Internet Information Server 6.0 kullanılmıştır. Sonuçlar her bir deney için 100 çalıştırmanın ortalaması alınarak hesaplanmıştır. Toplam 3800 farklı çalıştırmanın sonucu kaydedilmiş ve grafikler bu değerler kullanılarak elde edilmiştir.

Deneysel çalışmalar üç grup altında yapılmıştır. Her grupta toplam 100 farklı çalıştırma yapılarak sonuçlar elde edilmiştir. Birinci grupta farklı birey sayısına göre uygunluk değerlerinin değişimi minimum, ortalama ve maksimum değerleriyle elde edilmiştir. Bu uygulamada gerçek zamanlı çalışma olduğu için sonuca ulaşma süresi çok önemlidir. Bu nedenle, ikinci grupta farklı birey sayılarına göre çalışma sürelerinin değişimi minimum, ortalama ve

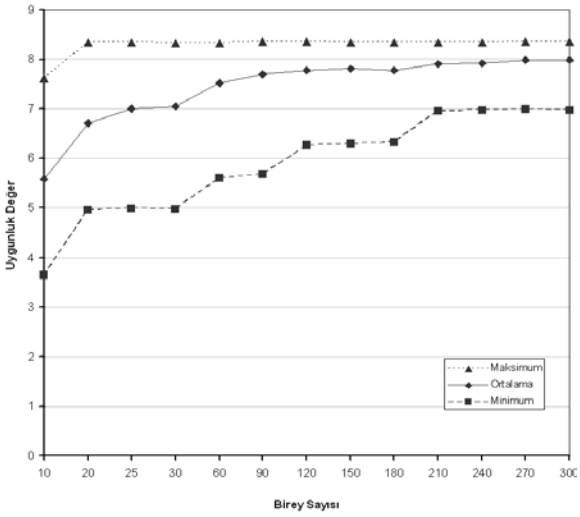
Çizelge 2. Değişik birey sayıları için minimum, maksimum ve ortalama uygunluk değerleri (Minimum, maximum and average fitness values for different choromosome number)

Birey Sayısı	Uygunluk Değeri (Min.)	Uygunluk Değeri (Mak.)	Uygunluk Değeri (Ort.)
10	3.64	7.61	5.59
20	4.96	8.34	6.71
25	4.99	8.34	7.01
30	4.98	8.32	7.06
60	5.61	8.33	7.53
90	5.68	8.35	7.69
120	6.27	8.35	7.78
150	6.29	8.34	7.81
180	6.32	8.34	7.78
210	6.95	8.34	7.91
240	6.97	8.34	7.92
270	6.99	8.35	7.99
300	6.97	8.35	7.98

Çizelge 3. Değişik birey sayıları için çalışma süreleri (Running time for different choromosome number

Birey Sayısı	Minimum Çalışma Süresi (sn)	Maksimum Çalışma Süresi (sn)	Ortalama Çalışma Süresi (sn)
10	0.12	0.34	0.25
20	0.25	0.53	0.37
25	0.31	0.60	0.44
30	0.33	0.60	0.44
60	0.64	1.00	0.71
90	0.94	1.21	1.01
120	1.23	1.56	1.33
150	1.51	2.18	1.66
180	1.82	2.18	1.95
210	2.14	2.63	2.28
240	2.48	3.36	2.69
270	2.83	3.27	2.98
300	3.13	3.81	3.33

maksimum olarak elde edilmiştir. Üçüncü grupta farklı mutasyon oranlarında uygunluk değişim değerleri ve çalışma süreleri alınmıştır.



Şekil 11. Değişik birey sayıları için minimum, maksimum ve ortalama uygunluk değerleri (Minimum, maximum and average fitness values for different choromosome number)

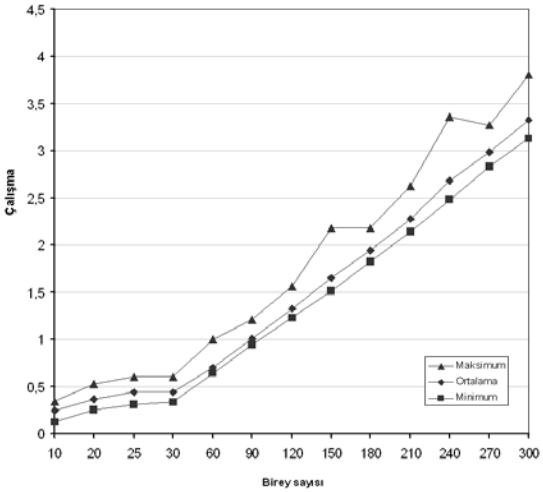
Uygunluk değeri değişimine ilişkin test değerleri farklı birey sayıları için 100 çalışmayı içermektedir. Ortalama değerler 100 çalıştırmanın sonucunda elde edilmiş ve değişim grafik olarak sunulmuştur. Çizelge 2'de 10, 20, 25, 30, 60, 90, 120, 150, 180, 210, 240, 270, 300 birey sayılarına sahip popülasyonlar için 100 farklı çalıştırmanın minimum, maksimum ve ortalama uygunluk değerleri görülmektedir. Şekil 11'de ise Çizelge 2'deki değerlere ait grafik görülmektedir.

Şekil 11'den görüldüğü gibi popülasyondaki birey sayısı arttıkça uygunluk değerinde artış olmaktadır. En iyi ortalama uygunluk değeri 270 birey için elde edilmiştir. Birey sayısı 300 değerinin üzerinde arttıkça ortalama uygunluk değerinde azalma olmaktadır.

Çizelge 3'te 10,20,25,30,60,90,120,150,180,210,240, 270,300 birey sayılarına sahip popülasyonlar için 100 farklı çalıştırmanın minimum, maksimum ve ortalama çalışma süreleri görülmektedir. Şekil 12'de ise Çizelge 3'teki değerlere ait değişim grafiği görülmektedir.

Şekil 12'den görüldüğü gibi popülasyondaki birey sayısı arttıkça çalışma süresinde artış olmaktadır. Birey sayısı 30'a kadar daha yavaş artış olmakta ancak 30'un üzerinde artış daha hızlı olmaktadır.
Mutasyon değerinin değişiminin sonuca etkisine yönelik olarak iki farklı popülasyon boyutu için deneyler yapılmıştır. İlk deneyin sonuçlarına göre en iyi uygunluk değerleri 150 ve 270 birey sayısına sahip popülasyonlarda elde edilmiştir. Bu yüzden farklı mutasyon oranlarına ait sonuçlar bu iki popülasyon için alınmıştır. Mutasyon oranı 0.0001,0.001 ve 0.005 olarak üç farklı değerde alınmıştır. Çizelge 4'te 150 ve 270 birey için 0.0001,0.001 ve 0.005 mutasyon oranlarında elde edilen minimum, ortalama ve maksimum uygunluk değerleri görülmektedir.

Çizelge 4'te görüldüğü gibi farklı mutasyon oranları uygunluk değerinde fazla iyileşme yapmamaktadır. Bu sonucun seçilen birey sayılarında genetik çeşitliliğin yeterli seviyede olmasından kaynaklandığı söylenebilir. Çizelge 5'te ise farklı mutasyon oranları için çalışma süreleri görülmektedir.



Şekil 12. Değişik birey sayıları için minimum, maksimum ve ortalama çalışma süreleri (Minimum, maximum and average running time for different choromosome number)

Çizelge 4. Farklı mutasyon oranlarında uygunluk değerleri (Fitness values for different mutation rate)

Mutasyon Oranı	Birey Sayısı	Uygunluk Değeri (Min.)	Uygunluk Değeri (Mak.)	Uygunluk Değeri (Ort.)
0.0001	150	6.32	8.33	7.63
	270	6.97	8.35	7.68
0.0010	150	6.30	8.34	7.62
	270	6.99	8.35	7.74
0.0050	150	5.66	8.34	7.69
	270	6.96	8.35	7.65

Çizelge 5. Farklı mutasyon oranlarında çalışma süreleri (Running time for different mutation rate)

Mutasyon Oranı	Birey Sayısı	Minimum Çalışma Süresi(sn)	Maksimum Çalışma Süresi(sn)	Ortalama Çalışma Süresi(sn)
0.0001	150	1.50	1.76	1.63
	270	2.84	4.11	3.19
0.0010	150	1.53	1.89	1.65
	270	2.94	4.18	3.27
0.0050	150	1.50	1.79	1.64
	270	2.87	4.24	3.16

Çizelge 5'te aynı birey sayısında mutasyon oranı değişiminin çalışma süresine etkisi olmadığı görülmektedir. Ancak farklı birey sayılarında aynı mutasyon değerleri için çalışma süreleri çok değişmektedir. Birey sayısı arttıkça gen seviyesinde yapılan mutasyon işleminin olma olasılığı artmaktadır ve her yapılan mutasyon çalışma süresini uzatmaktadır.

Şekil 13. Birey sayısı 10 için örnek sayfa (Sample page for 10 choromosomes)

Şekil 14. Birey sayısı 25 için örnek sayfa (Sample page for 25 choromosomes)

Şekil 15. Birey sayısı için 270 örnek sayfa (Sample page for 270 choromosomes)

Şekil 13'te 10 birey, Şekil 14'te 25 birey ve Şekil 15'te ise 270 birey sayısına sahip popülasyonlar için elde edilen örnek sayfalar görülmektedir.

Şekil 13 ve Şekil 14'te görüldüğü gibi 10 ve 25 birey sayılarında boş satırlar bulunmaktadır. Ancak Şekil 15'te görüldüğü gibi birey sayısı 270 alındığında sayfada en iyi görünüme ulaşılmakta ve boş satır kalmamaktadır.

Popülasyon boyutunun 150 olması durumunda çalışma süresinin 1,5 sn ve 270 olduğu durumda ise 3 sn olduğu görülmüştür. Popülasyon boyutunun 150'nin üzerine çıkması durumunda uygunluk değerinde çok fazla iyileşme olmamakta ancak çalışma süresinde artış çok olmaktadır. Elde edilen deneysel sonuçlardan mutasyon oranının 150 ve 270 birey sayısı için uygunluk değerine etki etmediği ancak çalışma süresini artırdığı görülmüştür. Birey sayısı çok azaltılarak yapılan mutasyon işlemininde

her çalıştırmada çok farklı sonuçlar oluştuğu ve sonuçtaki kararlılığın kaybolduğu gözlenmiştir. Deneysel sonuçlar genetik algoritmanın 150 birey sayısı ve 0,001 mutasyon oranıyla 1,5 sn'lik bir çalışma süresi elde edilerek bu tür problemlerde çok iyi sonuçlar verdiğini göstermiştir.

6. SONUÇLAR (CONCLUSIONS)

Bu çalışmada, genetik algoritma kullanılarak bir sayfada bulunan makale özetlerinin iki sütun halinde en az yer kaplayacak şekilde gerçek zamanlı olarak oluşturulması gerçekleştirilmiştir. Sayfalar oluşturulurken makalelerin güncelliğide göz alınmış ve daha güncel olanların daha yukarıda yer alması sağlanmıştır. Yapılan deneysel çalışmalar genetik algoritmanın popülasyon tabanlı bir algoritma olmasına karşın gerçek zamanlı İnternet sayfası optimizasyonunda başarılı olduğunu göstermiştir.

KAYNAKLAR (REFERENCES)

1. Hopper, E., **Two-Dimensional Packing Utilizing Evolutionary Algorithms and Other Meta-Heuristic Methods**, Ph.D.Thesis, Cardiff University, U.K., 2000.
2. Hopper, E., Turton, B.C.H, "An Empirical Investigation of Meta-Heuristic and Heuristic Algorithms for a 2D Packing Problem", **European Journal of Operational Research**, Vol.128, No.1, pp.34-57, 2001.
3. Hwang, S.M., Kao, C.Y., Horng, J.T., "On Solving Rectangle Bin Packing Problems Using Genetic Algorithms", **IEEE International Conference on Systems, Man, and Cybernetics-Humans, Information and Technology**, Vol.2, pp.1583-1590, 1994.
4. Kröger, B., Schwenderling, P., Vornberger, O., "Parallel Genetic Packing of Rectangles", **In Proc. of 1st PPSN'90**, pp.160-164, 1990.
5. Lesh, N.B., Marks, J.W., McMahon A., Mitzenmacher, M., "New Heuristic and Interactive Approaches to 2D Rectangular Strip Packing", **International Joint Conference on Artifical Intelligence (IJCAI), Workshop on Stochastic Search Algorithms**, TR2003-018, August 2003.
6. Lesh, N., Marks, J., McMahon, A., Mitzenmacher, M., "Exhaustive Approaches to 2D Rectangular Perfect Packings", **Inf. Process. Lett.**, Vol.90, No.1, pp.7-14, 2004.
7. Iori, M., Martello, S. and Monaci, M. 2003. "Metaheuristic Algorithms for the Strip Packing Problem", in P. M. Pardalos, V. Korotkikh, Eds., **Optimization and Industry: New Frontiers**, Kluwer Academic Publishers, Boston, MA, pp.159-179, 2003.
8. Bortfeldt, A., "A Genetic Algorithm for the Two-Dimensional Strip Packing Problem with Rectangular Pieces", **European Journal of Operational Research**, Vol.172, No.3, pp.814-837, 2006.
9. Hadjiconstantinou, E., Iori, M., "A genetic algorithm for the two dimensional knapsack problem", **Technical Report OR/04/5 DEIS**, University of Bologna, Italy, 2004.
10. Gonzalez, J., Rojas, I., Pomares, H., Salmeron, M., Merelo, J.J., "Web Newspaper Layout Using Simulated Annealing", **IEEE Transactions on Systems, Man And Cybernetics-Part B**, Vol.32, No.5, pp.686-692, 2002.
11. Canbek, G., Akcayol, M.A., "Internet Gazetesi Sayfa Düzeninin Gerçek Zamanlı Optimizasyonunun Tavlama Benzetimi Algoritmasıyla Gerçeklenmesi", **Journal of the Faculty of Engineering and Architecture of Gazi University**, Vol.21, No.2, pp.341-348, 2006.
12. Alander, J.T., "On Optimal Population Size of Genetic Algorithms", **In Proceedings of CompEuro 92**, IEEE Computer Society Press, pp.65-70, 1992.
13. Goldberg, D.E., "Optimal Population Size for Binary-Coded Genetic Algorithms", **TCGA Report, No.85001**, University of Alabama, 1985.
14. Khor, E.F., Tan, K.C., Wang, M.L. and Lee, T.H., "Evolutionary Algorithm with Dynamic Population Size for Multi-Objective Optimization", **26th Annual Conference of the IEEE Industrial Electronics Society**, Vol.3, pp.1686-1691, 2000.
15. Leung, K.S., and Liang, Y., "Adaptive Elitist-Population Based Genetic Algorithm for Multimodal Function Optimization", **Genetic and Evolutionary Computation Conference-GECCO 2003 Proceedings Part I**, Springer, Lecture Notes in Computer Science Vol.2723, pp.1160-1171, July 2003.